EXERCISES QUE TRABAJAREMOS EN LA CUE

- EXERCISE 1: CONOCIENDO GITHUB Y SUS COMANDOS BÁSICOS
- EXERCISE 2: TRABAJO COLABORATIVO CON GITHUB
- EXERCISE 3: USANDO GITHUB PAGES COMO HOSTING GRATUITO

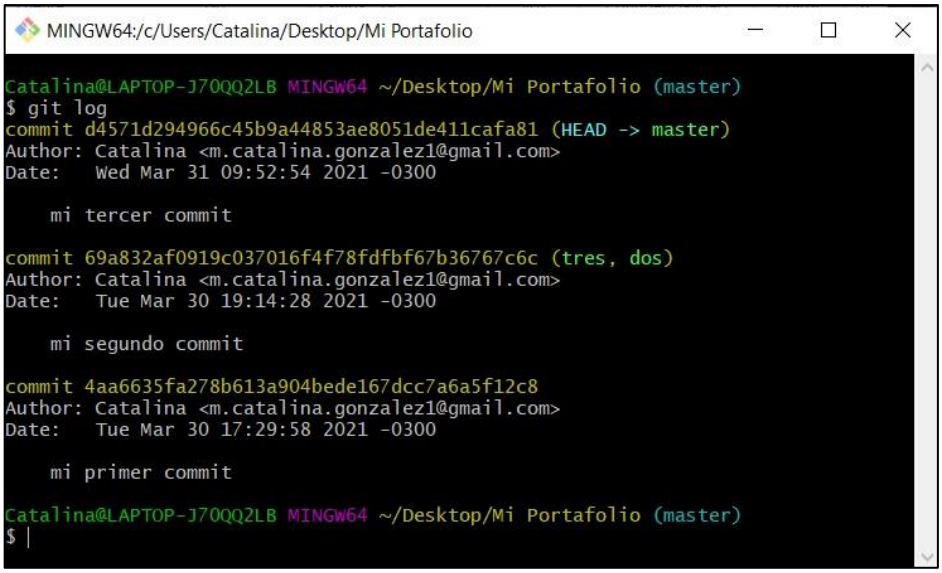
EXERCISE 1: CONOCIENDO GITHUB Y SUS COMANDOS BÁSICOS.

Hasta ahora conocimos **GIT** y trabajamos con el repositorio local. Continuaremos conociendo **GitHub** como repositorio remoto. Para eso, Trabajaremos en un proyecto simple llamado "Mi Portafolio" y veremos cómo podemos guardar los cambios que realicemos en él en **GitHub**.

Nuestro proyecto se encuentra en el siguiente estado:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Mi Portafolio</title>
</head>
<body>
    <h1>Portafolio</h1>

    <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
Consequatur eius rem
         aut praesentium, maiores illo placeat cumque aperiam cum
odit dignissimos vel
          quisquam, hic tenetur illum ex. Laudantium, ducimus
fuga.</p>
</body>
</html>
```

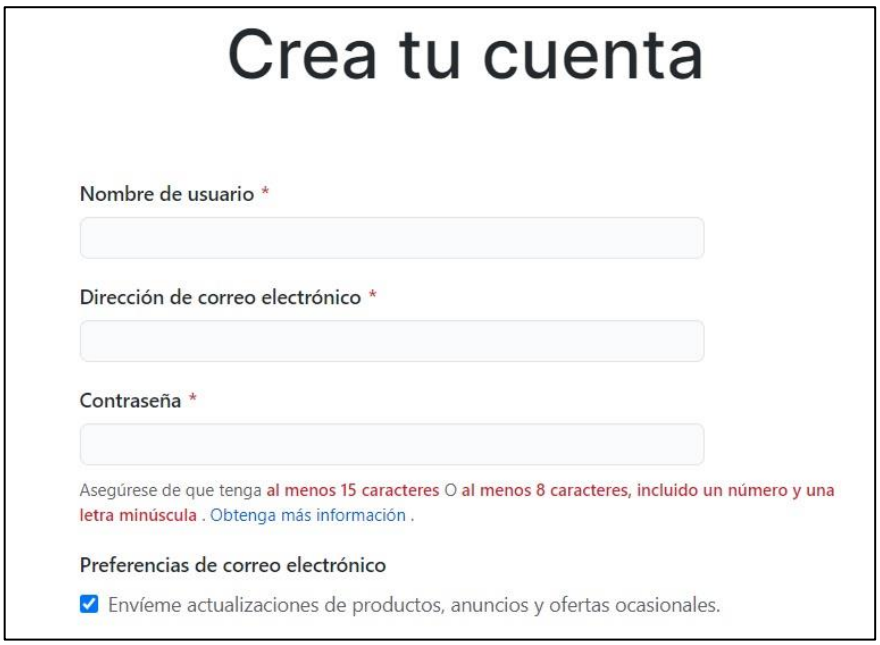
En él debemos realizar un commit. Para este ejemplo, se realizaron 3 commit. No importa si realizamos 1, 2 , 3 o 100 commit, el procedimiento será el mismo.



Ahora nos dirigiremos a la página de GitHub donde podremos acceder a muchas herramientas para desarrolladores.

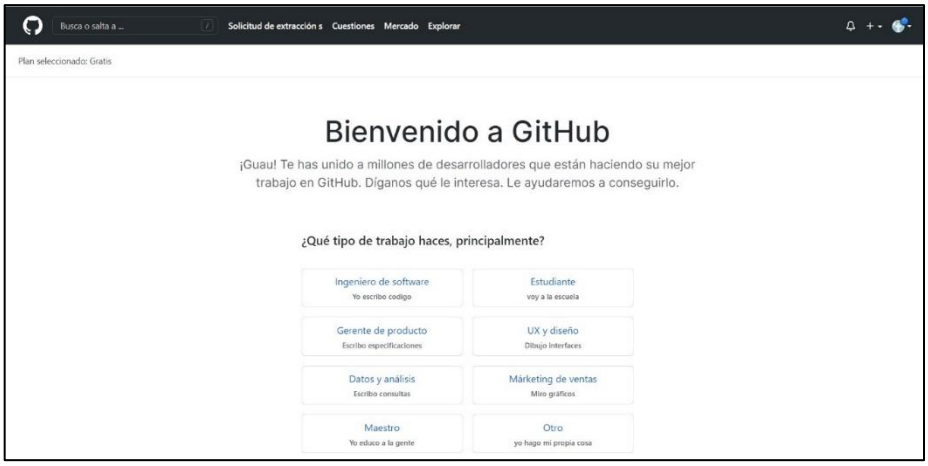
Acá utilizaremos el repositorio de código, una plataforma web donde podremos subir todos los códigos de nuestros proyectos que deseemos, incluso en su versión gratuita.

Lo primero que debemos hacer es dirigirnos a la pestaña de **registro** y registrar ahí nuestra cuenta. Si ya tenemos una cuenta creada solo debemos ingresar.

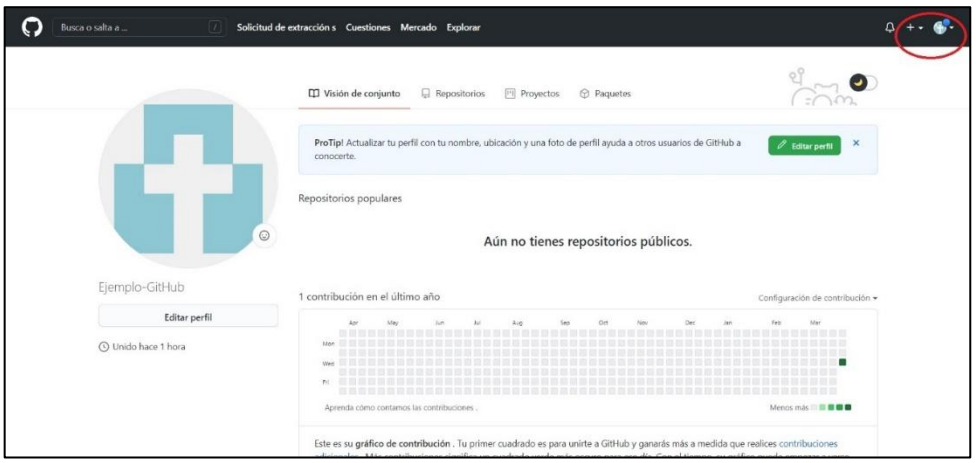


Una vez registrados, nos envía de inmediato a una pestaña donde elegiremos que función cumplimos principalmente. Podemos elegir, por ejemplo, estudiante o maestro.

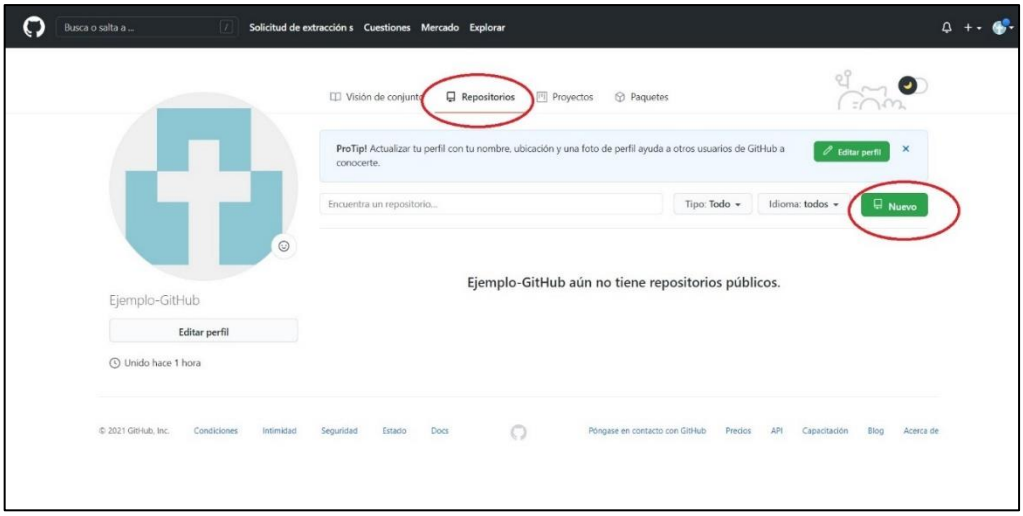
Nos pregunta también nuestra experiencia, para que usaremos **GitHub** y finalmente nos exige verificar nuestro correo electrónico.



Cuando ya nos hemos registrado podemos dirigirnos a nuestra cuenta seleccionando el icono de la parte superior derecha.

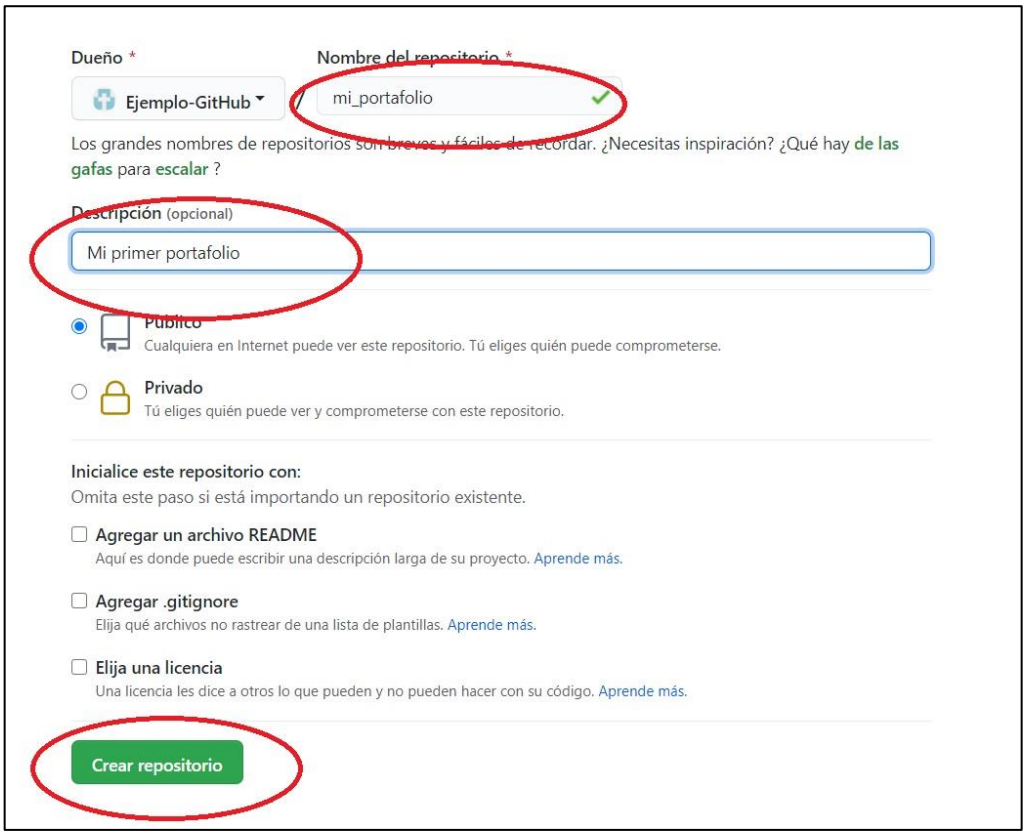


Vamos a agregar un nuevo repositorio dirigiéndonos a repositorio y seleccionando nuevo

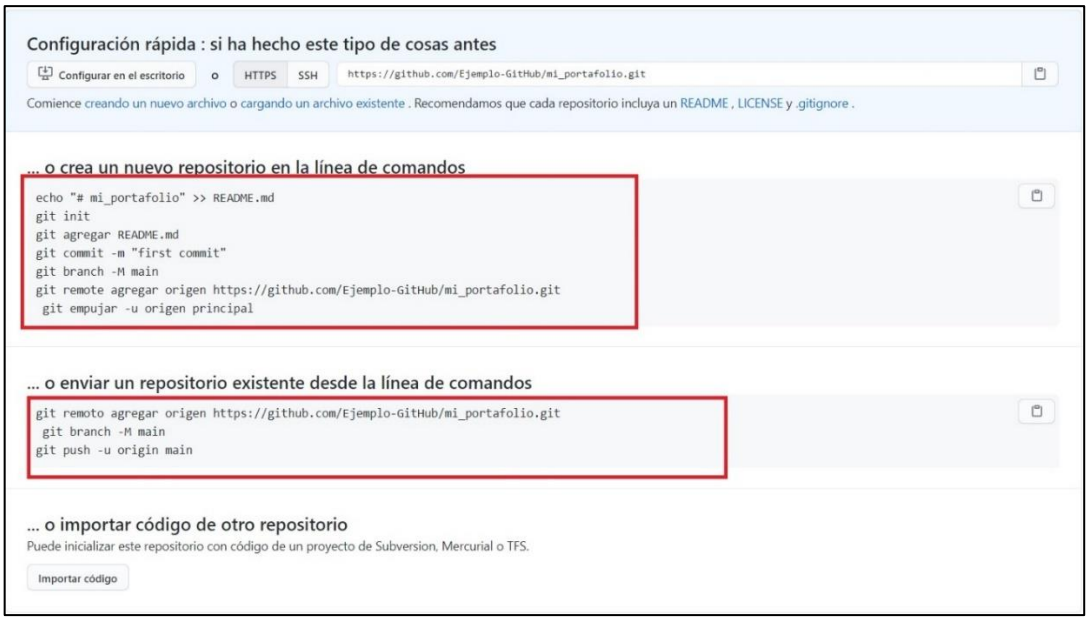


Ahora podremos colocar un nombre al nuevo repositorio, lo llamaremos “mi portafolio”. Podemos observar que nos explica que un repositorio contiene todos los archivos del proyecto, incluido el historial de versiones, ósea, lo que vemos cuando hacemos `git log`.

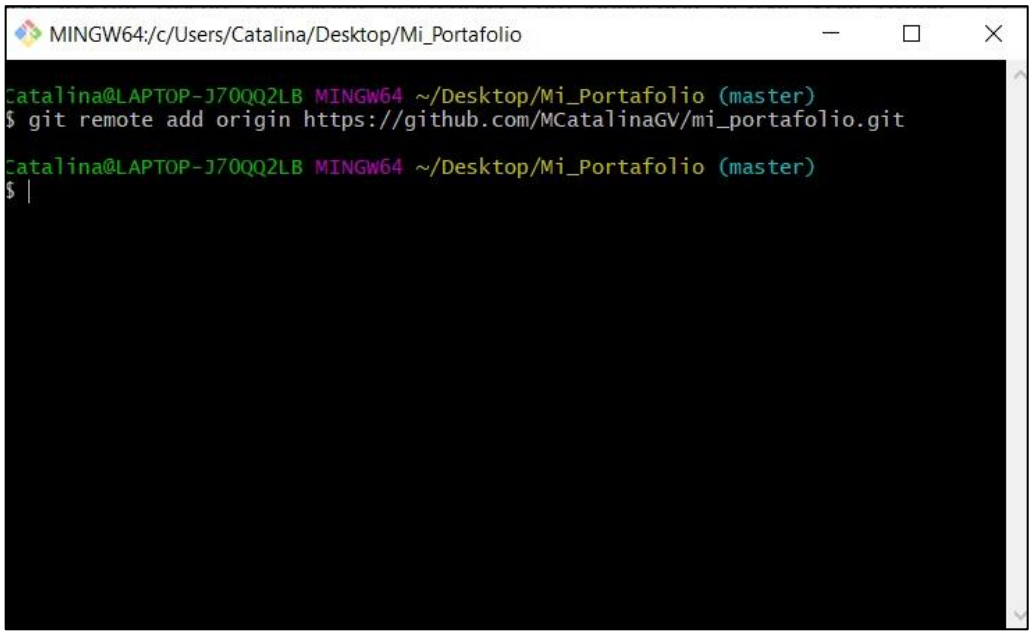
Podemos colocar una descripción del proyecto, pero esto es opcional. Lo dejaremos público (pero podemos pagar para que sea privado) y finalmente seleccionamos **Crear repositorio**.



Al momento de crearlo, inmediatamente nos muestra los pasos a seguir para poder usar nuestro repositorio. Los pasos en caso de un nuevo repositorio ya los hemos analizado en la clase anterior, por lo que continuaremos con aquellos pasos para enviar un repositorio existente.



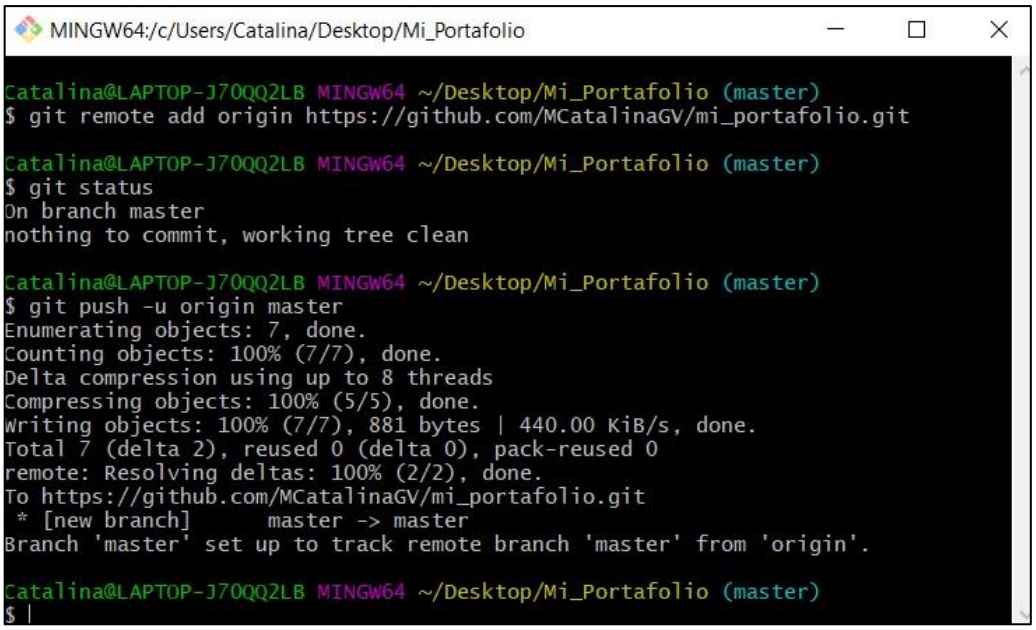
Copiamos y pegamos el comando que nos indica para agregar el repositorio



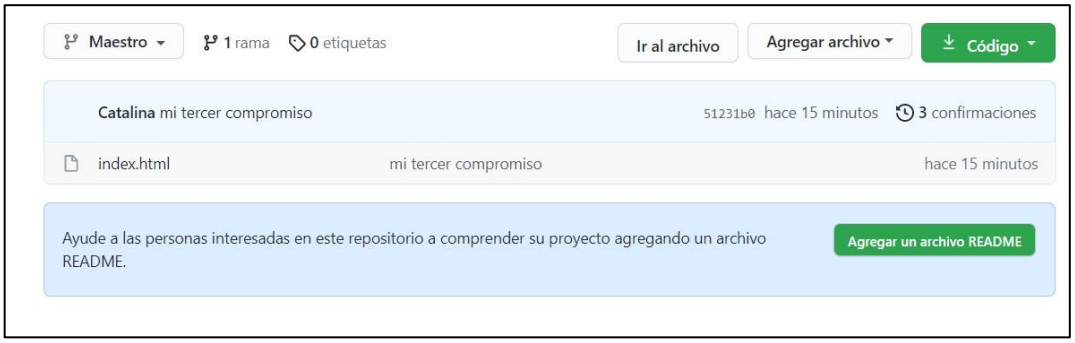
Copiamos el comando `git push -u origin main` y esperamos a ver que sucede (es necesario cambiar el comando “main” a “master” si nuestra rama principal se llama “master”).

Si es la primera vez que usaremos **GitHub** nos mostrará una pestaña donde nos solicitará colocar nuestro usuario y contraseña, de lo contrario, tomará el usuario ya registrado previamente.

Posteriormente nos mostrará el ingreso de la información a nuestro repositorio remoto.



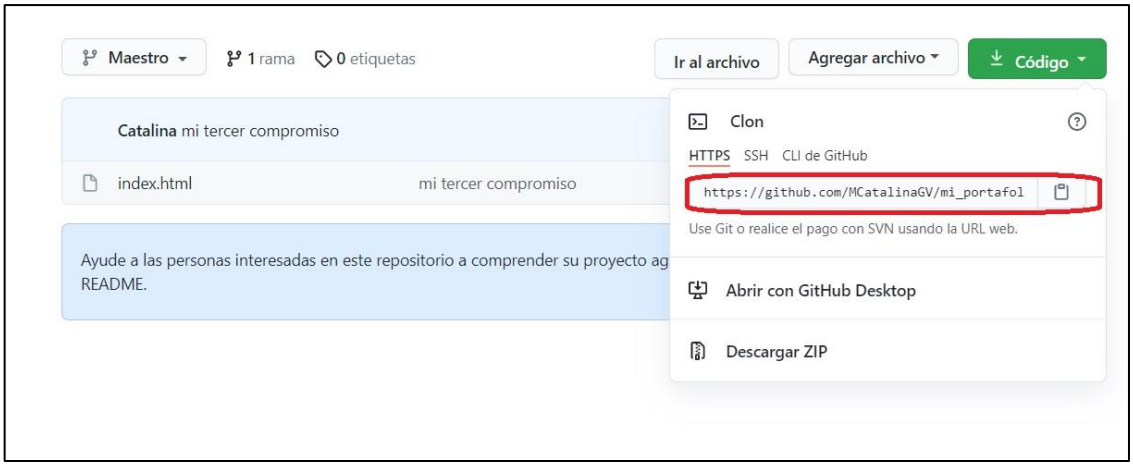
Si actualizamos la página podemos encontrar nuestro proyecto



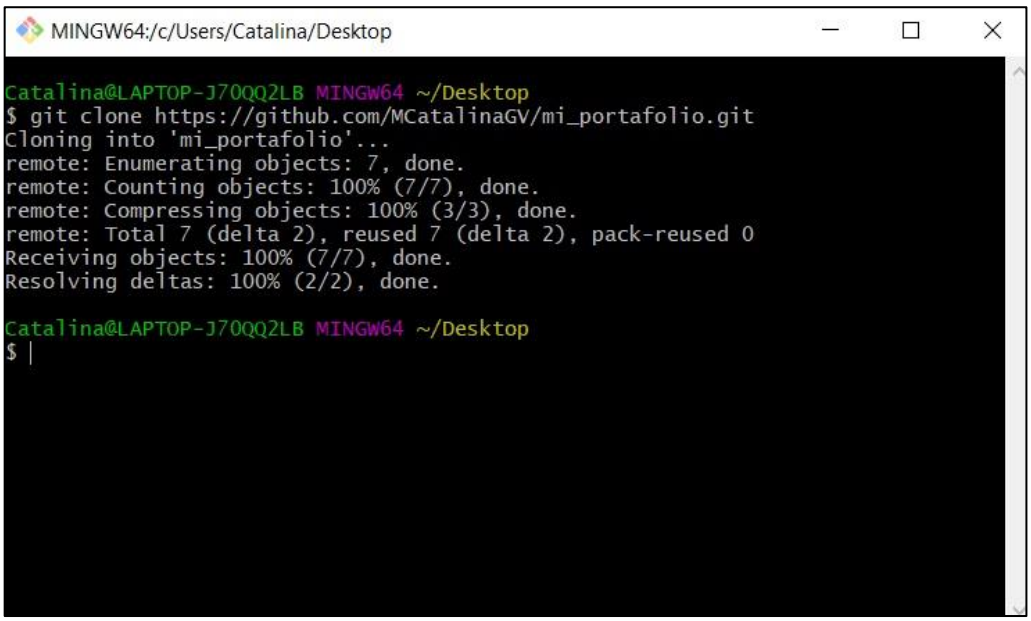
Podemos agregar un archivo para que las personas puedan saber qué intención tiene nuestro proyecto, esto lo hacemos agregando el archivo **readme**. Es opcional, podemos colocarlo o no.

CLONAR EL REPOSITORIO

Eliminamos el proyecto que habíamos creado y en **GitHub** copiaremos la **URL** que aparece al hacer clic en el botón código.

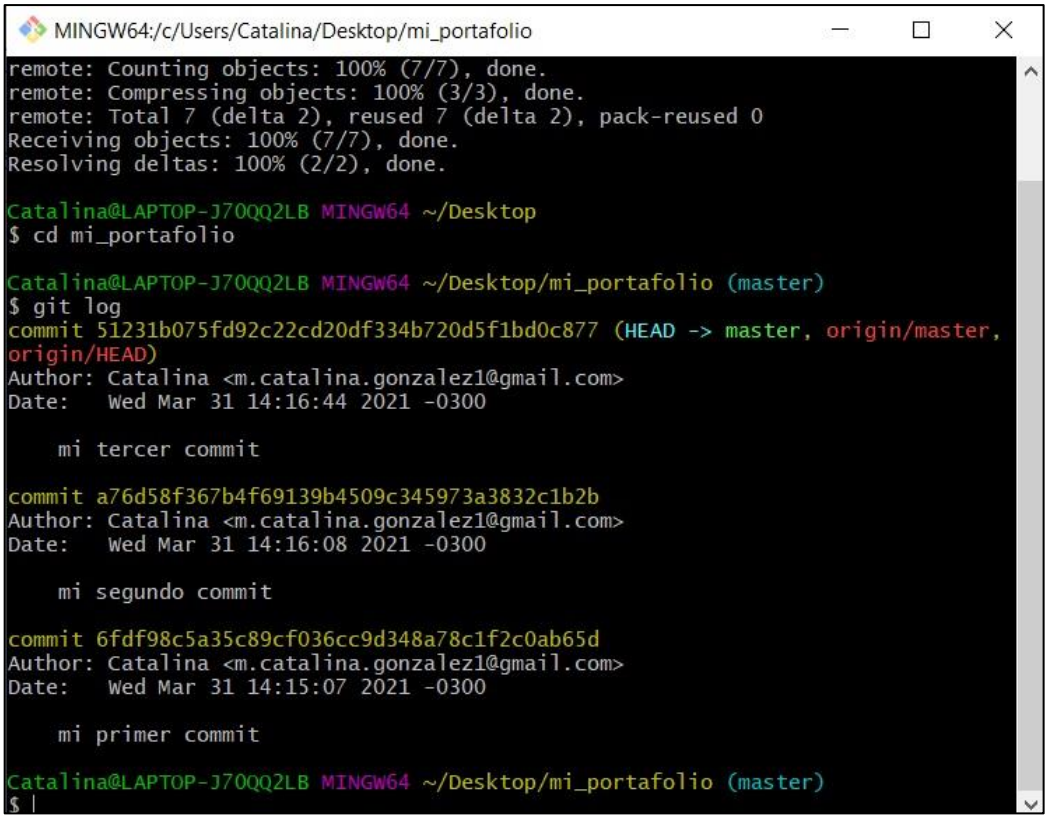


Abrimos nuevamente **Git Bash**, para este ejemplo fue abierto en el escritorio, pero puede abrirse en el lugar donde deseemos guardar el repositorio que clonaremos o navegar a través de los directorios, con el comando `cd` para posicionarnos donde gustemos. Posterior a eso escribiremos el comando `git clone` y pegaremos la **URL** que copiamos en **GitHub**.



De esta forma hemos clonado el repositorio y ahora podemos trabajarlo en nuestro ambiente local sin modificar el archivo que tenemos en el repositorio remoto.

Podemos corroborarlo ingresando al proyecto y haciendo un `git log` que nos muestra los **commit** del proyecto.

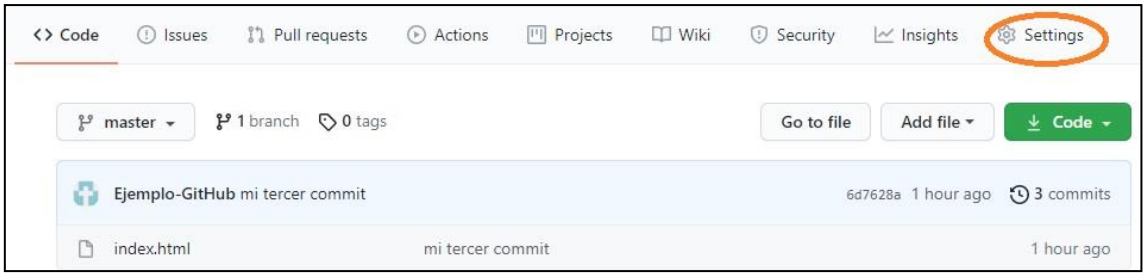


EXERCISE 2: TRABAJO COLABORATIVO CON GITHUB

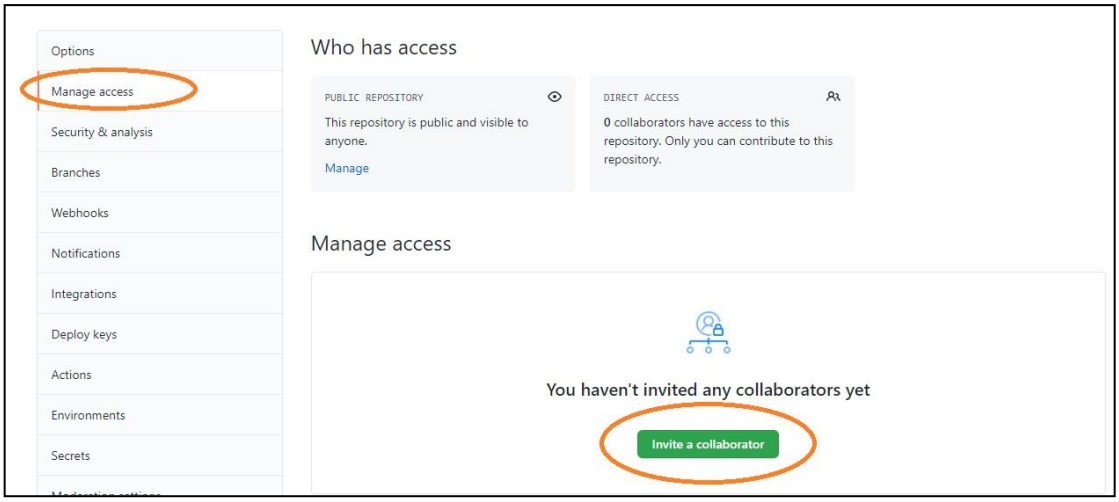
Ya aprendimos a subir nuestros proyectos a un repositorio remoto como es **GitHub**, pero mucho hemos hablado de como **GitHub** nos permite llevas a cabo trabajo colaborativo, por lo que continuaremos viendo cómo podemos utilizar esta función.

Para realizar esto, debemos tener un segundo usuario con el cual interactuar en **GitHub**. Hemos creado el usuario **colaboradorEjemplo** con el cual realizaremos una colaboración con el repositorio que hemos estado trabajando.

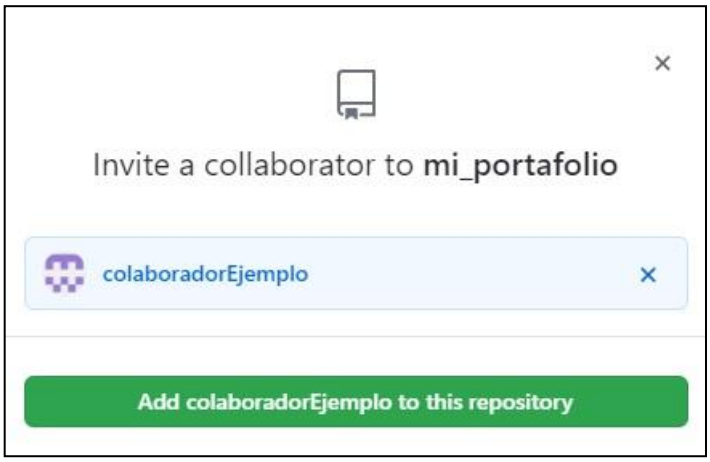
Para hacer eso nos dirigiremos a ajustes



Seleccionaremos la opción **Gestionar** los accesos y posteriormente invitar a colaborador.

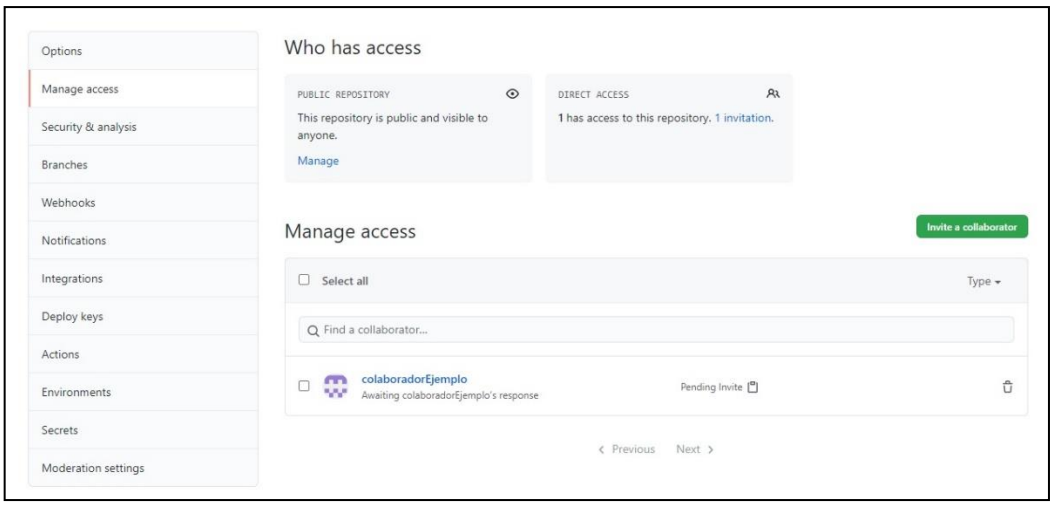


En la ventana que se abre, debemos colocar el nombre con el que el usuario se registró en **GitHub**.



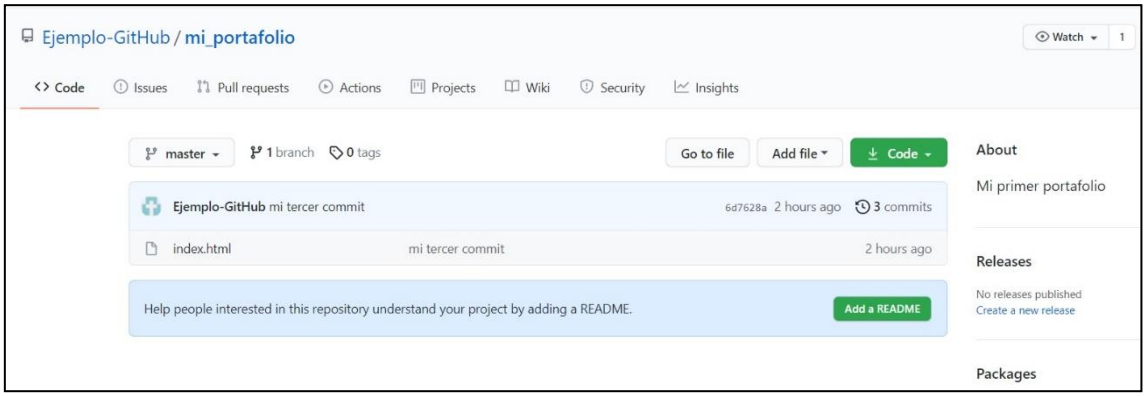
Finalmente debemos invitar al colaborador a trabajar con nosotros en nuestro portafolio especifico.

Cuando ya tenemos lista la invitación al colaborador debemos esperar que la apruebe.

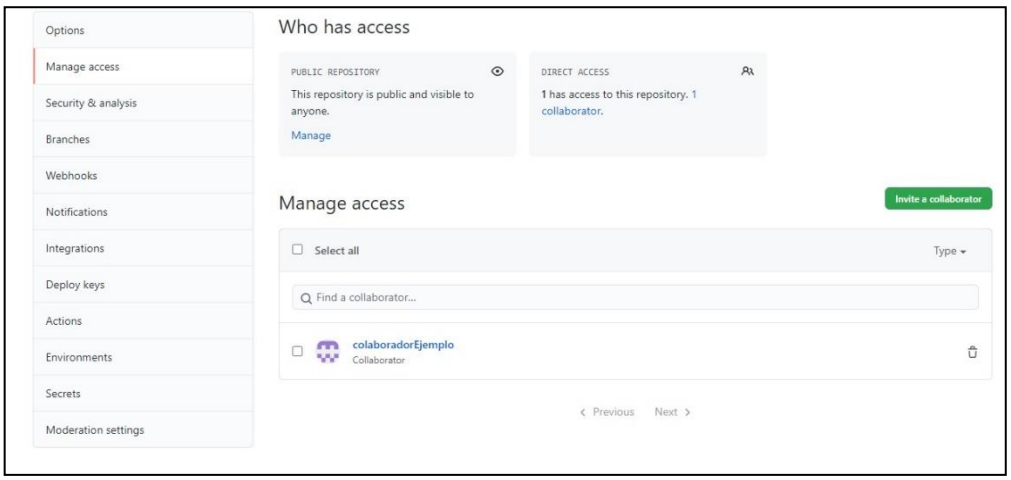


El colaborador invitado recibirá un correo con la invitación y el enlace para aceptarla.

Una vez que acepta la invitación tendrá acceso al repositorio compartido.



El dueño del repositorio podrá ver que el colaborador ahora está incluido como miembro colaborador y puede realizar cambios en el proyecto.

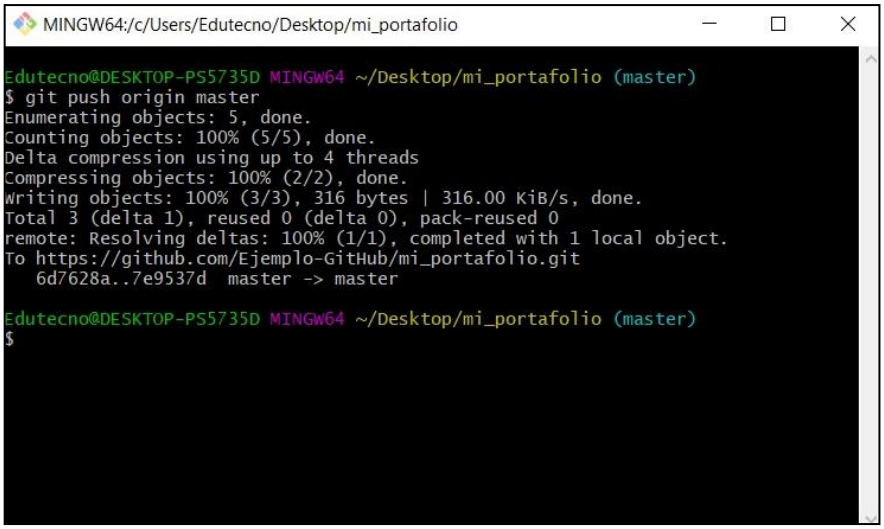


Ahora el colaborador puede clonar el repositorio y trabajar en él.

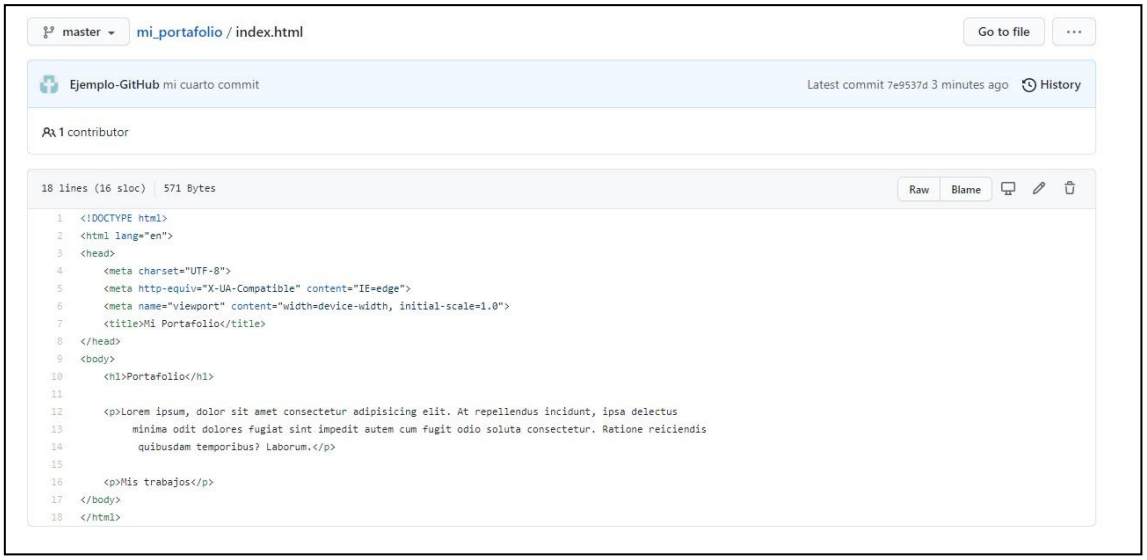
Vamos a realizar un cambio en nuestro proyecto

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Mi Portafolio</title>
</head>
<body>
    <h1>Portafolio</h1>

    <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. At
repellendus incidunt, ipsa delectus
         minima odit dolores fugiat sint impedit autem cum fugit odio
soluta consectetur. Ratione reiciendis
          quibusdam temporibus? Laborum.</p>

    <p>Mis trabajos</p>
</body>
</html>
```

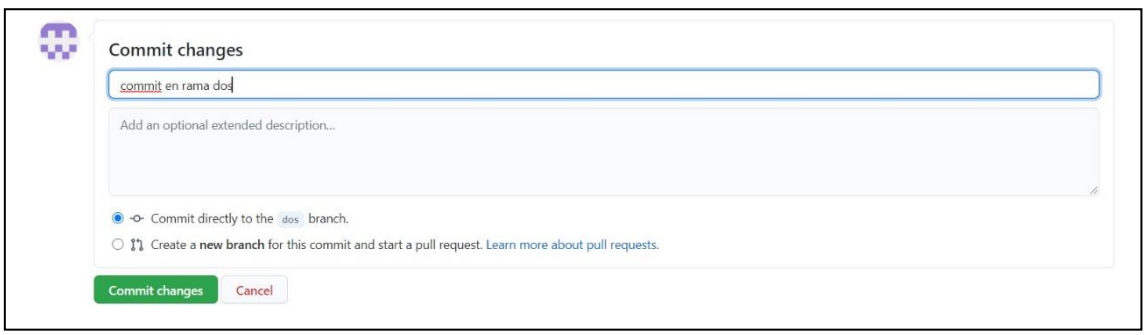
Haremos un cuarto **commit** y usando el comando `git push origin master` subimos los cambios a la rama principal.



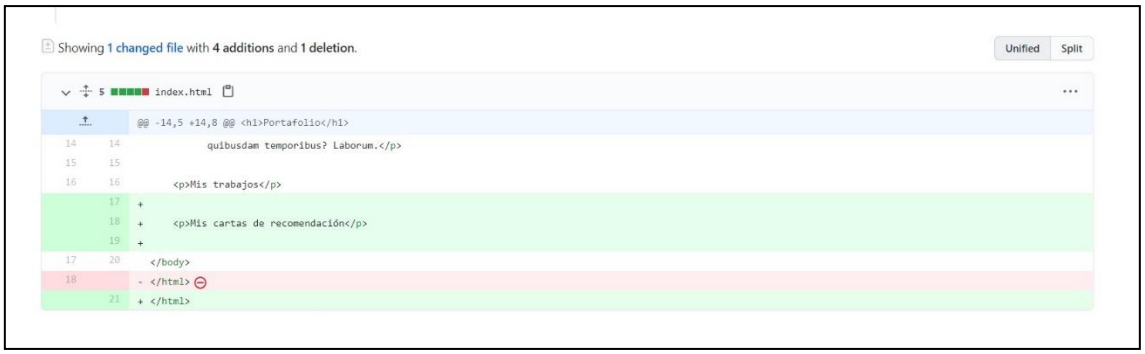
Si vemos nuestro proyecto podemos ver como el cambio de ha realizado



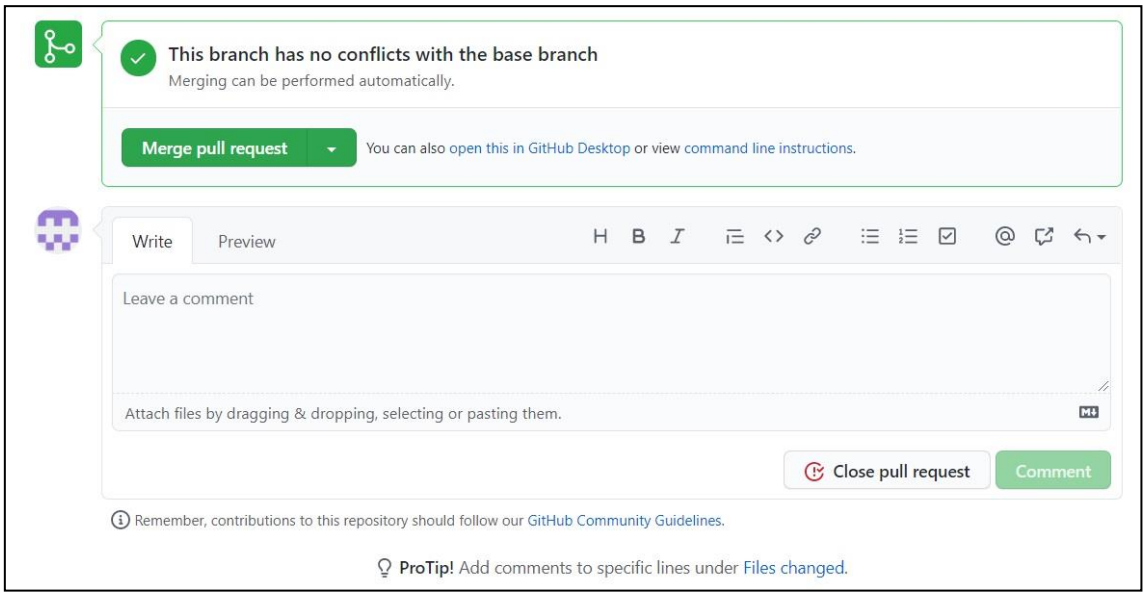
Supongamos ahora que el colaborador realiza una segunda rama para poder hacer cambios en ella sin afectar a la rama master y realiza un commit.



Seleccionando comparar podremos comparar las diferencias entre la rama master y la rama dos.

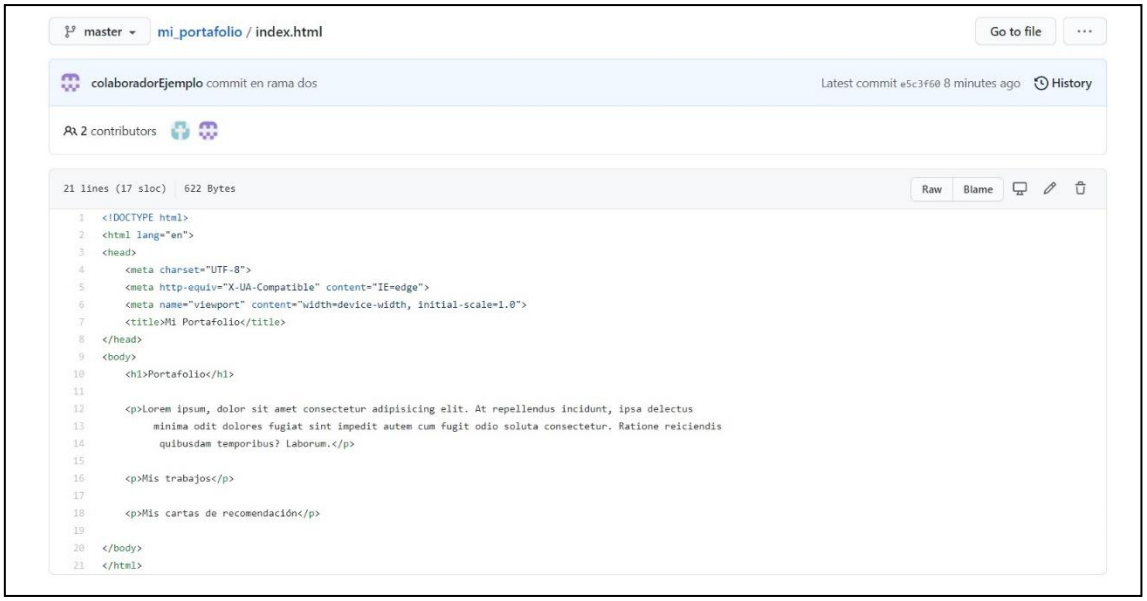


Podemos realizar la fusión de ambas ramas.



Y confirmamos el merge pull request o combinación.

Si nos dirigimos ahora a la rama master, podemos encontrar los cambios realizados y fusionados.



El colaborador original puede revisar los cambios realizados en la rama master y puede trabajar también en la rama dos.

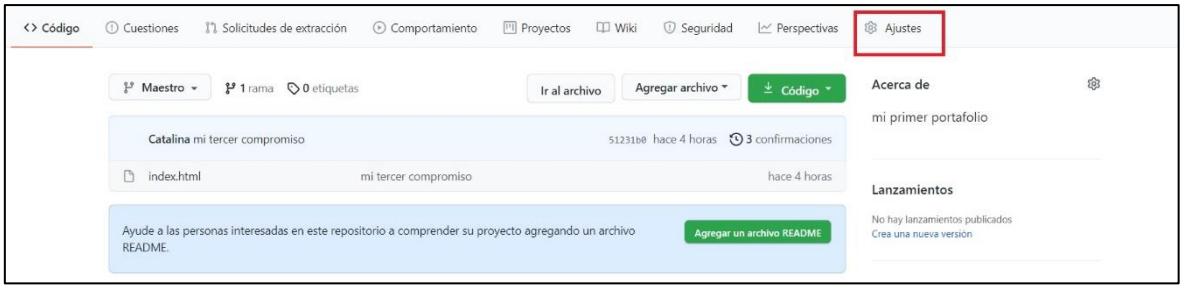
EXERCISE 3: USANDO GITHUB PAGES COMO HOSTING GRATUITO

Hemos dado los primeros pasos para guardar proyectos en un repositorio remoto como es **GitHub**, pero cuando tengamos listo el portafolio ¿Qué haremos para que sea visible a los reclutadores y puedan conocer nuestros trabajos y códigos?

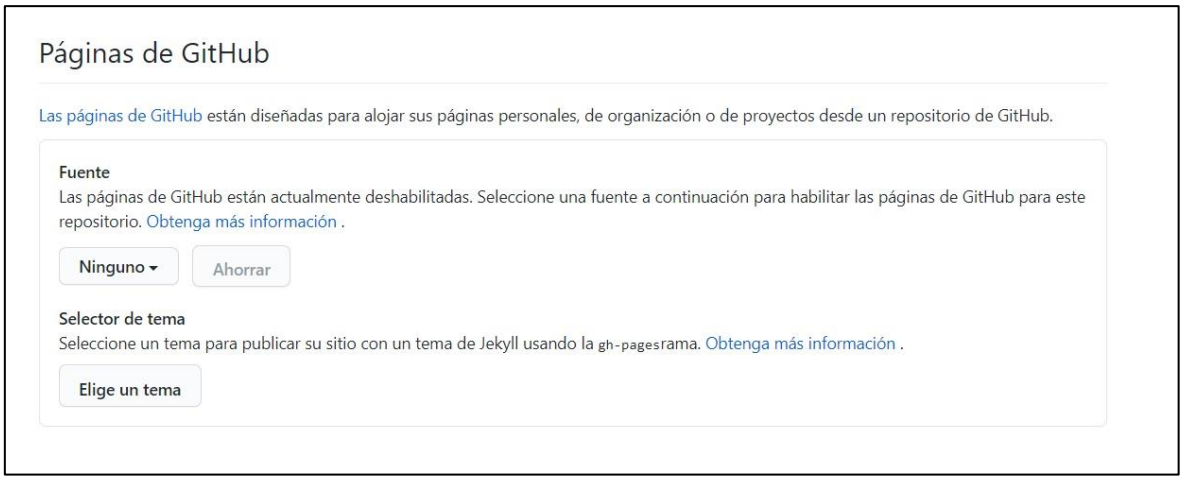
Para hacer que sea visible utilizaremos **GitHub Page**. Retomaremos nuestro ejercicio “mi portafolio” dirigiéndonos a nuestro usuario en **GitHub** y a nuestros repositorios.

Nos abrirá enseguida nuestro repositorio en el cual podemos ver los documentos que hemos guardados, en este caso, el archivo **index.html**.

Nos dirigiremos a ajustes.



Al ingresar ahí encontraremos las configuraciones de nuestro repositorio. Nos detendremos en la sección **GitHub Pages**.

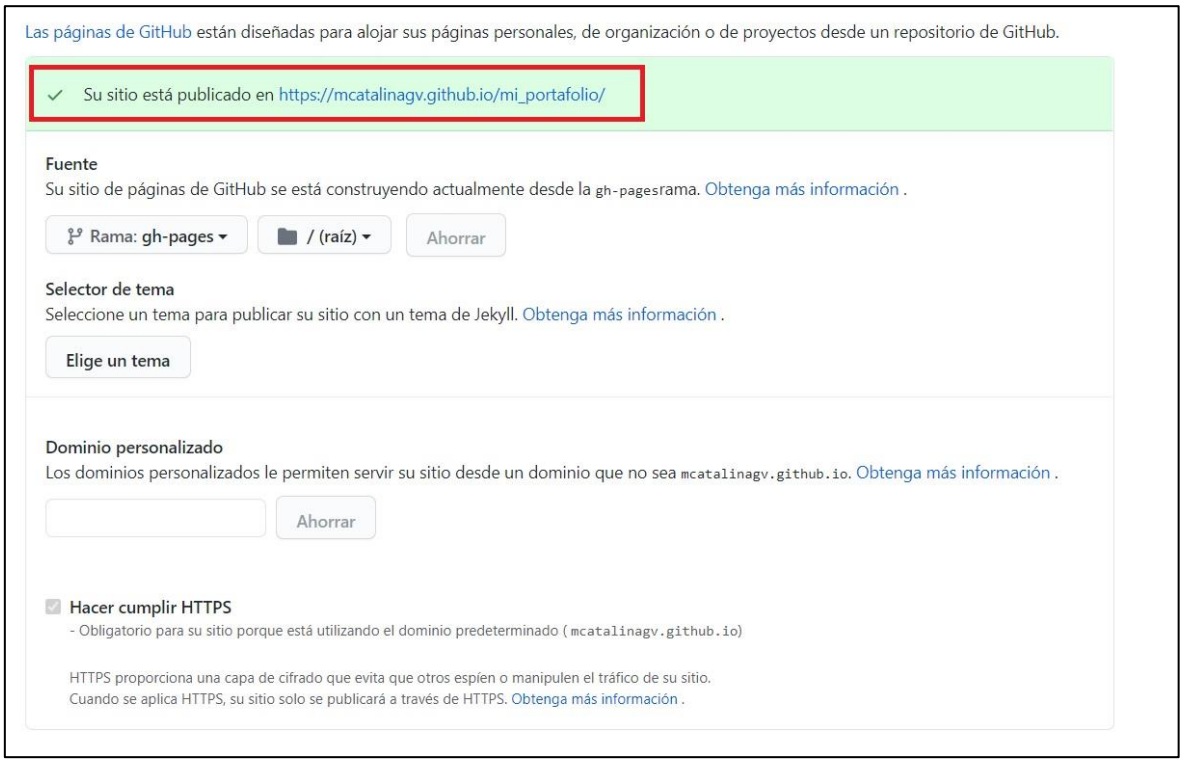


Acá nos da la opción de activar o desactivar **GitHub Pages**.

Para activarlo iremos a nuestro código, nos dirigiremos a la opción master y crearemos una nueva rama. La llamaremos **gh-pages**. Este nombre es necesario porque **GitHub** lo buscará con él y desde ahí tomará el código.

Ahora tenemos dos ramas con exactamente el mismo código.

Ahora que hemos creado esta nueva rama, nos dirigiremos nuevamente a ajustes, hasta la sección **GitHub Pages** y podemos observar que nos aparece una **URL** y nos dice que nuestro sitio web esta publicado en esa dirección.



Si hacemos clic en la **URL** ingresamos a una página web que se encuentra disponible para ser visitada por cualquier persona, ya que la hemos levantado en el hosting gratuito **GitHub Pages**.

Podemos conocer mucho más sobre GitHub Pages en su página web https://pages.github.com/