



# FACULTAD DE INGENIERÍA

---

## INGENIERÍA DE SISTEMAS COMPUTACIONALES

### “Informe: Colaboración en Git y GitHub”

Práctica de Campo 2

**Grupo 12:**

ASMAT DE LA CRUZ, ANTHONI BRYAN (N00405749)

CAMPOS LIPA, JOHN LENIN (N00400607)

LACHIRA PINGO, MIGUEL ORLANDO (N00268489)

ROJAS YSLA, DAVID ALEXANDER (N00331763)

ROMERO BARRETO, JUAN LUIS (N00399596)

**Curso:**

Técnicas de Programación Orientada a Objetos

**Docente:**

Martín Eduardo Torres Martínez

**Lima – Perú**

**2025-2**

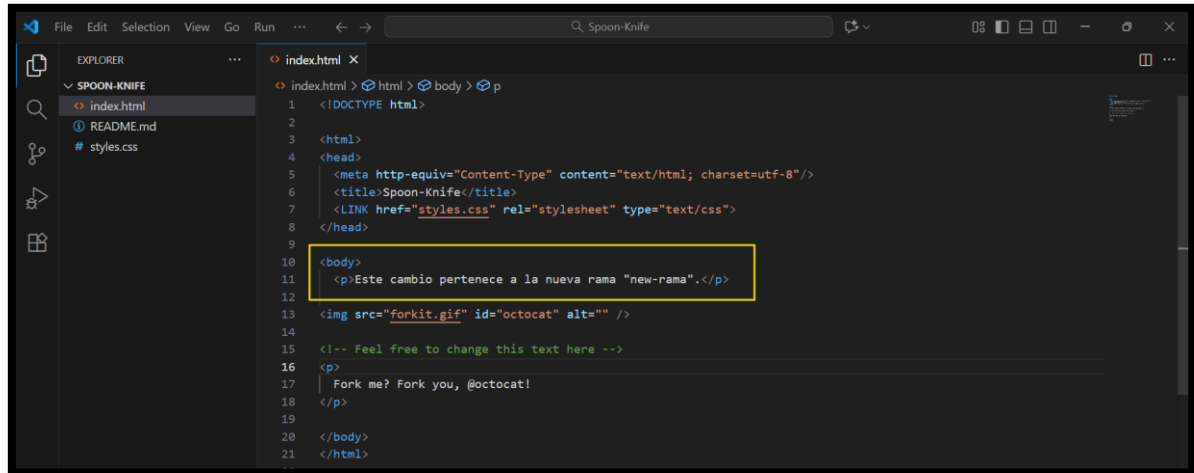
## Contenido

<b>Cambios y fusión de ramas: Merge .....</b>	<b>2</b>
<b>Resolución de Conflictos .....</b>	<b>4</b>
<b>Colaboración con GitHub.....</b>	<b>9</b>
1. Invitar a Compañeros a Colaborar en el Repositorio .....	11
2. Clonar un Repositorio Compartido.....	12
3. Pull request: realizar cambios en una rama y crear un Pull Request en GitHub. ....	13
4. Evidenciar Revisión de Código.....	16
<b>Repositorio del Grupo 12 .....</b>	<b>18</b>
<b>Evidencias del trabajo colaborativo .....</b>	<b>18</b>
1. Portada del repositorio.....	18
2. Colaboradores .....	19
3. Colaboración real: Caso 1 “Lectura de datos simples con Scanner” .....	19
4. Colaboración real: Caso 2 “Clase Estudiante con atributos privados” .....	20
5. Colaboración real: Caso 3 “Clase CuentaBancaria con validación” .....	20

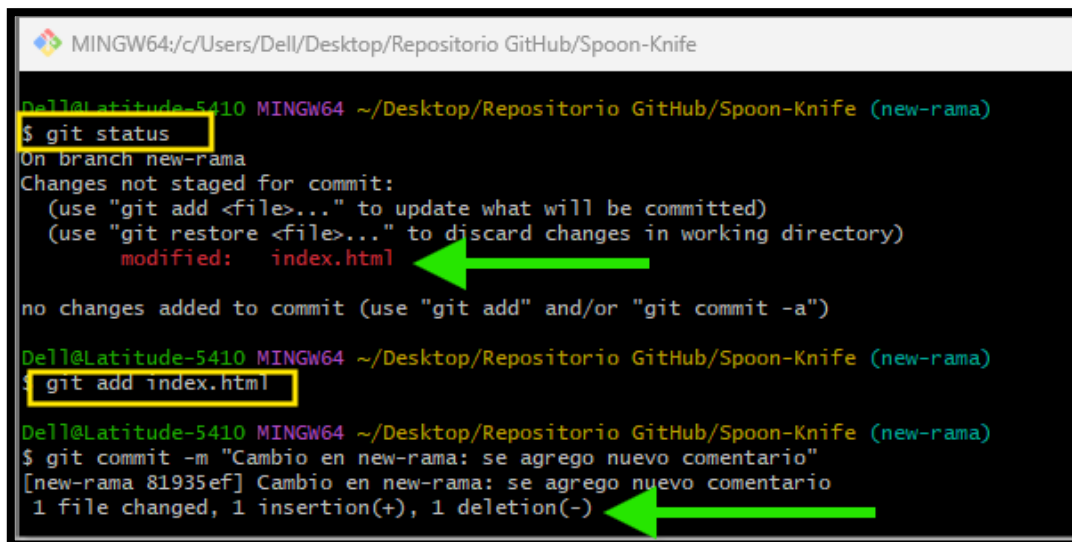
## Cambios y fusión de ramas: Merge

Ahora realizaremos los merge de Branch, es decir, la fusión de ramas, para ello abrimos el proyecto “Spoon-Knife” en visual studio code y modificamos nuevamente el archivo index.html.

Ojo: esta modificación se realiza estando en la nueva rama “new-rama”



Ahora validamos el repositorio:



- ➔ **Git status** (validamos el estado del repositorio y efectivamente nos indica que hay un cambio realizado en el archivo index.html, pero no se encuentra en STAGING).
- ➔ **Git add** (Agregamos los cambios al área de preparación (STAGING)).
- ➔ **Git commit -m “comentario”** (Registramos con un mensaje descriptivo).

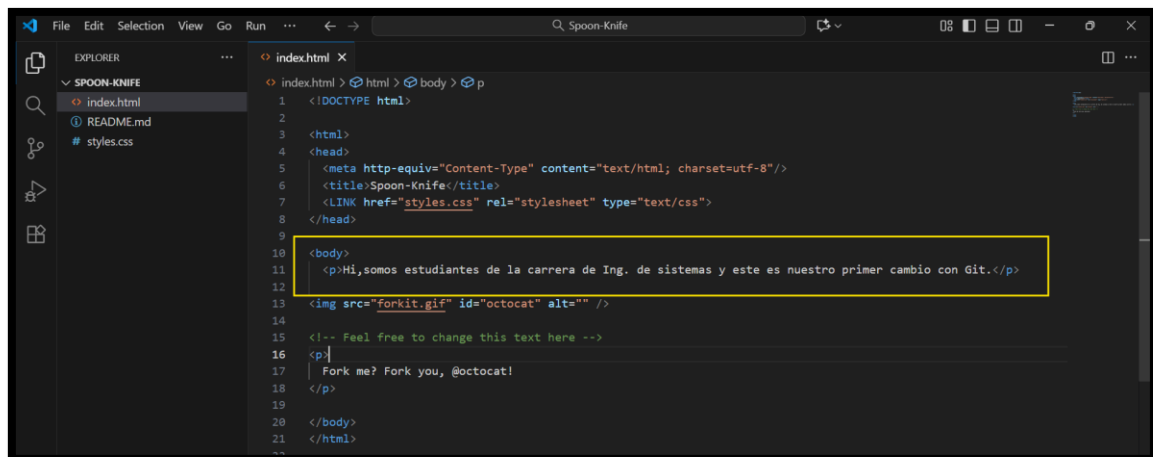
Ahora volvemos a la rama “main”

```
Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (new-rama)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main)
$ git branch
* main
  new-rama
```

- ➔ **Git checkout main** (Volvemos a la rama main).
- ➔ **Git branch** (validamos en que rama nos encontramos y efectivamente nos muestra las ramas que tenemos, el asterisco en la rama main (\*) nos indica que es la rama actual).

Ahora que nos encontramos nuevamente en la rama principal “main”, abrimos nuevamente el archivo index.html en VS Code, podemos observar que el cambio de la rama nueva no aparece aun, esto pasa porque nos encontramos en otra rama.



Ahora realizaremos merge(fusionar) la rama nueva “new-rama” con la rama principal “main”

```
Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main)
$ git merge new-rama
Updating 8b583fb..81935ef
Fast-forward
 index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

- ➔ **Git merge new-rama** (Este comando nos permite fusionar las ramas existentes “new-rama” y “main”, el mensaje nos indica actualizando y al final nos indica 1 archivo modificado, 1 inserción, 1 eliminación).

Ahora podemos validar si efectivamente se fusionaron correctamente ambas ramas, para ello abriremos nuevamente el archivo index.html y efectivamente ahora ya podemos ver el cambio del comentario en <body>.

```

1  <!DOCTYPE html>
2
3  <html>
4  <head>
5    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
6    <title>Spoon-Knife</title>
7    <link href="styles.css" rel="stylesheet" type="text/css">
8  </head>
9
10 <body>
11   <p>Este cambio pertenece a la nueva rama "new-rama".</p>
12
13   
14
15   <!-- Feel free to change this text here -->
16   <p>
17     Fork me? Fork you, @octocat!
18   </p>
19
20 </body>
21 </html>

```

## Resolución de Conflictos

Este punto es importante porque representa una situación real de trabajo en equipo:

- Dos personas (o dos ramas) modifican la misma parte de un archivo.
- Git no sabe cuál versión conservar y te pide resolverlo manualmente.

Ahora vamos a simular un conflicto.

Crearemos una nueva rama “rama-conflicto”

```

DellLatitude 5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main)
$ git checkout -b rama-conflicto
Switched to a new branch 'rama-conflicto'

```

➔ **Git checkout -b rama-conflicto** (El parámetro -b significa crear y moverme a una nueva rama, en este caso “rama-conflicto”).

Ahora modificamos el archivo index.html, la siguiente línea y guardamos:

**<body>**

**<p>Texto modificado en rama-conflicto. </p>**

Ahora en la terminal:

```
Dell@latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (rama-conflicto)
$ git status
On branch rama-conflicto
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:   index.html

no changes added to commit (use "git add" and/or "git commit -a")

Dell@latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (rama-conflicto)
$ git add index.html
```

- ➔ **Git status** (Validamos el estado del repositorio, en este caso nos indica que se modificó el archivo index.html, pero que no está en STAGING).
- ➔ **Git add index.html** (Agregamos el archivo a STAGING, ahora está listo para el commit).

Ahora realizamos el commit:

```
Dell@latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (rama-conflicto)
$ git commit -m "Modificación rama-conflicto"
[rama-conflicto 10a6a2d] Modificación rama-conflicto
1 file changed, 1 insertion(+), 1 deletion(-)

Dell@latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (rama-conflicto)
$ git log
commit 10a6a2d133e285bdee08c88641e4423e7bc97025 (HEAD -> rama-conflicto)
Author: Lenin Campos <jcampos1810@gmail.com>
Date: Tue Oct 21 16:56:39 2025 -0500

    Modificación rama-conflicto
```

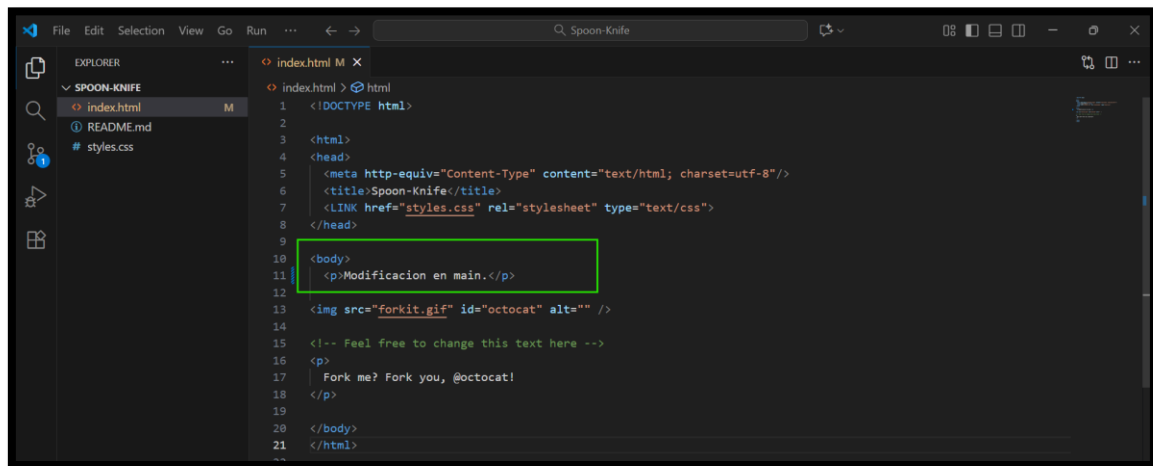
- ➔ **Git commit -m "Modificación rama-conflicto"** (Registramos oficialmente el cambio, junto a ello un comentario descriptivo de dicho cambio).

Ahora regresamos a la rama principal, en este caso a la rama main:

```
MINGW64:/c:/Users/Dell/Desktop/Repositorio GitHub/Spoon-Knife
Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (rama-conflicto)
$ git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)
```

➔ *Git checkout main* (Regresamos a la rama principal “main”).

Ahora en la rama principal, editamos la misma línea del mismo archivo index.html



```
index.html M
1 <!DOCTYPE html>
2
3 <html>
4 <head>
5   <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
6   <title>Spoon-Knife</title>
7   <LINK href="styles.css" rel="stylesheet" type="text/css">
8 </head>
9
10 <body>
11   <p>Modificacion en main.</p>
12
13   
14
15   <!-- Feel free to change this text here -->
16   <p>
17     Fork me? Fork you, @octocat!
18   </p>
19
20 </body>
21 </html>
```

Ahora en la terminal:

```
Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)

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:   index.html
no changes added to commit (use "git add" and/or "git commit -a")

Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main)
$ git add index.html
```

➔ *Git status* (Validamos el estado del repositorio y nos indica una modificación en el archivo index.html).

➔ *Git add index.html* (Agregamos a STAGING index.html).

Realizamos el commit:

```
Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main)
$ git commit -m "Cambio en Main"
[main 0845d3d] Cambio en Main
1 file changed, 1 insertion(+), 1 deletion(-)
```

→ *Git commit -m "Cambio en el main"* (Registramos oficialmente el cambio, junto a ello un comentario descriptivo de dicho cambio).

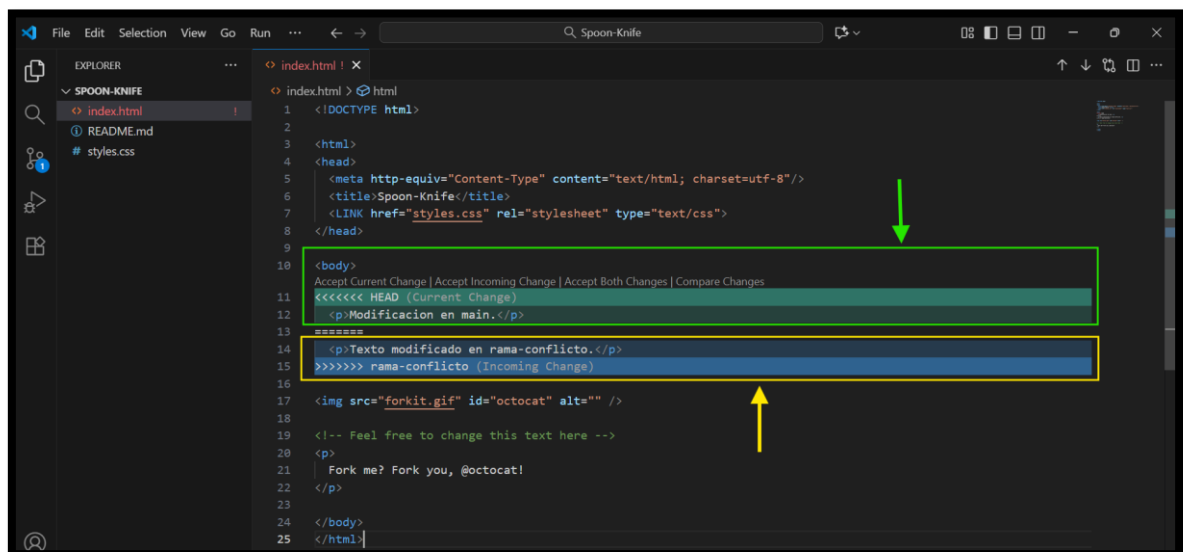
Ahora intentamos fusionar las ramas:

```
Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main)
$ git merge rama-conflicto
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

→ *Git merge rama-conflicto* (Al momento de fusionar las ramas, git nos indica que hay un conflicto(contenido) en el archivo index.html y nos indica corregir los conflictos y enviemos los resultados).

Abrimos el archivo index.html y vemos los siguiente:

Nos muestra los cambios realizamos en la misma línea, pero que se realizaron en diferentes ramas "main" & "rama-conflicto", aquí tenemos que corregir manualmente el conflicto, es decir, eliminamos la línea de conflicto que queramos.





Ahora procedemos a resolver el conflicto manualmente y para este caso quedara de siguiente manera:

Ahora corregido manualmente el conflicto, procedemos a realizar el commit:

```
Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main|MERGING)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
(use "git push" to publish your local commits)

You have unmerged paths.
(fix conflicts and run "git commit")
(use "git merge --abort" to abort the merge)

Unmerged paths:
(use "git add <file>..." to mark resolution)
    both modified:   index.html

no changes added to commit (use "git add" and/or "git commit -a")

Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main|MERGING)
$ git add index.html

Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main|MERGING)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
(use "git push" to publish your local commits)

All conflicts fixed but you are still merging.
(use "git commit" to conclude merge)

Changes to be committed:
    modified:   index.html

Dell@Latitude-5410 MINGW64 ~/Desktop/Repositorio GitHub/Spoon-Knife (main|MERGING)
$ git commit -m "Conflicto resuelto entre main y rama-conflicto"
[main 3623037] Conflicto resuelto entre main y rama-conflicto
```

- ➔ **Git status** (Validamos el repositorio, nos indica que hay un archivo modificado, pero que no está en STAGING).
- ➔ **Git add index.html** (Agregamos el archivo index.html a STAGING).

➔ *Git commit -m “Conflicto resuelto entre main y rama-conflicto”* (Registramos oficialmente el cambio, junto a ello un comentario descriptivo de dicho cambio y git nos confirma que el conflicto fue resuelto).

## Colaboración con GitHub

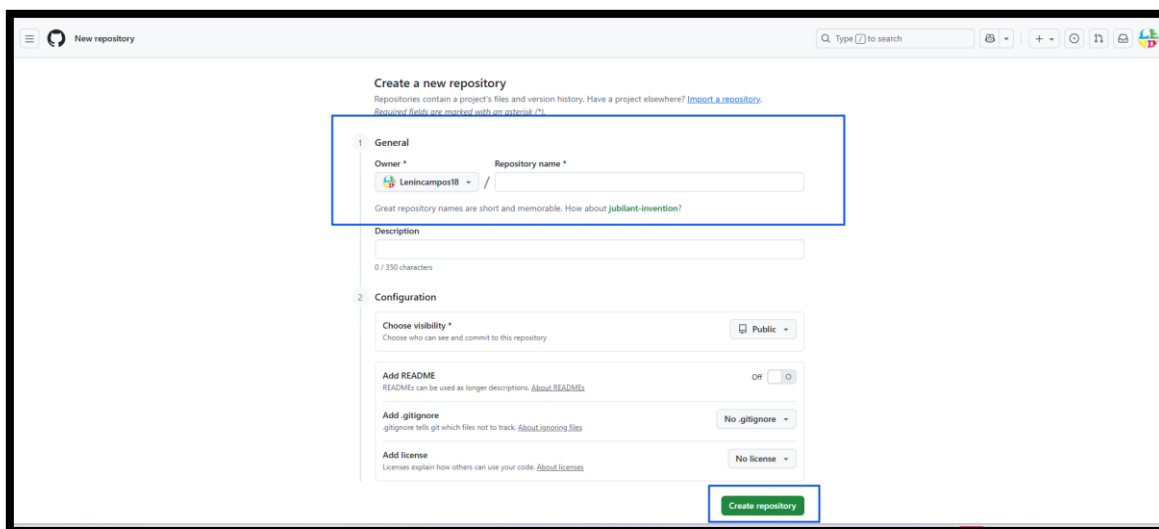
Ahora para colaborar en GitHub con miembros del equipo necesitamos subir nuestro repositorio local a la plataforma de GitHub.

Para ello utilizaremos el repositorio local el cual tiene como nombre “Proyecto\_Python”.

Primero nos dirigimos a la plataforma GitHub, creamos un repositorio vacío:

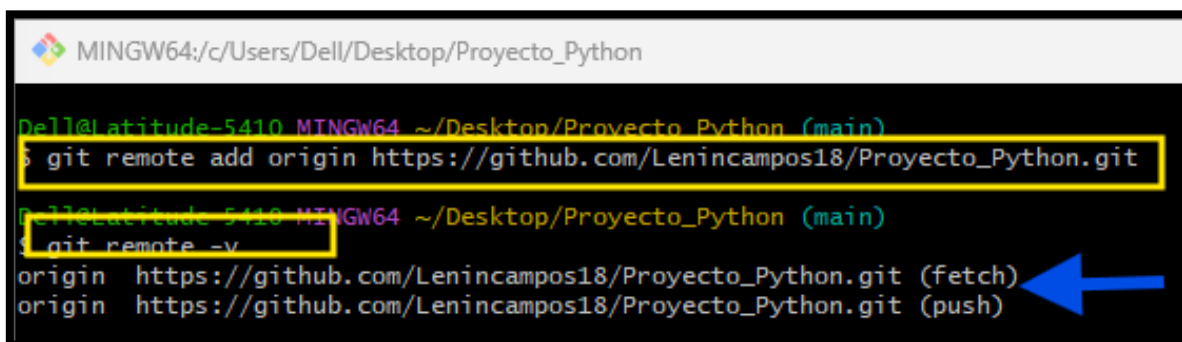
<https://github.com/>

En la esquina superior derecha, clic en New repository



- ➔ Ponemos el nombre tal cual, del repositorio local, para nuestro caso “Proyecto\_Python”
- ➔ Clic CREATE REPOSITORY.

Listo una vez creado el repositorio, ahora necesitamos vincular el repositorio local con el de GitHub, para ello nos dirigimos al terminal Git bash:



```

MINGW64:/c/Users/Dell/Desktop/Proyecto_Python
Dell@Latitude-5410 MINGW64 ~/Desktop/Proyecto_Python (main)
$ git remote add origin https://github.com/Lenincampos18/Proyecto_Python.git
Dell@Latitude-5410 MINGW64 ~/Desktop/Proyecto_Python (main)
$ git remote -v
origin https://github.com/Lenincampos18/Proyecto_Python.git (fetch)
origin https://github.com/Lenincampos18/Proyecto_Python.git (push)
  
```

- ➔ **Git remote add + URL del repositorio en GitHub** (Este comando nos permite vincular el repositorio local con el de GitHub).
- ➔ **Git remote -v** (Nos permite validar las conexiones remotas que tiene nuestro repositorio, ahí nos indica que tenemos permiso para descargar (Fetch) como para subir (push) cambios al repositorio, y la URL es la misma para ambas operaciones).

Ahora necesitamos subir el proyecto a GitHub:

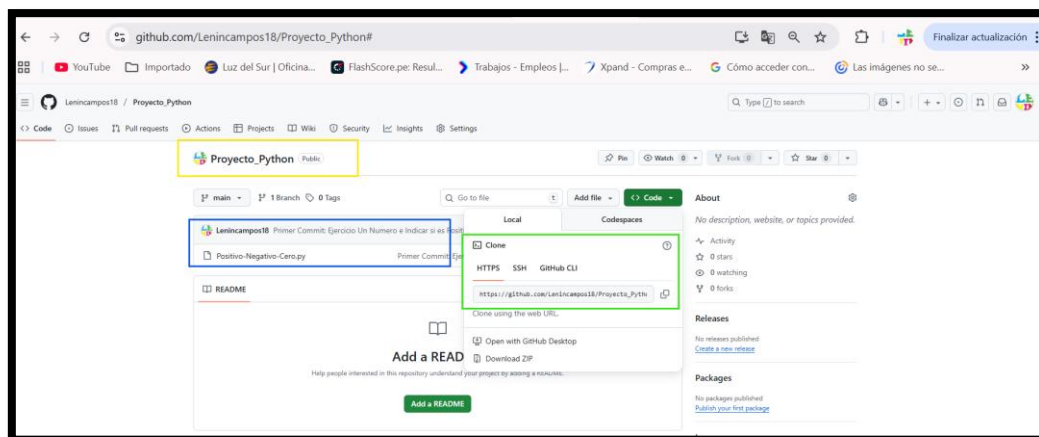
```
Dell@Latitude-5410 MINGW64 ~/Desktop/Proyecto_Python (main)
$ git branch -M main

Dell@Latitude-5410 MINGW64 ~/Desktop/Proyecto_Python (main)
$ git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 448 bytes | 448.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Lenincampos18/Proyecto_Python.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

- ➔ **Git branch -M main** (Renombra la rama local actual (se encontraba como Master) a main. El uso de -M asegura que el cambio se realice sin importar si la rama main existía o no).
- ➔ **Git push -u origin main** (Sube por primera vez la rama main de nuestro repositorio local al repositorio remoto llamado origin y configura una conexión de seguimiento para simplificar los futuros comando push y pull). **Aquí te pedirá por primera vez iniciar sesión o autorizar tu cuenta.**

Ahora verificamos en GitHub:

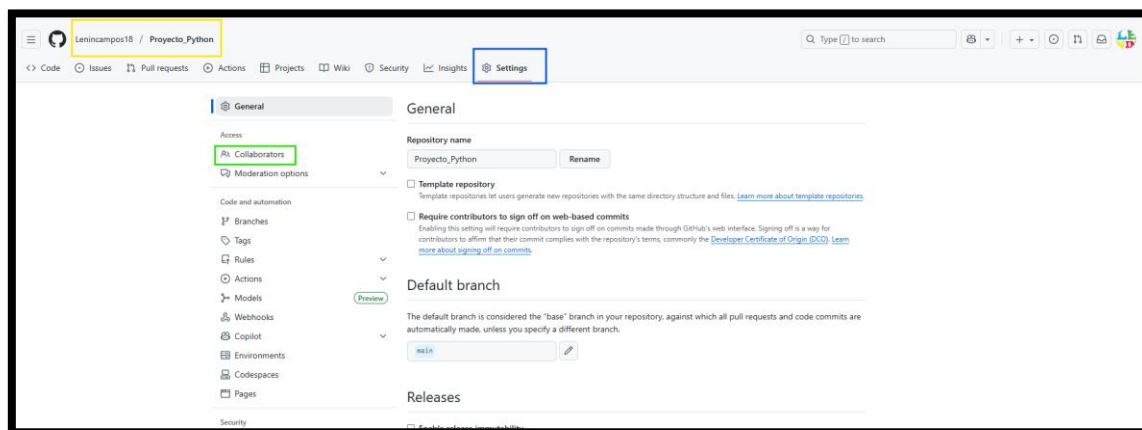
Efectivamente ya tenemos el repositorio local “Proyecto\_Python”, se observan los archivos del repositorio, la rama “main” y cuando nos dirigimos a código encontramos el URL del repositorio.



## 1. Invitar a Compañeros a Colaborar en el Repositorio

Ahora agregaremos colaboradores en GitHub, en este caso a los miembros que conforman el proyecto:

Para ello necesitamos enviar las invitaciones desde la plataforma GitHub



- ➔ Ingresamos al repositorio en GitHub (Proyecto\_Python).
- ➔ Pestaña “Settings” (Configuración).
- ➔ En el menú lateral, seleccionamos **Collaborators** y clic en “Add People”.
- ➔ En “Add People” escribimos los nombres de usuario de GitHub o el correo con el que se registraron de los integrantes del equipo (Miguel Lachira, David Rojas, Anthony Asmat y Juan Romero)
- ➔ Agregados a todos los colaboradores clic “Add to this repository”.

Verificamos que los colaborados acepten la invitación

Para ello cada compañero debe:

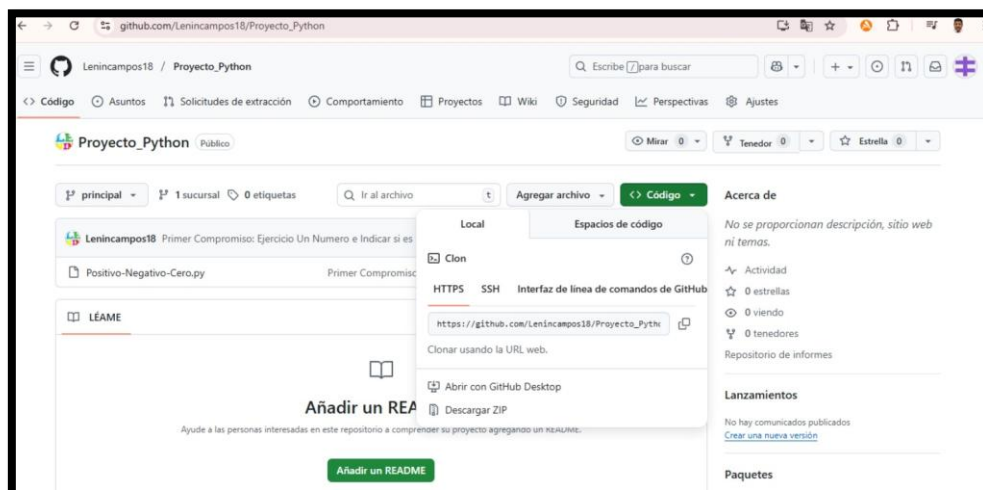
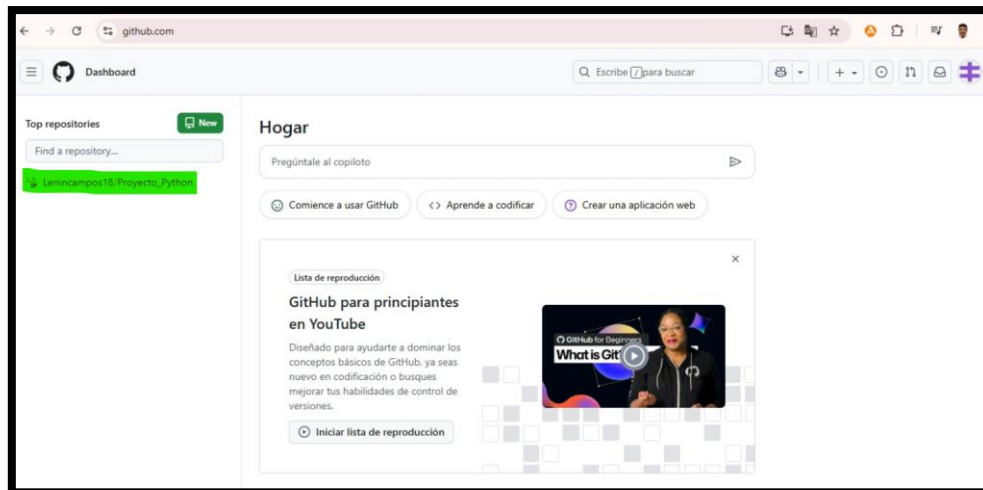
- ✓ Iniciar sesión en GitHub
- ✓ Aceptar la invitación al repositorio

Una vez aceptada, podrán:

- Clonar el repositorio con Git Clone
- Crear ramas
- Subir cambios(push)
- Hacer commit

Aceptación de invitación al repositorio de los miembros del equipo:

**Ejemplo: miembro del equipo: Miguel Lachira**



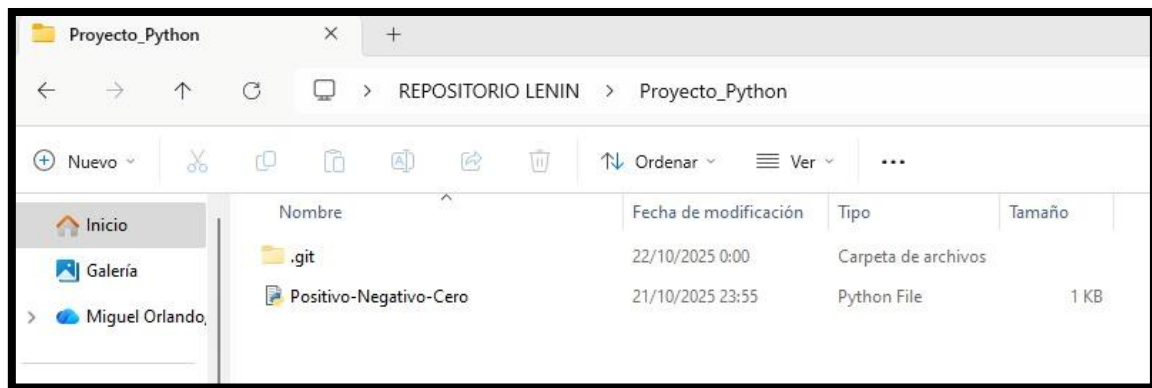
## 2. Clonar un Repositorio Compartido.

Ahora los miembros del equipo pueden clonar el repositorio compartido "Proyecto\_Python" localmente, mediante el terminal Gitbash.

**Miembro del Equipo: Miguel Lachira**

```
Miguel Lachira P@DESKTOP-FATQ7KC MINGW64 ~/Desktop/repositorio lenin
$ git clone https://github.com/Lenincampos18/Proyecto_Python.git
Cloning into 'Proyecto_Python'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
```

➔ Git clone [https://github.com/Lenincampos18/Proyecto\\_Python.git](https://github.com/Lenincampos18/Proyecto_Python.git) (Aquí se clona el repositorio remoto al equipo localmente)



➔ Aquí validamos que efectivamente que se clono correctamente el repositorio remoto localmente, podemos observar la carpeta oculta (.git).

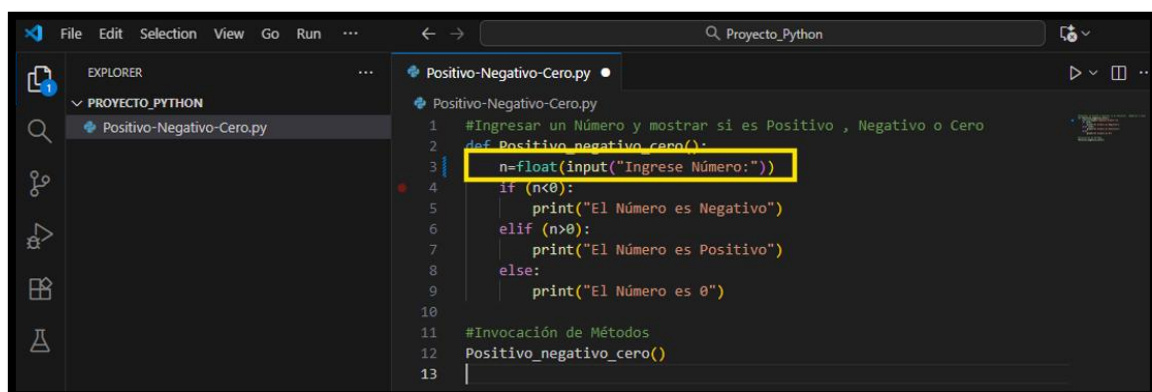
### 3. Pull request: realizar cambios en una rama y crear un Pull Request en GitHub.

Ahora para solicitar un Pull request (PR), es una solicitud formal que hace el desarrollador para integrar o fusionar los cambios de una rama de código a otra principal (generalmente main o master) dentro de un repositorio Git remoto (como GitHub).

En esta actividad, un miembro del equipo creo una nueva rama en el repositorio remoto y realizo modificaciones en el código.

#### **Miembro del Equipo: Miguel Lachira**

➔ Creo una nueva rama “modificación-miguel”, desde el terminal Gitbash de su localmente, mediante el comando *git checkout -b modificación-miguel*.



➔ En el proyecto realizo la modificación del tipo de dato a la variable “n” Float a Int ya que se manejarán número enteros.



```
Miguel Lachira P@DESKTOP-FATO7KC MINGW64 ~/Desktop/repositorio lenin/Proyecto_Python (modificacion-miguel)
$ git commit -m "realice el cambio del tipo de dato de float a int en la línea 3"
[modificacion-miguel 0408f0c] realice el cambio del tipo de dato de float a int en la línea 3
1 file changed, 1 insertion(+), 1 deletion(-)

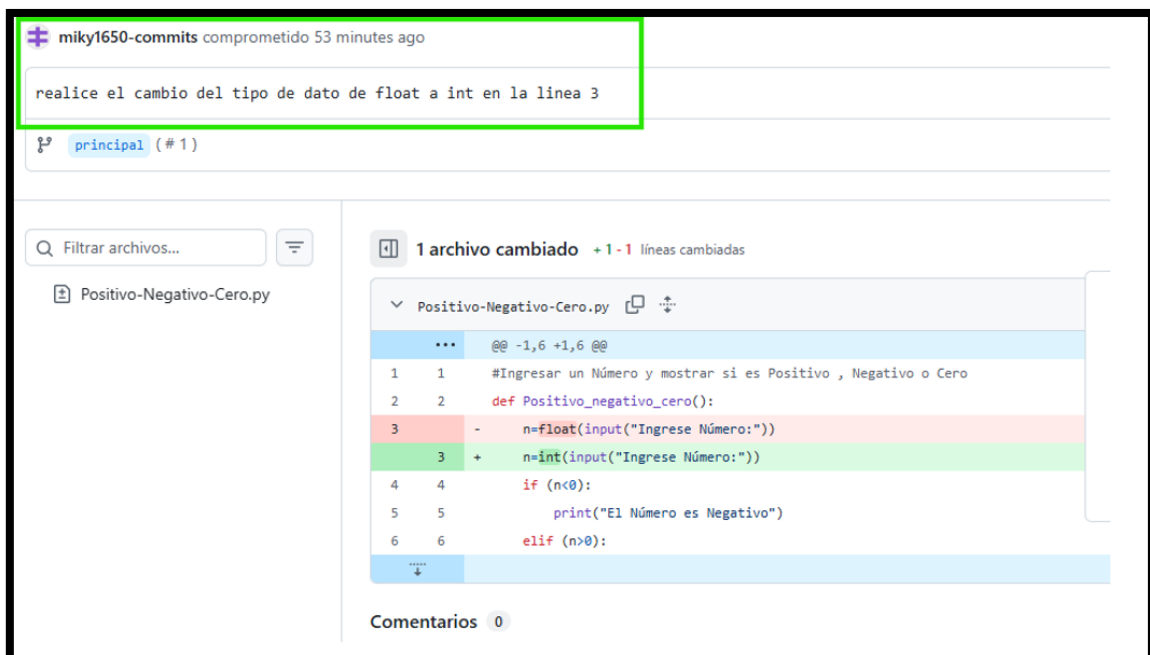
Miguel Lachira P@DESKTOP-FATO7KC MINGW64 ~/Desktop/repositorio lenin/Proyecto_Python (modificacion-miguel)
$ git log
commit 0408f0cac073317729086fb4483cc4950edaa7fe (HEAD -> modificacion-miguel)
Author: Miguel Lachira <miky1650@gmail.com>
Date: Wed Oct 22 01:34:29 2025 -0500
    realice el cambio del tipo de dato de float a int en la línea 3
```

- ➔ Se guardaron los cambios en la rama en Gitbash, mediante “*Git add Positivo-Negativo-Cero.py*” y *Git commit -m “Realice el cambio del tipo de dato de float a int – línea 3”*”.

```
Miguel Lachira P@DESKTOP-FATO7KC MINGW64 ~/Desktop/repositorio lenin/Proyecto_Python (modificacion-miguel)
$ git push origin modificacion-miguel
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 322 bytes | 322.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'modificacion-miguel' on GitHub by visiting:
remote:   https://github.com/Lenincampos18/Proyecto_Python/pull/new/modificacion-miguel
remote:
To https://github.com/Lenincampos18/Proyecto_Python.git
 * [new branch]      modificacion-miguel -> modificacion-miguel
```

- ➔ Ahora toca subir la rama al repositorio remoto “GitHub”, mediante “*Git push origin modificación-miguel*”. Ni bien mandes este comando automáticamente GitHub te indicara que inicies sesión en GitHub y le das autorizar, una vez se hallas autorizado se abrirá la plataforma de GitHub donde te mostrara un mensaje “Compare & Pull Request”.

Ahora procedemos a crear el Pull Request:



miky1650-commits comprometido 53 minutos ago

realice el cambio del tipo de dato de float a int en la línea 3

principal (#1)

Filtrar archivos...

Positivo-Negativo-Cero.py

1 archivo cambiado +1-1 líneas cambiadas

```

... @@ -1,6 +1,6 @@
1 1 #Ingresar un Número y mostrar si es Positivo , Negativo o Cero
2 2 def Positivo_negativo_cero():
3 - n=float(input("Ingrese Número:"))
3 + n=int(input("Ingrese Número:"))
4 4 if (n<0):
5 5     print("El Número es Negativo")
6 6 elif (n>0):

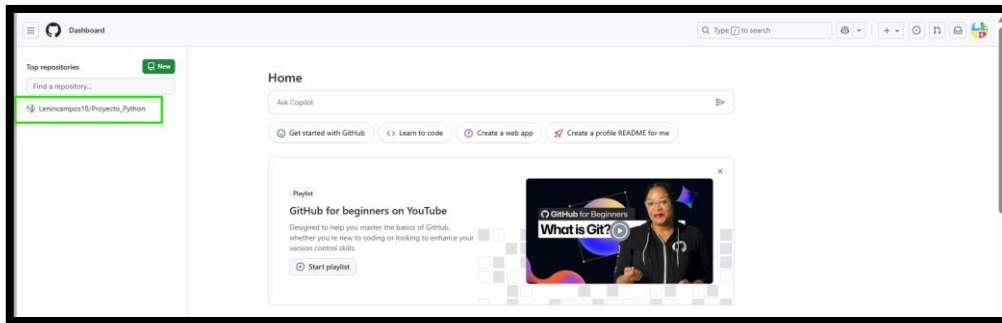
```

Comentarios 0

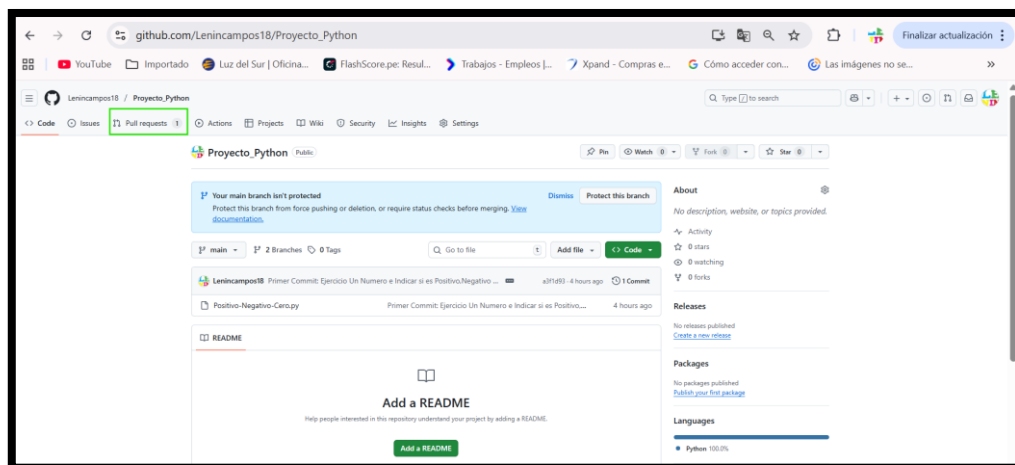
- ➔ Escribimos un título claro **“Realice el cambio del tipo de dato de float a int en la línea 3”**.
- ➔ Se agrega una descripción breve de los cambios que se realizaron.
- ➔ Por último, clic en **“Create Pull Request”**.

El miembro Miguel Lachira envió correctamente su solicitud de PULL REQUEST, ahora el propietario o el miembro que tiene permiso, puede realizar el “Merge Pull Request”.

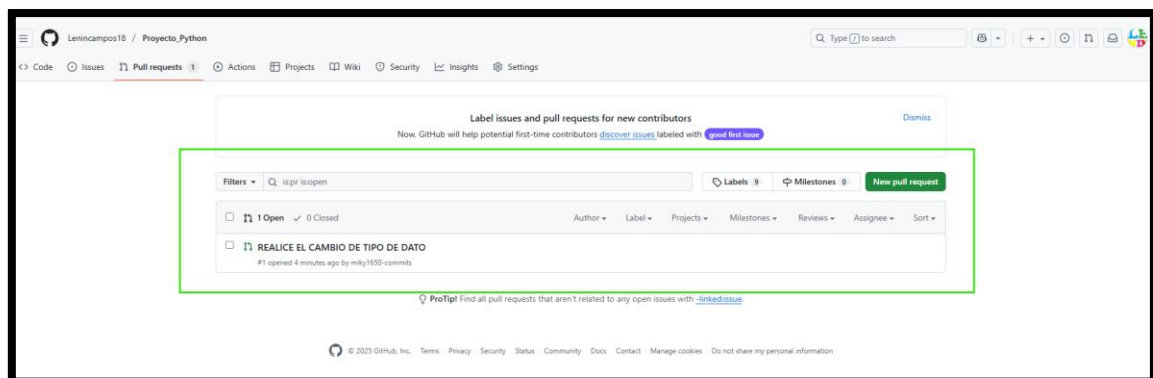
En este caso lo realizara **Lenin Campos**, miembro del equipo:



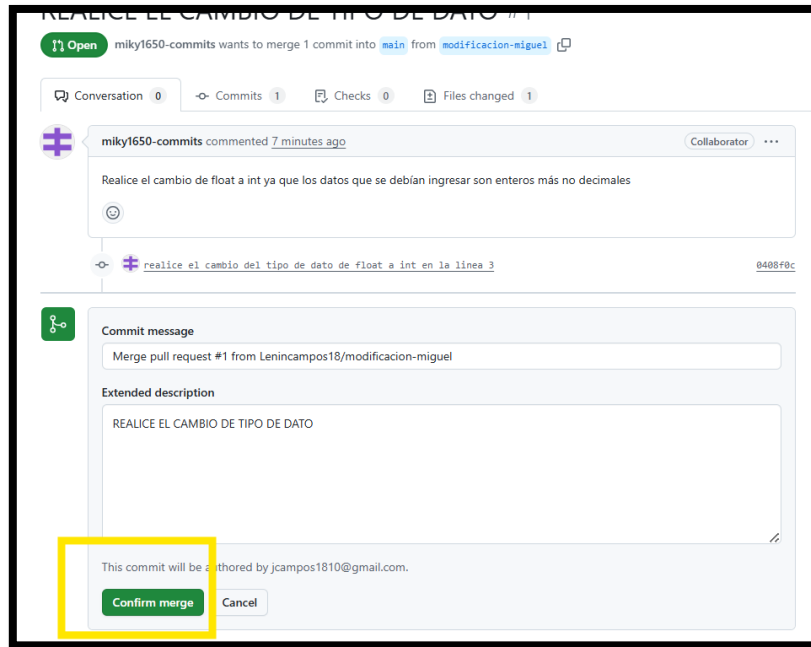
- ➔ Ingresamos a la plataforma GitHub e ingresamos al repositorio **“Proyecto\_Python”**.



- ➔ Podemos validar que tenemos un Pull Request, ingresamos







- ➔ Ingresamos y vemos que este Pull Request corresponde a Miguel Lachira, donde indica que realizo un cambio del código del tipo de dato en la variable “n” en el archivo *Positivo-Negativo-Cero.py*.
- ➔ Una vez revisado procedemos a confirmar la fusión de ramas, clic “**Confirm merge**”.

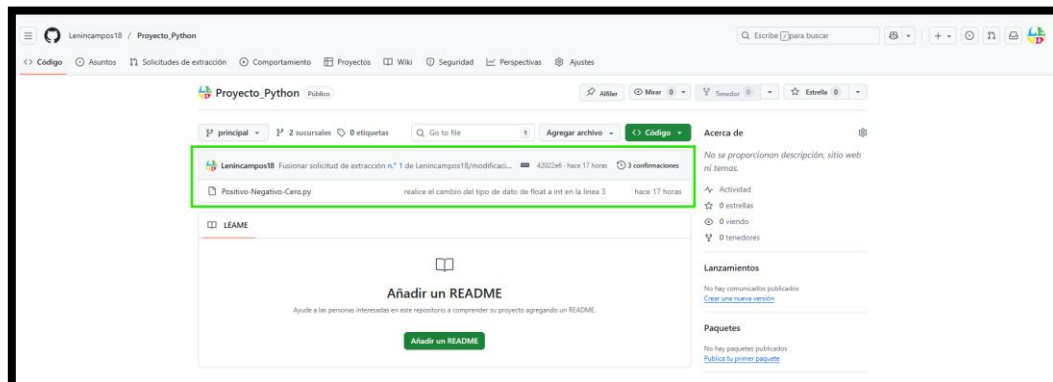
#### 4. Evidenciar Revisión de Código

En un proyecto colaborativo, cuando alguien crea un Pull Request, los demás miembros del equipo pueden:

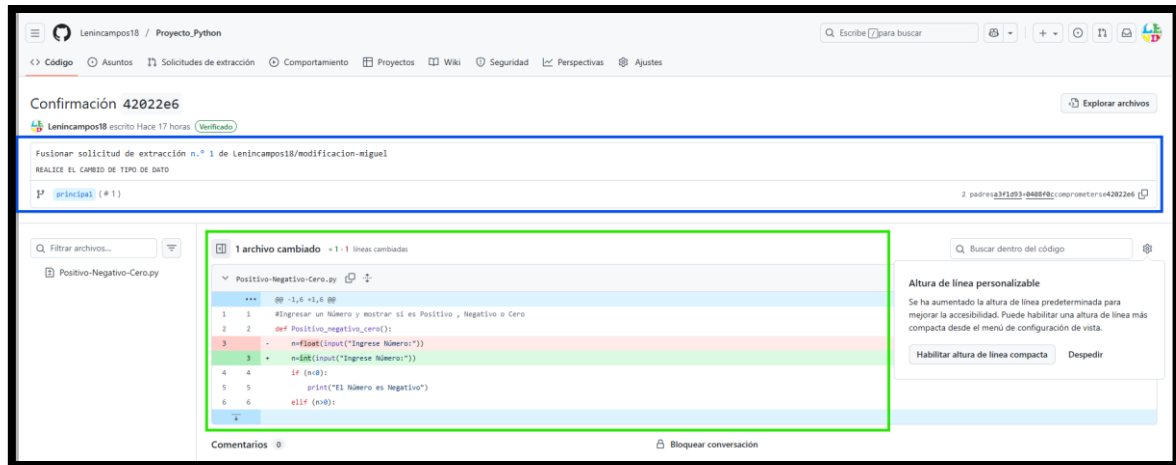
- ✓ Revisar los cambios propuestos
- ✓ Comentar líneas específicas del código
- ✓ Sugerir mejoras
- ✓ Aprobar o rechazar los cambios

En GitHub, eso se le llama Code Review (Revisión de Código).

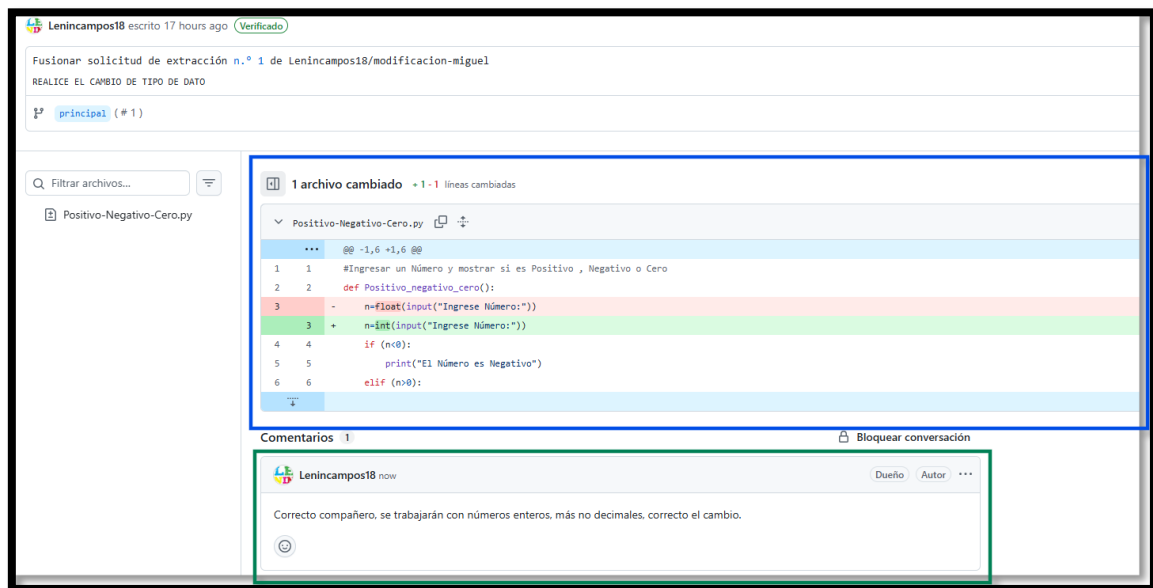
Ahora procederemos a evidenciarlo:



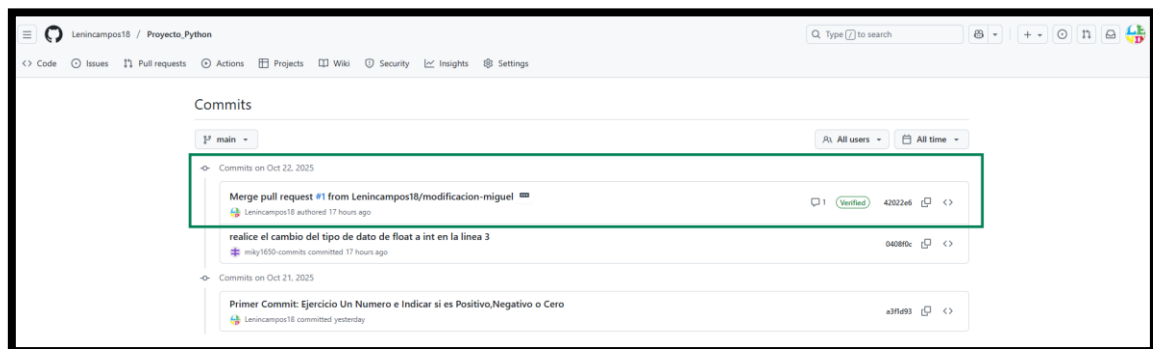
➔ Ingresamos al PR que solicito Miguel Lachira e ingresamos.



➔ Aquí podemos ver las líneas del código cambiaron (marcadas en verde y rojo).



➔ Aquí podemos agregar comentarios en las líneas y dejar observaciones, así como lo realizo un miembro del equipo Lenin Campos.



➔ “Pull Request Merged Successfully”.

## Repositorio del Grupo 12

Nombre: POO-Git-Casos

Link: <https://github.com/Lenincampos18/POO-Git-Casos.git>

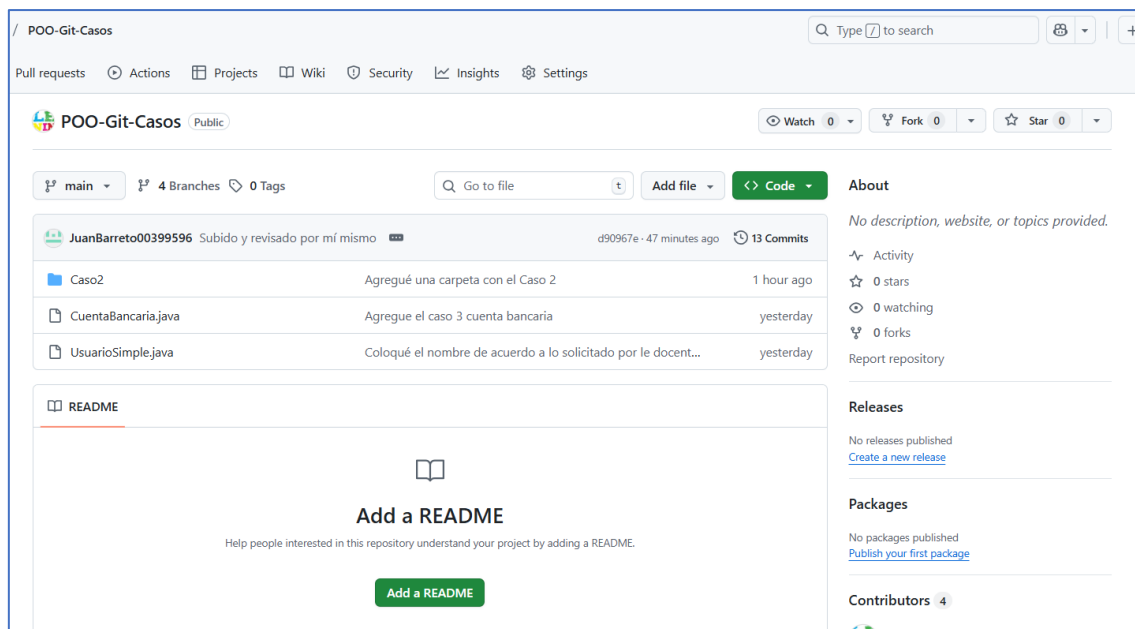
Creador: Lenin Campos

Colaboradores:

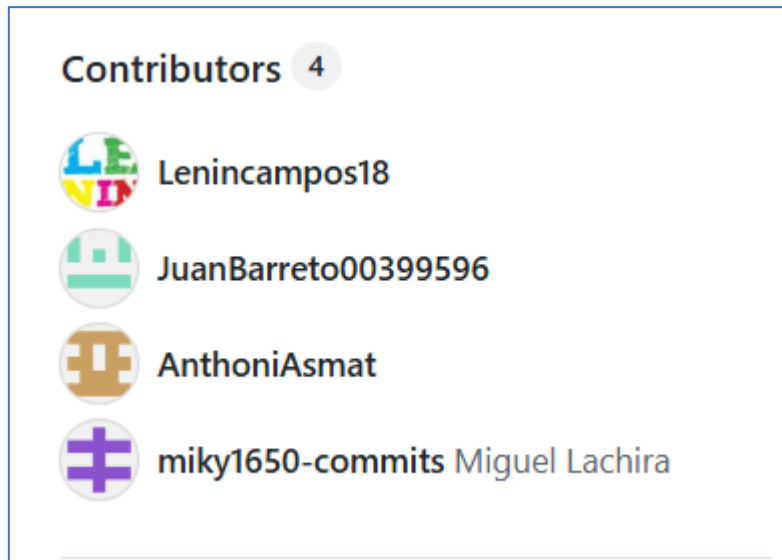
- Anthoni Asmat
- David Rojas
- Juan Romero
- Miguel Lachira

### Evidencias del trabajo colaborativo

1. Portada del repositorio



## 2. Colaboradores



## 3. Colaboración real: Caso 1 “Lectura de datos simples con Scanner”

```

Dell@DESKTOP-3TIVFUF MINGW64 ~/Documents/GitHub/POO-Git-Casos (main)
$ cd ..

Dell@DESKTOP-3TIVFUF MINGW64 ~/Documents/GitHub
$ cd POO-Git-Casos

Dell@DESKTOP-3TIVFUF MINGW64 ~/Documents/GitHub/POO-Git-Casos (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  POO-Git-Casos/
  UsuarioSimple.java

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

Dell@DESKTOP-3TIVFUF MINGW64 ~/Documents/GitHub/POO-Git-Casos (main)
$ git add UsuarioSimple.java

Dell@DESKTOP-3TIVFUF MINGW64 ~/Documents/GitHub/POO-Git-Casos (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
  new file:   UsuarioSimple.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  POO-Git-Casos/

Dell@DESKTOP-3TIVFUF MINGW64 ~/Documents/GitHub/POO-Git-Casos (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
  new file:   UsuarioSimple.java

Dell@DESKTOP-3TIVFUF MINGW64 ~/Documents/GitHub/POO-Git-Casos (main)
$ git

```

Se cargó al repositorio el archivo con el código en Java del caso 2; se trabajó en Git Bash desde la rama principal (main).

#### 4. Colaboración real: Caso 2 “Clase Estudiante con atributos privados”

```

Caso2 > J Main.java > {} Caso2
5 public class Main{
6     public static void main(String[] args) {
7
8         //Se solicita al usuario ingresar los datos del estudiante
9         System.out.println("=== REGISTRO DE NUEVO ESTUDIANTE ===");
10
11         //Código
12         System.out.println("Ingrese el código del estudiante: ");
13         int codigo = sc.nextInt();
14         sc.nextLine(); // Consumir el salto de línea pendiente
15
16         //Nombre
17         System.out.println("Ingrese el nombre del estudiante: ");
18         String nombre = sc.nextLine();
19
20         //Apellido
21         System.out.println("Ingrese el apellido del estudiante: ");
22         String apellido = sc.nextLine();
23
24         //Promedio
25         System.out.println("Ingrese el promedio del estudiante: ");
26         double promedio = sc.nextInt();
27         sc.nextLine();
28
29         //Crear objeto Estudiante
30         Estudiante estudiante = new Estudiante(codigo, nombre, apellido, promedio);
31         estudiante.mostrarDatos();
32     }
33 }

```

Se cargó al repositorio el proyecto “Caso 2”; al hacerse un merge, en el Visual Code ya se puede visibilizar el proyecto y se puede ejecutar el código.

#### 5. Colaboración real: Caso 3 “Clase CuentaBancaria con validación”

```

Usuario@Anthoni MINGW64 ~/Desktop/Proyecto/POO-Git-Casos (main)
$ git checkout Asmat
Switched to branch 'Asmat'

Usuario@Anthoni MINGW64 ~/Desktop/Proyecto/POO-Git-Casos (Asmat)
$ git status
On branch Asmat
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    CuentaBancaria.java

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

Usuario@Anthoni MINGW64 ~/Desktop/Proyecto/POO-Git-Casos (Asmat)
$ git add CuentaBancaria.java

Usuario@Anthoni MINGW64 ~/Desktop/Proyecto/POO-Git-Casos (Asmat)
$ git status
On branch Asmat
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   CuentaBancaria.java

Usuario@Anthoni MINGW64 ~/Desktop/Proyecto/POO-Git-Casos (Asmat)
$ git commit -m "Agregue el caso 3 cuenta bancaria"
[Asmat 3e13c54] Agregue el caso 3 cuenta bancaria
1 file changed, 88 insertions(+)
create mode 100644 CuentaBancaria.java

Usuario@Anthoni MINGW64 ~/Desktop/Proyecto/POO-Git-Casos (Asmat)
$ |

```

Se cargó al repositorio el archivo con el código en Java del caso 3; se trabajó en Git Bash desde la rama Asmat (realizada por el compañero Anthoni Asmat).