UT3 - Actividad 07 – Git Remoto

Contenido

[1 Github, Git Remote, Push y Pull 2](#_Toc124153180)

[1.1 Control de acceso al repositorio remoto. 2](#_Toc124153181)

[1.2 GitHub 4](#_Toc124153182)

[1.3 Creación de una cuenta en github 5](#_Toc124153183)

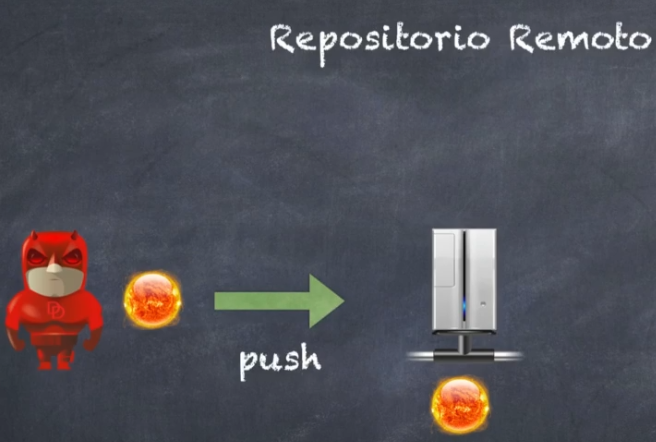
[1.4 Push a GitHub 6](#_Toc124153184)

[2 Comando pull 13](#_Toc124153185)

**A continuación, completa este manual con tus capturas tras la ejecución de los comandos, sustituyendo cuando sea necesario las imágenes por las tuyas.**

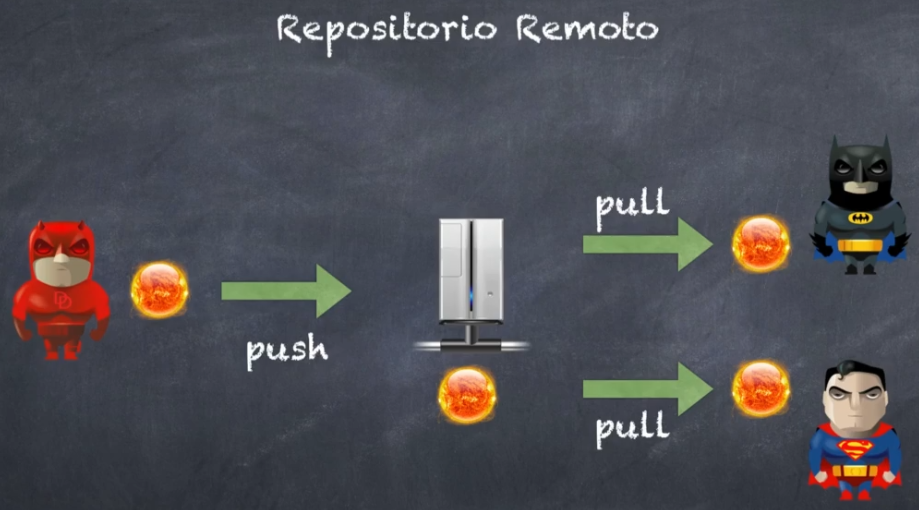
# Github, Git Remote, Push y Pull

Si por alguna razón se nos estropea el disco duro donde tenemos nuestro repositorio, por mucho que tengamos muchos commits perderemos toda la información. Por eso es recomendable tener una copia en un repositorio remoto, es decir en un repositorio que no sea nuestro equipo.



Para subir una copia de nuestro repositorio a un repositorio remoto se usa el **comando** **push**.

Por otra parte, si varias personas están trabajando sobre el mismo repositorio y alguien ha utilizado el **comando** **push** para subir cambios, las demás personas podrán actualizar su repositorio local con estos cambios utilizando el **comando pull**:



## Control de acceso al repositorio remoto.

Git no se encarga de controlar el acceso al repositorio remoto, es decir de controlar que usuario tiene permiso para modificar ciertos archivos y cuáles no. De esto se encargan otras herramientas.

Tenemos dos tipos de herramientas para controlar el acceso al repositorio remoto:

**Hosted services**: en este caso usamos los servicios que nos proporciona alguna plataforma.

**Servicios gestionados por nosotros**: utilizamos una herramienta desde nuestro equipo la cual nos permite configurar todo por nosotros mismos.



## GitHub

GitHub es una plataforma de desarrollo colaborativo de software para alojar proyectos.

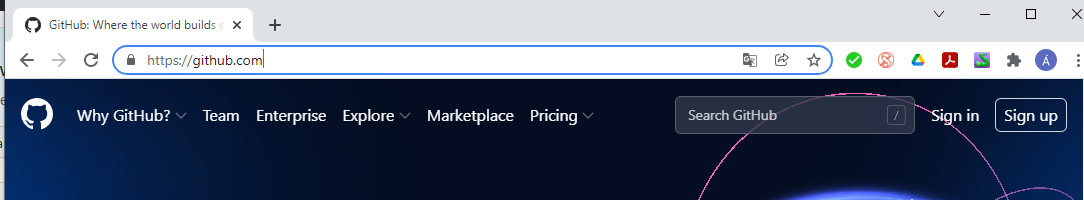
GitHub es una plataforma muy popular porque su versión gratuita nos proporciona muchas características como:

* Repositorios ilimitados
* Páginas HTML, CSS y JS ilimitadas
* Push, Pull, Clones ilimitados
* Issues, Wikis, estadísticas ilimitadas
* Organizaciones ilimitadas
* Desde 2019 permite crear repositorios privados además de públicos.

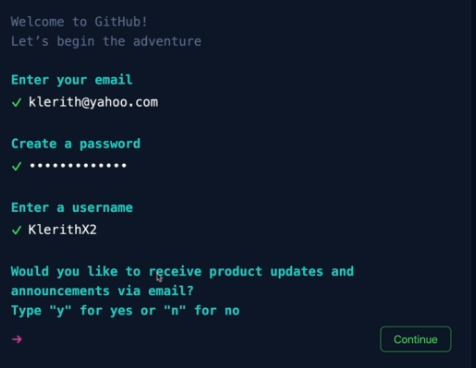
## Creación de una cuenta en github

Para crear una cuenta iremos a la web de GitHub: <https://github.com/>

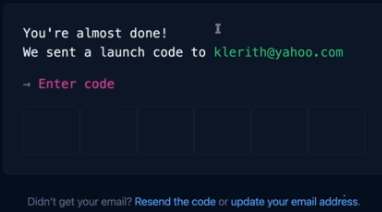
Y pulsaremos en el **enlace** **Sign Up:**



A continuación tendremos que rellenar un formulario con datos introduciendo entre otros un correo electrónico al que nos mandarán datos para verificar la cuenta:

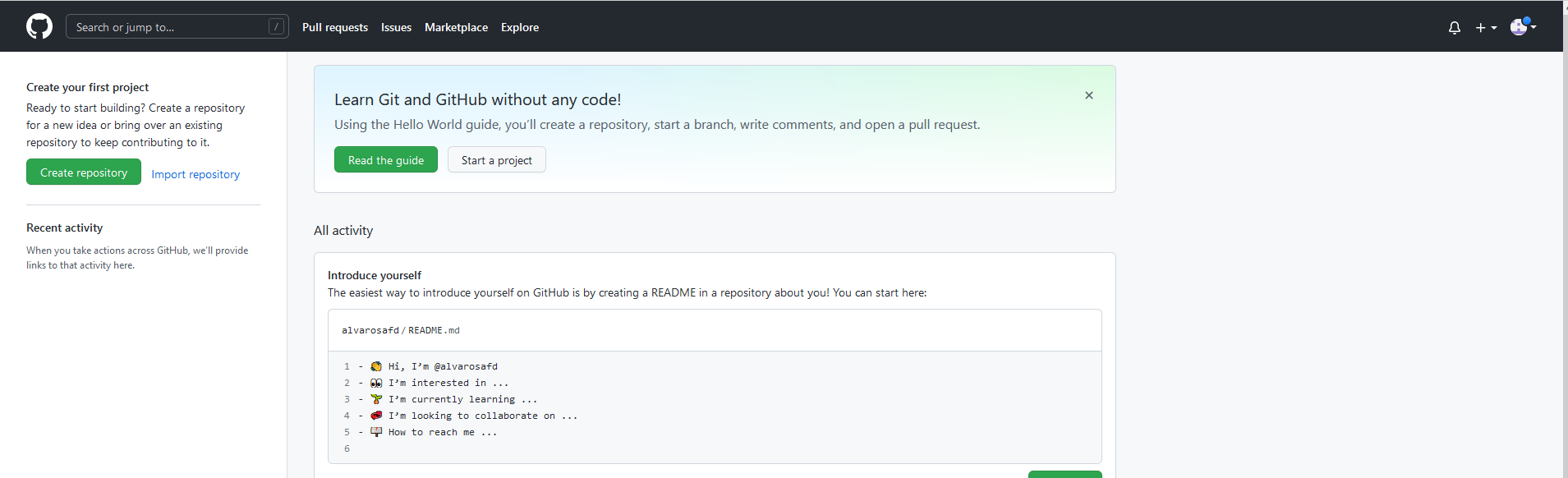


Después de rellenar el formulario y completar un cátchup tendremos que verificar la cuenta introduciendo el código recibido en el correo:



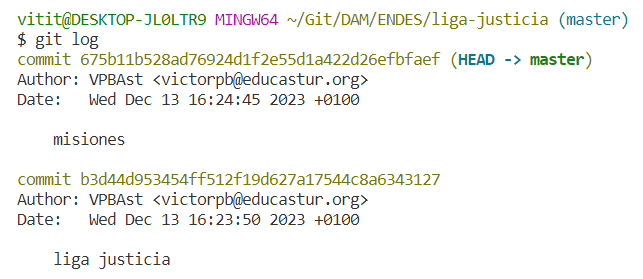
En algún punto del proceso nos permitirá elegir si queremos una cuenta de estudiante lo cual nos proporcionará acceso a funciones avanzadas de forma gratuita. Para ello será necesario adjuntar mediante foto un documento acreditativo de ser estudiante. Este documento os lo habrá facilitado el tutor con anterioridad.

Una vez esté la cuenta creada y estemos logueados veremos algo así:



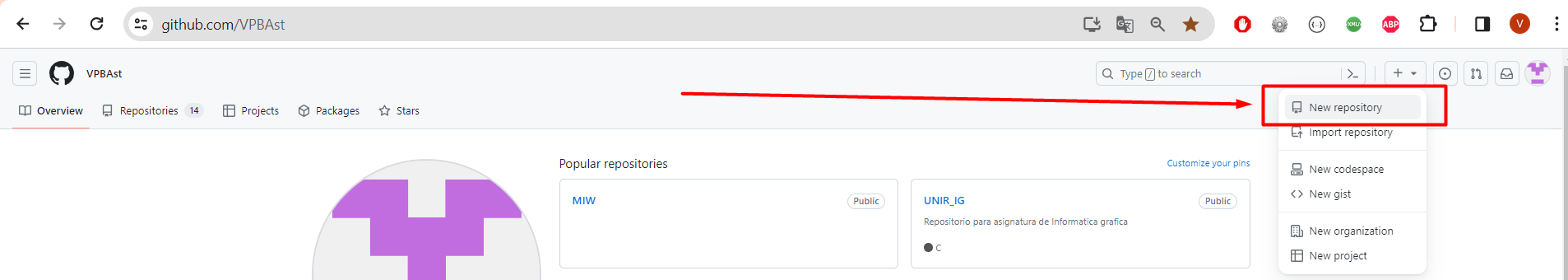
## Push a GitHub

Tomamos nuestro repositorio ya creado en local donde ya está git iniciado es decir la ya venía con el directorio .git y commits realizados, podemos comprobarlo haciendo git log:



…

Vamos a GitHub y creamos un repositorio:

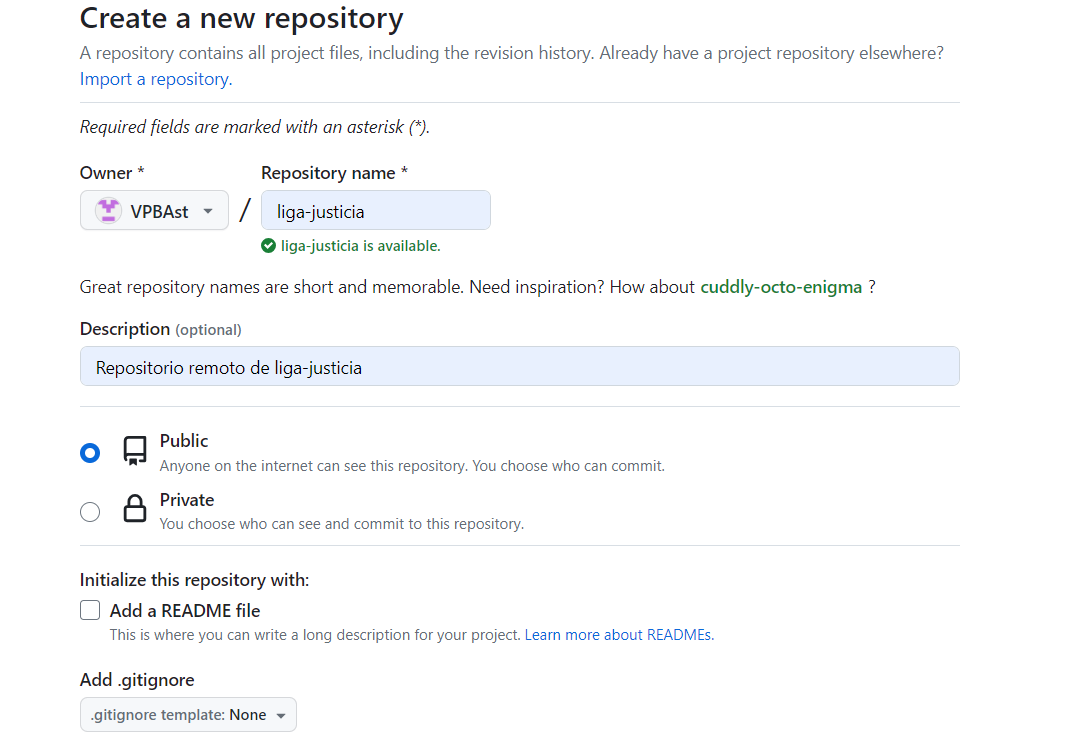


Le damos un nombre que debe ser único dentro de nuestra cuenta.

Rellenamos la descripción del repositorio

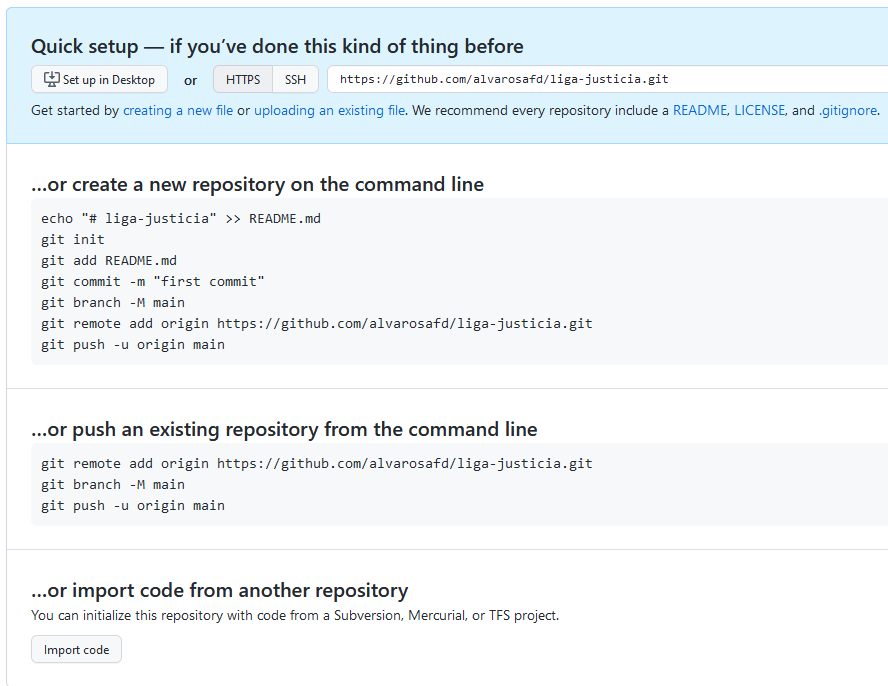
Elegimos que queremos que sea publico

Podemos marcar que se creen ciertos archivos como un archivo README o el archivo .gitignore, en nuestro caso estos dos archivos ya existen en nuestro repositorio local que queremos subir a GitHub, así que no marcamos la opción.



**Si elegimos que el repositorio sea público cualquiera podrá verlo y hacer una copia, pero no modificarlo salvo que nosotros lo permitamos.**

Una vez le damos al botón de crear el repositorio GitHub nos muestra opciones para rellenarlo con archivos:



Aquí podemos elegir si queremos usar HTTPS o SSH para la versión escritorio

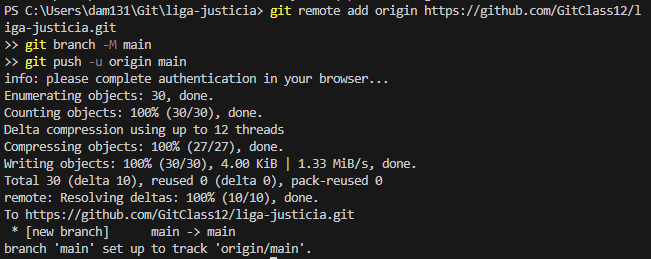
Esta opción nos muestra los comandos para crear un nuevo repositorio local y subirlo a GitHub

Por último también tenemos la opción de rellenar nuestro repositorio cogiendo código de otro repositorio remoto

Vamos a usar la segunda opción **push an existing repository from command line**, subir un repositorio ya existente a GitHub. Copiaremos los tres comandos que nos proporciona GitHub y los ejecutaremos desde nuestro equipo:

Añadimos a nuestro repositorio local un repositorio remoto, **add origin** está añadiendo el repositorio remoto y le está dando el nombre local de **origin** este es un nombre estándar que se suele dar al primer repositorio remoto que añadimos pero podríamos ponerle otro.

git remote add origin …



A continuación se indica el nombre de la rama principal

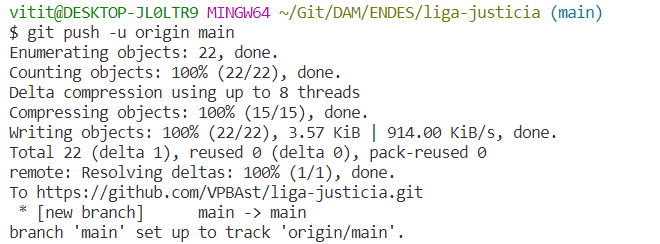
git branch -M main



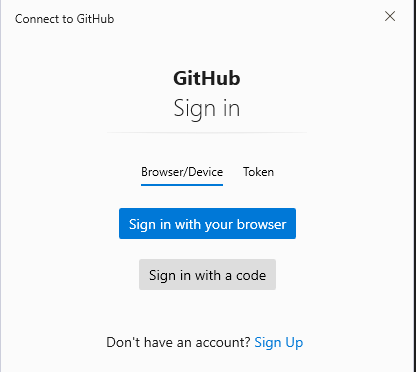
Finalmente hacemos el push del repositorio (es decir lo subimos a GitHub), la opción -u hace que a partir de ahora no tengamos que indicar el nombre del repositorio al que vamos a hacer el push ya que estableceremos que será por defecto el que indiquemos en este comando. En este caso es el repositorio origin

git push -u origin main

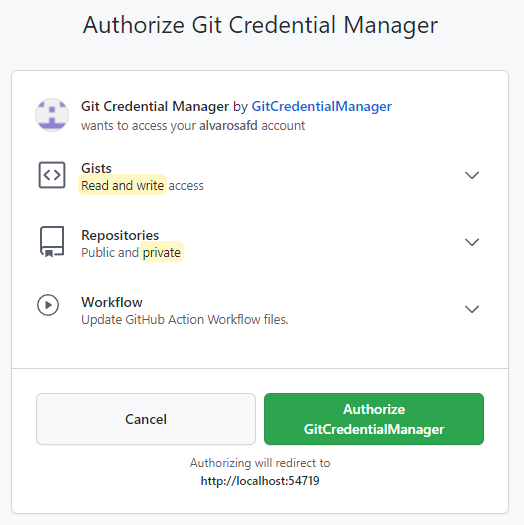
Si ya estamos logueados en GitHub subirá los ficheros y el histórico de commits:



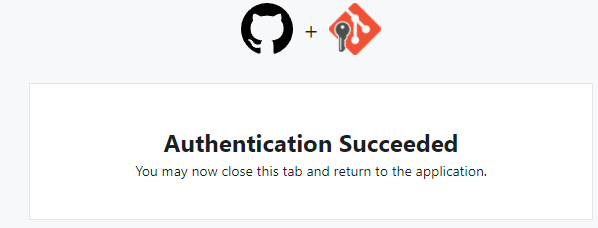
Si no estamos logueados, al ejecutar el último comando nos pedirá que nos identifiquemos en GitHub:



Una vez que nos identifiquemos por ejemplo con la opción de usar el navegador, nos pedirá que demos permisos de acceso:

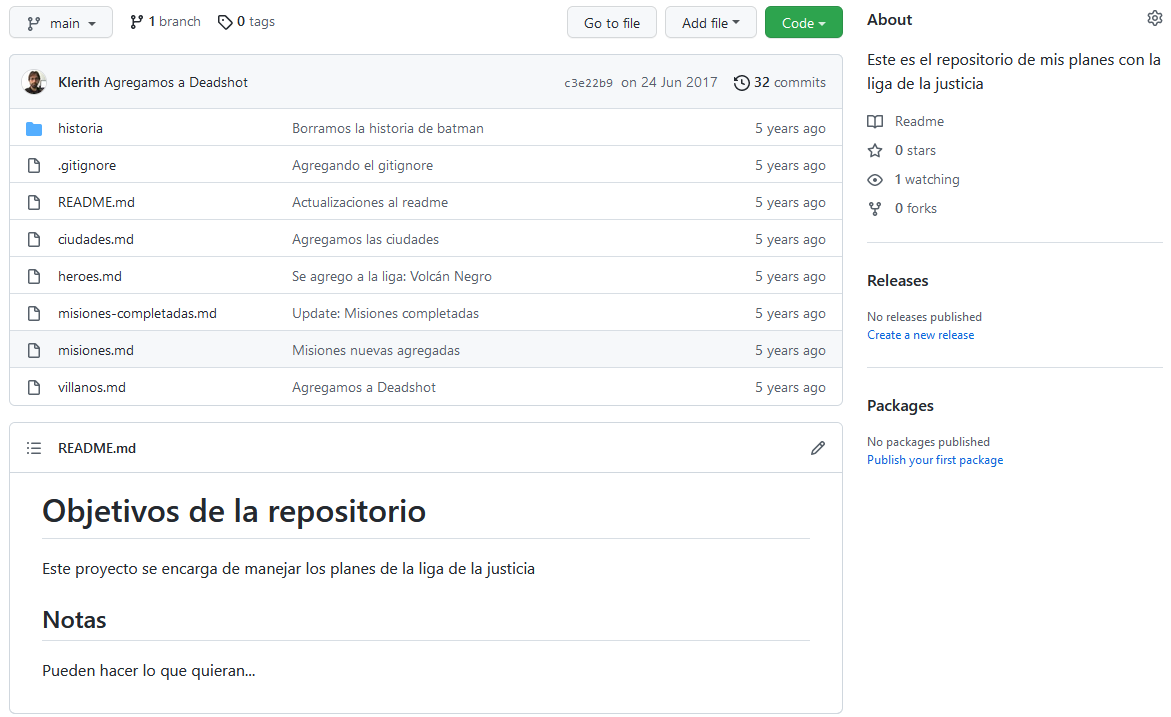


Una vez que le demos al botón de autorizar nos mostrará el siguiente mensaje si todo ha ido bien:



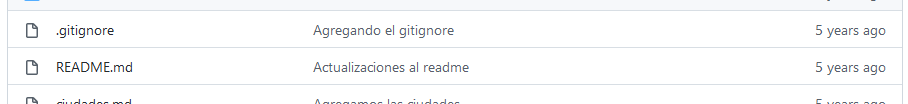
Nos indica que se realizó con éxito la autentificación y que podemos cerrar esta pestaña.

Si ahora vamos a ver nuestro repositorio en GitHub veremos que se ha subido el contenido que teníamos en nuestro repositorio local:

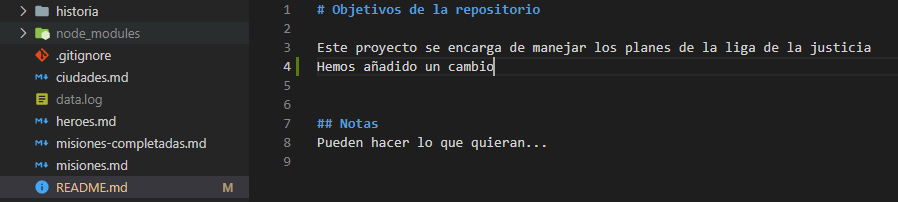


Como vemos git identifica el usuario creador de los archivos del repositorio y además nos indica que en este repositorio tiene 32 commits.

Si pinchamos en cualquier archivo podemos ver el contenido:



Vamos a hacer un cambio en el repositorio local y subirlo a GitHub. Editamos el archivo README.md



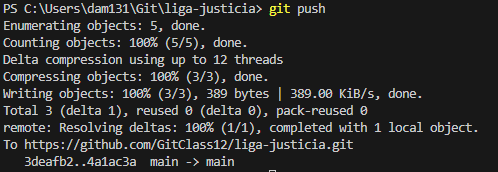
Hacemos commit de este cambio:

git commit -am "README.md modificado"



Y subimos el cambio a GitHub

git push



Como anteriormente usamos git push -u origin main dejamos establecido el repositorio remoto y por eso ahora no nos hace falta especificarlo de nuevo.

Si abrimos de nuevo en GitHub el fichero README.md:

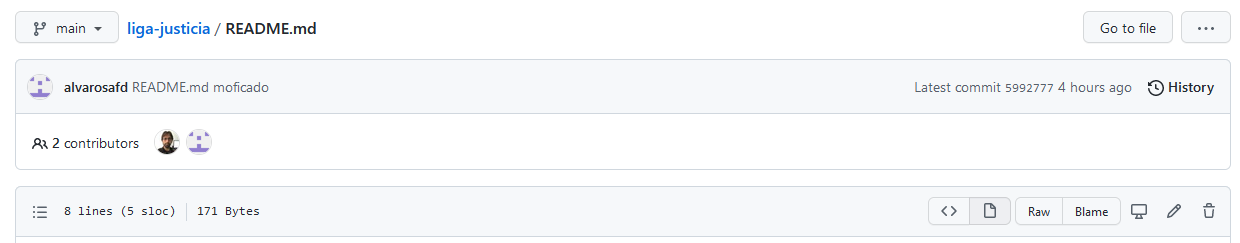


Vemos que en efecto el archivo se actualizó.

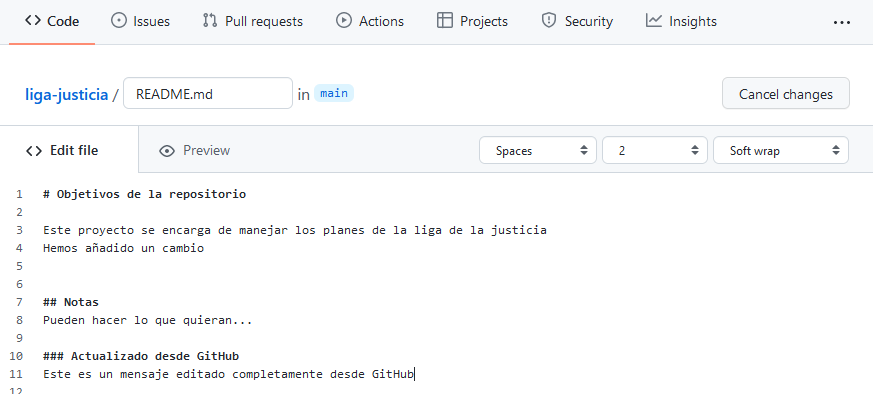
# Comando pull

Desde el propio GitHub podemos editar los archivos que se encuentran en el repositorio remoto. Para actualizar estos cambios en nuestro repositorio local usaremos el comando pull.

Vamos a modificar desde GitHub el archivo README.md

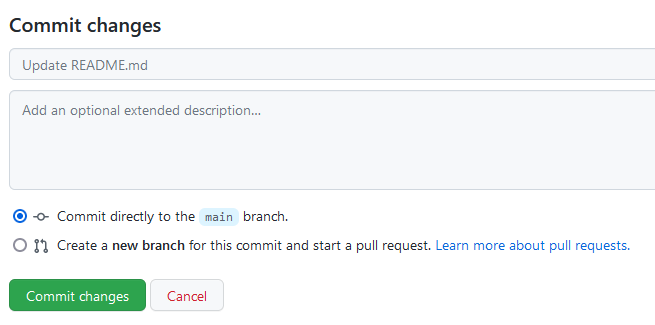


Al pulsar el botón de editar pasaremos al modo de edición:

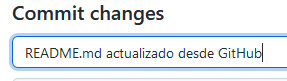


Realizamos los cambios

Para guardar los cambios nos desplazamos hacia abajo en la página web:



Para guardar los cambios GitHub crea un nuevo commit con el mensaje que establezcamos. En este caso si no ponemos nada el mensaje será por defecto “Update README.md”. Vamos a poner “README.md actualizado desde GitHub”:

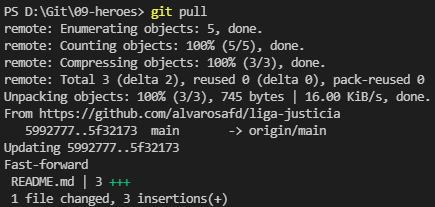
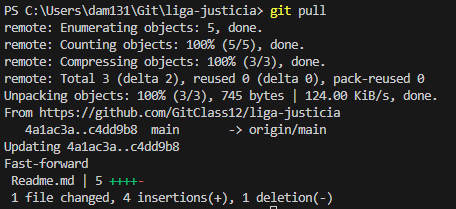


A continuación pulsamos el botón de Commit Changes:

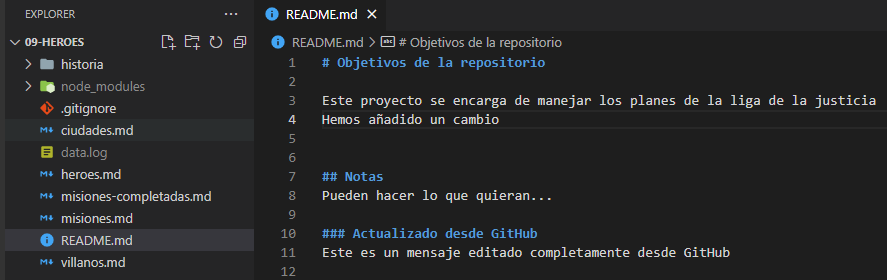
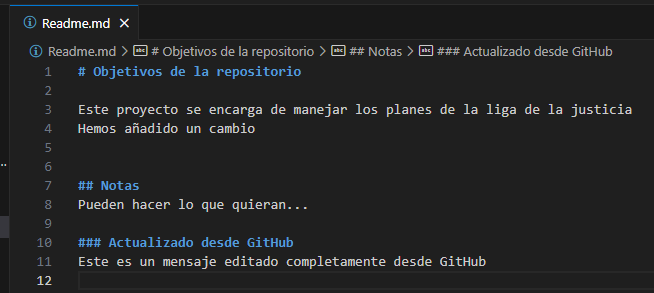


Para traer este cambio realizado en GitHub a nuestro repositorio local ejecutamos el siguiente comando desde nuestro repositorio local:

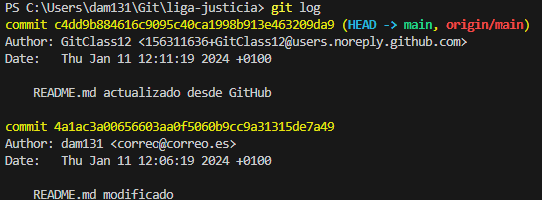
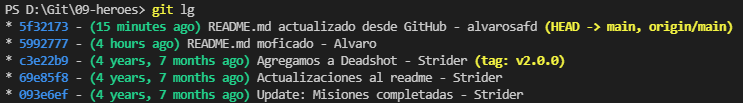
git pull



Si ahora editamos el archivo desde visual studio code veremos que se han añadido los cambios realizados desde GitHub:



Y si hacemos git log veremos que se ha añadido el commit realizado desde GitHub:



# URL de tu repositorio liga-justicia

Adjunta la URL de tu repositorio liga-justicia , que debe estar configurado como **público**:

<https://github.com/GitClass12/liga-justicia>