**PUSH**

**FETCH**

**PULL**

**Tarea 4 - PISA**

**Trabajo realizado por: Cristina Quiñones Castellanos**

**ÍNDICE**

[**DEFINICIONES**](#_1xib8u3imka6) **3**

[➔ PUSH](#_s429etdz1f1) 3

[➔ FETCH](#_a2fnjevtdjxw) 5

[➔ PULL](#_uljlyth0mmx8) 7

# **DEFINICIONES**

## **PUSH**

El comando **push** en Git se utiliza para enviar los cambios realizados en la rama local hacia el repositorio remoto. Esta acción asegura que los archivos y commits en los que has trabajado se sincronizan con el servidor remoto, permitiendo que otros colaboradores tengan acceso a la última versión de tu trabajo.

Es importante destacar que el comando push solamente transfiere los datos desde tu repositorio local hacia el remoto, pero no actualiza tu rama local con los cambios realizados por otros desarrolladores en el remoto. Por esta razón, se recomienda combinar el uso de push con otros comandos como pull o fetch para mantener una sincronización completa entre ambos entornos.

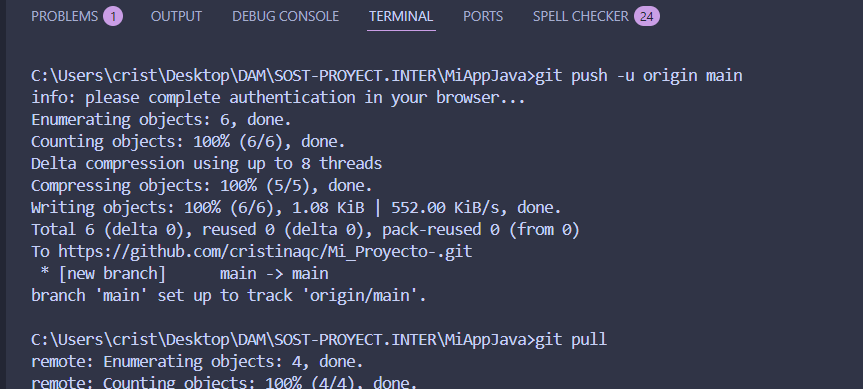
**Sintaxis básica**

La sintaxis básica para hacer uso del push es:

git push <remote> <branch>

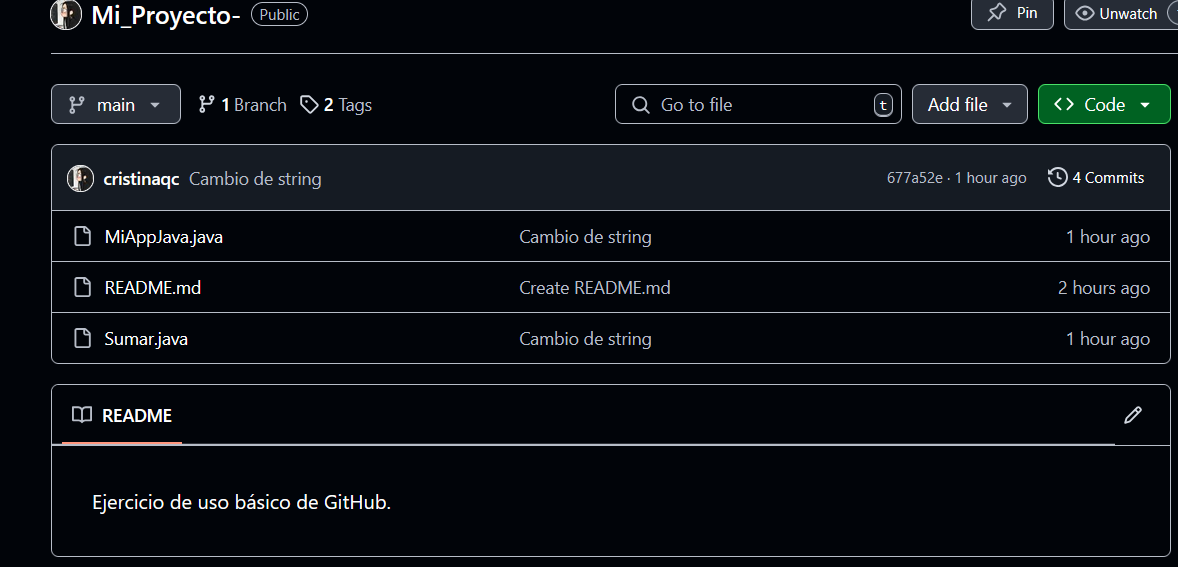
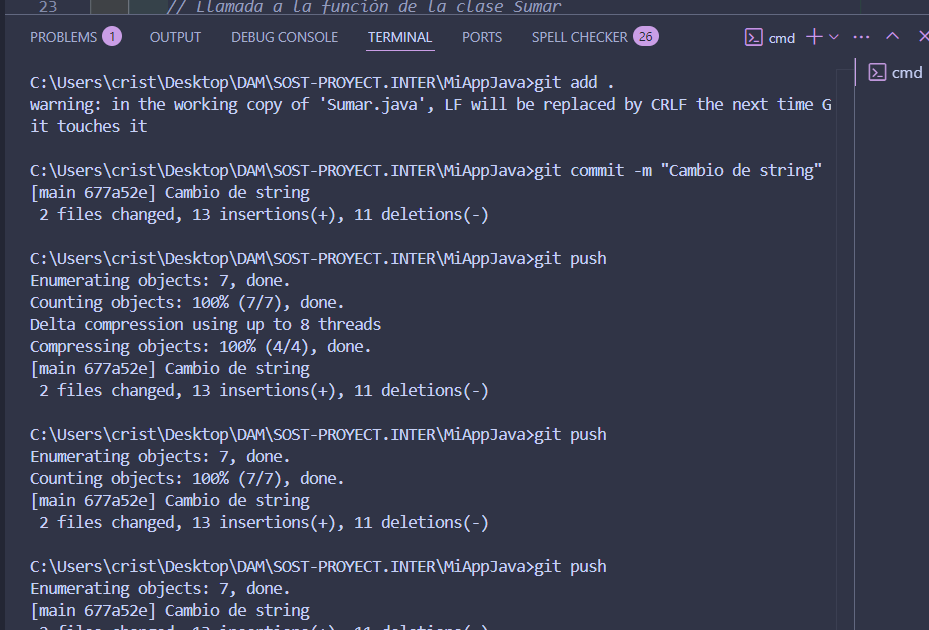
**Sintaxis cuando la rama local no existe en el repositorio remoto**

git push -u <remote> <branch>

El -u establece una conexión predeterminada entre tu rama local y la rama remota para futuros usos del comando.

**Flujo de trabajo típico hasta llegar a un push**

Antes de realizar un push, hay que realizar una serie de pasos previos:

* Primero se realizan cambios en los archivos.
* Luego se agregan al área de staging con **git add** y se confirman con **git commit**.
* Por último, se envían al repositorio remoto con **git push**.

## **FETCH**

Es un comando que permite descargar todos los cambios realizados en un repositorio remoto, pero sin combinarlos automáticamente con tu rama local. Esto significa que puedes traer información actualizada desde el servidor remoto, como nuevas ramas o commits, para revisarlos primero, antes de decidir si los integras con tu código local. Este comportamiento es muy útil cuando necesitas mantener un control preciso sobre lo que se fusiona en tu trabajo.

Cuando realizas un fetch, Git descarga todos los datos del remoto, como los objetos y referencias, y los almacena en tu repositorio local.

Sus usos principales son:

* Inspeccionar el progreso del equipo antes de actualizar tu rama.
* Identificar posibles posibles conflictos antes de integrar los cambios del remoto.
* Obtener las últimas actualizaciones del remoto sin interferir en tu entorno de trabajo actual.

**Sintaxis**

La sintaxis básica para hacer uso del fetch es:

git fetch <remote> 

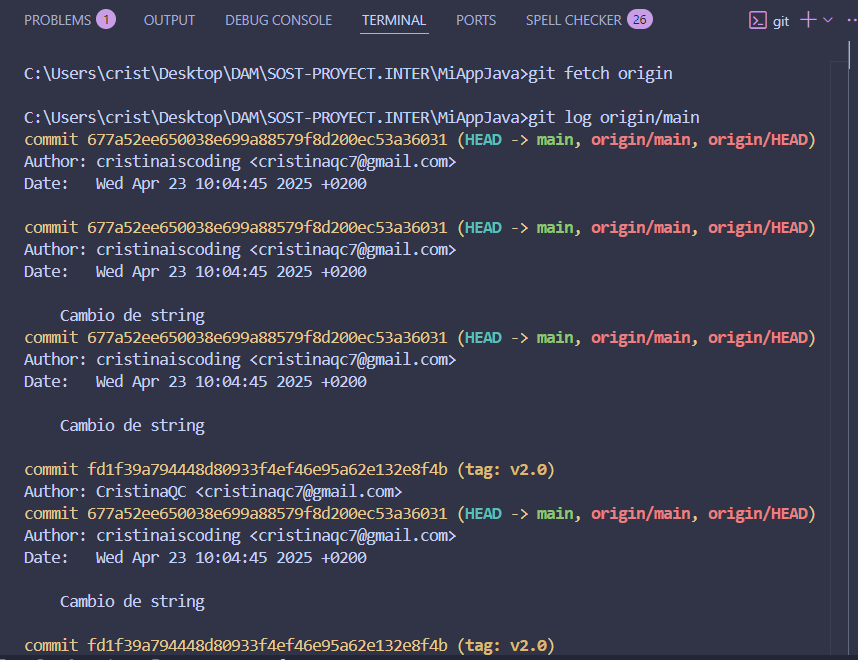
**Ejemplo práctico**

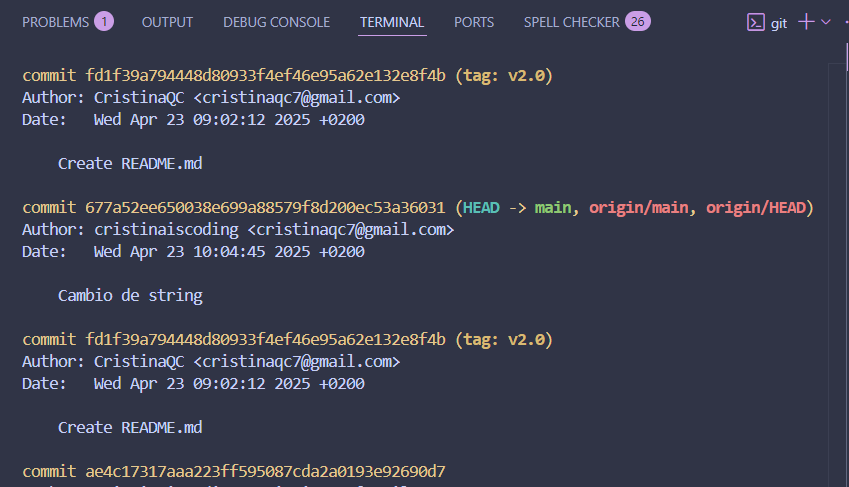
Nos descargamos los cambios en remoto sin aplicarlos con

git fetch origin

Luego, visualizamos los cambios que hemos descargado con

git log origin/main





## **PULL**

Es un comando en Git que combina dos operaciones principales: fetch y merge. Primero, descarga los cambios desde el repositorio remoto a tu repositorio local (fetch) y luego los integra automáticamente en la rama activa (merge). Esto lo convierte en una herramienta clave para mantener la rama local actualizada con los últimos cambios realizados en el remoto, especialmente en entornos colaborativos.

El objetivo principal de pull es asegurarte de que el trabajo local esté sincronizado con el progreso del equipo. Es una forma rápida y eficiente de traer los cambios del remoto y aplicarlos directamente.

Sus usos principales son:

* Actualizar tu rama local con los últimos cambios del remoto.
* Simplificar la sincronización en equipos colaborativos, evitando el uso manual de fetch seguido de un merge.
* Incorporar nuevos commits, ramas, o correcciones realizadas por otros miembros del equipo.

**Sintaxis básica**

La sintaxis básica para hacer uso del pull es:

git pull <remote> <branch>

**Ejemplo práctico**

Para descargar y fusionar los cambios realizados en la rama main del remoto origin se haría:

