

Tecnicatura Universitaria en Programación

Programación I

Comisión 7

Profesor: Bruselario, Sebastián

Tutor: Candia, Verónica

Alumna: Córdoba, Natalia

Trabajo Práctico Unidad 2:

Git y GitHub

Consignas y Respuestas:

1) *Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas):*

1. ¿Qué es GitHub?

Es una plataforma en línea para almacenar proyectos que utilizan el sistema de control de versiones Git. Permite a los desarrolladores almacenar y compartir sus proyectos, así como también, trabajar de manera colaborativa con otros desarrolladores, ya sea en proyectos privados como de código abierto. GitHub resulta muy útil a la hora de realizar trabajos de forma colaborativa ya que cuenta con herramientas que permiten organizar, discutir, revisar y aprobar cambios en los proyectos que allí se alojan.

2. ¿Cómo crear un repositorio en GitHub?

Lo primero de todo es crearse una cuenta en GitHub. Una vez realizado esto, podemos iniciar sesión en GitHub. Para crear un repositorio debemos hacer clic en el signo más “+” que se encuentra en la esquina superior derecha y seleccionar "Nuevo repositorio".

A screenshot of a GitHub profile page for 'Natalia-Cordoba'. The top navigation bar shows tabs for Overview, Repositories (14), Projects, Packages, and Starred. A context menu is open in the top right corner, with the 'New repository' option highlighted by a yellow circle. Other options in the menu include Import repository, New codespace, New gist, New organization, New project, and Javascript.

Para crearlo deberemos: ponerle un nombre al repositorio, opcionalmente una descripción, elegir si será un repositorio público o privado, y finalmente, hacer clic en "Crear repositorio".

A screenshot of the 'Create a new repository' form on GitHub. The 'Repository name' field contains 'repo_prueba'. The 'Description (optional)' field contains 'repositorio de prueba'. The 'Public' radio button is selected. At the bottom, the 'Create repository' button is highlighted by a yellow circle.

Así se ve el repositorio una vez creado

The screenshot shows a GitHub repository page for 'repo_prueba'. The browser tabs at the top are 'Tema: Práctica', 'TP-Unidad2-Córdoba', and 'Natalia-Cordoba/repo_prueba'. The main content area displays a 'Quick setup' section with instructions for setting up GitHub Copilot, adding collaborators, and creating or pushing repositories via command line. It includes code snippets for initializing a local repository and pushing it to GitHub.

3. ¿Cómo crear una rama en Git?

Para crear una nueva rama en Git, lo primero es asegurarte de que el repositorio ya haya sido inicializado con git init o clonado desde GitHub. Una vez controlado eso, debes utilizar el siguiente comando:

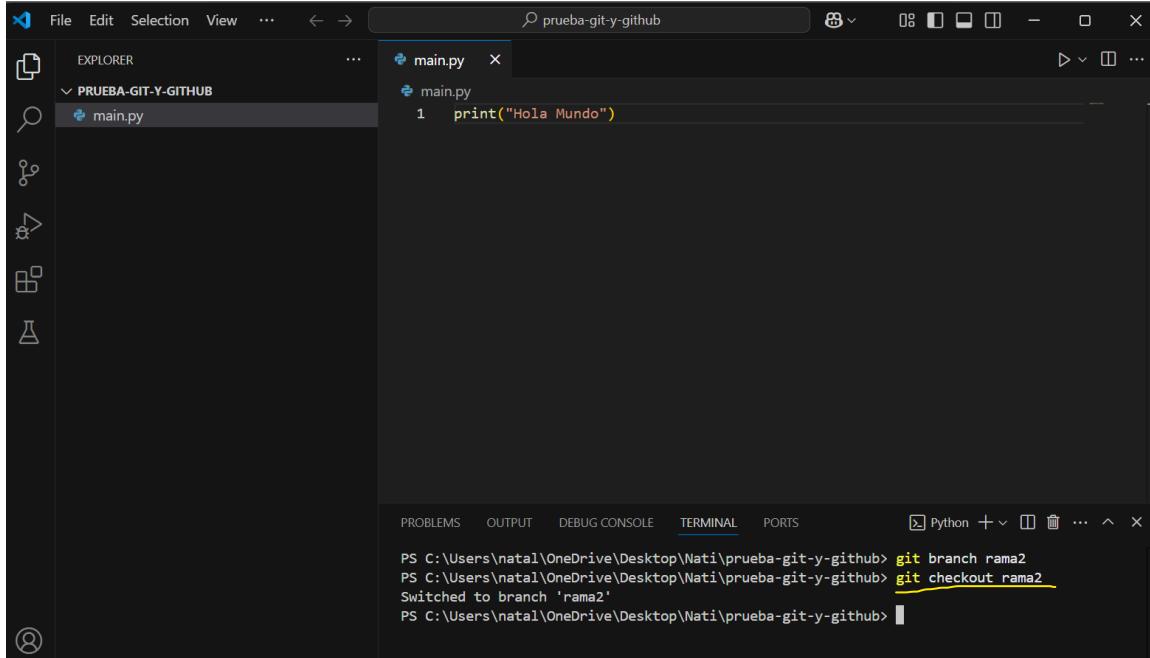
git + espacio + branch + espacio + nombre-de-la-nueva-rama

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar has icons for Explorer, Search, Find, Open, and Outline. The main workspace shows a file named 'main.py' with the content 'print("Hola Mundo")'. Below the workspace is a terminal window with the following text:
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git branch rama2
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github>
A yellow arrow points to the command 'git branch rama2' in the terminal.

4. ¿Cómo cambiar a una rama en Git?

En Git si queremos cambiar de una rama a otra debemos utilizar el siguiente comando:

git + espacio + checkout + espacio + nombre-de-rama-a-la-que-queremos-cambiar



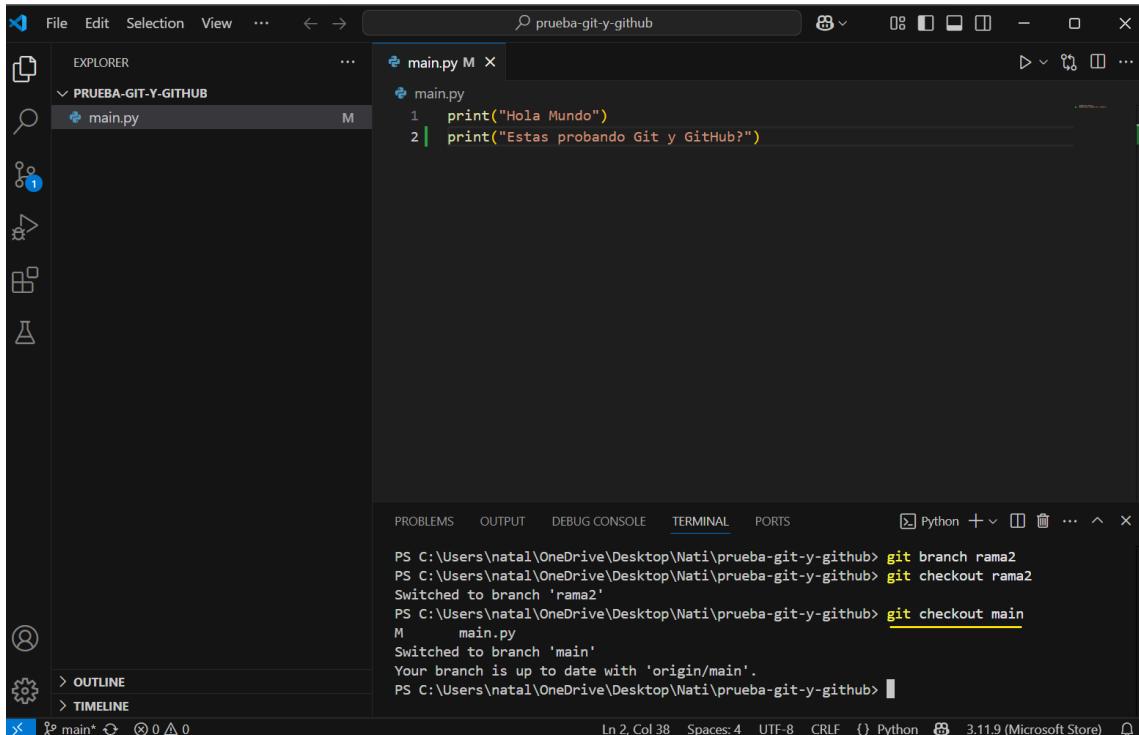
The screenshot shows the VS Code interface with the terminal tab selected. The terminal window displays the following command sequence:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git branch rama2
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git checkout rama2
Switched to branch 'rama2'
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github>
```

5. ¿Cómo fusionar ramas en Git?

Para fusionar ramas en Git lo primero que debes hacer es posicionarte en la rama donde deseas fusionar los cambios. Si no estás en ella utiliza el comando:

git + espacio + checkout + espacio + nombre-de-la-rama-destino



The screenshot shows the VS Code interface with the terminal tab selected. The terminal window displays the following command sequence, showing the creation of a new branch 'rama2' and its subsequent merge into the 'main' branch:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git branch rama2
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git checkout rama2
Switched to branch 'rama2'
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git checkout main
M     main.py
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github>
```

Luego, para fusionar la rama actual con otra rama, debemos utilizar el siguiente comando:

git + espacio + merge + nombre-de-la-rama-a-fusionar

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. On the left is the Explorer sidebar with a project named 'PRUEBA-GIT-Y-GITHUB' containing a file 'main.py'. The main area shows the content of 'main.py':

```
1 print("Hola Mundo")
2 print("Estas probando Git y GitHub?")
```

Below the editor is the Terminal tab, which displays the following command history:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git checkout rama2
Switched to branch 'rama2'
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git checkout main
M     main.py
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git merge rama2
Already up to date.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github>
```

The status bar at the bottom indicates the terminal has 2 lines, column 38, 4 spaces, UTF-8 encoding, and is using Python 3.11.9.

6. ¿Cómo crear un commit en Git?

Lo principal para hacer un commit es tener cambios en nuestro proyecto para poder hacer el commit. En este caso podemos guardar la fusión de ambas ramas. Para ello, comenzamos con el comando:

git + espacio + add + .

Este comando sirve para preparar los cambios antes del commit. El punto implica que queremos agregar todos los archivos. Si quisieramos hacer un commit de un solo archivo tendríamos que poner el siguiente comando:

git + espacio + add + nombre-del-archivo

```
main.py M X
main.py
1 print("Hola Mundo")
2 print("Estas probando Git y GitHub?")
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python + ⚡ ... ^ x
Switched to branch 'rama2'
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git checkout main
M main.py
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git merge rama2
Already up to date.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git add .
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github>
Not Committed Yet (Staged) Ln 2, Col 38 Spaces:4 UTF-8 CRLF {} Python 3.11.9 (Microsoft Store)
```

Ahora sí, ya podemos realizar el commit y confirmar los cambios:

git + espacio + commit + espacio + -m + espacio + "Mensaje del commit"

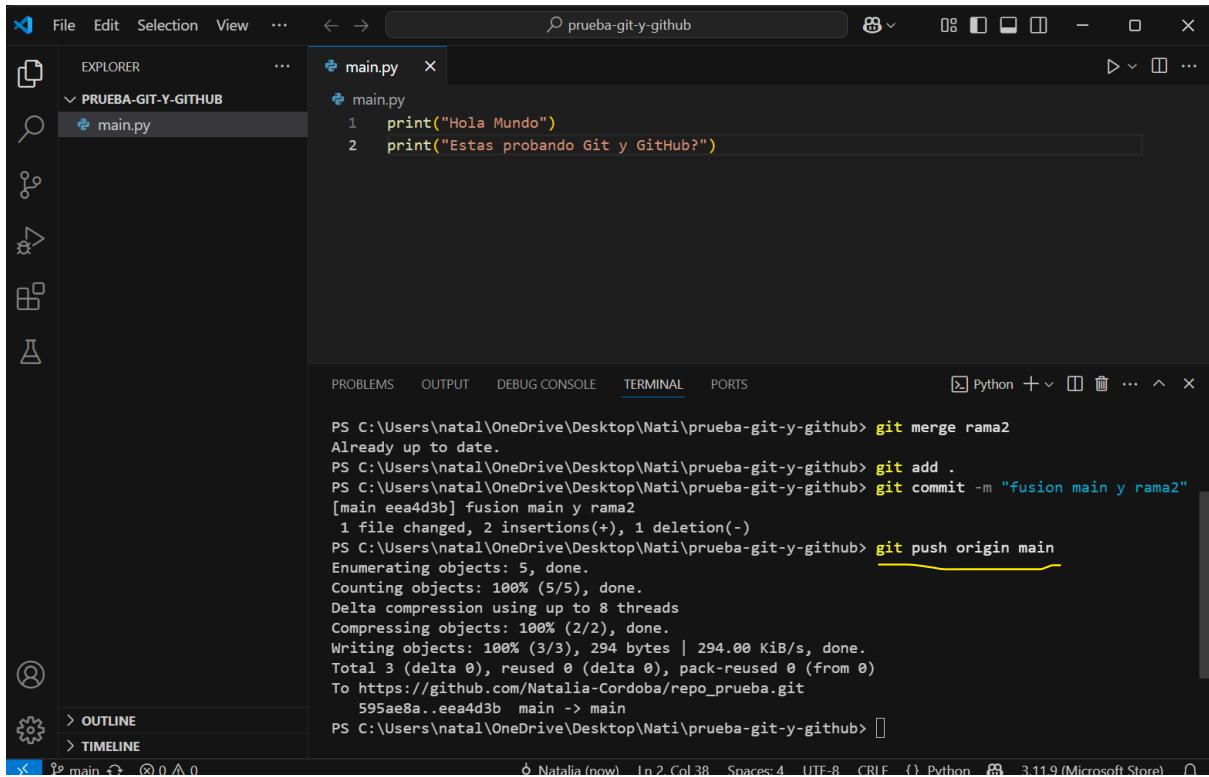
```
main.py M X
main.py
1 print("Hola Mundo")
2 print("Estas probando Git y GitHub?")
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python + ⚡ ... ^ x
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git merge rama2
Already up to date.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git add .
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git commit -m "fusion main y rama2"
[main eea4d3b] fusion main y rama2
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github>
```

7. ¿Cómo enviar un commit a GitHub?

Luego de ejecutar el comando `git commit` en nuestro repositorio local, podemos enviar ese commit a nuestro repositorio remoto en GitHub, para ello debemos usar el siguiente comando:

git + espacio + push + origin + nombre-de-rama



The screenshot shows the Visual Studio Code interface. On the left is the Explorer sidebar with a project named 'PRUEBA-GIT-Y-GITHUB' containing a file 'main.py'. The main area shows the content of 'main.py':

```
1 print("Hola Mundo")
2 print("Estas probando Git y GitHub?")
```

Below the editor is the Terminal tab, which displays the following command-line session:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git merge rama2
Already up to date.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git add .
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git commit -m "fusion main y rama2"
[main eea4d3b] fusion main y rama2
 1 file changed, 2 insertions(+), 1 deletion(-)
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github> git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 294 bytes | 294.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Natalia-Cordoba/repo_prueba.git
 595ae8a..eea4d3b main -> main
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-y-github>
```

A yellow underline highlights the command `git push origin main`.

8. ¿Qué es un repositorio remoto?

Un repositorio remoto es una versión de un repositorio local (el que manejamos con Git por ejemplo) que se encuentra alojada en Internet o en una red. GitHub es una de las plataformas más populares para almacenar repositorios de manera remota.

Tener un repositorio remoto, además de local, nos permite compartir nuestros códigos con otros desarrolladores, realizar trabajos en equipo de manera colaborativa y guardar una copia del proyecto en la nube, lo cual es beneficioso para la seguridad de un proyecto.

9. ¿Cómo agregar un repositorio remoto a Git?

Si tengo un repositorio remoto creado desde GitHub, puedo agregarlo a un repositorio local en Git. Comenzamos con la creación de repositorio remoto.

The screenshot shows a GitHub repository page for 'repo-prueba'. At the top, there are sections for 'Set up GitHub Copilot' and 'Add collaborators to this repository'. Below these, a 'Quick setup' section provides instructions for cloning or pushing code. It includes a command-line example for creating a new repository:

```
echo "# repo-prueba" >> README.md  
git init  
git add README.md  
git commit -m "first commit"  
git branch -M main  
git remote add origin https://github.com/Natalia-Cordoba/repo-prueba.git  
git push -u origin main
```

It also includes a command-line example for pushing an existing repository:

```
git remote add origin https://github.com/Natalia-Cordoba/repo-prueba.git  
git branch -M main  
git push -u origin main
```

A 'ProTip!' note at the bottom suggests using the URL for adding GitHub as a remote.

Antes de agregar un repositorio remoto a Git, es importante asegurarse de que el proyecto ya esté inicializado con Git y tenga al menos un commit.

Si no está inicializado debemos abrir la terminal, en este caso voy a utilizar la que está incorporada en Visual Studio Code, y allí inicializamos el repositorio local con el comando:

git + espacio + init

The screenshot shows a Visual Studio Code interface. The left sidebar has icons for Explorer, Search, Problems, Outline, and Timeline. The main workspace shows a Python file named 'main.py' with the code:

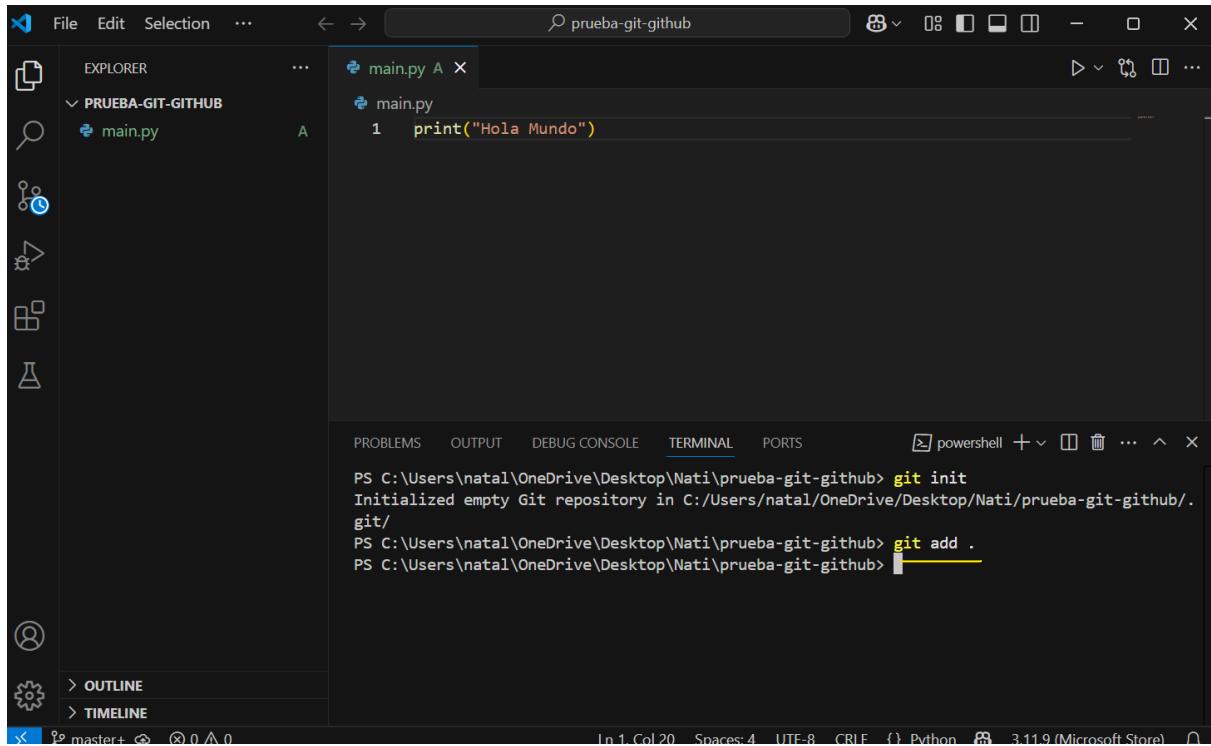
```
print("Hola Mundo")
```

. The bottom right corner shows a terminal window with the following output:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git init  
Initialized empty Git repository in C:/Users/natal/OneDrive/Desktop/Nati/prueba-git-github/.git/  
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

Luego realizamos el comando para agregar los archivos:

git + espacio + add + espacio + .



A screenshot of the Visual Studio Code interface. The left sidebar shows a project named 'PRUEBA-GIT-GITHUB' containing a file 'main.py'. The code in 'main.py' is:

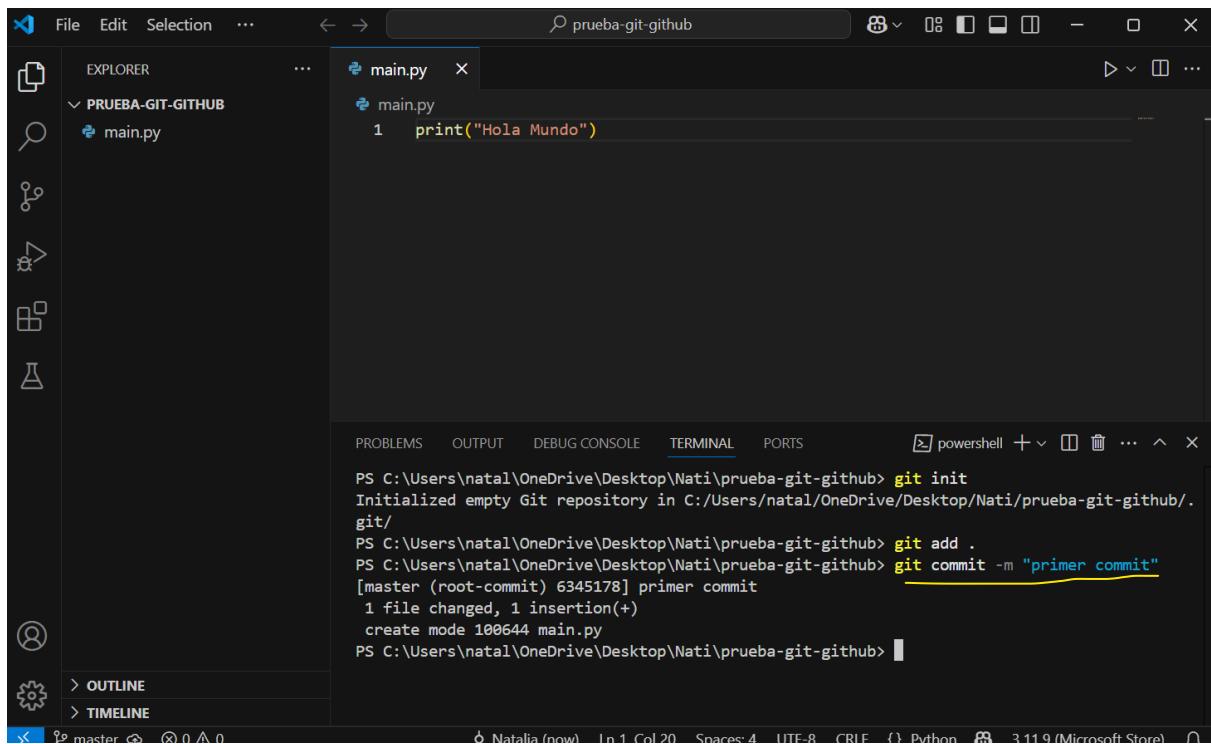
```
1 print("Hola Mundo")
```

The bottom right panel is the terminal, showing the following PowerShell session:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git init
Initialized empty Git repository in C:/Users/natal/OneDrive/Desktop/Nati/prueba-git-github/.git/
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git add .
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

El siguiente paso sería realizar el primer commit, para ello usamos:

git + espacio + commit + espacio + -m + espacio + mensaje-commit



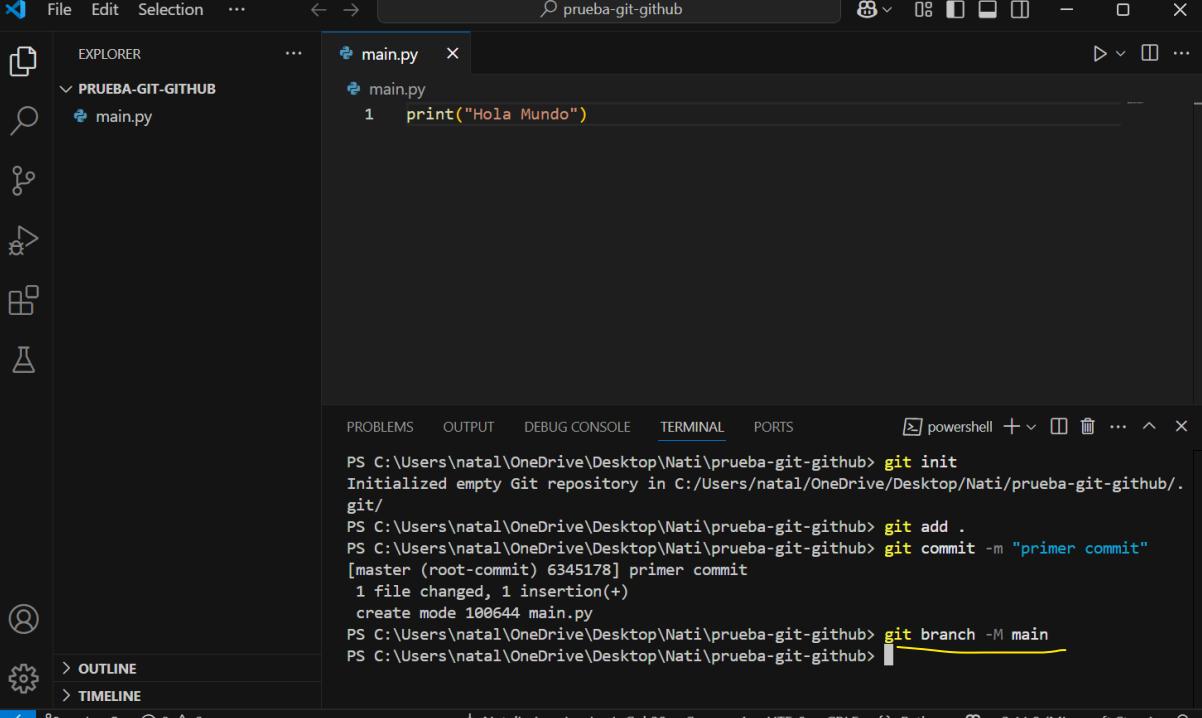
A screenshot of the Visual Studio Code interface, similar to the previous one but with more terminal history.

The terminal shows the following PowerShell session:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git init
Initialized empty Git repository in C:/Users/natal/OneDrive/Desktop/Nati/prueba-git-github/.git/
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git add .
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git commit -m "primer commit"
[master (root-commit) 6345178] primer commit
 1 file changed, 1 insertion(+)
 create mode 100644 main.py
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

Nombramos a nuestra rama principal como main, con el comando:

git + espacio + branch + espacio + -M + main



A screenshot of the Visual Studio Code interface. The left sidebar shows a project folder named 'PRUEBA-GIT-GITHUB' containing a file 'main.py'. The code editor shows the following Python script:

```
1 print("Hola Mundo")
```

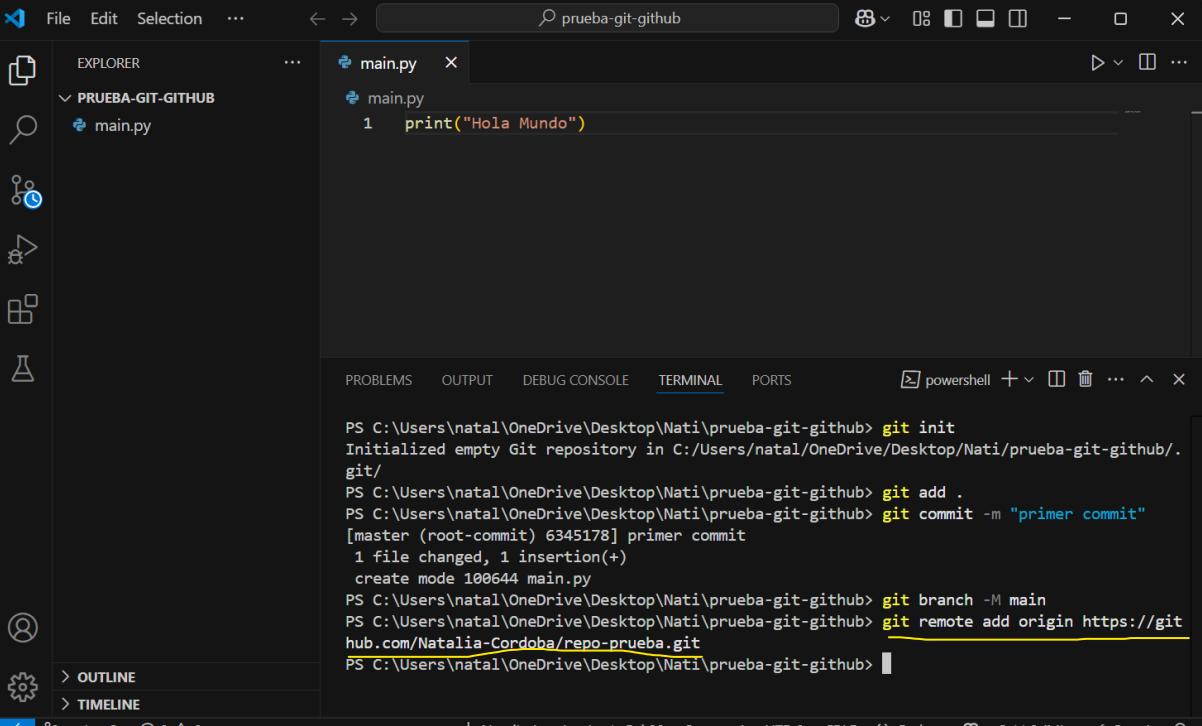
The terminal at the bottom displays the following Git commands:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git init
Initialized empty Git repository in C:/Users/natal/OneDrive/Desktop/Nati/prueba-git-github/ .
git/
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git add .
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git commit -m "primer commit"
[master (root-commit) 6345178] primer commit
 1 file changed, 1 insertion(+)
 create mode 100644 main.py
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git branch -M main
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

Ahora si, podemos utilizar el comando para agregar el repositorio remoto a Git:

git + espacio + remote + espacio + add + espacio + nombre-remoto + espacio + URL-del-repositorio

Si nuestro repositorio local en Git ya había sido inicializado y ya se había realizado al menos un commit, solo deberíamos ejecutar este comando para agregar el repositorio remoto a Git.



A screenshot of the Visual Studio Code interface. The left sidebar shows a project folder named 'PRUEBA-GIT-GITHUB' containing a file 'main.py'. The code editor shows the same Python script as before:

```
1 print("Hola Mundo")
```

The terminal at the bottom displays the following Git commands, with the command to add the remote repository highlighted in yellow:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git init
Initialized empty Git repository in C:/Users/natal/OneDrive/Desktop/Nati/prueba-git-github/ .
git/
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git add .
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git commit -m "primer commit"
[master (root-commit) 6345178] primer commit
 1 file changed, 1 insertion(+)
 create mode 100644 main.py
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git branch -M main
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git remote add origin https://github.com/Natalia-Cordoba/repo-prueba.git
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

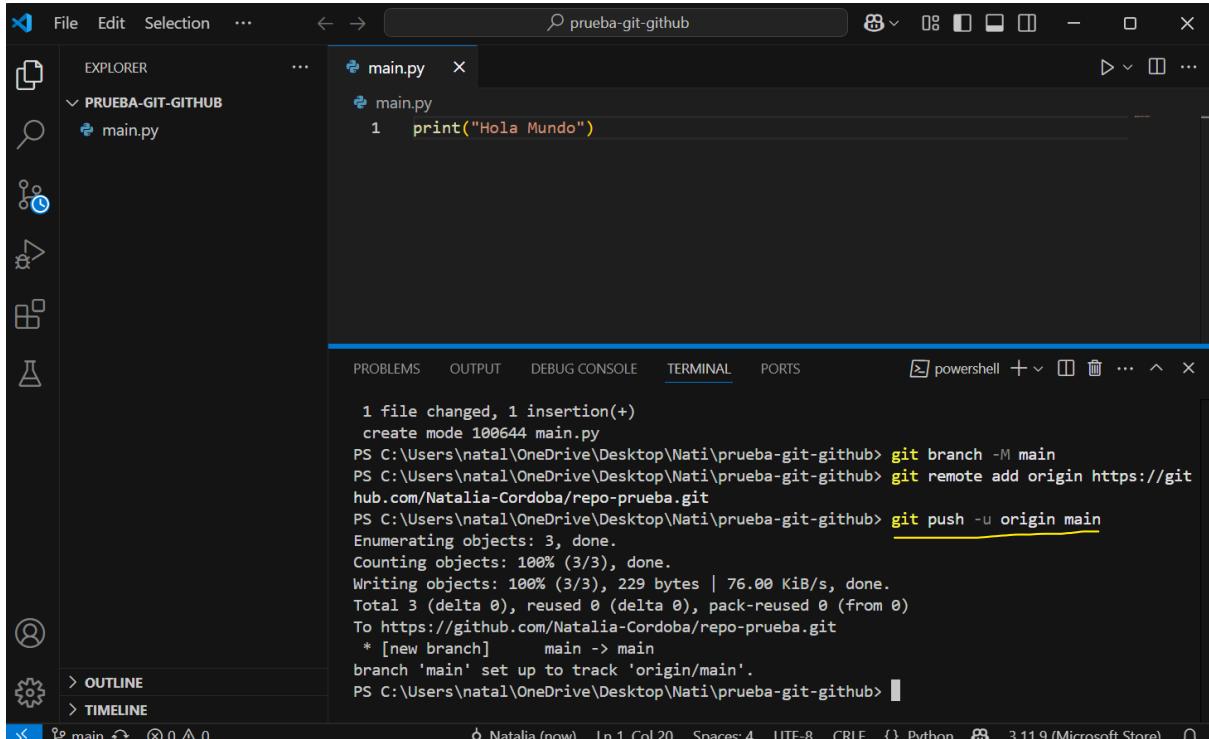
10. ¿Cómo empujar cambios a un repositorio remoto?

Para enviar cambios al repositorio remoto, luego de hacer un commit, debemos ejecutar el siguiente comando:

git + espacio + push + espacio + origin + espacio + nombre-rama

Si lo estamos haciendo por primera vez, luego de agregar el repositorio local en Git, debemos agregar -u antes de origin:

git + espacio + push + espacio + -u + espacio + origin + espacio + nombre-rama



```
1 file changed, 1 insertion(+)
create mode 100644 main.py
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git branch -M main
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git remote add origin https://github.com/Natalia-Cordoba/repo-prueba.git
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 229 bytes | 76.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Natalia-Cordoba/repo-prueba.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

11. ¿Cómo tirar de cambios de un repositorio remoto?

Así como podemos enviar los cambios del repositorio local al remoto podemos realizar el paso contrario y tirar los cambios del repositorio remoto a nuestro repositorio local. Así podremos obtener y fusionar cambios desde un repositorio remoto a la rama local. Para ello, tenemos que ejecutar el comando:

git + espacio + pull + espacio + origin + espacio + rama-remota

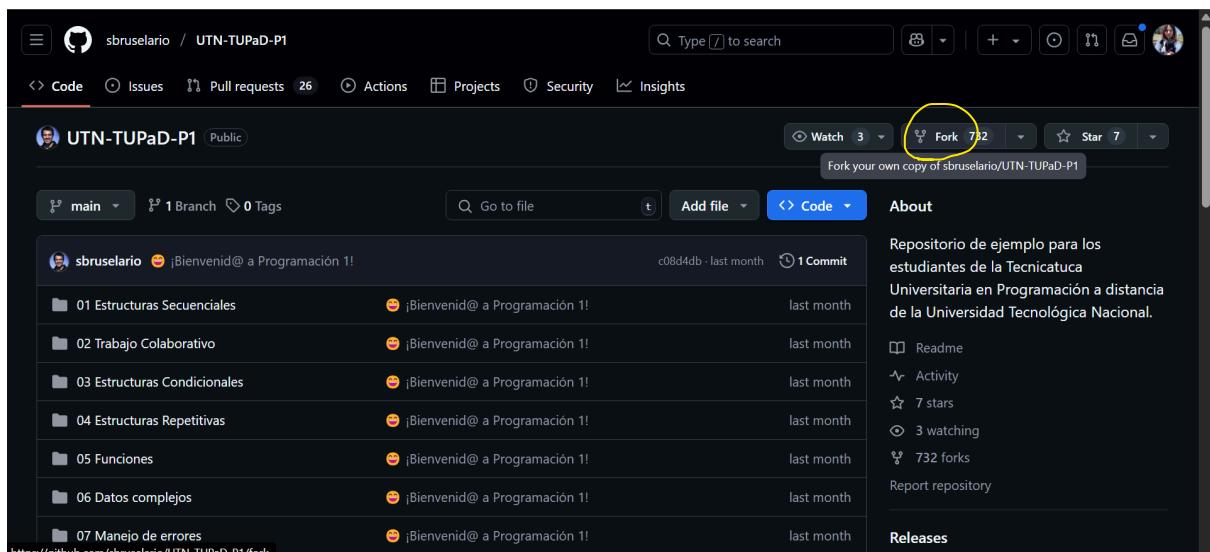
```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git pull origin main
From https://github.com/Natalia-Cordoba/repo-prueba
 * branch            main      -> FETCH_HEAD
Already up to date.
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

12. ¿Qué es un fork de repositorio?

Un fork es una copia de un repositorio ajeno que se hace desde GitHub a nuestra cuenta. Es muy útil ya que nos permite: hacer cambios en un proyecto que no es nuestro sin afectar el repositorio original, proponer mejoras al propietario del repositorio (por ejemplo, haciendo un *pull request*), analizar cómo está hecho un proyecto. Una vez que realizamos el fork de un repositorio, podemos clonarlo, modificarlo, y trabajar en él como si fuera nuestro.

13. ¿Cómo crear un fork de un repositorio?

Primero debemos ir al repositorio en el que quieres realizar el fork. Luego hacemos clic en el botón que dice **Fork**, este se encuentra en la esquina superior derecha de la página del repositorio.



Si es la primera vez que hacemos un fork, GitHub tendremos que elegir en qué cuenta queremos que se copie el repositorio. Una vez seleccionada nuestra cuenta, GitHub nos va a crear una copia del repositorio en nuestra cuenta, y al terminar, nos redirige a la página de tu nuevo repositorio en tu cuenta de GitHub.

14. ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

Luego de realizar los cambios en nuestra copia del repositorio, nos dirigimos al repositorio original en el que hicimos el Fork y seleccionamos la pestaña **Pull requests**. Luego debemos dar clic en el botón **New pull request**.

En este paso, GitHub nos dirigirá a una pantalla donde podremos comparar los cambios realizados entre nuestro repositorio y el repositorio original. También podremos agregar un asunto que explique qué cambios realizamos y detallar por qué creemos que esos cambios serían buenos para el proyecto original.

El último paso es dar clic en el botón que dice **Create pull request**.

15. ¿Cómo aceptar una solicitud de extracción?

El creador del repositorio original, una vez que nosotros realicemos la solicitud de extracción recibirá una notificación. En la sección Pull request, podrá ver nuestra solicitud, al ingresar en ella deberá analizar si acepta o no los cambios que propusimos para su proyecto. Si está de acuerdo con los cambios, deberá presionar el botón en la parte superior del pull request que dice **Merge pull request** y luego confirmar que acepta la fusión.

16. ¿Qué es una etiqueta en Git?

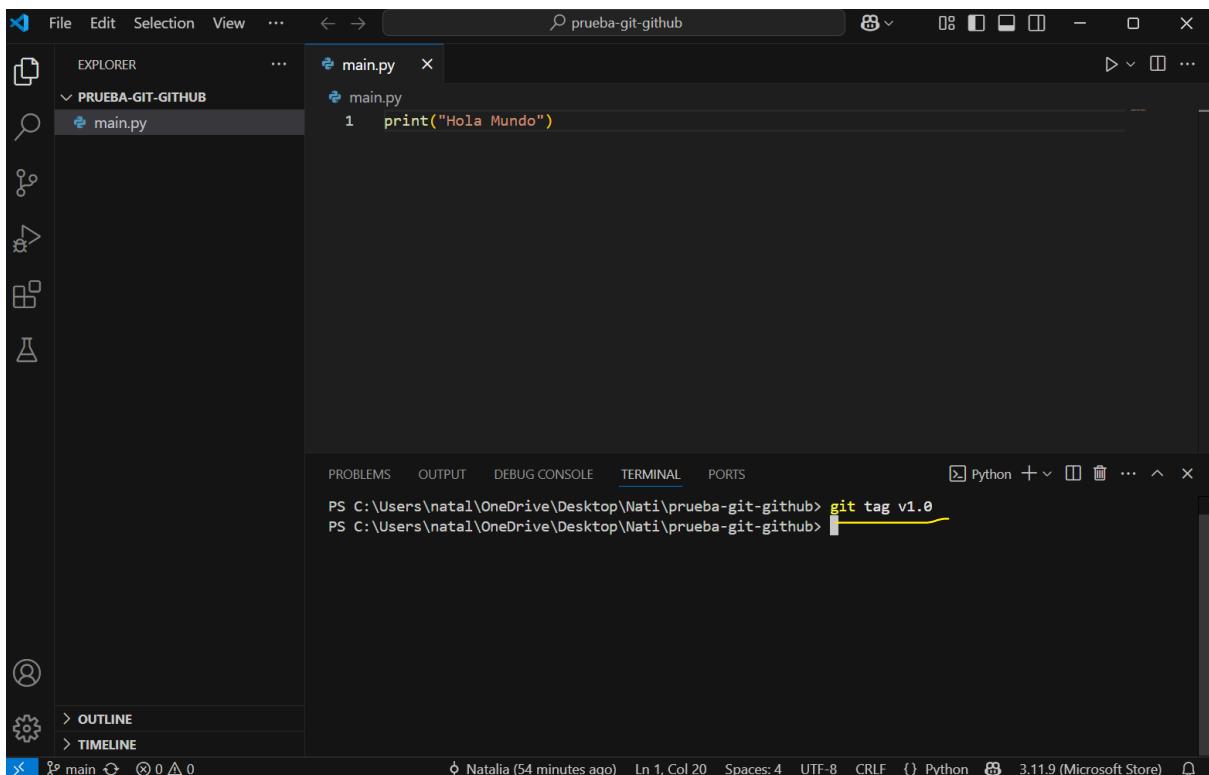
Una etiqueta en Git es una referencia que señala un punto específico en la historia del repositorio. Principalmente las etiquetas se utilizan para marcar versiones.

17. ¿Cómo crear una etiqueta en Git?

Para poder crear una etiqueta, tenemos dos opciones diferentes: las etiquetas ligeras y las etiquetas anotadas.

Las etiquetas ligeras: son una simple referencia a un commit, sin ninguna información adicional. Las podemos realizar ejecutando el comando:

git + espacio + tag + espacio + nombre-de-la-etiqueta



```
git tag v1.0
```

Las etiquetas anotadas: son más completas, almacenan el autor, la fecha y un mensaje de confirmación. Se realizan mediante el siguiente comando:

git + espacio + tag + espacio + -a + espacio + nombre-de-la-etiqueta + espacio + -m + espacio + mensaje-de-la-etiqueta

The screenshot shows the Visual Studio Code interface. The left sidebar has icons for Explorer, Search, Problems, and others. The Explorer view shows a folder named 'PRUEBA-GIT-GITHUB' containing a file 'main.py'. The main editor area displays the following Python code:

```
1 print("Hola Mundo")
```

Below the editor is a terminal window with the following history:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git tag v1.0
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git tag -a v1.1 -m "Versión 1.1 - prueba"
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

18. ¿Cómo enviar una etiqueta a GitHub?

Luego de crear una o varias etiquetas en nuestro repositorio local, podemos enviarlas a nuestro repositorio remoto en GitHub con el siguiente comando:

git + espacio + push + espacio + origin + espacio + nombre-de-la-etiqueta

The screenshot shows a Microsoft Visual Studio Code (VS Code) interface. The left sidebar has icons for Explorer, Search, Issues, Projects, and Outline. The Explorer view shows a folder named 'PRUEBA-GIT-GITHUB' containing a file 'main.py'. The main editor area displays the following Python code:

```
1 print("Hola Mundo")
```

The terminal at the bottom shows the following command-line session:

```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git tag -a v1.1 -m "Versión 1.1 - prueba"
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git push origin v1.0
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Natalia-Cordoba/repo-prueba.git
 * [new tag]           v1.0 -> v1.0
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git push origin v1.1
Enumerating objects: 1, done.
Counting objects: 100% (1/1), done.
Writing objects: 100% (1/1), 174 bytes | 87.00 KiB/s, done.
Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Natalia-Cordoba/repo-prueba.git
 * [new tag]           v1.1 -> v1.1
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

19. ¿Qué es un historial de Git?

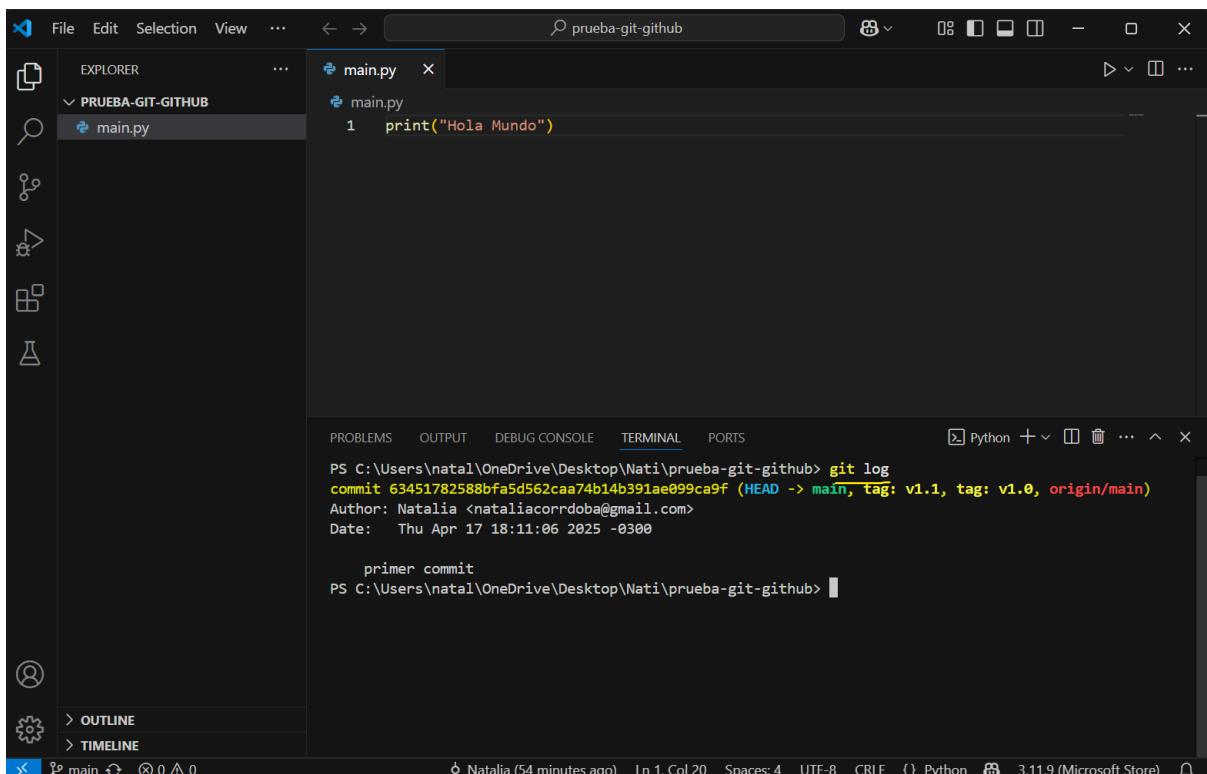
El historial de Git es el registro de todos los cambios realizados en un repositorio a lo largo del tiempo. Cada vez que hacemos un commit, Git guarda una instantánea del estado del proyecto, junto con información importante como: qué archivos cambiaron, quién hizo el cambio, cuándo se hizo y un mensaje que explica por qué se hizo el cambio.

El historial nos sirve para conocer la evolución de nuestro proyecto, si trabajamos en equipo para ver qué hizo cada uno, y también para volver hacia atrás si cometimos algún error.

20. ¿Cómo ver el historial de Git?

Podemos consultar el historial de Git utilizando el comando:

git + espacio + log



```
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git log
commit 63451782588bfa5d562caa74b14b391ae099ca9f (HEAD -> main, tag: v1.1, tag: v1.0, origin/main)
Author: Natalia <nataliacorrdoba@gmail.com>
Date:   Thu Apr 17 18:11:06 2025 -0300

    primer commit
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>
```

21. ¿Cómo buscar en el historial de Git?

Para buscar dentro del historial tenemos varias opciones. Algunas son:

Podemos buscar por mensaje de commit, por ejemplo, si recordamos parte del mensaje que usamos en un commit usamos:

git + espacio + log + espacio + --grep="texto-a-buscar"

```

PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git log
commit 63451782588bfa5d562caa74b14b391ae099ca9f (HEAD -> main, tag: v1.1, tag: v1.0, origin/main)
Author: Natalia <nataliacorrdoba@gmail.com>
Date:   Thu Apr 17 18:11:06 2025 -0300

    primer commit
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github> git log --grep="primer"
commit 63451782588bfa5d562caa74b14b391ae099ca9f (HEAD -> main, tag: v1.1, tag: v1.0, origin/main)
Author: Natalia <nataliacorrdoba@gmail.com>
Date:   Thu Apr 17 18:11:06 2025 -0300

    primer commit
PS C:\Users\natal\OneDrive\Desktop\Nati\prueba-git-github>

```

También podemos buscar por autor del commit:

git + espacio + log + espacio + --author="nombre"

Otra opción es buscar cambios en un archivo específico:

git + espacio + log + espacio + nombre-del-archivo

22. ¿Cómo borrar el historial de Git?

En Git tenemos dos formas de trabajar con el historial, dependiendo de lo que necesitemos: modificar el historial o eliminarlo por completo.

Si solo queremos modificar el historial reciente, por ejemplo, eliminar un commit, corregir un mensaje o quitar un archivo sensible, podemos usar algunos comandos como:

- para deshacer commits recientes

git + espacio + reset

git + espacio + reset + espacio + nombre-archivo

git + espacio + reset + espacio + nombre-carpeta

- para editar los últimos *n* commits

git + espacio + rebase + espacio + -i + espacio + HEAD~n

- para deshacer cambios sin borrar historial

git + espacio + restore

o

git + espacio + revert

Si queremos empezar desde cero y eliminar todo el historial anterior, podemos ejecutar los siguientes comandos:

Primero debemos borrar la carpeta interna de Git:

rm + espacio + -rf + espacio + .git

Segundo, volvemos a inicializar el repositorio:

git + espacio + init

git + espacio + add + espacio + .

git + espacio + commit + espacio + -m + espacio + mensaje-commit

Si el repositorio está en GitHub, tendremos que forzar el push con el siguiente comando:

**git + espacio + remote + espacio + add + espacio + origin + espacio +
URL-del-repositorio**

git + espacio + push + espacio + -f + espacio + origin + espacio + nombre-rama

23. ¿Qué es un repositorio privado en GitHub?

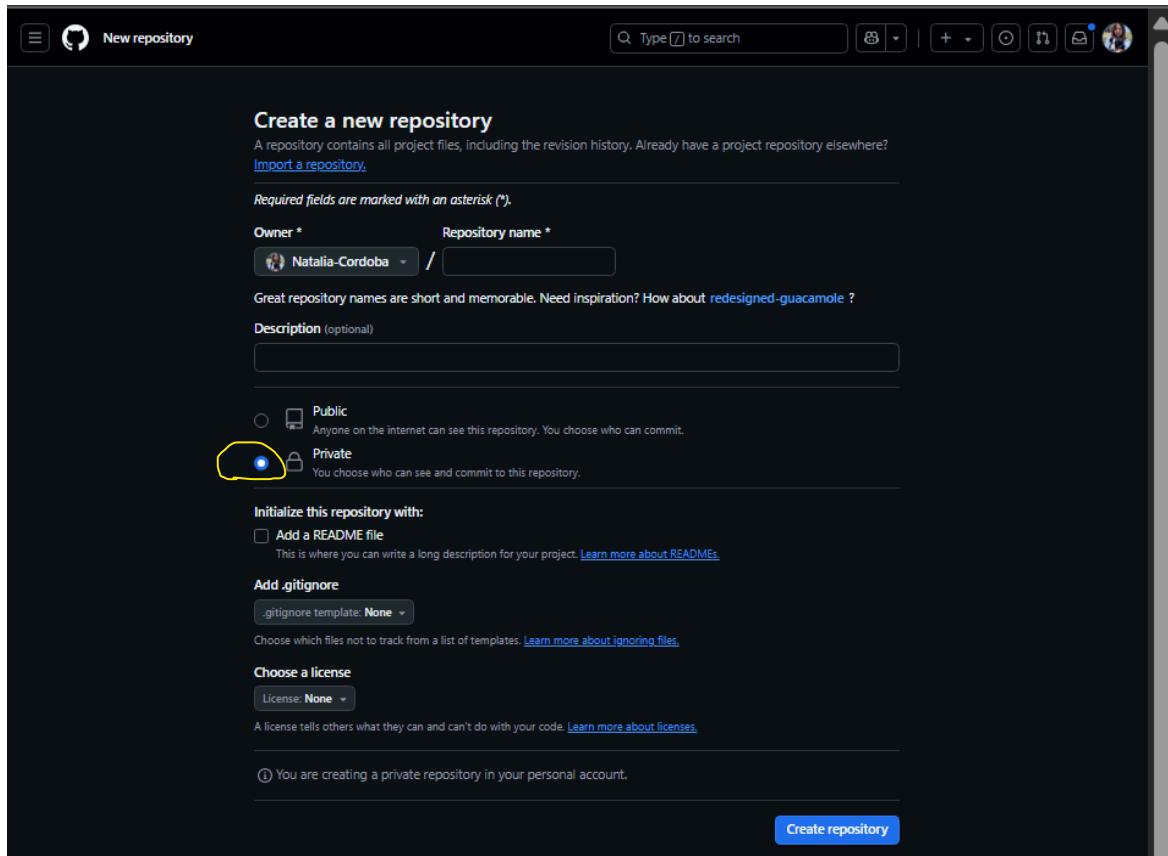
Un repositorio privado en GitHub es un repositorio que solo pueden ver o acceder aquellos colaboradores que estén autorizados.

24. ¿Cómo crear un repositorio privado en GitHub?

Durante la creación de un repositorio en GitHub, proceso que explicamos en la resolución de la pregunta 2, debemos elegir la opción de visibilidad privada.

Resumen del proceso:

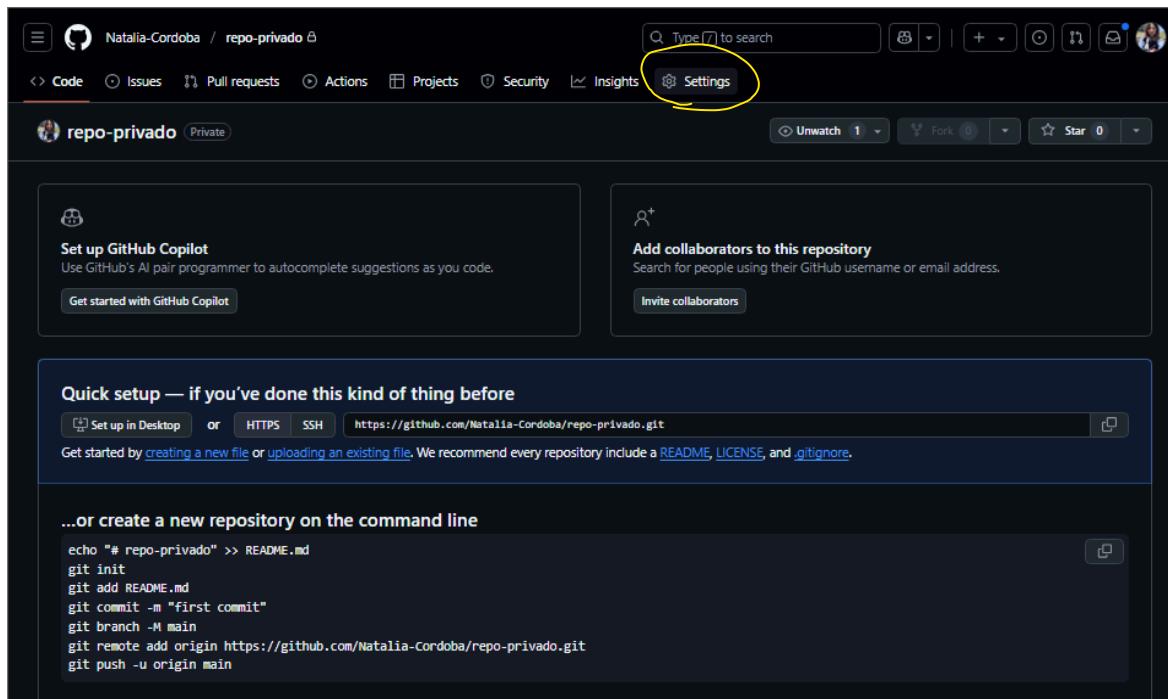
- crearse una cuenta en GitHub.
- iniciar sesión en GitHub.
- hacer clic en el signo más "+" que se encuentra en la esquina superior derecha y seleccionar "Nuevo repositorio".
- ponerle nombre al repositorio, opcionalmente una descripción, elegir como visibilidad la opción privado, y hacer clic en "Crear repositorio".



25. ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Los pasos a seguir que debemos hacer para invitar a una persona a nuestro repositorio privado son los siguientes:

- entramos al repositorio privado en GitHub.
- hacemos clic en la pestaña **Settings**



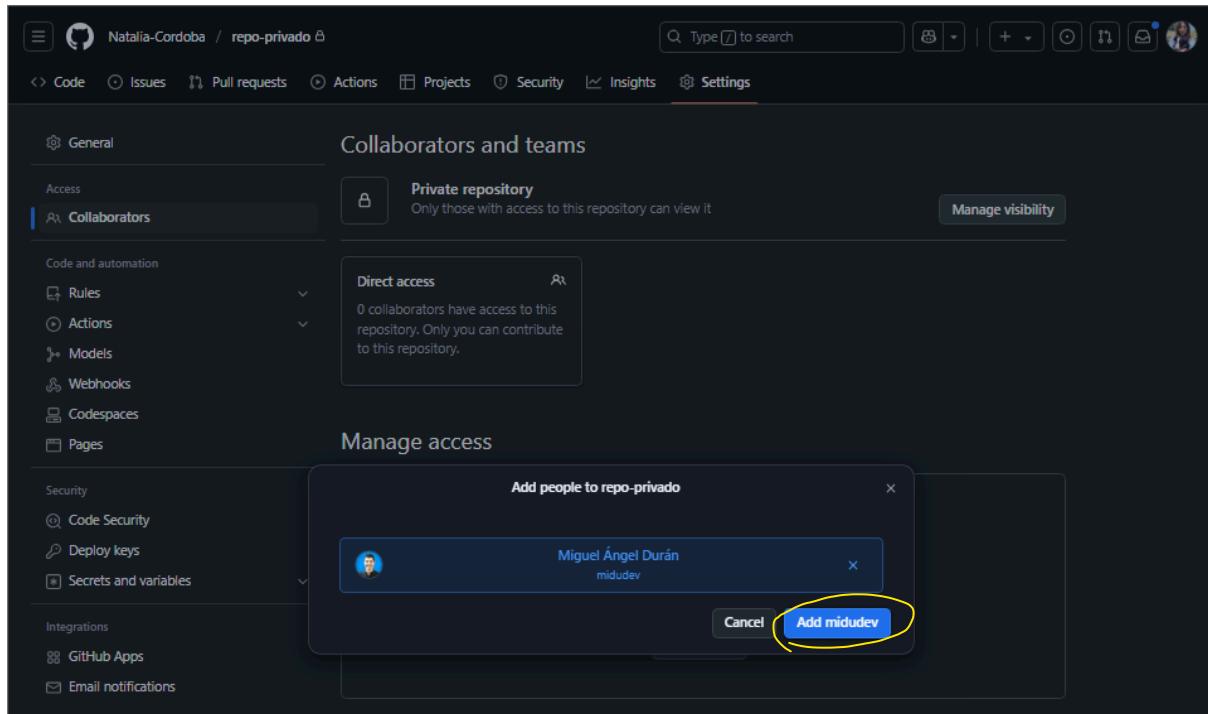
- en el menú lateral, seleccionamos **Collaborators**

The screenshot shows the GitHub repository settings page for 'repo-privado'. The left sidebar has a 'Collaborators' section highlighted with a yellow circle. The main area shows repository details like 'Repository name: repo-privado' and various configuration options under 'General' and 'Features' sections.

- hacemos clic en el botón **Add people**.

The screenshot shows the 'Collaborators and teams' section of the repository settings. The 'Collaborators' tab is selected. A large button labeled 'Add people' is highlighted with a yellow circle. Below it, a message says 'You haven't invited any collaborators yet'.

- escribimos el nombre de usuario de GitHub de la persona que queremos invitar y luego damos clic en **Add to repository**, para que esa persona reciba una invitación por correo o en su perfil de GitHub, la cual debe aceptar para poder acceder



26. ¿Qué es un repositorio público en GitHub?

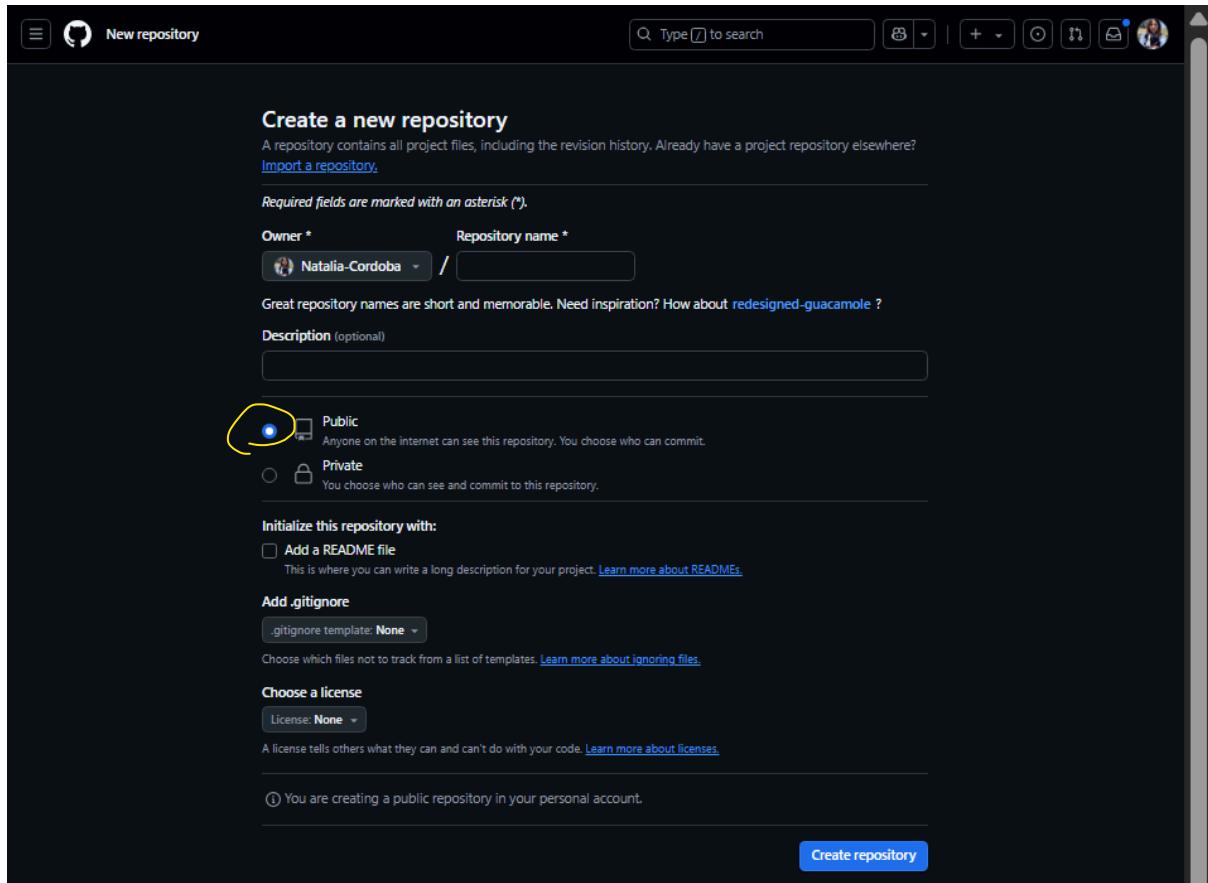
A diferencia del repositorio privado, donde el acceso se restringe a determinados desarrolladores. En un repositorio público en GitHub, este repositorio se puede ver y acceder por cualquier persona. Los repositorios públicos, además, pueden ser clonados por quienes accedan a ellos, recibir pull request e incluso contribuciones si quien accede cuenta con los permisos para ello.

27. ¿Cómo crear un repositorio público en GitHub?

Durante la creación de un repositorio en GitHub, proceso que explicamos en la resolución de la pregunta 2, debemos elegir la opción de visibilidad pública.

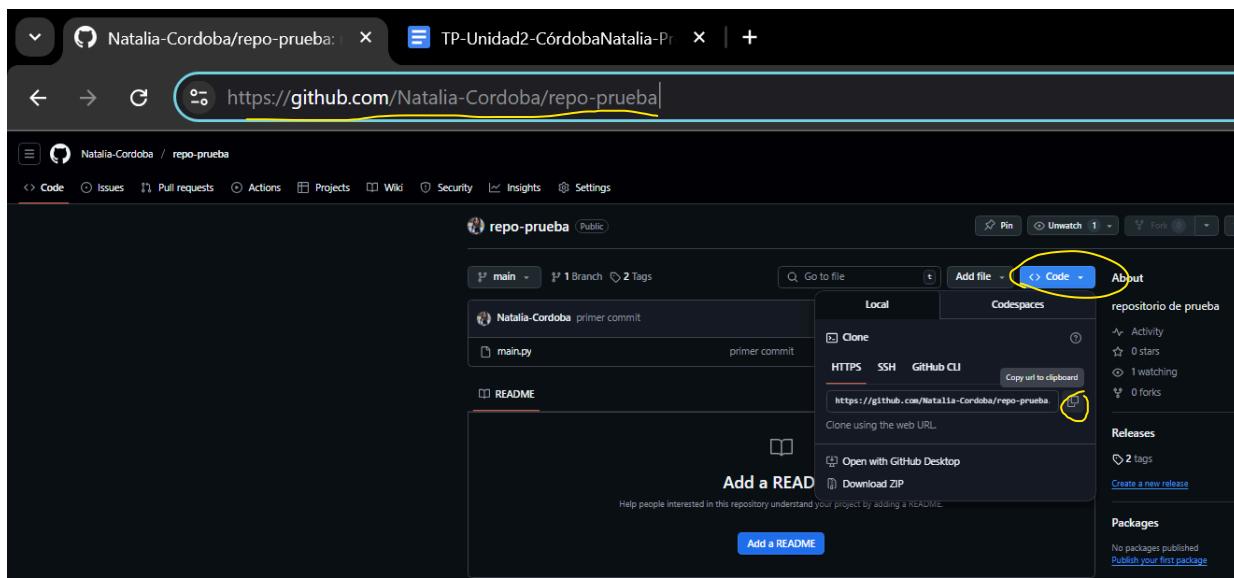
Resumen del proceso:

- crearse una cuenta en GitHub.
- iniciar sesión en GitHub.
- hacer clic en el signo más “+” que se encuentra en la esquina superior derecha y seleccionar “Nuevo repositorio”.
- ponerle nombre al repositorio, opcionalmente una descripción, elegir como visibilidad la opción público, y hacer clic en “Crear repositorio”.



28. ¿Cómo compartir un repositorio público en GitHub?

La forma más sencilla de compartir un repositorio público en GitHub, es compartiendo el enlace URL del repositorio. Para ello, primero debemos ingresar al repositorio que queremos compartir. Podemos copiar el enlace directamente desde la barra con la URL, en la parte superior del repositorio; o presionar el botón **<> Code** y luego el botón **Copiar**.

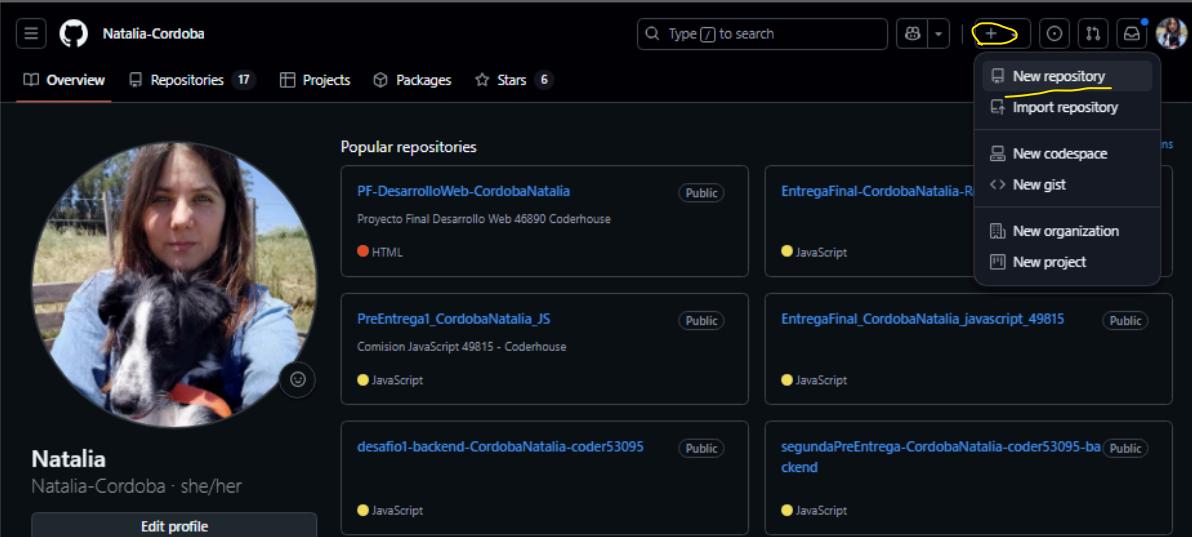


Otra forma en que podemos invitar a otros colaboradores a nuestro repositorio público es mediante el mismo proceso que realizamos con los repositorios privados.

2) Realizar la siguiente actividad:

Crear un repositorio.

- Dale un nombre al repositorio.
- Elige que el repositorio sea público.
- Inicializa el repositorio con un archivo.



Natalia-Cordoba

Overview Repositories 17 Projects Packages Stars 6

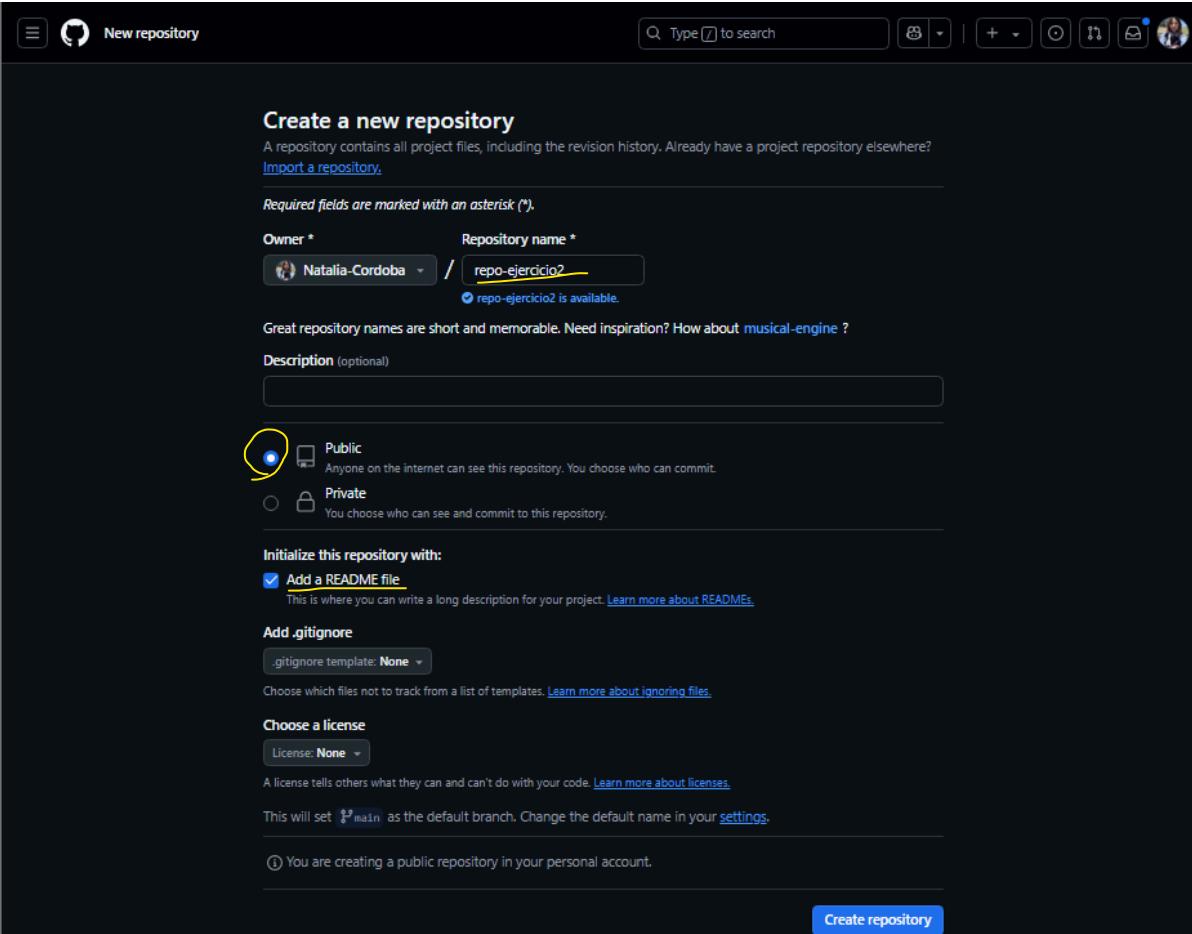
Type to search

Popular repositories

- PF-DesarrolloWeb-CordobaNatalia (Public) HTML
- EntregaFinal-CordobaNatalia-R (JavaScript)
- PreEntrega1_CordobaNatalia_JS (Public) JavaScript
- EntregaFinal_CordobaNatalia_javascript_49815 (JavaScript)
- desafio1-backend-CordobaNatalia-coder53095 (Public) JavaScript
- segundaPreEntrega-CordobaNatalia-coder53095-ba (JavaScript)

Natalia
Natalia-Cordoba - she/her

Edit profile



New repository

Type to search

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 * Repository name *

Natalia-Cordoba / repo-ejercicio2 is available.

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

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:

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

Add .gitignore

.gitignore template:

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

Choose a license

License:

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](#).

You are creating a public repository in your personal account.

Create repository

The screenshot shows a GitHub repository named 'repo-ejercicio2'. The repository has 1 branch ('main') and 0 tags. It contains a single commit from 'Natalia-Cordoba' titled 'Initial commit' at 'f061e9d · now'. The commit message is 'Initial commit'. The repository has 0 stars, 1 watching, and 0 forks. There are no releases published.

Agregando un Archivo

- Crea un archivo simple, por ejemplo, "mi-archivo.txt".
- Realiza los comandos `git add .` y `git commit -m "Agregando mi-archivo.txt"` en la línea de comandos.
- Sube los cambios al repositorio en GitHub con `git push origin main` (o el nombre de la rama correspondiente).

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

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\natal> git clone https://github.com/Natalia-Cordoba/repo-ejercicio2.git
Cloning into 'repo-ejercicio2'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd repo-ejercicio2
PS C:\Users\natal\repo-ejercicio2> echo "Nuevo archivo de texto" > mi-archivo.txt
PS C:\Users\natal\repo-ejercicio2> git add .
PS C:\Users\natal\repo-ejercicio2> git commit -m "Agregando mi-archivo.txt"
[main d0a91b7] Agregando mi-archivo.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 mi-archivo.txt
PS C:\Users\natal\repo-ejercicio2> git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 333 bytes | 166.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Natalia-Cordoba/repo-ejercicio2.git
  f061e9d..d0a91b7  main -> main
PS C:\Users\natal\repo-ejercicio2>
```

Creando Branchs

- Crear una Branch
- Realizar cambios o agregar un archivo
- Subir la Branch

```

Windows PowerShell
To https://github.com/Natalia-Cordoba/repo-ejercicio2.git
f061e9d..d0a91b7  main -> main
PS C:\Users\natal\repo-ejercicio2> git branch nueva-rama
PS C:\Users\natal\repo-ejercicio2> git checkout nueva-rama
git: 'ckeckout' is not a git command. See 'git --help'.
The most similar command is
    checkout
PS C:\Users\natal\repo-ejercicio2> git checkout nueva-rama
Switched to branch 'nueva-rama'
PS C:\Users\natal\repo-ejercicio2> echo "Segundo archivo de texto" > mi-nuevo-archivo.txt
PS C:\Users\natal\repo-ejercicio2> git add .
PS C:\Users\natal\repo-ejercicio2> git commit -m "Agregando mi-nuevo-archivo.txt"
[nueva-rama 90d3a16] Agregando mi-nuevo-archivo.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 mi-nuevo-archivo.txt
PS C:\Users\natal\repo-ejercicio2> git push origin nueva-rama
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 373 bytes | 373.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'nueva-rama' on GitHub by visiting:
remote:     https://github.com/Natalia-Cordoba/repo-ejercicio2/pull/new/nueva-rama
remote:
To https://github.com/Natalia-Cordoba/repo-ejercicio2.git
 * [new branch]      nueva-rama -> nueva-rama
PS C:\Users\natal\repo-ejercicio2>

```

A screenshot of a GitHub repository page for 'repo-ejercicio2'. The main navigation bar includes 'Code', 'Issues', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. Below the navigation is a search bar and a header for the repository 'repo-ejercicio2' (Public). A notification bar at the top says 'nueva-rama had recent pushes 37 seconds ago' with a 'Compare & pull request' button. The repository summary shows 'main' branch, '2 Branches' (circled in yellow), '0 Tags', and a commit history for 'Natalia-Cordoba' adding 'mi-archivo.txt'. The 'README' file is also visible. On the right side, there's an 'About' section with a placeholder 'No description, website, or topics provided.', and sections for 'Readme', 'Activity', 'Stars', 'Watching', and 'Forks'. There are also 'Releases' and 'Packages' sections.

A screenshot of the GitHub 'Branches' page for the same repository. The top navigation bar is identical. The main content area is titled 'Branches' with a 'New branch' button. It includes tabs for 'Overview', 'Yours', 'Active', 'Stale', and 'All', and a search bar. The 'Default' section shows the 'main' branch with an update time of '9 minutes ago' and a status of 'Default'. The 'Your branches' section shows the 'nueva-rama' branch with an update time of '1 minute ago' and a status of '0 | 1'. The 'Active branches' section also shows the 'nueva-rama' branch with an update time of '1 minute ago' and a status of '0 | 1'.

The screenshot shows a GitHub repository page for 'repo-ejercicio2'. At the top, there's a banner indicating 'nueva-rama had recent pushes 1 minute ago'. Below this, the 'Code' tab is selected. A yellow box highlights the 'nueva-rama' dropdown menu and the list of commits. The commits are as follows:

- Natalia-Cordoba Agregando mi-nuevo-archivo.txt
- Initial commit
- 90d3a16 · 2 minutes ago
- 3 Commits
- README.md
- Initial commit
- 16 minutes ago
- mi-archivo.txt
- Agregando mi-archivo.txt
- 9 minutes ago
- mi-nuevo-archivo.txt
- Agregando mi-nuevo-archivo.txt
- 2 minutes ago

On the right side, there are sections for 'About', 'Readme', 'Activity', 'Contributors', 'Releases', and 'Packages'.

3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

- Ve a GitHub e inicia sesión en tu cuenta.
- Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
- Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
- Opcionalmente, añade una descripción.
- Marca la opción "Initialize this repository with a README".
- Haz clic en "Create repository".

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 * **Repository name ***

Natalia-Cordoba / conflict-exercise conflict-exercise is available.

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

Description (optional)

repositorio para el ejercicio 3 de la unidad 2 repository for exercise 3 of unit 2

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](#).

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](#).

You are creating a public repository in your personal account.

Create repository

Natalia-Cordoba / conflict-exercise

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

conflict-exercise Public

main 1 Branch 0 Tags

Go to file + Code

Natalia-Cordoba Initial commit 2c58862 · now 1 Commit

README.md Initial commit now

README

conflict-exercise

repositorio para el ejercicio 3 de la unidad 2

About

repository for exercise 3 of unit 2

Readme Activity 0 stars 1 watching 0 forks

Releases

No releases published [Create a new release](#)

Paso 2: Clonar el repositorio a tu máquina local

- Copia la URL del repositorio
(usualmente algo como <https://github.com/tuusuuario/conflict-exercise.git>)
- Abre la terminal o línea de comandos en tu máquina.
- Clona el repositorio usando el comando:
`git clone https://github.com/tuusuuario/conflict-exercise.git`
- Entra en el directorio del repositorio: `cd conflict-exercise`

Natalia-Cordoba / conflict-exercise

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

conflict-exercise Public

main 1 Branch 0 Tags

Natalia-Cordoba Initial commit

README.md Initial commit

README

conflict-exercise

repositorio para el ejercicio 3 de la unidad 2

About

repository para el ejercicio 3 de la unidad 2

Readme Activity 0 stars 1 watching 0 forks

Releases

No releases published Create a new release

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

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras.
https://aka.ms/PSWindows

PS C:\Users\natal> git clone https://github.com/Natalia-Cordoba/conflict-exercise.git
Cloning into 'conflict-exercise'...
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 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd conflict-exercise
PS C:\Users\natal\conflict-exercise>
```

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch:
git checkout -b feature-branch
- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:
Este es un cambio en la feature branch.
- Guarda los cambios y haz un commit:
git add README.md
git commit -m "Added a line in feature-branch"

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

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras.
https://aka.ms/PSWindows

PS C:\Users\natal> git clone https://github.com/Natalia-Cordoba/conflict-exercise.git
Cloning into 'conflict-exercise'...
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 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd conflict-exercise
PS C:\Users\natal\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\natal\conflict-exercise> |
```

```
① README.md X

C: > Users > natal > conflict-exercise > ① README.md > abc # conflict-exercise
  1  # conflict-exercise
  2  repositorio para el ejercicio 3 de la unidad 2
  3  Este es un cambio en la feature branch.|
```

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

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras.
https://aka.ms/PSWindows

PS C:\Users\natal> git clone https://github.com/Natalia-Cordoba/conflict-exercise.git
Cloning into 'conflict-exercise'...
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 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd conflict-exercise
PS C:\Users\natal\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch 3b1edc3] Added a line in feature-branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise>
```

Paso 4: Volver a la rama principal y editar el mismo archivo

- Cambia de vuelta a la rama principal (main):
git checkout main
- Edita el archivo README.md de nuevo, añadiendo una línea diferente:
Este es un cambio en la main branch.

- Guarda los cambios y haz un commit:

```
git add README.md
```

```
git commit -m "Added a line in main branch"
```

```
Windows PowerShell

Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\natal> git clone https://github.com/Natalia-Cordoba/conflict-exercise.git
Cloning into 'conflict-exercise'...
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 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd conflict-exercise
PS C:\Users\natal\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch 3b1edc3] Added a line in feature-branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\conflict-exercise>
```

```
README.md X

C: > Users > natal > conflict-exercise > README.md > # conflict-exercise
 1 # conflict-exercise
 2 repositorio para el ejercicio 3 de la unidad 2
 3 Este es un cambio en la main branch.
```

```
PS C:\Users\natal> git clone https://github.com/Natalia-Cordoba/conflict-exercise.git
Cloning into 'conflict-exercise'...
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 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd conflict-exercise
PS C:\Users\natal\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch 3b1edc3] Added a line in feature-branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in main branch"
[main c0f3486] Added a line in main branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise>
```

Paso 5: Hacer un merge y generar un conflicto

- Intenta hacer un merge de la feature-branch en la rama main:
git merge feature-branch
- Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

```
PS C:\Users\natal> git clone https://github.com/Natalia-Cordoba/conflict-exercise.git
Cloning into 'conflict-exercise'...
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 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd conflict-exercise
PS C:\Users\natal\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch 3b1edc3] Added a line in feature-branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in main branch"
[main c0f3486] Added a line in main branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise> git merge feature-branch
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
PS C:\Users\natal\conflict-exercise>
```

Paso 6: Resolver el conflicto

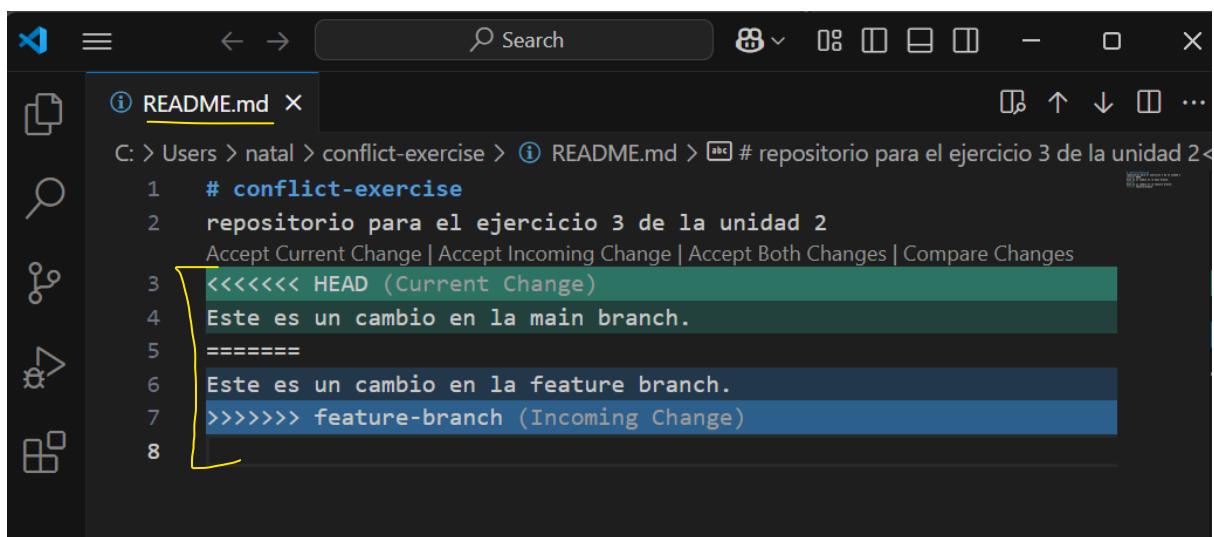
- Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:
<<<<< HEAD
Este es un cambio en la main branch.

=====

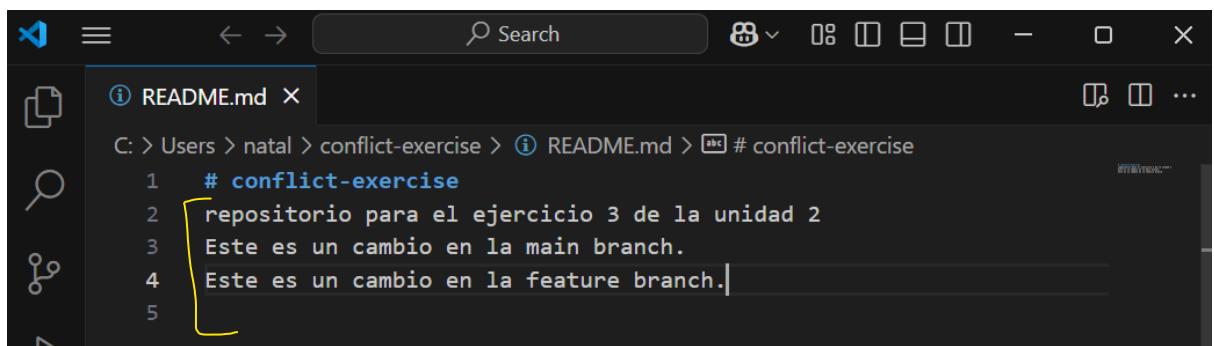
Este es un cambio en la feature branch.

>>>>> feature-branch

- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
- Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estés solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).
- Añade el archivo resuelto y completa el merge:
git add README.md
git commit -m "Resolved merge conflict"



```
C: > Users > natal > conflict-exercise > ⓘ README.md > abc # repositorio para el ejercicio 3 de la unidad 2<
1 # conflict-exercise
2 repositorio para el ejercicio 3 de la unidad 2
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
3 <<<<< HEAD (Current Change)
4 Este es un cambio en la main branch.
5 =====
6 Este es un cambio en la feature branch.
7 >>>>> feature-branch (Incoming Change)
8
```



```
C: > Users > natal > conflict-exercise > ⓘ README.md > abc # conflict-exercise
1 # conflict-exercise
2 repositorio para el ejercicio 3 de la unidad 2
3 Este es un cambio en la main branch.
4 Este es un cambio en la feature branch.
5
```

```
Windows PowerShell

remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\natal> cd conflict-exercise
PS C:\Users\natal\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch 3b1edc3] Added a line in feature-branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Added a line in main branch"
[main c0f3486] Added a line in main branch
 1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise> git merge feature-branch
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Resolved merge conflict"
[main bb5d717] Resolved merge conflict
PS C:\Users\natal\conflict-exercise>
```

Paso 7: Subir los cambios a GitHub

- Sube los cambios de la rama main al repositorio remoto en GitHub:
git push origin main
- También sube la feature-branch si deseas:
git push origin feature-branch

```
Windows PowerShell

1 file changed, 1 insertion(+)
PS C:\Users\natal\conflict-exercise> git merge feature-branch
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
PS C:\Users\natal\conflict-exercise> git add README.md
PS C:\Users\natal\conflict-exercise> git commit -m "Resolved merge conflict"
[main bb5d717] Resolved merge conflict
PS C:\Users\natal\conflict-exercise> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 797 bytes | 199.00 KiB/s, done.
Total 9 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/Natalia-Cordoba/conflict-exercise.git
 2c58862..bb5d717 main -> main
PS C:\Users\natal\conflict-exercise> git push origin feature-branch
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature-branch' on GitHub by visiting:
remote:     https://github.com/Natalia-Cordoba/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/Natalia-Cordoba/conflict-exercise.git
 * [new branch]      feature-branch -> feature-branch
PS C:\Users\natal\conflict-exercise>
```

Paso 8: Verificar en GitHub

- Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
- Puedes revisar el historial de commits para ver el conflicto y su resolución.

The screenshot shows the GitHub repository 'conflict-exercise'. At the top, there are navigation links: Code, Issues, Pull requests, Actions, Projects, Wiki, Security, Insights, and Settings. Below the header, the repository name 'conflict-exercise' is shown as public. A dropdown menu indicates the 'main' branch is selected. A circled area highlights the '2 Branches' link. The repository details show 'Natalia-Cordoba' as the owner, a commit from 'bb5d717' resolved a merge conflict 9 minutes ago, and a file 'READMEmd' was updated 9 minutes ago. The 'About' section includes a description of the repository as 'repositorio para el ejercicio 3 de la unidad 2', and the 'Releases' section indicates 'No releases published' with a link to 'Create a new release'. The 'Packages' section is also visible.

The screenshot shows the 'README.md' file in the 'conflict-exercise' repository. The left sidebar shows the 'Files' tab with 'main' selected, and the 'README.md' file is highlighted. The main content area displays the file's content: 'conflict-exercise' and 'repositorio para el ejercicio 3 de la unidad 2 Este es un cambio en la main branch. Este es un cambio en la feature branch.' A yellow bracket highlights the second sentence.

The screenshot shows the commit history for the 'README.md' file on the 'main' branch. A yellow bracket highlights the first commit, which is a 'Resolved merge conflict' by 'Natalia-Cordoba' committed 10 minutes ago. The commit history also includes: 'Added a line in main branch' (17 minutes ago), 'Added a line in feature-branch' (23 minutes ago), and 'Initial commit' (42 minutes ago, authored by 'Natalia-Cordoba'). A blue button labeled 'Verified' is present next to the initial commit. The footer of the commit list says 'End of commit history for this file'.

