

‘Actividad Evaluable

Módulo	Servicios de red
Nombre y Apellidos:	DAVID RECIO RAMIREZ
Nombre y Apellidos:	DOMENICO ROSAS
Nombre y Apellidos:	MIGUEL ANGEL DIAZ
Estas soluciones deben ser entregadas en PDF.	

FECHA DE ENTREGA: --/--/21

Objetivos

El objetivo de esta actividad individual es la de repasar, asentar y adquirir un mayor conocimiento de lo impartido en clase.

El archivo debe nombrarse como: **nombre_apellido1_apellido2.PDF**

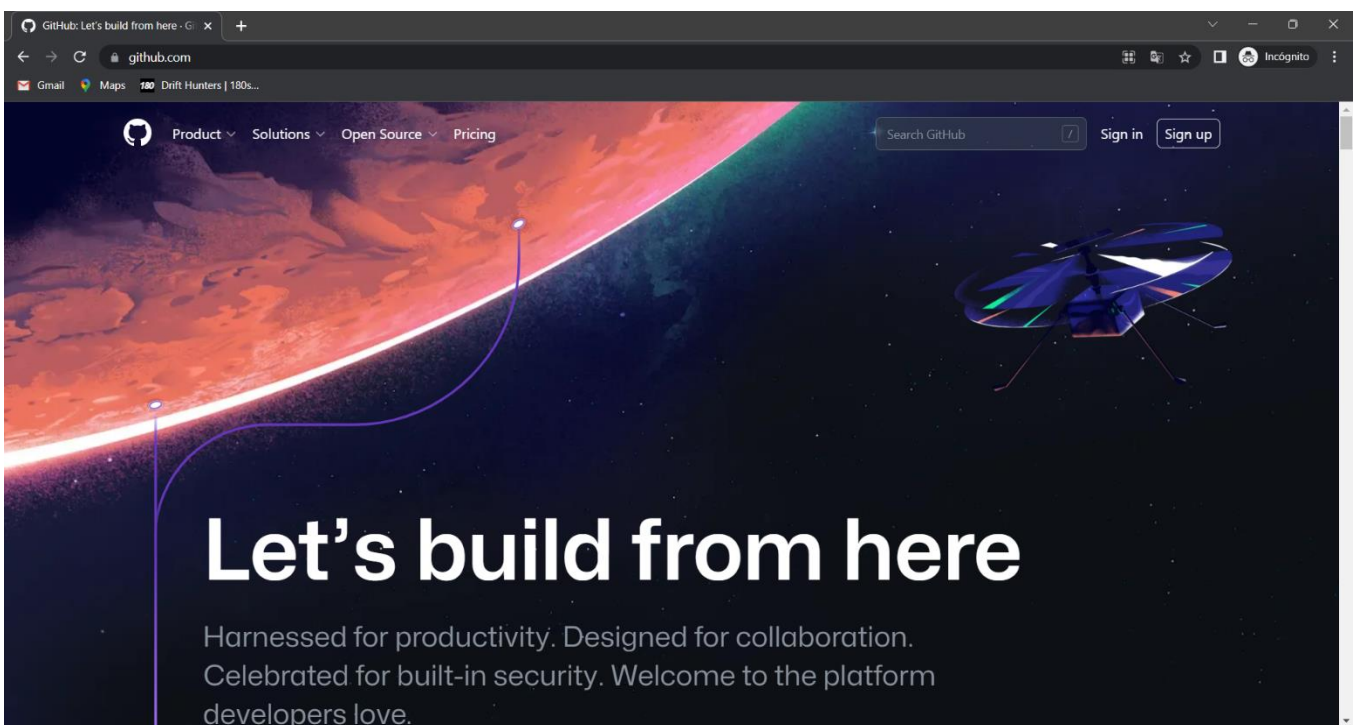
Ej: Belen_Vargas_Sabater.pdf

Cuestiones

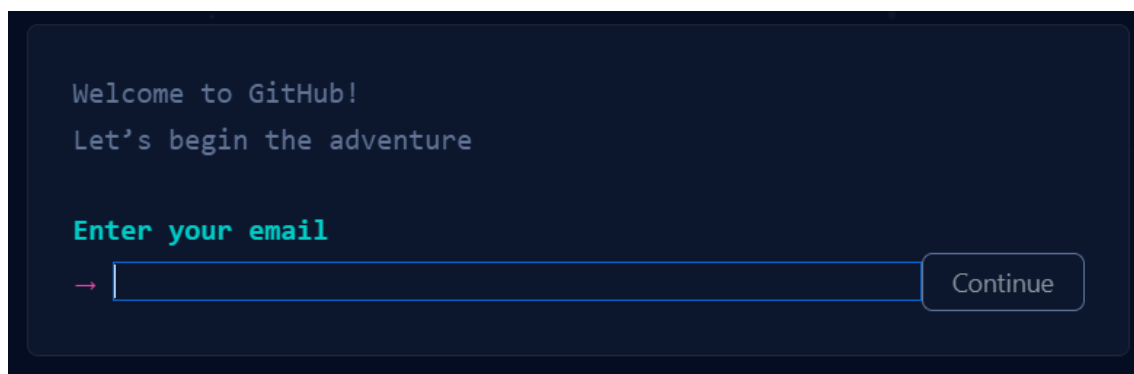
EJERCICIO 1: **Práctica:** Realizar una guía práctica a modo de tutorial, sobre el uso de GitHub, algunos de los elementos que debe cubrir son:

- Creación de repositorio y conexión con repositorio personal de GitHub mediante terminal con ejemplos.

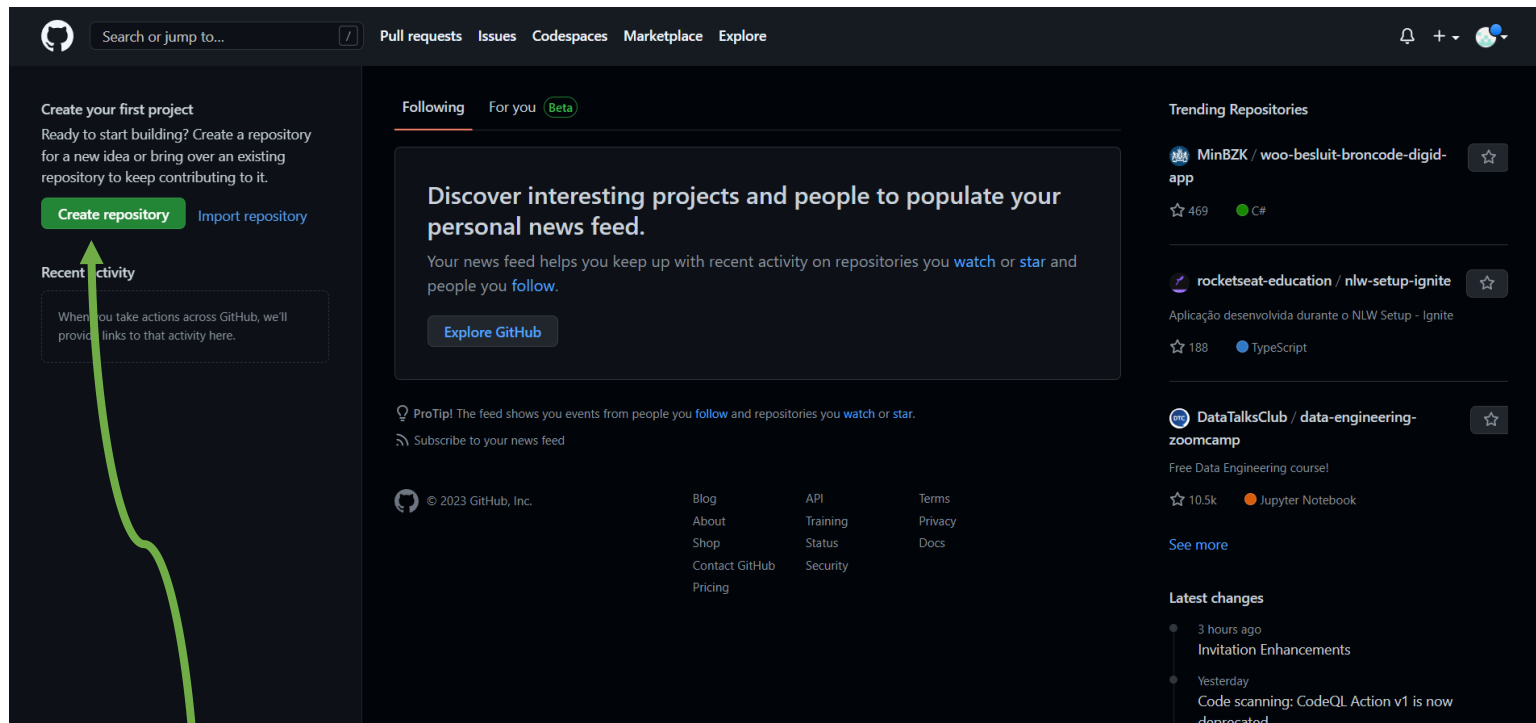
-Lo primero que tenemos que hacer para crear un repositorio es dirigirnos a la web de GitHub → <https://github.com>



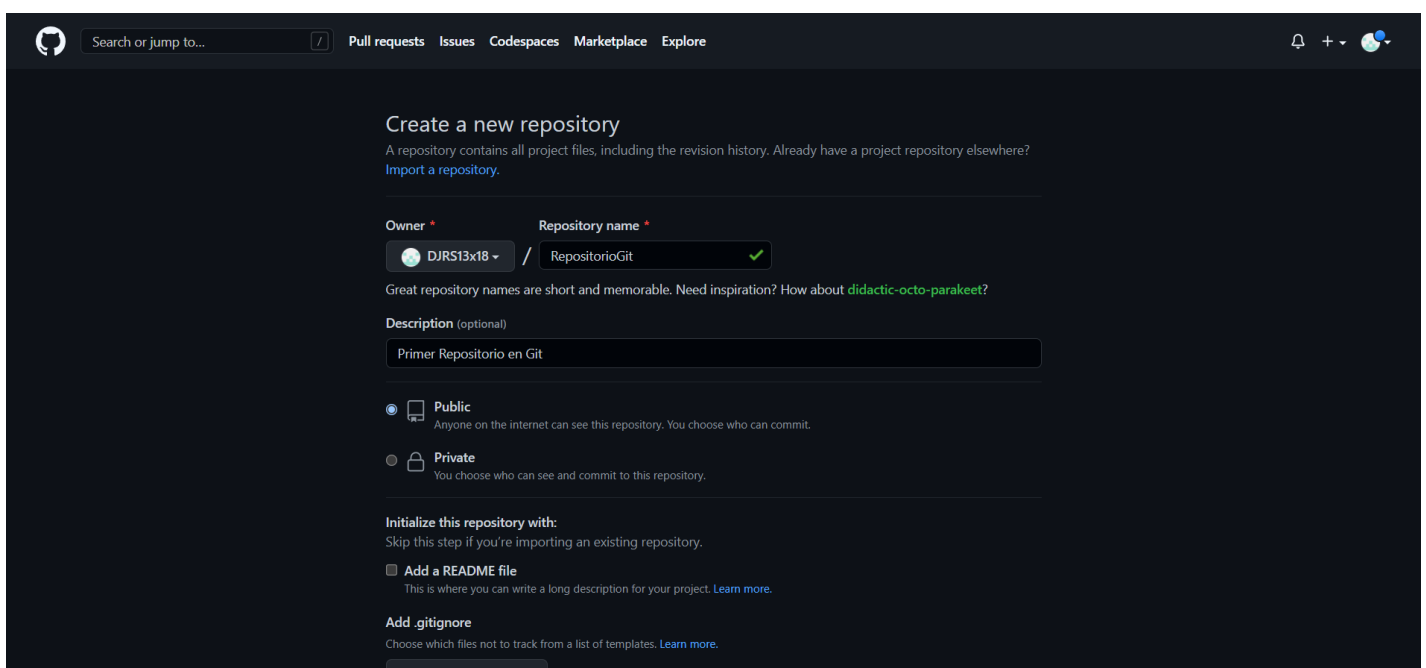
Una vez dentro de la página de GitHub para crear el repositorio necesitaremos una cuenta, para ello haremos click en el botón **Sign up**. Allí dentro nos encontraremos con una interfaz bastante intuitiva y llamativa. Nos pedirán algunos datos para poder gestionar el alta en la página.



Introducimos los datos que nos solicita la página, creamos la cuenta y la verificamos con el código que enviarán al email que se ha usado para crear la cuenta. Una vez que hayamos hecho todo esto la página nos dejará en este punto.



A partir de aquí para crear el repositorio solo tendremos que hacer click en **Create repository**. Nos redireccionará a una pestaña donde crearemos nuestro repositorio, tenemos que colocarle un nombre y una descripción, dependiendo de la finalidad para la que lo vayamos a utilizar.



Initialize this repository with:

Skip this step if you're importing an existing repository.

☒ **Add a README file**

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

Add .gitignore


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

.gitignore template: **None** ▾

Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

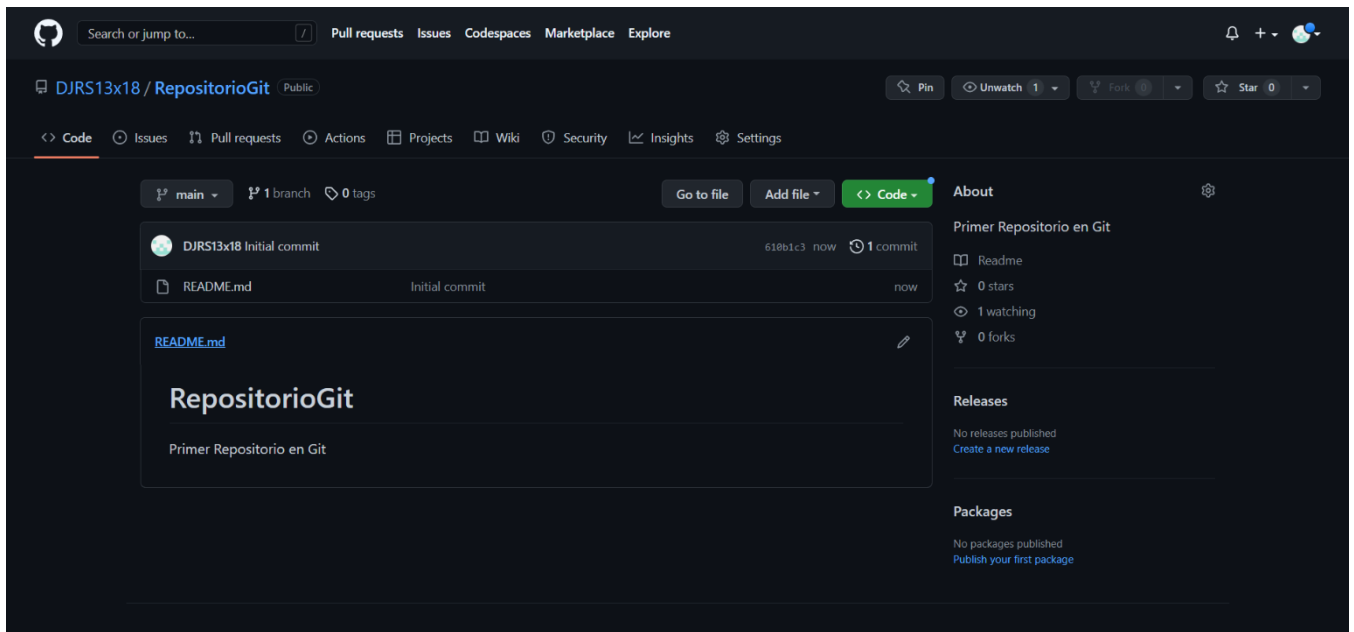
License: **None** ▾

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

Luego de añadir un nombre y una descripción a nuestro repositorio, hacemos click en [Add a README file](#) este archivo es para escribir una descripción del proyecto dentro del repositorio. y hacemos click en [Create repository](#).

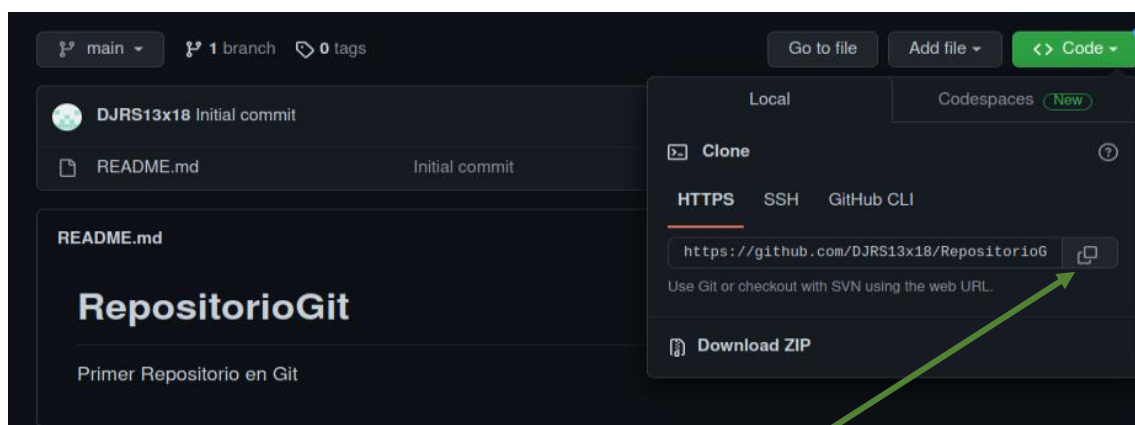


-Seremos redirigidos a nuestro repositorio. Lo siguiente que tenemos que hacer para conectar nuestro repositorio a nuestro terminal y poder subir los cambios desde allí será **clonar nuestro repositorio en el lugar donde querremos trabajar en nuestro proyecto**. En este caso en un portátil de trabajo con Linux como sistema operativo.

-Para ello necesitaremos el **link de nuestro repositorio**, que bien podemos copiarlo desde la URL de la página del mismo.

   <https://github.com/DJRS13x18/RepositorioGit>

La página de GitHub también nos da otra opción en la que tendríamos que hacer click en el botón **Code**.



Y hacer click en el botón que esta al final de la URL.

Una vez tengamos la URL de nuestro repositorio, crearemos una carpeta en la cual clonaremos el repositorio que es lo primero que necesitamos hacer para tener conexión con nuestro repositorio en GitHub y poder subir los cambios y se vean reflejados en la página de GitHub.

```
domenico@domenico-VirtualBox:~/Escritorio$ mkdir CarpetaRepositorioGit
```

Lo siguiente que haremos será entrar en la carpeta para clonar el repositorio dentro de la misma.

```
domenico@domenico-VirtualBox:~/Escritorio$ cd CarpetaRepositorioGit
```

Para clonar el repositorio dentro de la carpeta usaremos el comando git clone añadiendo la URL de nuestro repositorio.

```
$ git clone https://github.com/DJRS13x18/RepositorioGit.git
```

ejecutamos el comando y nos aparecerá lo siguiente. Indicando que nuestro repositorio ha sido clonado con éxito.

```
Clonando en 'RepositorioGit'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Recibiendo objetos: 100% (3/3), listo.
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepositorioGit$
```

El repositorio estaría ubicado dentro de la carpeta donde lo clonamos.

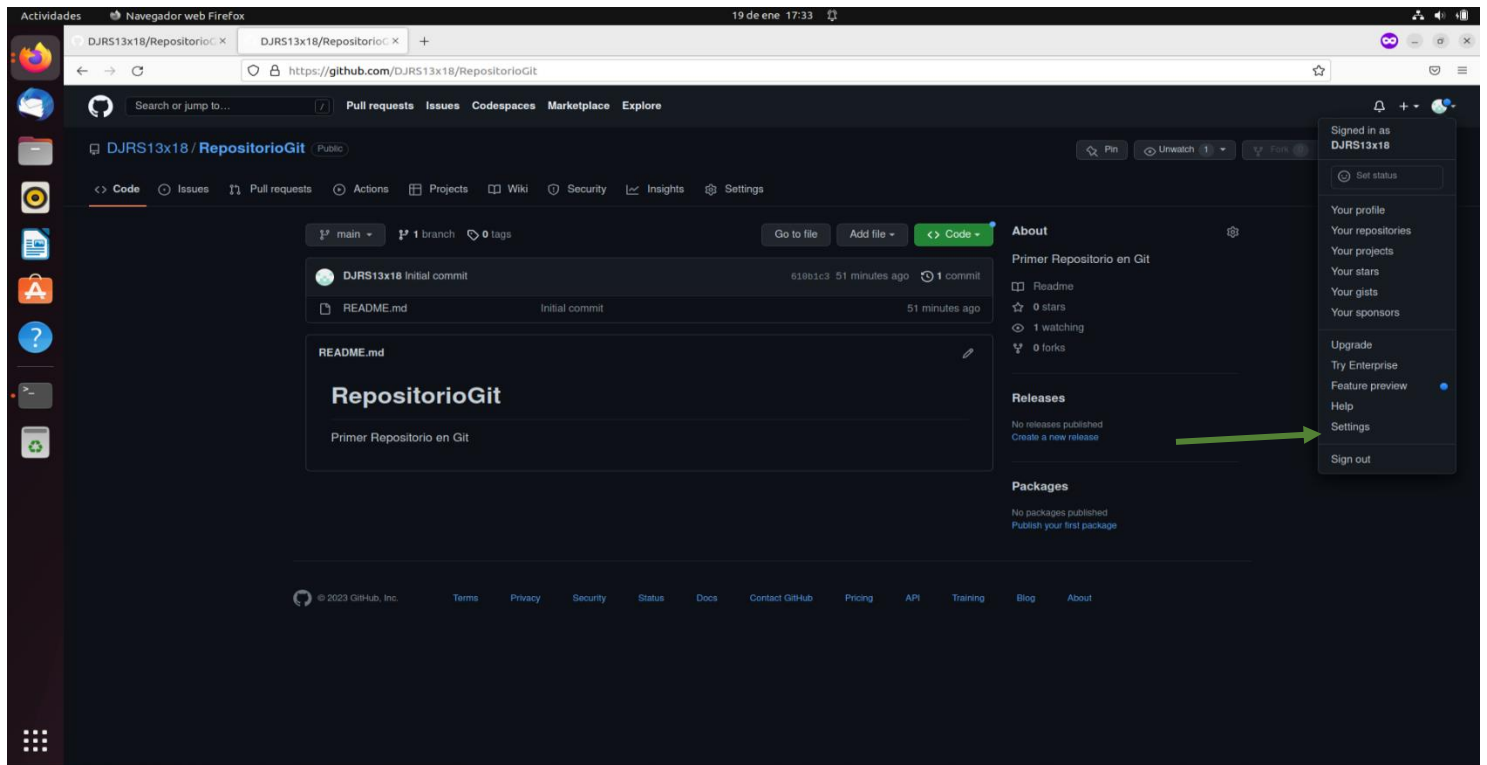
```
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepositorioGit$ ls
RepositorioGit
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepositorioGit$
```

Entramos dentro de nuestro repositorio

```
cd RepositorioGit
```

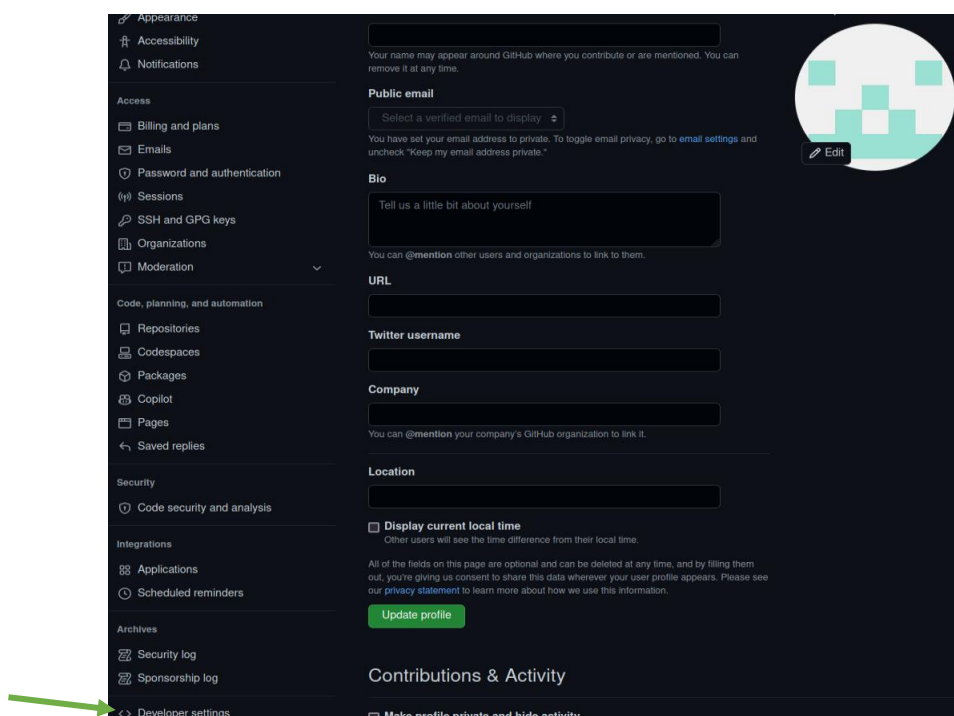
Y dentro estará todo tal cual como en el repositorio central que sería el de GitHub, la clonación es la forma más habitual en la que los desarrolladores obtienen una copia del trabajo.

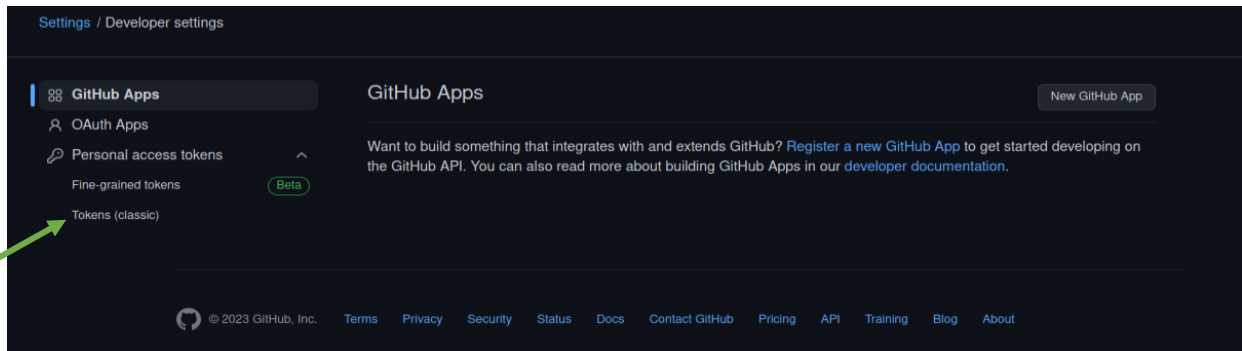
Tendremos que crear un token que será los que nos dará el acceso remoto a nuestro repositorio. Para ello tendremos que volver a la página de GitHub.



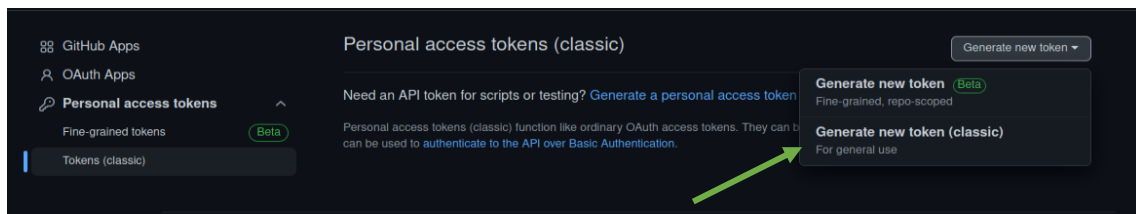
Estando en la página de GitHub hacemos click en la foto de nuestro usuario en la esquina superior derecha de la pantalla, luego hacemos click en el apartado **Settings**.

Dentro del apartado settings tendremos que ir a la última opción. **Developer Settings**.





Dentro del apartado **Developer Settings**, hacemos click en **Personal Access tokens** y luego click en **Tokens (classic)**.



Hacemos click en **Generate new token** y luego click en **Generate new token (classic)**.

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

Note

TokenRepositorio

What's this token for?

Expiration

30 days The token will expire on Sat, Feb 18 2023

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

- ☒ **repo** Full control of private repositories
 - ☒ repo:status Access commit status
 - ☒ repo_deployment Access deployment status
 - ☒ public_repo Access public repositories
 - ☒ repo:invite Access repository invitations
 - ☒ security_events Read and write security events
- ☐ **workflow** Update GitHub Action workflows
- ☐ **write:packages** Upload packages to GitHub Package Registry
 - ☐ read:packages Download packages from GitHub Package Registry
- ☐ **delete:packages** Delete packages from GitHub Package Registry
- ☐ **admin:org** Full control of orgs and teams, read and write org projects
 - ☐ write:org Read and write org and team membership, read and write org projects
 - ☐ read:org Read org and team membership, read org projects
 - ☐ manage_runners:org Manage org runners and runner groups
- ☐ **admin:public_key** Full control of user public keys

Tendremos que colocarle un nombre al Token, y añadir los **Scopes**.

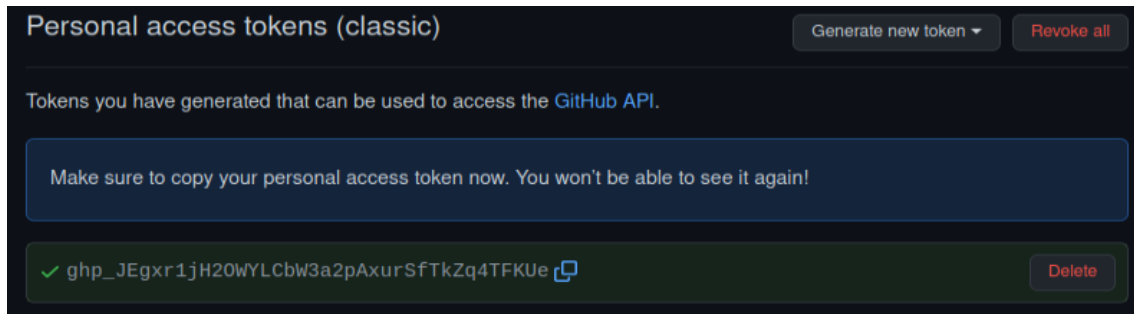
Los **Scopes** son simplemente operaciones que estarán o no permitidas desde este token. En este caso solo seleccionaremos la de **Repo**.

- ☐ **project** Full control of projects
 - ☐ read:project Read access of projects
- ☐ **admin:gpg_key** Full control of public user GPG keys
 - ☐ write:gpg_key Write public user GPG keys
 - ☐ read:gpg_key Read public user GPG keys
- ☐ **admin:ssh_signing_key** Full control of public user SSH signing keys
 - ☐ write:ssh_signing_key Write public user SSH signing keys
 - ☐ read:ssh_signing_key Read public user SSH signing keys

Generate token

Cancel

Al final de los Scopes podremos Generar el token que necesitamos para tener el acceso remoto al repositorio.



Esta sería la token ya generada que necesitamos para tener acceso remoto al repositorio, ahora lo que necesitamos es añadir el acceso remoto en el dispositivo en el cual se desea trabajar en el proyecto. En este caso que sería donde previamente clonamos el repositorio.

Para añadir el acceso remoto usaremos el comando

```
<git remote add [nombre del remoto] [URL del repositorio]>
```

Se vería de la siguiente manera

```
git remote add reporemoto https://github.com/DJRS13x18/RepositorioGit.git
```

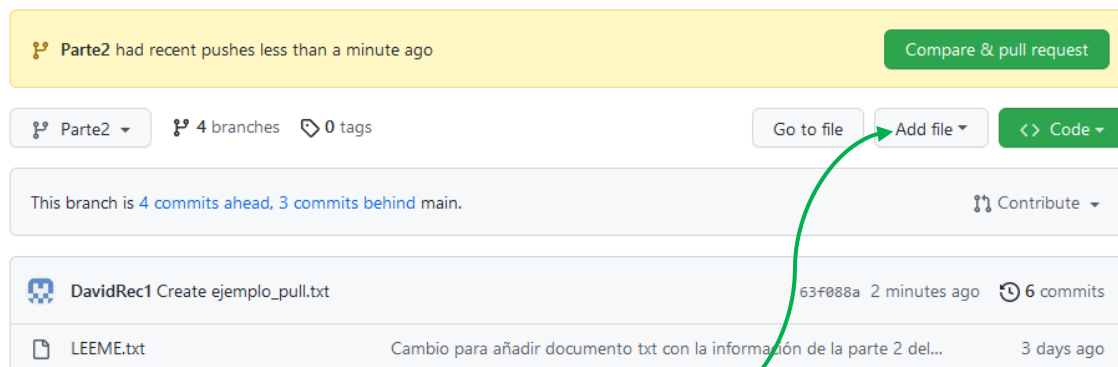
Lo siguiente sería verificar que el acceso remoto al repositorio está vinculado correctamente.

```
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepositorioGit/RepositorioGit$ git remote -v
origin https://github.com/DJRS13x18/RepositorioGit.git (fetch)
origin https://github.com/DJRS13x18/RepositorioGit.git (push)
reporemoto https://github.com/DJRS13x18/RepositorioGit.git (fetch)
reporemoto https://github.com/DJRS13x18/RepositorioGit.git (push)
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepositorioGit/RepositorioGit$
```

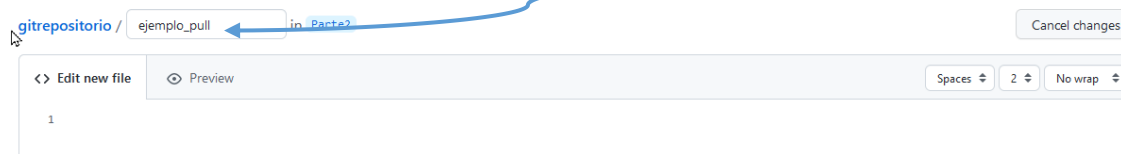
Con el comando `<git remote -v>` podemos ver el acceso remoto que hemos creado con el nombre que le hemos asignado. Y de esta manera podemos **crear un repositorio y conectarlo con nuestro repositorio personal en GitHub mediante la terminal de Linux.**

- Push Y pulls a los repositorios desde el terminal con ejemplos.

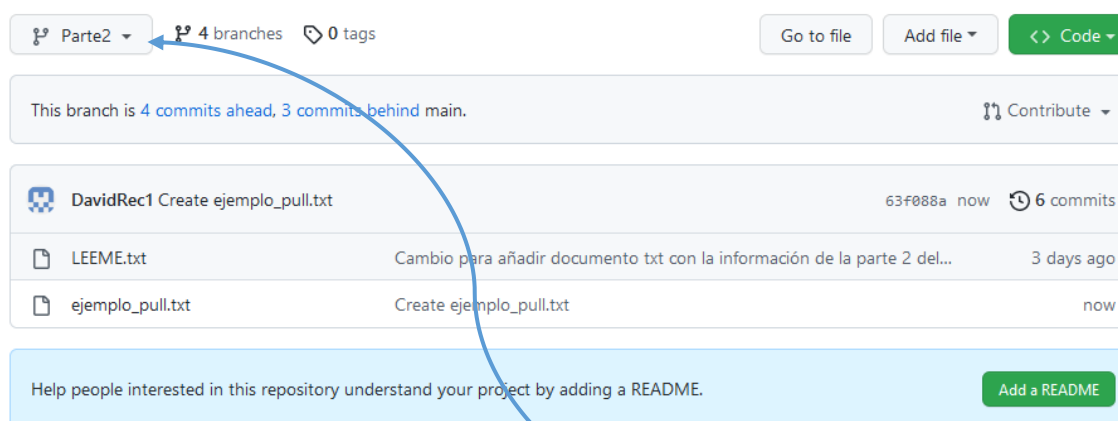
– Para poder mostrar cómo se realiza un **pull**, vamos a crear un nuevo archivo de texto directamente desde GitHub en el que le llamaremos ejemplo_pull.txt, la intención de crear este archivo de texto desde GitHub es para demostrar cómo descargarlo dicho archivo desde tu terminal.



Una vez seleccionado "añadir un nuevo archivo", lo siguiente será ponerle nombre, en este caso lo hemos llamado "ejemplo_pull.txt"



Al aceptarlo y añadirlo en nuestro repositorio de GitHub aparecerá el archivo de texto que acabamos de crear:



Tras añadir un nuevo documento de texto en nuestra **BRANCH (rama)**, el siguiente paso será descargarnos dicho documento de texto en nuestro ordenador desde la terminal.

Lo primero que debemos de tener en cuenta antes de hacer un pull en nuestra terminal, es crear una conexión entre nuestro repositorio local y el repositorio remoto (de GitHub).

El comando que debemos de utilizar para que dicha conexión entre los repositorios se realice es el siguiente:

```
<git pull --set-upstream [nombre del remoto] [nombre de la rama]>
```

En nuestro caso nosotros introducimos el comando de esta forma

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git pull --set-upstream reporemoto Parte2
```

Para mostrar que aún no tenemos dicho archivo en nuestro repositorio hacemos un `ls` y vemos el contenido de nuestra branch (rama)

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ ls  
LEEME.txt
```

Una vez configurada la conexión entre el repositorio local y el repositorio remoto lo único que tendremos que hacer es introducir el comando `git pull`

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git pull  
remote: Enumerating objects: 4, done.  
remote: Counting objects: 100% (4/4), done.  
remote: Compressing objects: 100% (3/3), done.  
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0  
Desempaquetando objetos: 100% (3/3), 761 bytes | 761.00 KiB/s, listo.  
Desde https://github.com/DavidRec1/gitrepositorio  
7c7137f..a0841ef Parte2 -> reporemoto/Parte2  
Actualizando 7c7137f..a0841ef  
Fast-forward  
 ejemplo_pull.txt | 1 +  
 1 file changed, 1 insertion(+)  
 create mode 100644 ejemplo_pull.txt  
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ ls  
ejemplo_pull.txt LEEME.txt
```

Tras haber hecho `git pull` el archivo que habíamos creado desde el repositorio remoto de GitHub ya está en nuestro repositorio local, volvemos a introducir el comando `ls` para mostraros los archivos nuevos descargados.

Y de esta manera queda ejemplificado el uso del comando pull, que es utilizado para descargar cambios desde un repositorio remoto a un repositorio local.

-Ahora vamos a explicar cómo hacer un push.

Lo primero que debemos de hacer es crear un archivo de texto, en este caso a dicho archivo lo hemos llamado "ejemplo_push.txt". Para ello introducimos en la terminal el comando `touch ejemplo_push.txt`.

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ touch ejemplo_push.txt
```

Una vez creado el archivo que vamos a "pushear" debemos añadirlo al área STAGIN introduciendo el comando `git add ejemplo_push.txt`

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git add ejemplo_push.txt
```

Ahora que lo hemos añadido al área STAGIN vamos a comprobar los cambios realizados con el comando `git status -s`

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git status -s
A  ejemplo_push.txt
```

La **A** al lado izquierdo del ejemplo_push.txt quiere decir que el archivo ya está añadido al Staging Area.

El siguiente paso es crear un commit al que vamos a añadir un comentario que se refiera a los cambios efectuados con el comando `<git commit -m "hacemos push para subir el archivo al repositorio remoto">`

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git commit -m "hacemos push para subir el archivo al repositorio remoto"
[Parte2 f6aca6d] hacemos push para subir el archivo al repositorio remoto
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 ejemplo_push.txt
```

Ahora sí, hacemos **push** para subir los datos seleccionados y añadidos anteriormente en el área STAGING para compartirlo en el repositorio remoto mediante el comando `git push`. Y de esta forma los cambios quedaran subidos a **GitHub**.

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git pull
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Desempaquetando objetos: 100% (3/3), 761 bytes | 761.00 KiB/s, listo.
Desde https://github.com/DavidRec1/gitrepositorio
7c7137f..a0841ef Parte2 -> reporemoto/Parte2
Actualizando 7c7137f..a0841ef
Fast-forward
 ejemplo_pull.txt | 1 +
1 file changed, 1 insertion(+)
create mode 100644 ejemplo_pull.txt
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ ls
ejemplo_pull.txt LEEME.txt
```

- Cada miembro del equipo debe realizar cambios en el repositorio propiedad de uno de los miembros, documenta el proceso.
- Para esta última parte del proyecto vamos a subir el archivo faltante de la Branch **Parte2** concretando al mismo tiempo la Branch **Parte3** documentando el proceso de los cambios que van a surgir a continuación.

Subimos el archivo faltante de la Branch Parte2 que sería el "Parte2.pdf"

Desde la terminal. Para esto emplearemos todos los comandos vistos anteriormente.

<git add [archivo a añadir]>

<git status -s>

<git commit -m "Comentario alusivo al cambio que se realizara">

<git push>

```
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git add Parte2.pdf
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git status -s
A Parte2.pdf
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git commit -m "Parte 2 del proyecto culminada"
[Parte2 f91fff5] Parte 2 del proyecto culminada
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Parte2.pdf
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$ git push
Username for 'https://github.com': DavidRec1
Password for 'https://DavidRec1@github.com':
Enumerando objetos: 4, listo.
Contando objetos: 100% (4/4), listo.
Compresión delta usando hasta 2 hilos
Comprimiendo objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 372.36 KiB | 11.28 MiB/s, listo.
Total 3 (delta 0), reusados 0 (delta 0), pack-reusados 0
To https://github.com/DavidRec1/gitrepositorio.git
   f6aca6d..f91fff5 Parte2 -> Parte2
david@PruebaPablo-VirtualBox:~/Escritorio/gitrepositorio$
```

Como la Parte2 ha sido culminada hay que actualizar el archivo “LEEME.txt” que se encuentra dentro de la Branch “Parte2” la cual esta creada por David Recio y ahora será modificada por Domenico Rosas mediante su terminal de Linux. Así se verían los cambios:

```
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$ ls
ejemplo_pull.txt  ejemplo_push.txt  LEEME.txt  Parte2.pdf
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$ nano LEEME.txt
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$ ls
ejemplo_pull.txt  ejemplo_push.txt  LEEME.txt  Parte2.pdf
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$ git add LEEME.txt
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$ git status -s
M LEEME.txt
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$ git commit -m "Parte2 finalizada y actualizada"
[Parte2 4ada485] Parte2 finalizada y actualizada
1 file changed, 9 insertions(+), 1 deletion(-)
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$ git push
Username for 'https://github.com': iRoot13x18
Password for 'https://iRoot13x18@github.com':
Enumerando objetos: 5, listo.
Contando objetos: 100% (5/5), listo.
Compresión delta usando hasta 6 hilos
Comprimiendo objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 550 bytes | 550.00 KiB/s, listo.
Total 3 (delta 0), reusados 0 (delta 0), pack-reusados 0
To https://github.com/DavidReci/gitrepositorio.git
f91fff5..4ada485 Parte2 -> Parte2
domenico@domenico-VirtualBox:~/Escritorio/CarpetaRepoD/gitrepositorio$
```

Estos cambios se verán reflejados en la Branch Parte2 en el archivo LEEME.txt de nuestro repositorio.

Dado así por completadas las “Parte2” y “Parte3” se subirá la Parte3.pdf a su Branch perteneciente y se actualizará también su archivo “LEEME.txt”