



---

# MANUAL GIT

---

DESPLIEGUE DE APLICACIONES WEB



NURIA NIETO CORRAL  
2 DAW

# ÍNDICE

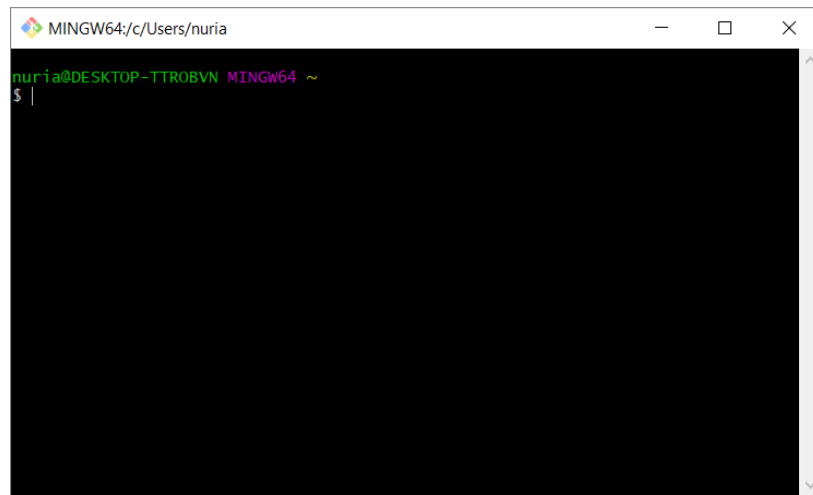
INSTALACIÓN DE GIT .....	2
INSTALACIÓN ATOM .....	2
CREACIÓN DE PROYECTO .....	3
UTILIZANDO COMANDOS.....	4
<b>COMPROBANDO DIRECTORIOS</b> .....	4
<b>PRIMEROS PASOS</b> .....	4
<b>CONFIGURACIÓN DE USER</b> .....	6
<b>COMMIT</b> .....	6
GUARDAR CAMBIOS EN NUESTRO PROYECTO .....	7
IGNORAR ARCHIVOS O CARPETAS .....	9
CREAR OTRA RAMA.....	9
GITHUB.....	10
<b>CREAR REPOSITORIO</b> .....	10
<b>SUBIR EL REPOSITORIO A GITHUB</b> .....	11
<b>DESCARGAR PROYECTO DE GITHUB</b> .....	13
GITKRAKEN .....	14
<b>INSTALACIÓN</b> .....	14
<b>TRABAJAR CON PROYECTO DE COMPAÑERO</b> .....	15
<b>UNIR RAMAS</b> .....	17

## INSTALACIÓN DE GIT

En primer lugar, vamos a instalar GIT desde su propia web:

<https://git-scm.com/>

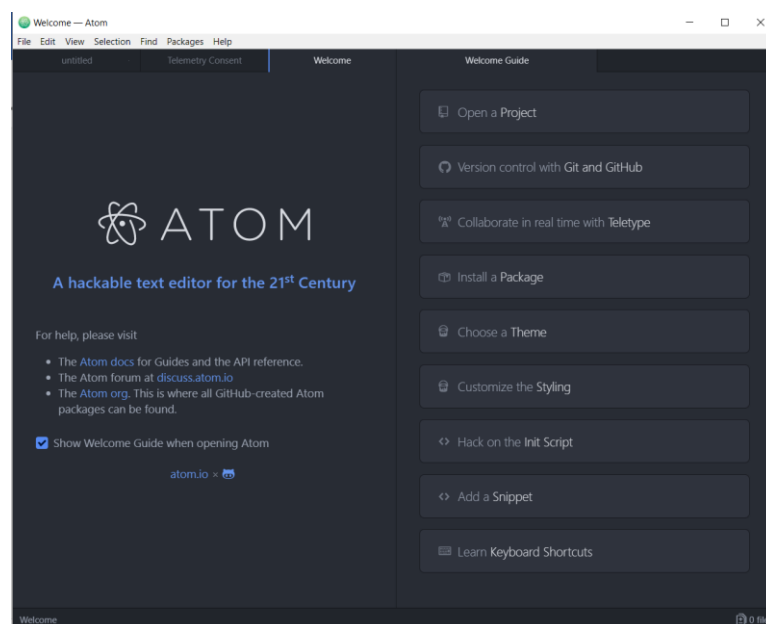
En mi caso ya estaba instalado de modo que no puedo agregar capturas de pantalla. Una vez que esta instalado podemos abrirlo y veremos esto:



## INSTALACIÓN ATOM

Lo descargamos de la web: <https://atom.io/>

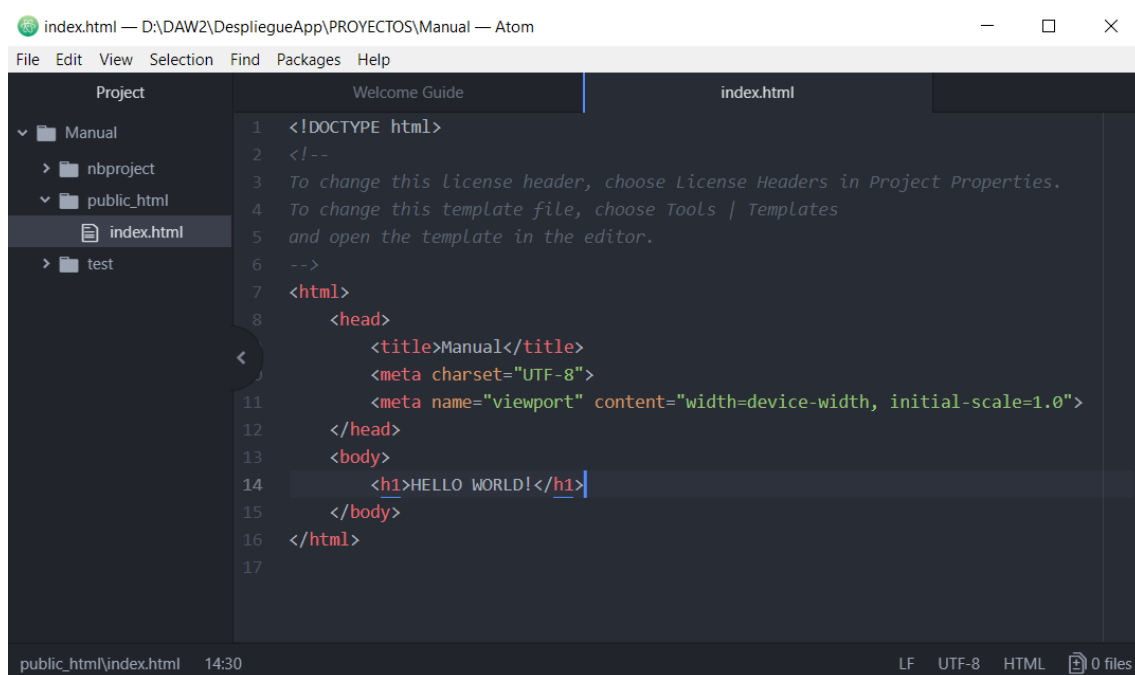
Se instala de forma directa y sencilla, puesto que una vez se descarga lo abrimos y está listo para utilizar.



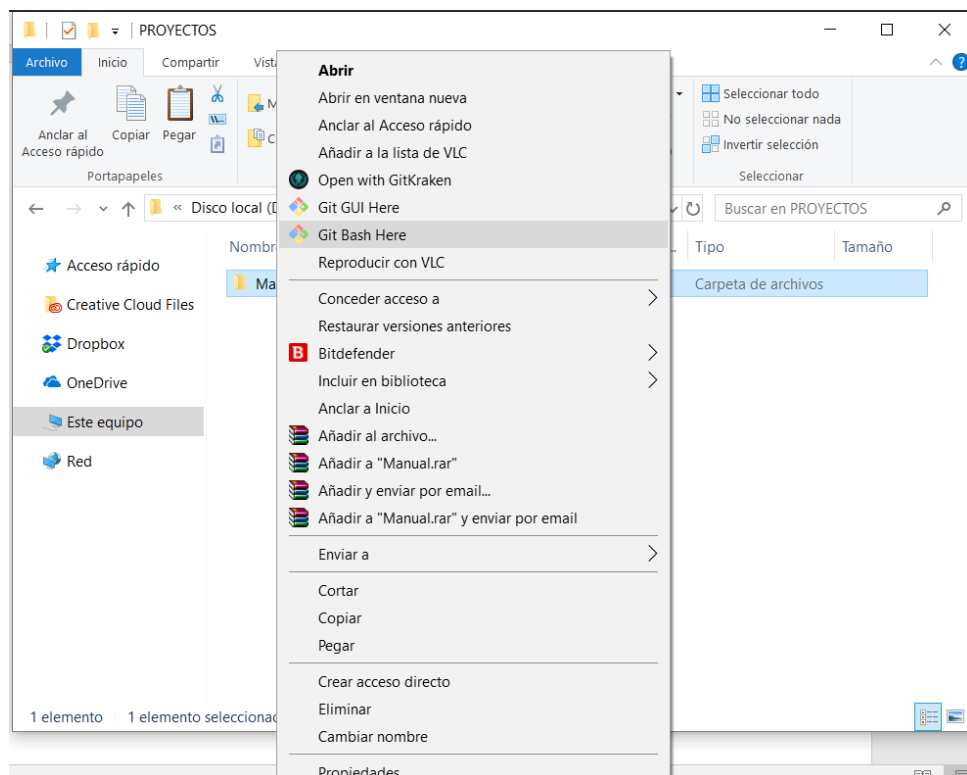
## CREACIÓN DE PROYECTO

Una vez hemos descargado las herramientas oportunas nos disponemos a crear un proyecto con el que trabajar.

En mi caso voy a crear el proyecto “Manual” en mi disco local y lo voy a arrastrar hasta Atom.



Después voy donde está mi proyecto y hago click derecho en él y le doy a “Git Bash Here”



## UTILIZANDO COMANDOS

A continuación, pongo los comandos que he ido utilizando, eso si teniendo en cuenta que esta consola diferencia entre mayúsculas y minúsculas.

### COMPROBANDO DIRECTORIOS

En primer lugar, utilizo el comando “**pwd**” para comprobar que estoy en el directorio correcto.

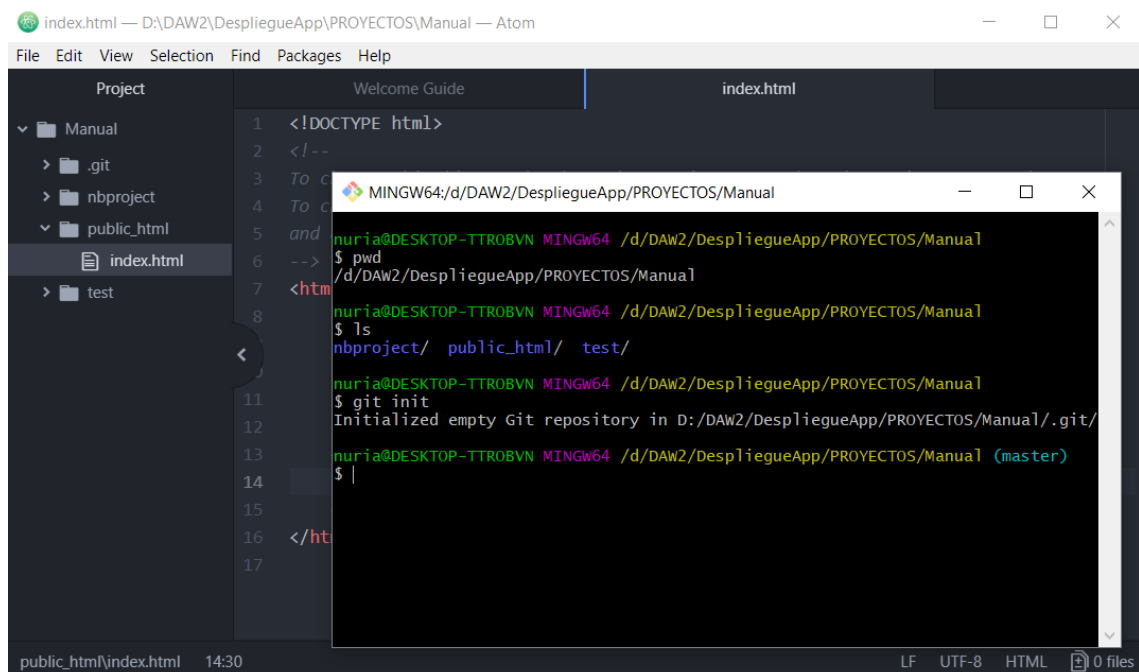
```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual
$ pwd
/d/DAW2/DespliegueApp/PROYECTOS/Manual
```

Después podemos usar “**ls**” para comprobar lo que hay dentro de nuestro directorio.

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual
$ ls
nbproject/  public_html/  test/
```

### PRIMEROS PASOS

Utilizamos el comando “**git ini**” para iniciar un proyecto, y como podemos ver en la siguiente imagen en nuestro proyecto se nos crea una nueva carpeta de git.



Si queremos ver que archivos estamos trabajando podemos utilizar el comando **“git status”**

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        nbproject/
        public_html/

nothing added to commit but untracked files present (use "git add" to track)
```

Si queremos agregar archivos a nuestro entorno de trabajo podemos usar el comando **“git add public\_html/”**

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual (master)
$ git add public_html/
warning: LF will be replaced by CRLF in public_html/index.html.
The file will have its original line endings in your working directory
```

De nuevo podemos utilizar el comando **“git status”** para comprobar que se ha añadido.

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

        new file:   public_html/index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        nbproject/
```

A continuación, quiero añadir un fichero css a mi proyecto, una vez lo tengo creado lo puedo añadir, utilizo el comando **“cd public\_html”** para acceder a la carpeta donde esta el fichero, después utilizo el comando **“git add style.css”**

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git add style.css
```



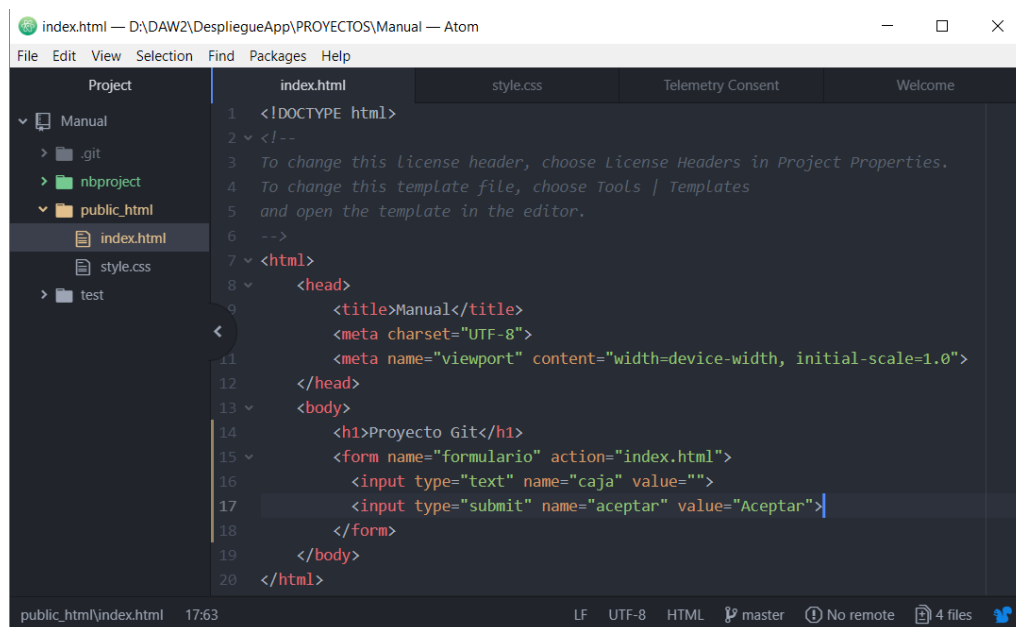
De modo que podemos utilizar el comando **"git log"** esto nos mostrara un código hash con el que poder diferenciar mi versión del proyecto, la persona (y el mail) que hizo el commit y la fecha;

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git log
commit cca8bf7ad403e73475db5acbae974377d6086757 (HEAD -> master)
Author: nuria <nuriartica@hotmail.com>
Date: Thu Sep 27 13:11:04 2018 +0200

    mi primer commit
```

## GUARDAR CAMBIOS EN NUESTRO PROYECTO

Queremos modificar ficheros de nuestro proyecto, añadir líneas de código, mejorarlo, etc. . Vamos a nuestro editor de texto y lo hacemos y guardamos.



Después vamos de nuevo a git y usamos el comando **"git status"** y nos dirá que el fichero ha sido modificado

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   index.html

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        ../nbproject/

no changes added to commit (use "git add" and/or "git commit -a")
```



Podemos descartar los cambios utilizando el comando **“git checkout – index.html”** de modo que volverá al estado anterior

```
<html>
  <head>
    <title>Manual</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <h1>HELLO WORLD!</h1>
  </body>
</html>
```

Voy a introducir de nuevo el texto anterior, y ahora vamos a utilizar el comando “git diff index.html” para ver que es lo que se ha modificado en el fichero. Como se puede comprobar en la siguiente imagen aparece un signo – y texto en rojo para lo que hemos quitado, y un símbolo + y en color verde lo nuevo que hemos agregado.

```
$ git diff index.html
warning: LF will be replaced by CRLF in public_html/index.html.
The file will have its original line endings in your working directory
diff --git a/public_html/index.html b/public_html/index.html
index d6f16a8..f134f0b 100644
--- a/public_html/index.html
+++ b/public_html/index.html
@@ -11,6 +11,10 @@ and open the template in the editor.
     <meta name="viewport" content="width=device-width, initial-scale=1.0">
   </head>
   <body>
-     <h1>HELLO WORLD!</h1>
+     <h1>Proyecto Git</h1>
+     <form name="formulario" action="index.html">
+       <input type="text" name="caja" value="">
+       <input type="submit" name="aceptar" value="Aceptar">
+     </form>
   </body>
</html>
```

Aunque esto nos aparece no hemos hecho un commit de modo que no ha sido guardado, para agregar los nuevos cambios utilizamos el comando “git add index.html” esto lo añade a mi entorno de trabajo, pero si de verdad quiero guardarlo debo hacer “git commit”, en vim añadimos algo descriptivo.

```
Insercion de formulario en el archivo html  
# Please enter the commit message for your changes. Lines starting  
# with '#' will be ignored, and an empty message aborts the commit.  
#  
# On branch master  
# Changes to be committed:  
#       modified:   index.html  
#  
# Untracked files:  
#       ../nbproject/  
~  
~  
~  
~  
~  
~  
~  
~
```

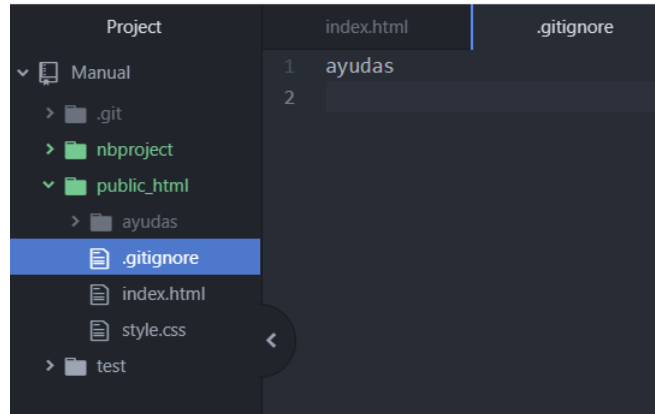
<eApp/PROYECTOS/Manual/.git/COMMIT\_EDITMSG[+] [unix] (19:51 30/09/2018)1,42 Todo : wq|

```
nuria@DESKTOP-TTROBYN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git commit
[master 199c0f2] Insercion de formulario en el archivo html
1 file changed, 5 insertions(+), 1 deletion(-)
```

## IGNORAR ARCHIVOS O CARPETAS

En nuestro proyecto podemos tener archivos o carpetas que queremos que GIT ignore para ello tenemos que crear en nuestro proyecto un archivo con el nombre **“.gitignore”** y en el escribir los archivos o carpetas que queremos que ignore.

Para que esto surta efecto debemos añadirlo con el comando **“git add .gitignore”**



Volvemos a hacer un commit para añadir este archivo, en esta ocasión voy a hacerlo de un modo distinto en el que introducimos directamente el mensaje sin tener que pasar por vim **“git commit -m “he agregado archivo .gitignore”**

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git commit -m "he agregado archivo .gitignore"
[master 32be545] he agregado archivo .gitignore
1 file changed, 1 insertion(+)
create mode 100644 public_html/.gitignore
```

## CREAR OTRA RAMA

Ahora vamos a utilizar el comando **“git branch”** y nos aparece lo que vemos en la imagen, en ella vemos que solo está la rama master.

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git branch
* master
```

Para crear otra rama usamos el comando **“git branch nuevarama”** y después el comando **“git checkout nuevarama”** para entrar en esa rama, y de nuevo usamos el comando **“git branch”** y como podemos comprobar ya estamos dentro de nuevarama

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(nuevarama)
$ git branch
  master
* nuevarama
```

Agregamos nuevos archivos a nuestro proyecto, primero creando y luego añadiendo a git en este caso vamos a poner el comando “**git add .**” poniendo el punto se agregan todos los archivos nuevos a la vez, así no tenemos que hacerlo uno a uno. Después ponemos “**git status**” para comprobar que se han añadido los archivos.

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(nuevarama)
$ git status
On branch nuevarama
Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   pag1.jsp
        new file:   pag2.jsp
        new file:   style2.css
```

De nuevo hacemos commit “**git commit -m “versión alternativa con nuevarama”**”

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(nuevarama)
$ git commit -m "version alternativa con nuevarama"
[nuevarama cf024cf] version alternativa con nuevarama
3 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 public_html/pag1.jsp
create mode 100644 public_html/pag2.jsp
create mode 100644 public_html/style2.css
```

Tenemos que tener en cuenta que estos cambios solo le afectan a **nuevarama** de modo que master esta como el ultimo commit que se hizo como master.

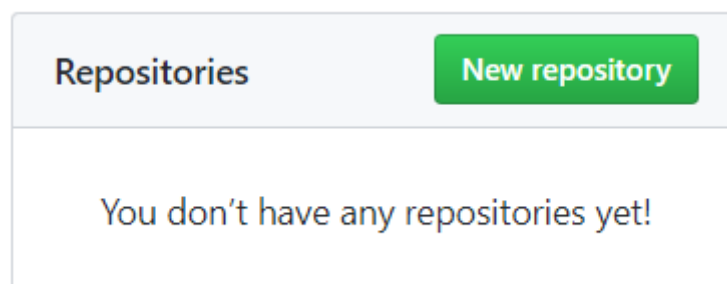
## GITHUB

Ahora queremos subir nuestro proyecto a github para que otros desarrolladores puedan participar en él.

El primer paso es crearse una cuenta de forma sencilla y gratuita, en mi caso ya la tengo con lo cual no voy a añadir en el manual como hacerse una cuenta.

### CREAR REPOSITORIO


Es hora de crear un nuevo repositorio y lo haremos dándole a “New repository”



Rellenamos los datos de nuestro repositorio y le damos a create repository.


## Create a new repository


A repository contains all the files for your project, including the revision history.

Owner:  Nuriani ▾ / Repository name:  ✓

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

Description (optional)

☒  **Public**  
Anyone can see this repository. You choose who can commit.

☐  **Private**  
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**  
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None** ▾

Add a license: **None** ▾




Create repository

## SUBIR EL REPOSITORIO A GITHUB

Al crear el repositorio este nos da un enlace web hasta ese repositorio.

### Quick setup — if you've done this kind of thing before

 Set up in Desktop or ☐ HTTPS ☒ SSH



We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

Y ahora desde la consola de git usamos el comando “git remote add origin <https://github.com/Nuriani/DespApp-DAW2-NuriaN.git>”

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git remote add origin https://github.com/Nuriani/DespApp-DAW2-NuriaN.git
```

Ahora escribimos el comand **"git push -u origin master"** y se nos abre otro programa en el que nos tenemos que loguear con nuestra cuenta de github

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git push -u origin master
```

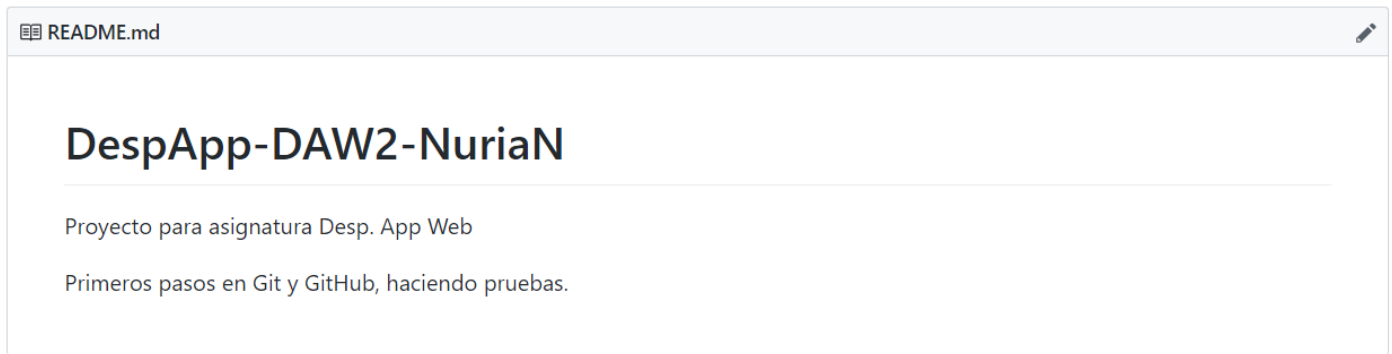
Una vez nos hemos autenticado en github en la consola podemos ver que se ha subido:

```
nuria@DESKTOP-TTROBVN MINGW64 /d/DAW2/DespliegueApp/PROYECTOS/Manual/public_html
(master)
$ git push -u origin master
Enumerating objects: 13, done.
Counting objects: 100% (13/13), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (13/13), 1.30 KiB | 265.00 KiB/s, done.
Total 13 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/Nuriani/DespApp-DAW2-Nurian/pull/new/master
remote:
To https://github.com/Nuriani/DespApp-DAW2-Nurian.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

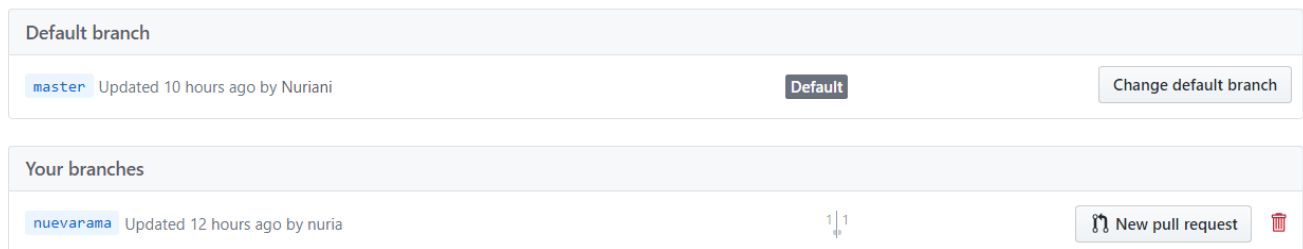
Si refrescamos la página de github del repositorio podemos ver que el proyecto esta subido con los ficheros y carpetas.

The screenshot shows the GitHub interface for a repository named 'DespApp-DAW2-Nurian' by user 'Nuriani'. At the top, there are buttons for 'Watch', 'Star', and 'Fork', each with a count of 0. Below this is a navigation bar with links to 'Code', 'Issues', 'Pull requests', 'Projects', 'Wiki', 'Insights', and 'Settings'. The repository description is 'Proyecto para asignatura Desp. App Web'. A summary bar shows '3 commits', '1 branch', '0 releases', and '0 contributors'. Below this, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. The commit history shows two commits: 'nuria he agregado archivo .gitignore' (latest commit 32be545, 3 hours ago) and 'public\_html he agregado archivo .gitignore' (3 hours ago). At the bottom, there is a prompt to 'Add a README'.

Si en esa misma ventana le damos a “Add a README” podemos escribir una descripción del proyecto para los otros usuarios que participen en él.

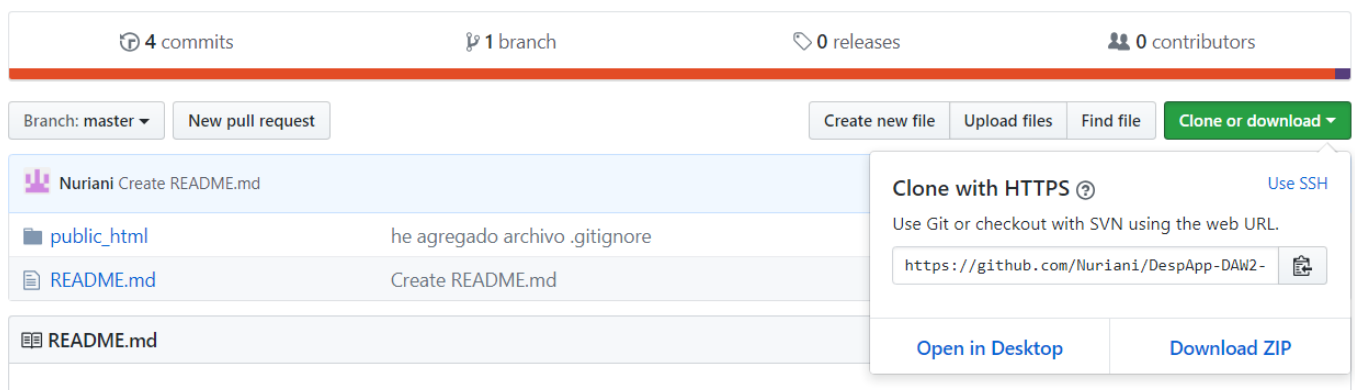


Ahora voy a subir la otra rama, para ello escribimos el comando “`git checkout nuevarama`” y después “`git remote add nuevarama https://github.com/Nuriani/DespApp-DAW2-NuriaN.git`” para indicarle la rama que vamos a subir y por ultimo “`git push -u nuevarama`” para subir la rama.



## DESCARGAR PROYECTO DE GITHUB

Si hemos borrado o queremos descargar nuestro proyecto en otro equipo podemos darle a clone or download para descargar el proyecto.



Podemos descargarlo directamente o podemos copiar el link que viene ahí y hacerlo desde la consola con el comando “`git clone https://github.com/Nuriani/DespApp-DAW2-NuriaN.git`”

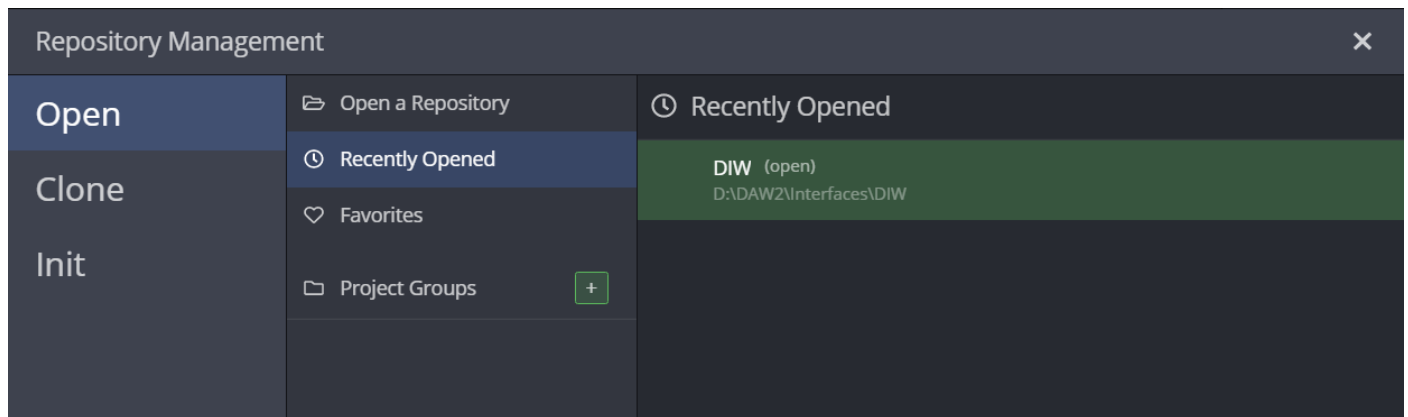
# GITKRAKEN

## INSTALACIÓN

En mi caso ya descargué de la web <https://www.gitkraken.com/> e instale esta aplicación, es realmente sencillo y gratuito.

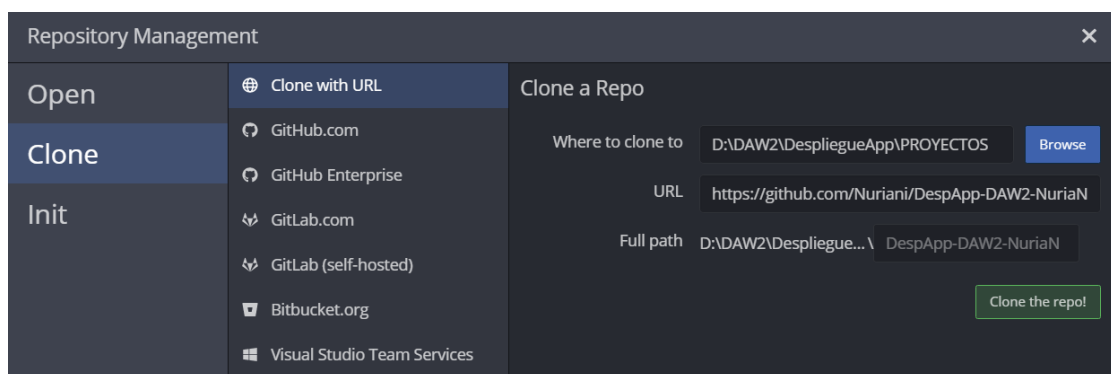
Tiene la ventaja de que podemos descargarlo tanto para Linux, Windows o Mac y además es compatible tanto con Github como Bitbucket (los repositorios más usados actualmente).

Podemos abrir (Open), clonar (Clone) o iniciar (Init) un nuevo proyecto como se puede ver en la próxima imagen.

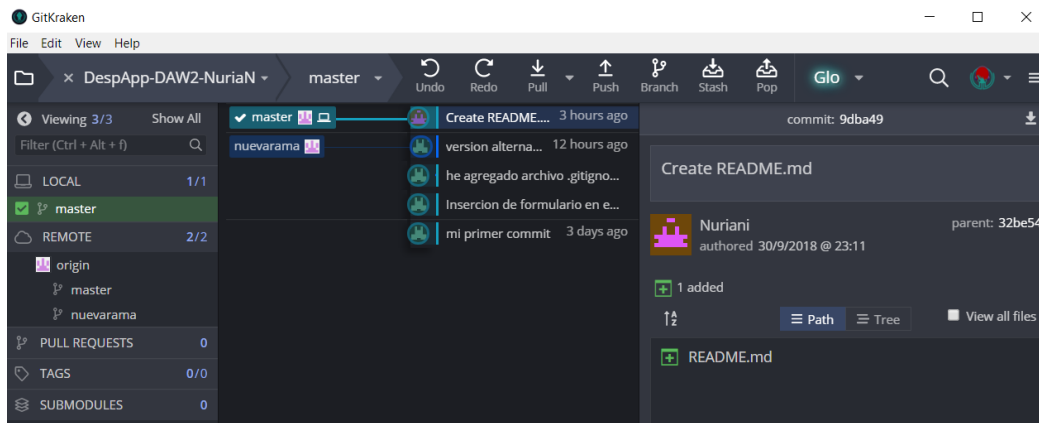


Como ya hemos iniciado el proyecto en la parte anterior del manual voy a clonar este mismo:

Primero le damos a clone y añadimos la url del proyecto y le indicamos donde queremos guardar el proyecto, y clonamos el repositorio:

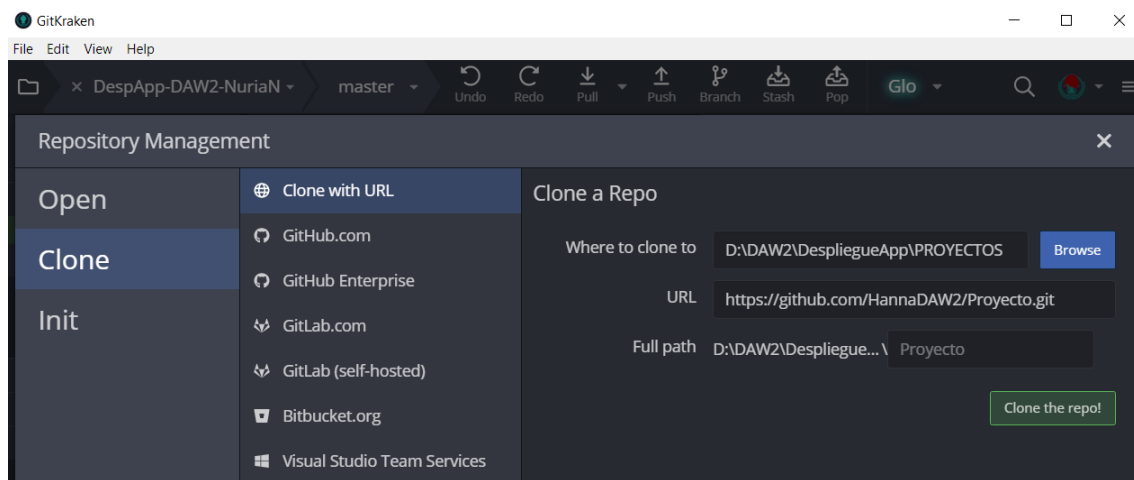


Y a continuación podemos ver el proyecto descargado y trabajar con el, podemos utilizar los botones pull para descargar actualizaciones y push para subir.

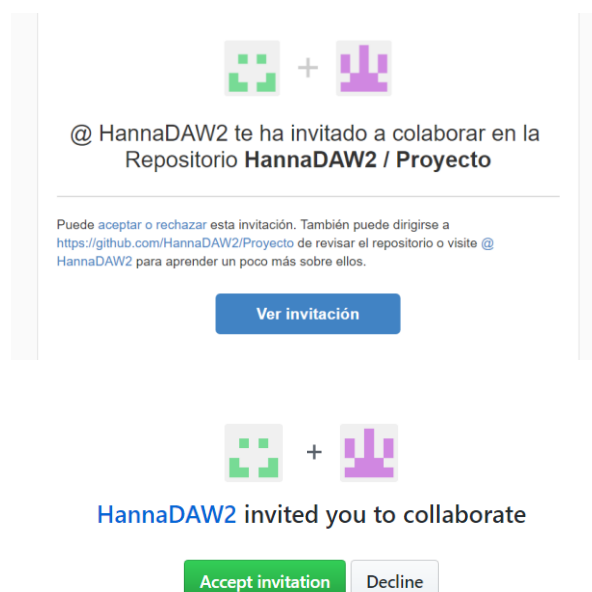


## TRABAJAR CON PROYECTO DE COMPAÑERO

Sigo los mismos pasos que en el apartado anterior esta vez con la dirección de mi compañera Hanna que es:

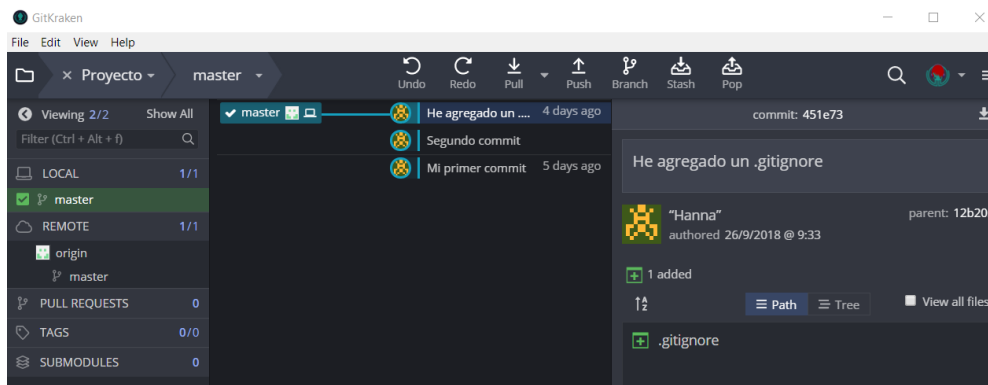


Para poder colaborar mi compañera tiene que invitarme a colaborar y yo debo aceptar la invitación, esta se envía al correo.

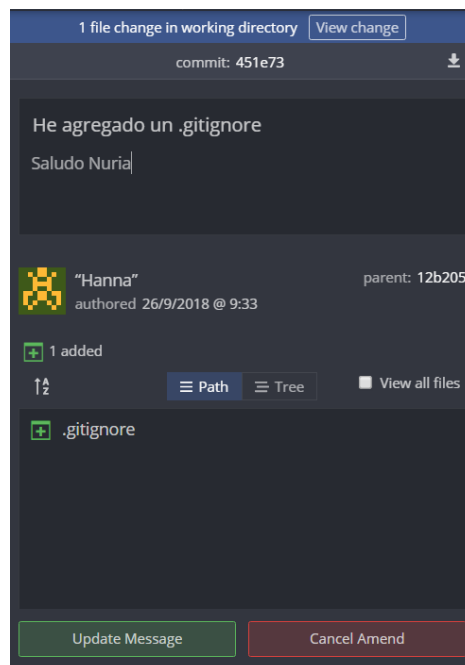




Como podemos comprobar en la siguiente imagen el repositorio ya se ha descargado



Modificamos lo que queramos del proyecto y hacemos click derecho en la rama master, a continuación, le damos a Edit commit message, escribimos algo descriptivo y le damos a Update Message.



Después hacemos un **“push”** para subir los cambios.



Como podemos comprobar es bastante sencillo e intuitivo, podemos realizar más cosas como:

**Pull:** para descargar las actualizaciones que hacen otros compañeros.

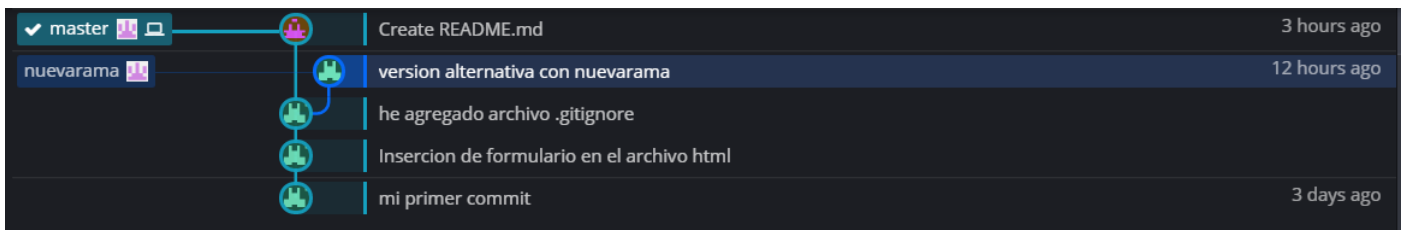
**Undo:** podemos revertir el ultimo commit

También podemos cambiar de rama.

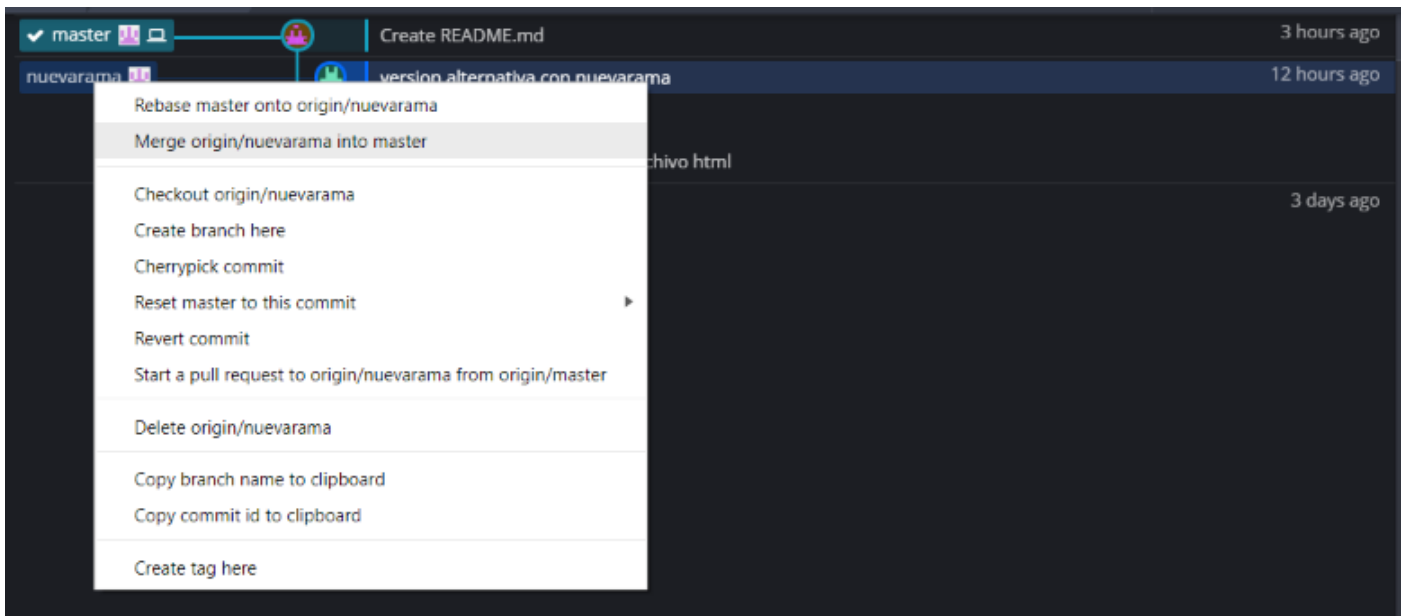
**Redo:** podemos revertir el ultimo undo.

## UNIR RAMAS

Vuelvo de nuevo a mi proyecto en el que tengo dos ramas como se puede comprobar en la siguiente imagen:



Para ello es necesario realizar un merge, nos ponemos encima de la rama y hacemos click derecho en “**Merge origin/nuevarama into master**”



Y vemos como estas quedan fusionadas

