

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:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <meta name="viewport" content="width=device-width, initial-
7 scale=1.0">
8     <title>Mi Portafolio</title>
9 </head>
10 <body>
11     <h1>Portafolio</h1>
12
13     <p>Lorem ipsum dolor sit amet, consectetur adipisicing elit.
14 Consequatur eius rem
15         aut praesentium, maiores illo placeat cumque aperiam cum
16 odit dignissimos vel
17         quisquam, hic tenetur illum ex. Laudantium, ducimus
18 fuga.</p>
19 </body>
20 </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.

```

MINGW64:/c/Users/Catalina/Desktop/Mi Portafolio
Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/Mi Portafolio (master)
$ git log
commit d4571d294966c45b9a44853ae8051de411cafa81 (HEAD -> master)
Author: Catalina <m.catalina.gonzalez1@gmail.com>
Date:   Wed Mar 31 09:52:54 2021 -0300

    mi tercer commit

commit 69a832af0919c037016f4f78fd6bf67b36767c6c (tres, dos)
Author: Catalina <m.catalina.gonzalez1@gmail.com>
Date:   Tue Mar 30 19:14:28 2021 -0300

    mi segundo commit

commit 4aa6635fa278b613a904bede167dcc7a6a5f12c8
Author: Catalina <m.catalina.gonzalez1@gmail.com>
Date:   Tue Mar 30 17:29:58 2021 -0300

    mi primer commit

Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/Mi Portafolio (master)
$ |
  
```

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.

Crea tu cuenta

Nombre de usuario *

Dirección de correo electrónico *

Contraseña *

Asegúrese de que tenga **al menos 15 caracteres** O **al menos 8 caracteres, incluido un número y una letra minúscula**. [Obtenga más información](#).

Preferencias de correo electrónico

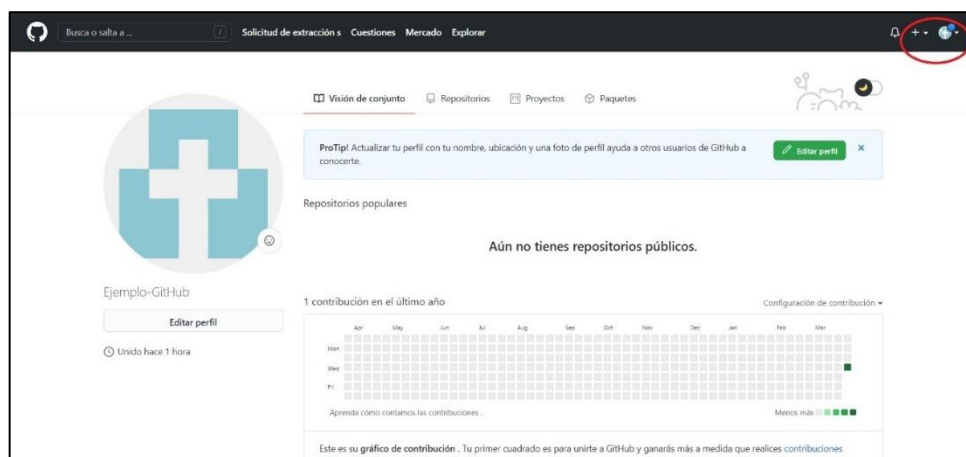
☒ Envíeme actualizaciones de productos, anuncios y ofertas ocasionales.

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

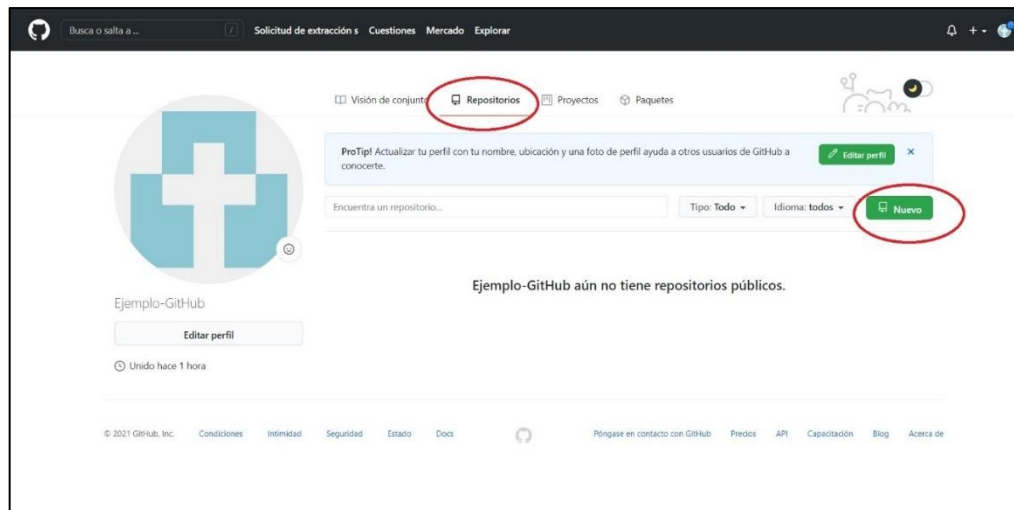
Nos pregunta también nuestra experiencia, para que usaremos **GitHub** y finalmente nos exige verificar nuestro correo electrónico.



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**.

Dueño * Ejemplo-GitHub

Nombre del repositorio * mi_portafolio ✓

Los grandes nombres de repositorios son breves y fáciles de recordar. ¿Necesitas inspiración? ¿Qué hay de las gafas para escalar?

Descripción (opcional)
Mi primer portafolio

☒ **Público**
Cualquiera en Internet puede ver este repositorio. Tú eliges quién puede comprometerse.

☐ **Privado**
Tú eliges quién puede ver y comprometerse con este repositorio.

Inicialice este repositorio con:
Omita este paso si está importando un repositorio existente.

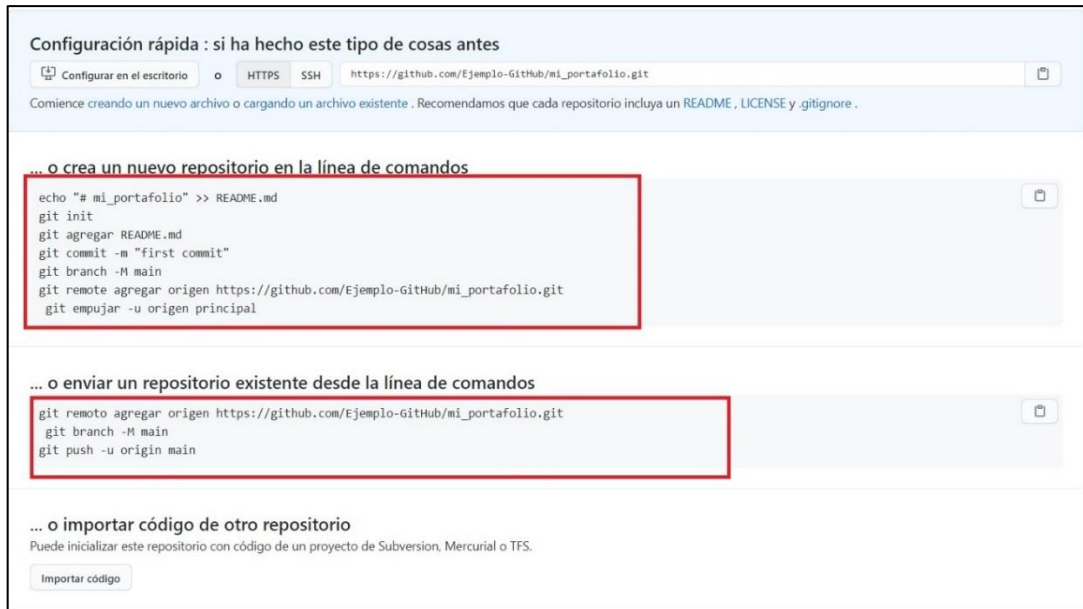
☐ **Agregar un archivo README**
Aquí es donde puede escribir una descripción larga de su proyecto. [Aprende más.](#)

☐ **Agregar .gitignore**
Elija qué archivos no rastrear de una lista de plantillas. [Aprende más.](#)

☐ **Elija una licencia**
Una licencia les dice a otros lo que pueden y no pueden hacer con su código. [Aprende más.](#)

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 origen 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.

```

MINGW64:/c/Users/Catalina/Desktop/Mi_Portafolio

Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/Mi_Portafolio (master)
$ git remote add origin https://github.com/MCatalinaGV/mi_portafolio.git

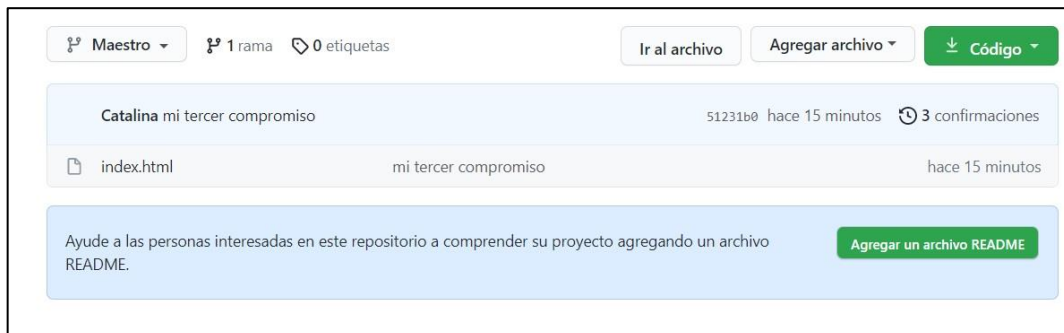
Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/Mi_Portafolio (master)
$ git status
On branch master
nothing to commit, working tree clean

Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/Mi_Portafolio (master)
$ git push -u origin master
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 881 bytes | 440.00 KiB/s, done.
Total 7 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/MCatalinaGV/mi_portafolio.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.

Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/Mi_Portafolio (master)
$ |

```

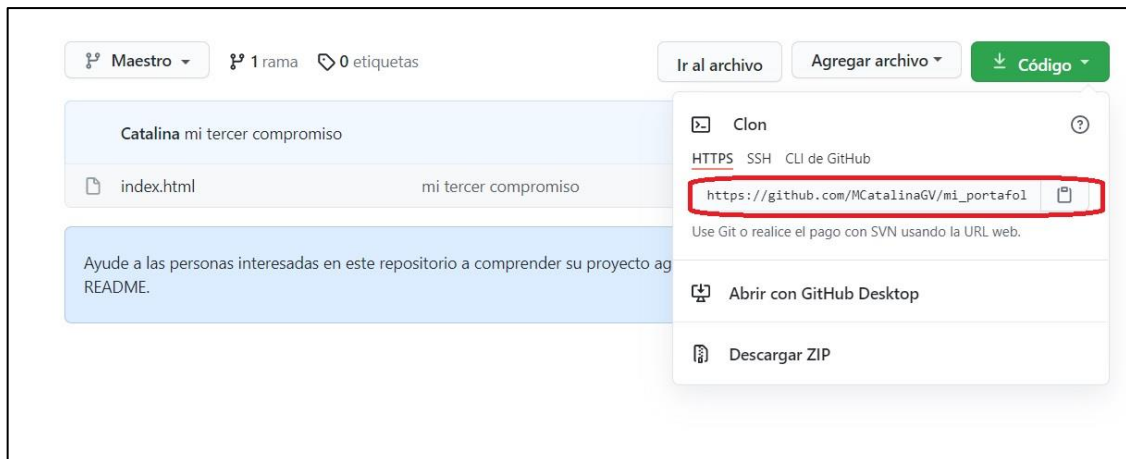
Si actualizamos la página podemos encontrar nuestro proyecto



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**.

```
MINGW64:/c/Users/Catalina/Desktop
Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop
$ git clone https://github.com/MCatalinaGV/mi_portafolio.git
Cloning into 'mi_portafolio'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 7 (delta 2), reused 7 (delta 2), pack-reused 0
Receiving objects: 100% (7/7), done.
Resolving deltas: 100% (2/2), done.

Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop
$ |
```

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.

```
MINGW64:/c/Users/Catalina/Desktop/mi_portafolio
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 7 (delta 2), reused 7 (delta 2), pack-reused 0
Receiving objects: 100% (7/7), done.
Resolving deltas: 100% (2/2), done.

Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop
$ cd mi_portafolio

Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/mi_portafolio (master)
$ git log
commit 51231b075fd92c22cd20df334b720d5f1bd0c877 (HEAD -> master, origin/master,
origin/HEAD)
Author: Catalina <m.catalina.gonzalez1@gmail.com>
Date: Wed Mar 31 14:16:44 2021 -0300

    mi tercer commit

commit a76d58f367b4f69139b4509c345973a3832c1b2b
Author: Catalina <m.catalina.gonzalez1@gmail.com>
Date: Wed Mar 31 14:16:08 2021 -0300

    mi segundo commit

commit 6fdf98c5a35c89cf036cc9d348a78c1f2c0ab65d
Author: Catalina <m.catalina.gonzalez1@gmail.com>
Date: Wed Mar 31 14:15:07 2021 -0300

    mi primer commit

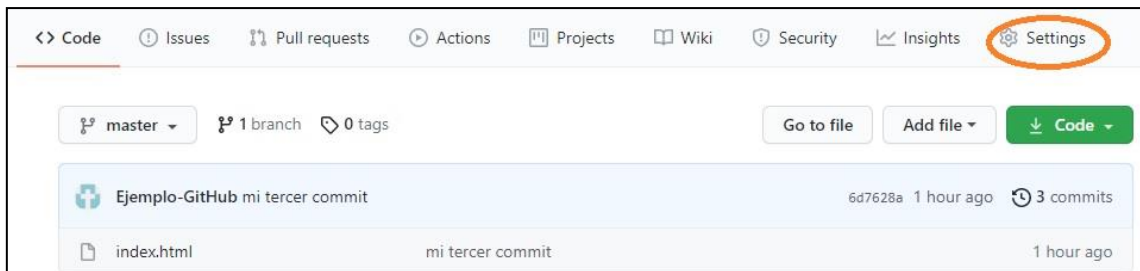
Catalina@LAPTOP-J70QQ2LB MINGW64 ~/Desktop/mi_portafolio (master)
$ |
```

EXERCISE 2: TRABAJO COLABORATIVO CON GITHUB

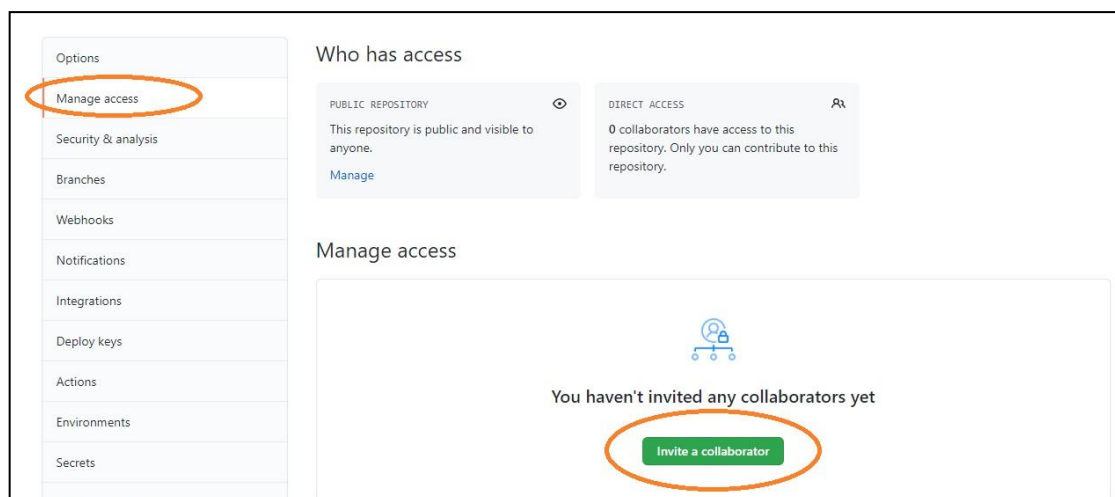
Ya aprendimos a subir nuestros proyectos a un repositorio remoto como es **GitHub**, pero mucho hemos hablado de como **GitHub** nos permite llevar a cabo trabajo colaborativo, por lo que continuaremos viendo cómo podemos utilizar esta función.

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.

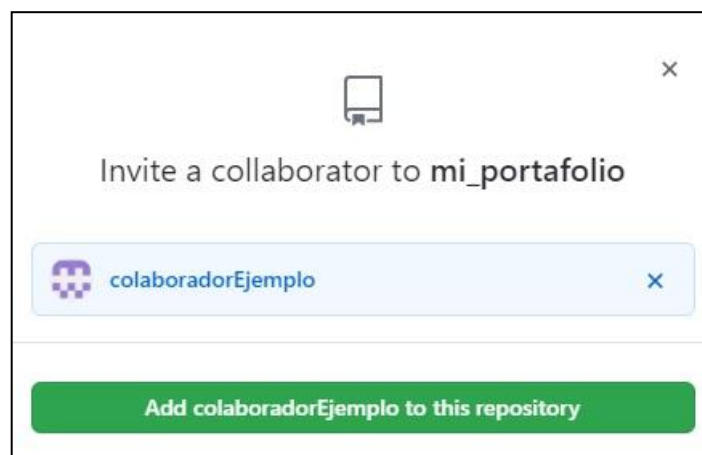
Para hacer eso nos dirigiremos a ajustes



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

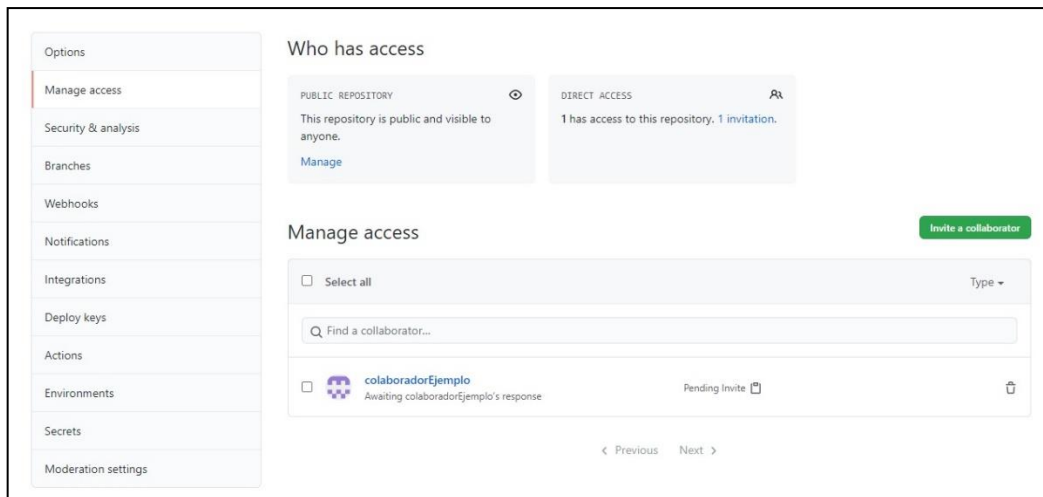


En la ventana que se abre, debemos colocar el nombre con el que el usuario se registró en **GitHub**.



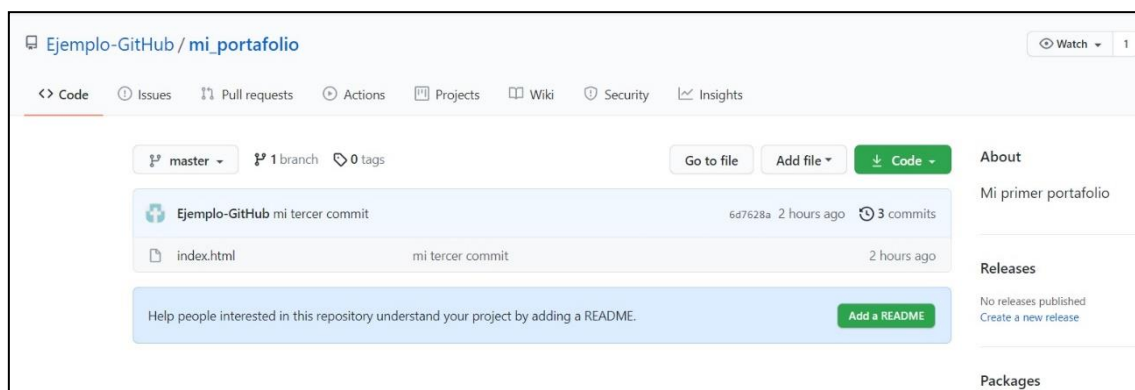
Finalmente debemos invitar al colaborador a trabajar con nosotros en nuestro portafolio específico.

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

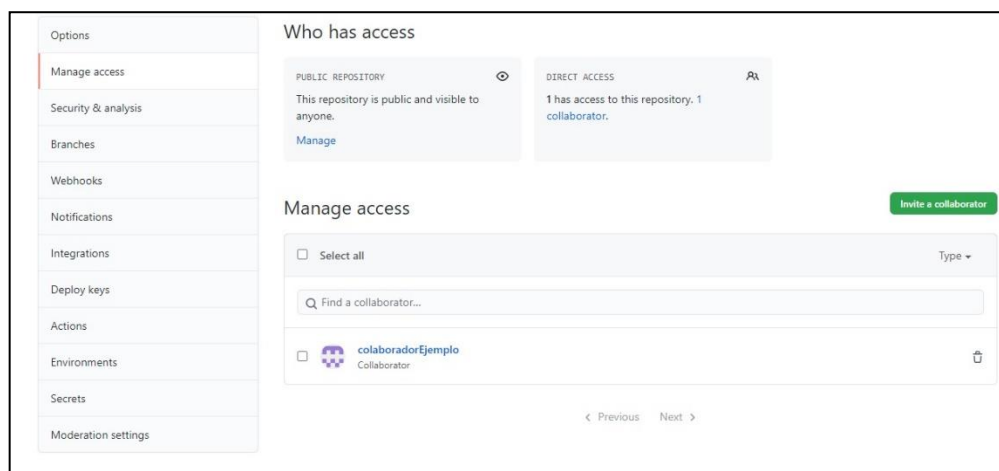


El colaborador invitado recibirá un correo con la invitación y el enlace para aceptarla.

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

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-
7 scale=1.0">
8   <title>Mi Portafolio</title>
9 </head>
10 <body>
11   <h1>Portafolio</h1>
12
13   <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. At
14 repellendus incidunt, ipsa delectus
15     minima odit dolores fugiat sint impedit autem cum fugit odio
16 soluta consectetur. Ratione reiciendis
17     quibusdam temporibus? Laborum.</p>
18
19   <p>Mis trabajos</p>
20 </body>
21 </html>
22
```

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

```

MINGW64:/c/Users/Edutecno/Desktop/mi_portafolio

Edutecno@DESKTOP-PS5735D MINGW64 ~/Desktop/mi_portafolio (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 316.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Ejemplo-GitHub/mi_portafolio.git
   6d7628a..7e9537d  master -> master

Edutecno@DESKTOP-PS5735D MINGW64 ~/Desktop/mi_portafolio (master)
$
  
```

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

master mi_portafolio / index.html

Ejemplo-GitHub mi cuarto commit Latest commit 7e9537d 3 minutes ago History

1 contributor

18 lines (16 sloc) 571 Bytes

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Mi Portafolio</title>
8 </head>
9 <body>
10  <h1>Portafolio</h1>
11
12  <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. At repellendus incidunt, ipsa delectus
13  minima odit dolores fugiat sint impedit autem cum fugit odio soluta consectetur. Ratione reiciendis
14  quibusdam temporibus? Laborum.</p>
15
16  <p>Mis trabajos</p>
17 </body>
18 </html>
  
```

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.

Commit changes

commit en rama dos

Add an optional extended description...

☒ Commit directly to the dos branch.

☐ Create a new branch for this commit and start a pull request. Learn more about pull requests.

Commit changes Cancel

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

```

Showing 1 changed file with 4 additions and 1 deletion.

index.html
@@ -14,5 +14,8 @@ <h1>Portafolio</h1>
    quibusdam temporibus? Laborum.</p>
    <p>Mis trabajos</p>
+   <p>Mis cartas de recomendación</p>
+   <p>Mis cartas de recomendación</p>
  </body>
- </html>
+ </html>
  
```

Podemos realizar la fusión de ambas ramas.

✓ This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request You can also open this in [GitHub Desktop](#) or view [command line instructions](#).

Write Preview H B I ≡ <> 🔗 ≡ ≡ ☑ @ 🗨 ⏪

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.

Close pull request **Comment**

Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).

ProTip! Add comments to specific lines under [Files changed](#).

Y confirmamos el merge pull request o combinación.

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

The screenshot shows the GitHub interface for a repository named 'mi_portafolio'. The file 'index.html' is selected, showing its content in a code editor. The file has 21 lines, 17 slots, and 622 bytes. The content is an HTML document with a head section containing meta tags for charset, language, and viewport, and a body section with a heading 'Portafolio' and three paragraphs of placeholder text.

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Mi Portafolio</title>
8 </head>
9 <body>
10  <h1>Portafolio</h1>
11
12  <p>Lorem ipsum, dolor sit amet consectetur adipisicing elit. At repellendus incidunt, ipsa delectus
13    minima odit dolores fugiat sint impedit autem cum fugit odio soluta consectetur. Ratione reiciendis
14    quibusdam temporibus? Laborum.</p>
15
16  <p>Mis trabajos</p>
17
18  <p>Mis cartas de recomendación</p>
19
20 </body>
21 </html>

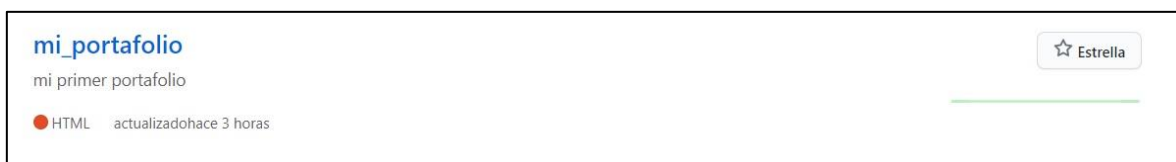
```

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

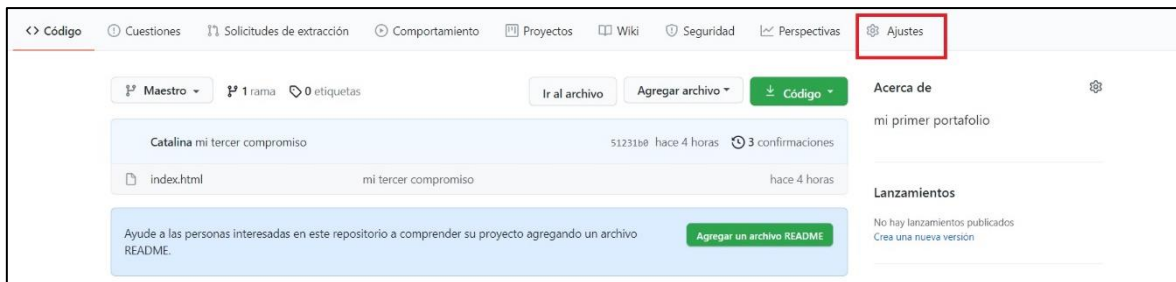
Hemos dado los primeros pasos para guardar proyectos en un repositorio remoto como es **GitHub**, pero cuando tengamos listo el portafolio ¿Qué haremos para que sea visible a los reclutadores y puedan conocer nuestros trabajos y códigos?

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

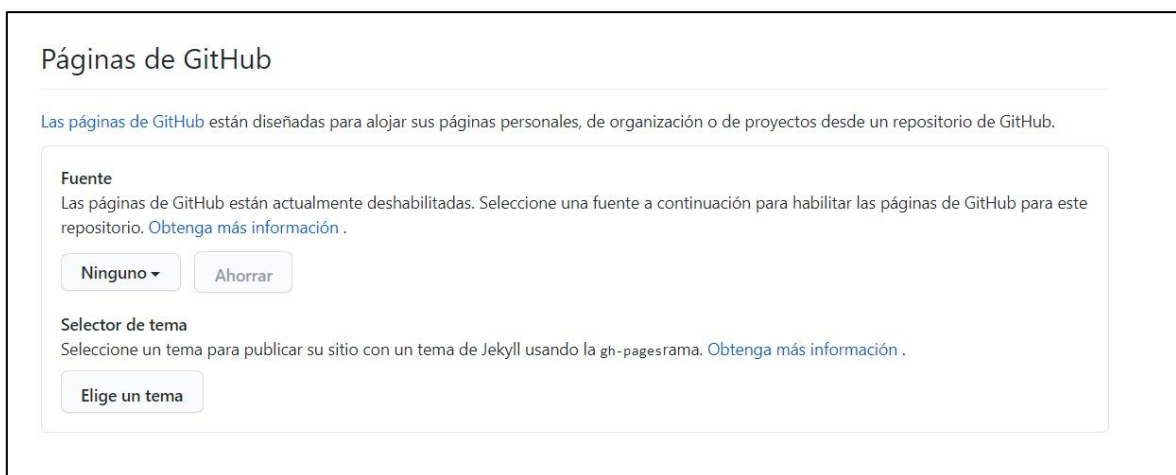


Nos abrirá enseguida nuestro repositorio en el cual podemos ver los documentos que hemos guardados, en este caso, el archivo **index.html**.

Nos dirigiremos a ajustes.

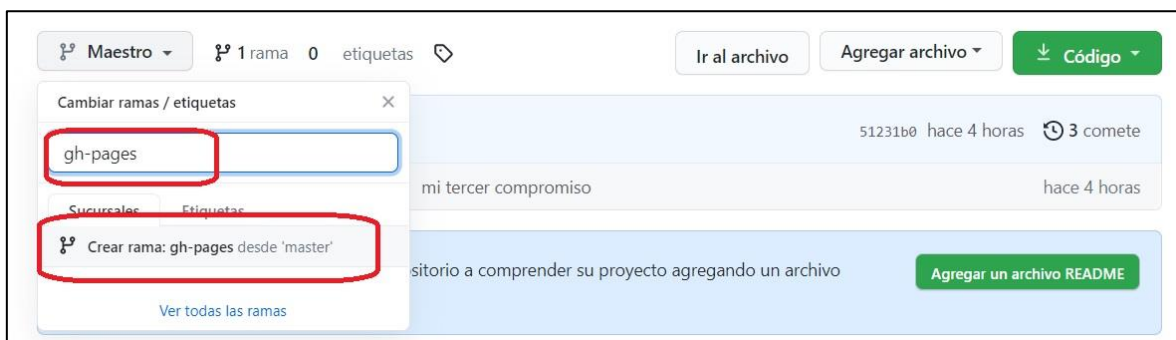


Al ingresar ahí encontraremos las configuraciones de nuestro repositorio. Nos detendremos en la sección **GitHub Pages**.

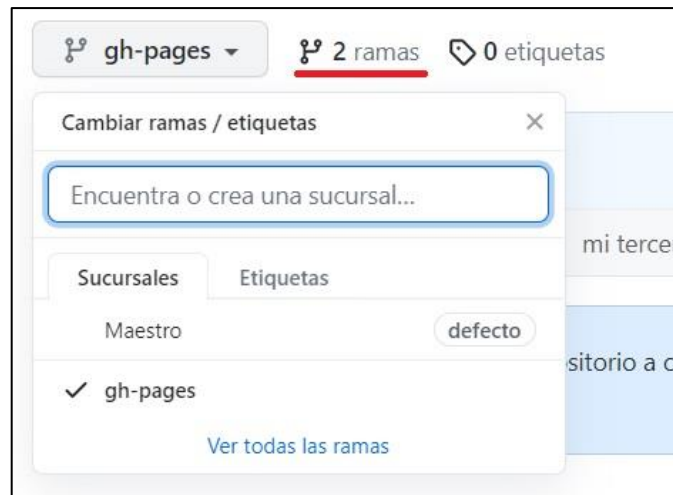


Acá nos da la opción de activar o desactivar **GitHub Pages**.

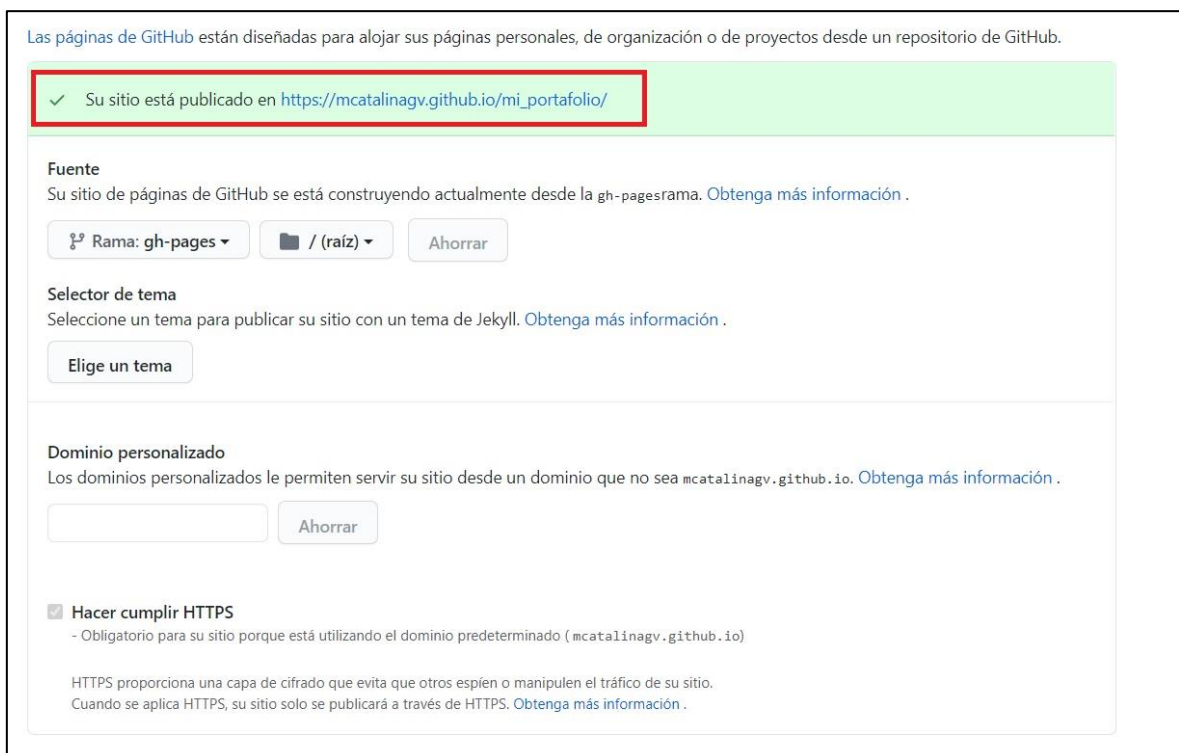
Para activarlo iremos a nuestro código, nos dirigiremos a la opción master y crearemos una nueva rama. La llamaremos **gh-pages**. Este nombre es necesario porque **GitHub** lo buscará con él y desde ahí tomará el código.



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/>