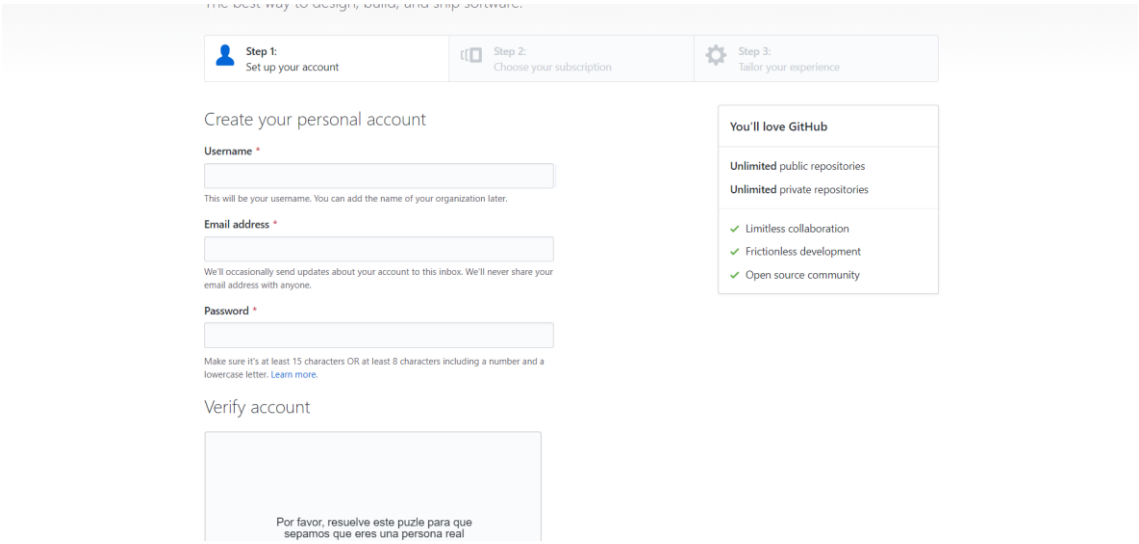
CONTROL DE VERSIONES GIT – GITHUB

Jesús Rodríguez – Desarrollador Aplicaciones Multiplataforma

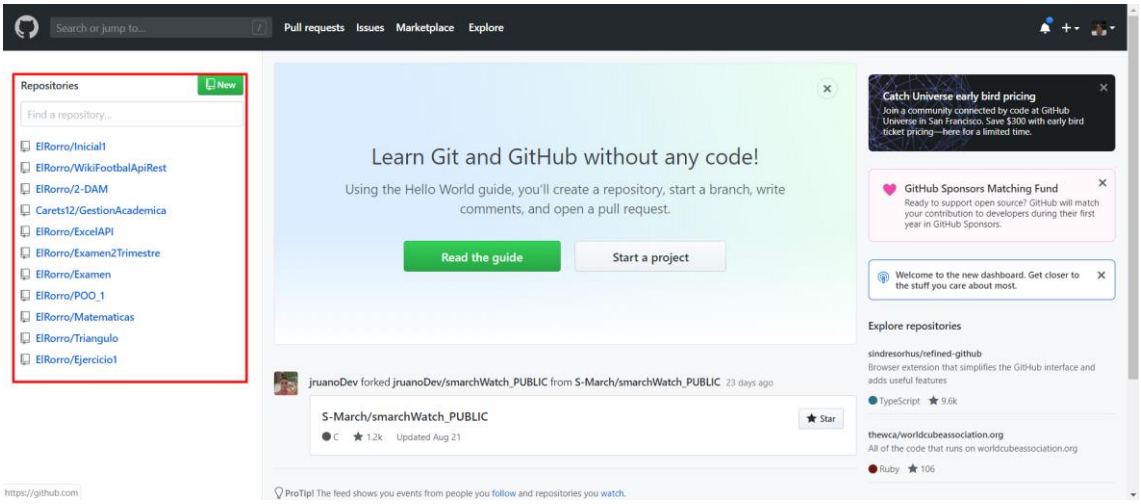
ÍNDICE

1.-a.- Registro en control de versiones de git

Accedemos a la página web de git en mi caso github https://github.com/, y nos registramos en ella.

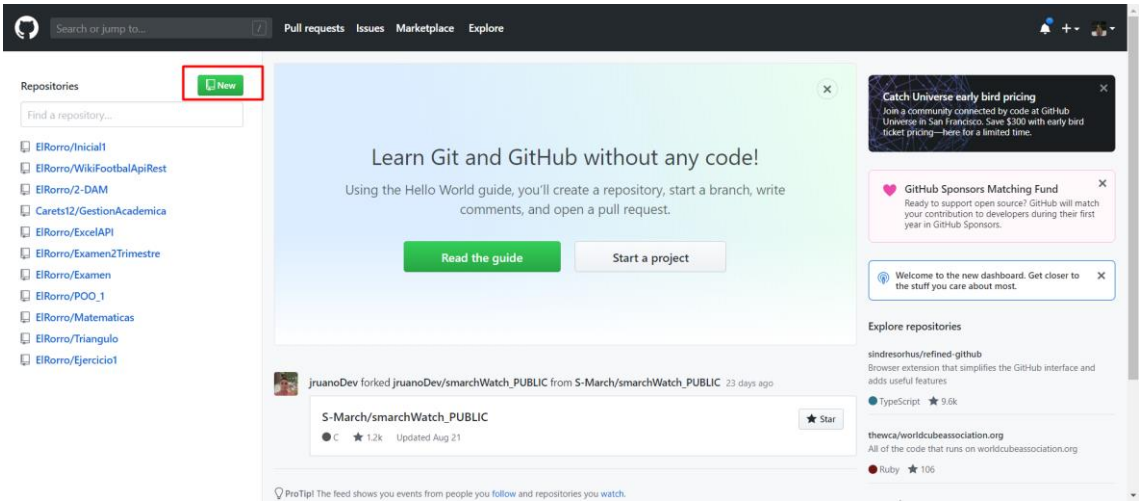


Una vez que nos hemos registrado y hemos realizado todos los pasos de registro, accederemos a la interfaz de git. Dónde encontraremos todos los proyectos asociados a nuestra cuenta, en primer lugar, no tendremos ninguno, pero en mi caso si que tendré por lo que seleccionaré en rojo dónde aparecen.

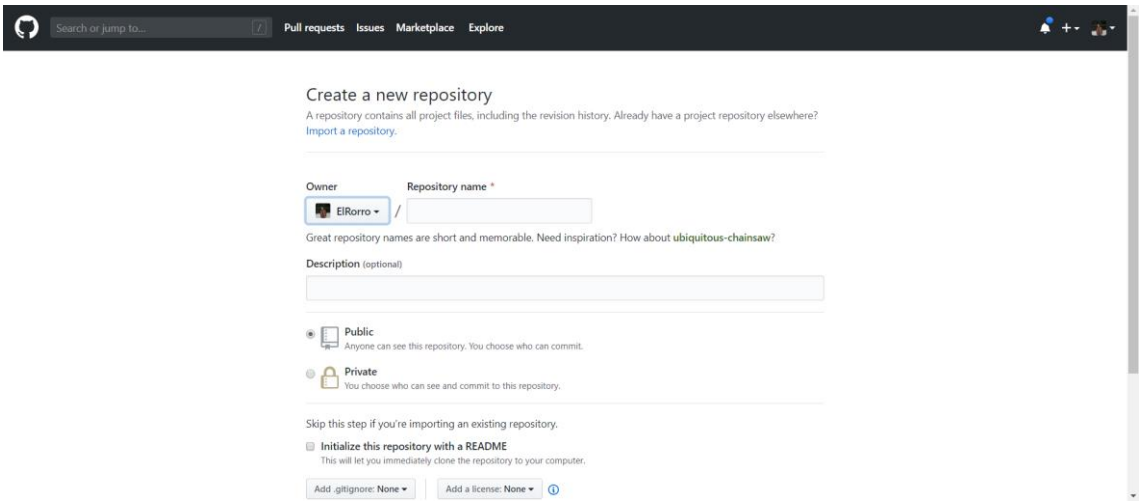


1.-b.- Creación nuevo repositorio

Ahora procedemos a crear el nuevo repositorio, para ello pulsamos el botón que mostraré a continuación.



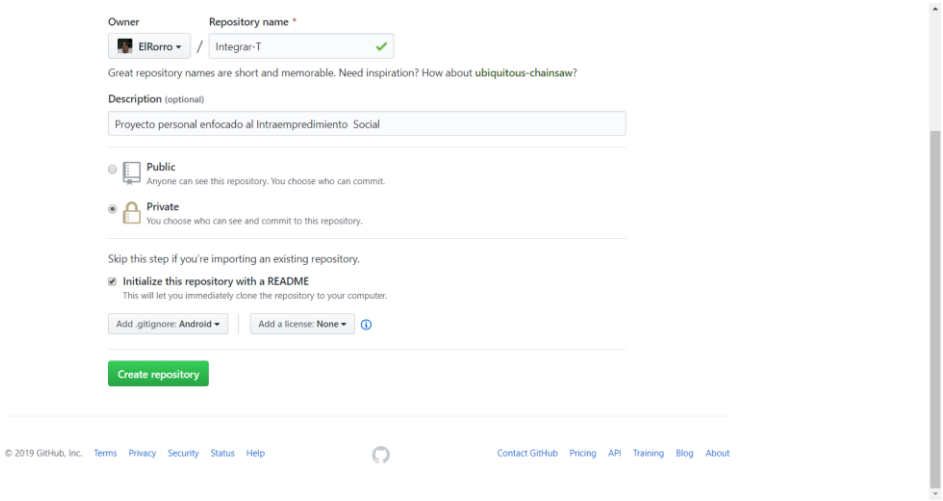
Nos aparecerá la siguiente pantalla:



Rellenamos los diferentes campos, como vemos algunos son opcionales, pero otros obligatorios (aquellos que tienen un *), según como queramos que sea el trabajo realizado en nuestro proyecto, el repositorio puedes ser público o privado.

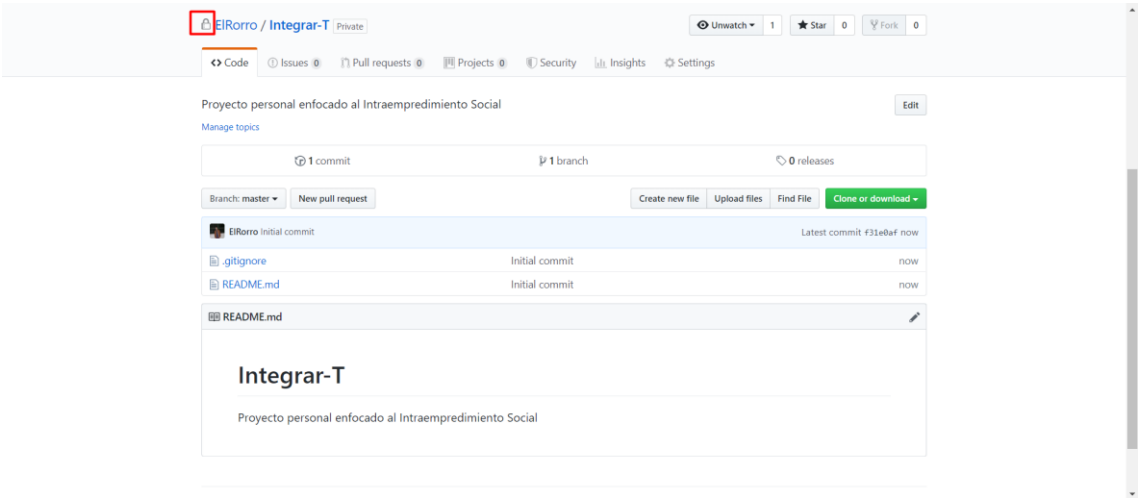
En mi caso, la creación de mi repositorio ha quedado de la siguiente manera:

- Podemos destacar dos puntos, el fichero **README**, es simplemente un fichero de información cuyo objetivo es facilitar una guía rápida a los que acaban de descubrir nuestra aplicación, API o librería de cómo empezar a usarla. Este fichero puede ser modificado una vez hemos creado el repositorio.
- El fichero **.ignore** he seleccionado Android, debido a que mi proyecto se va a realizar en ese lenguaje de programación el objetivo de este fichero es no subir ningún fichero del proyecto que no sea útil, es decir, todos aquellos ficheros basura que por ejemplo se crean cuando se realiza una ejecución del mismo. No obstante, una vez creado el proyecto podemos modificar el fichero para añadir o quitar las extensiones de los ficheros que son ignorados.

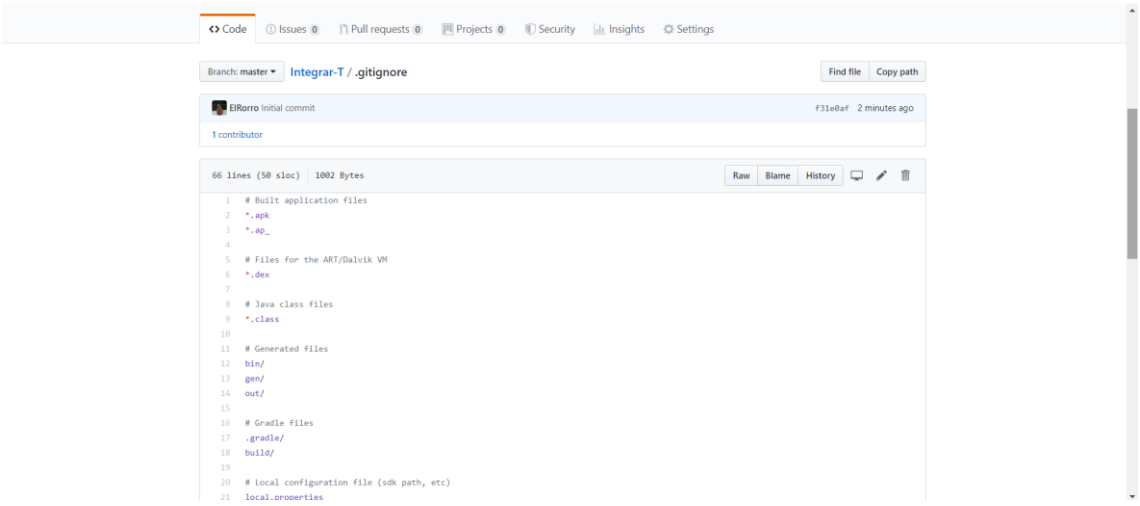


Una vez que tenemos todo completado pulsamos el botón verde **Create repository**.

Nos aparecerá la siguiente pantalla, como podemos ver al lado del nombre del repositorio sale un candado, esto quiere decir que nuestro repositorio es privado. Además, podemos ver que se han generado automáticamente los ficheros **.ignore** y **README**.



Contenido fichero **.ignore**, el contenido de este fichero podemos ver tantos ficheros como rutas, todo aquello que esté contenido aquí será ignorada a la hora de hacer una subida, por lo que si algo nos interesa subirlo a nuestro repositorio deberíamos quitar esa línea del **.ignore**.

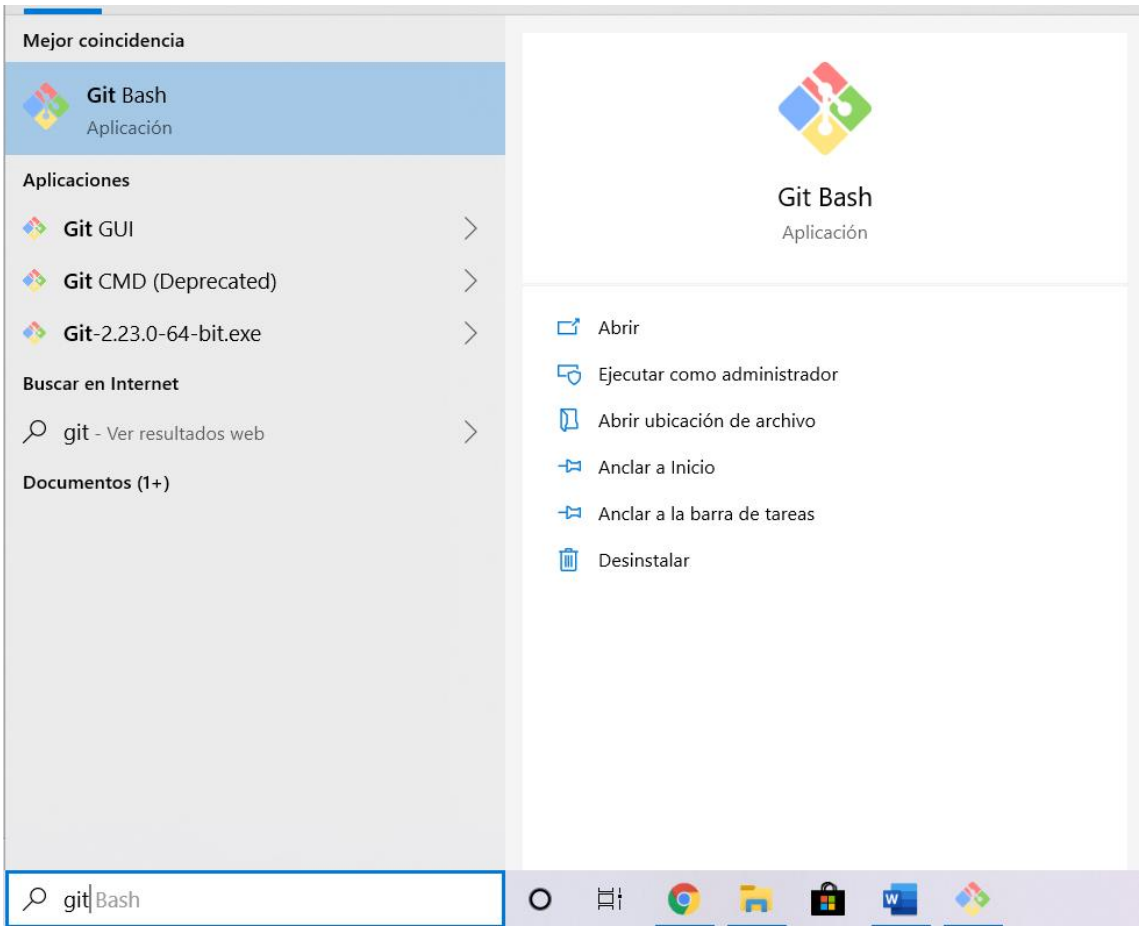


1.-c.- Herramienta GIT-SCM

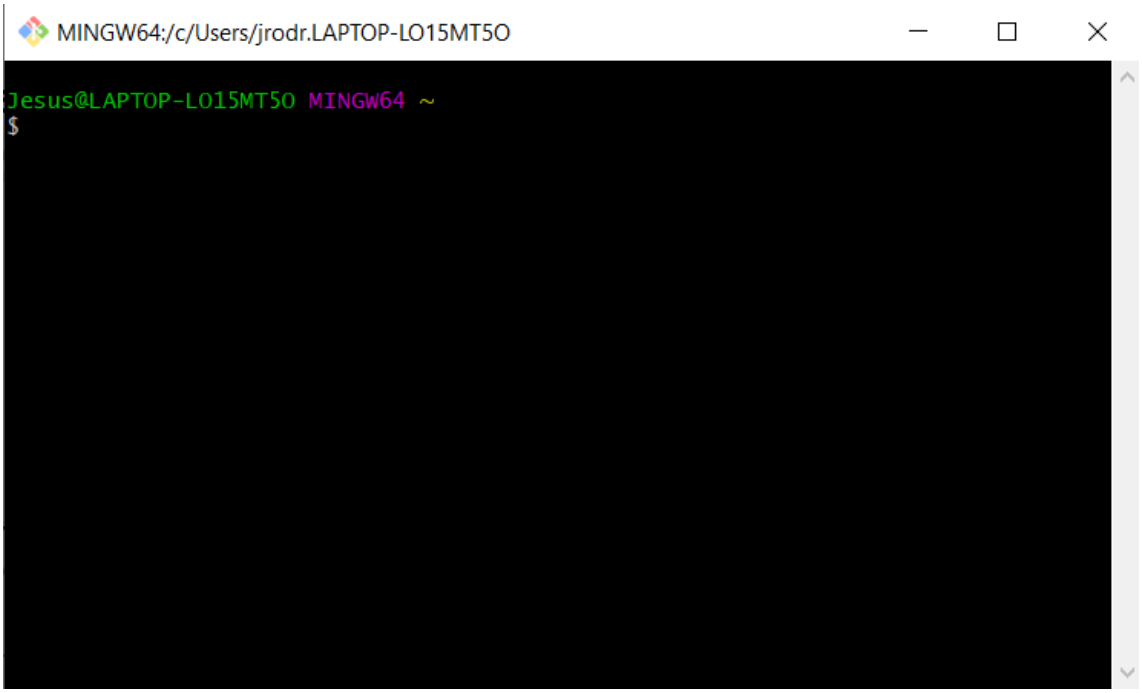
Instalamos la siguiente herramienta **git-scm**, para poder subir nuestro proyecto que tenemos en local al repositorio, la herramienta la podemos encontrar en la siguiente página web [https://git-scm.com.](https://git-scm.com)

1.-d.- Subir proyecto de local a repositorio

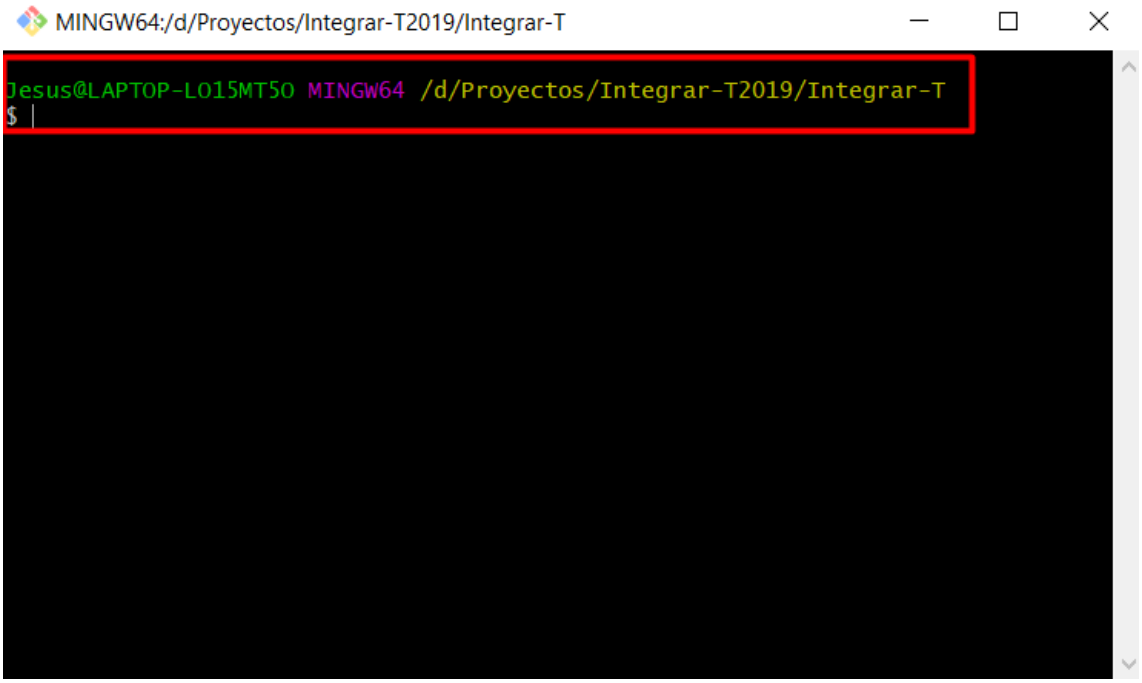
Abrimos la herramienta **git Bash** que hemos instalado anteriormente.



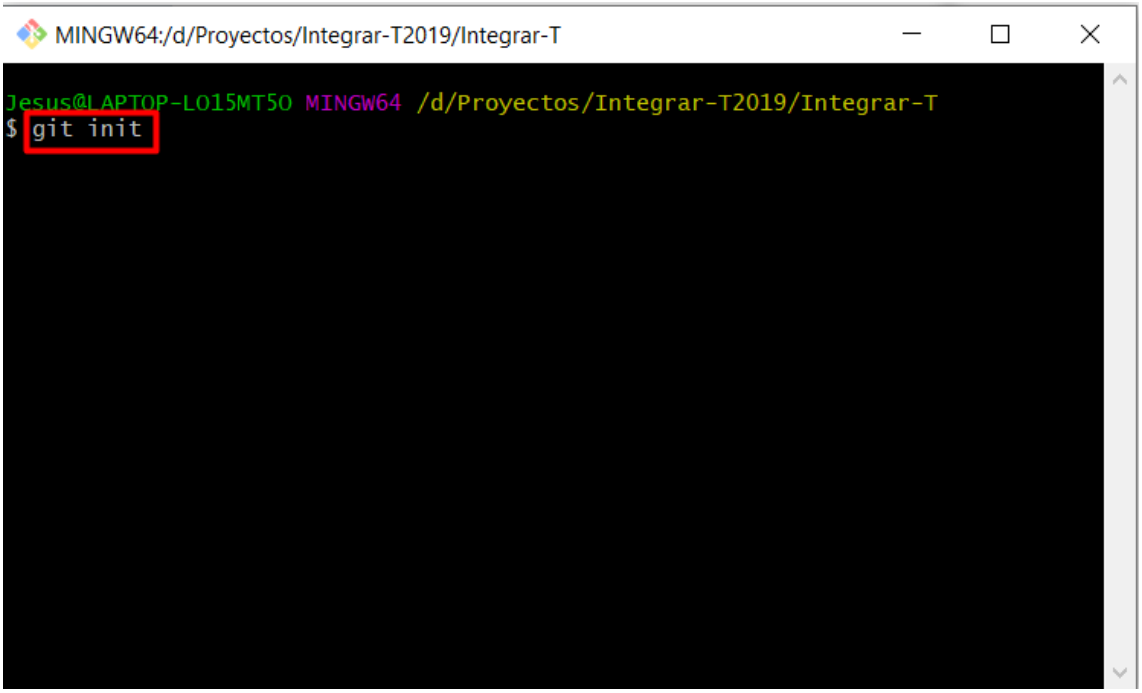
Se nos abrirá la siguiente terminal:



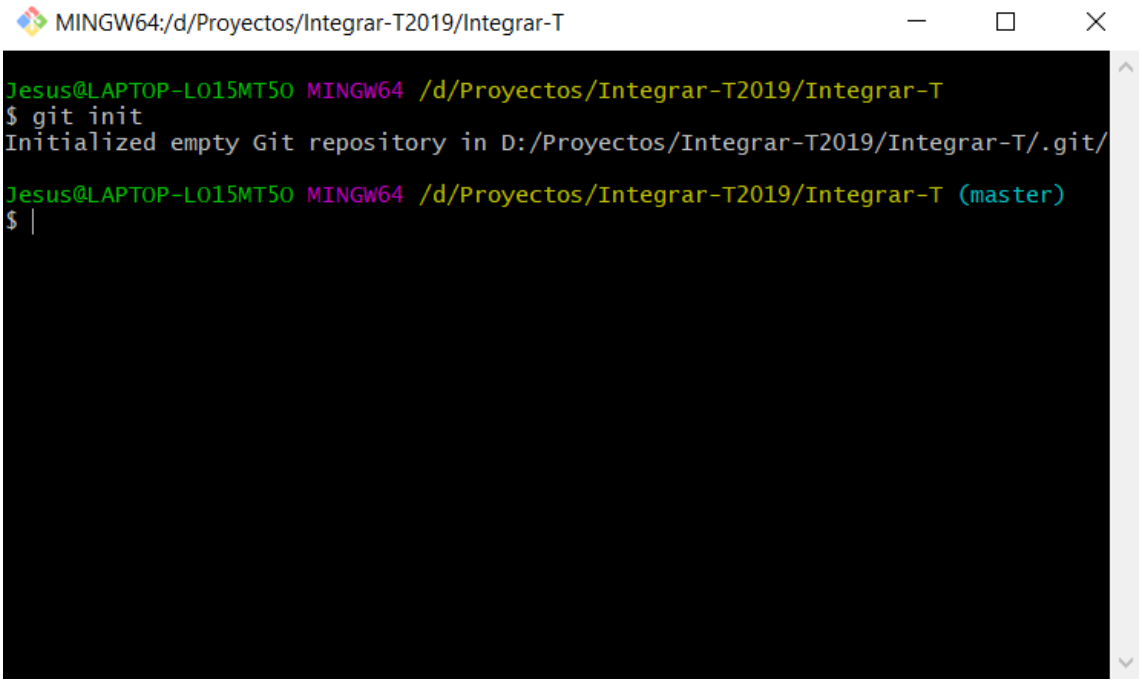
Accedemos a la ruta del proyecto que queremos subir mediante el comando **cd**.



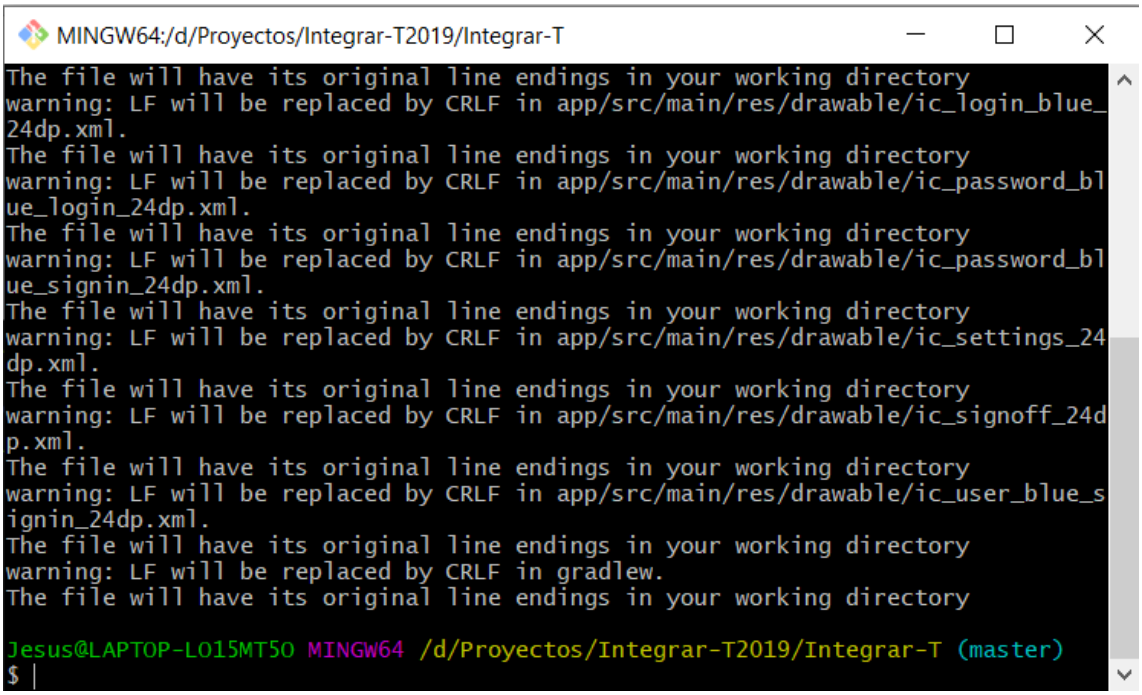
Una vez que nos encontramos en la ruta debemos inicializar el repositorio con el comando **git init**.



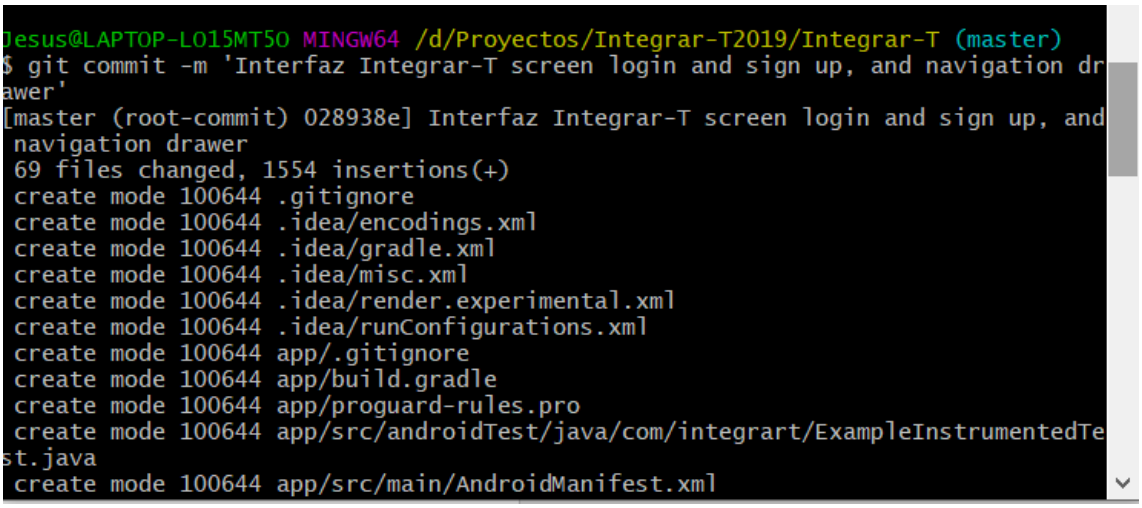
Nos debe aparecer que se ha inicializado el repositorio en la ruta que hemos especificado.



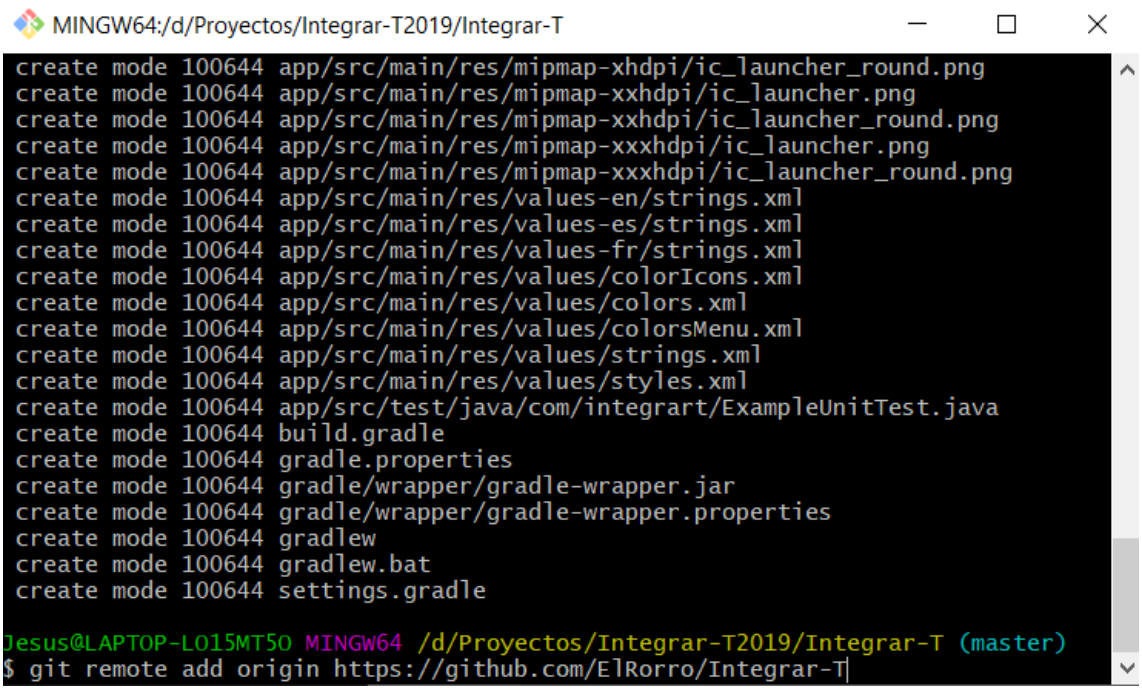
Ahora procedemos a la subida del proyecto para ello ejecutamos el comando **git add .**, este comando lo que realiza es seleccionar todos los ficheros del proyecto para prepararlos para la subida al repositorio si quitamos el punto y ponemos el nombre de un fichero solo realizará la preparación de subida de ese fichero.



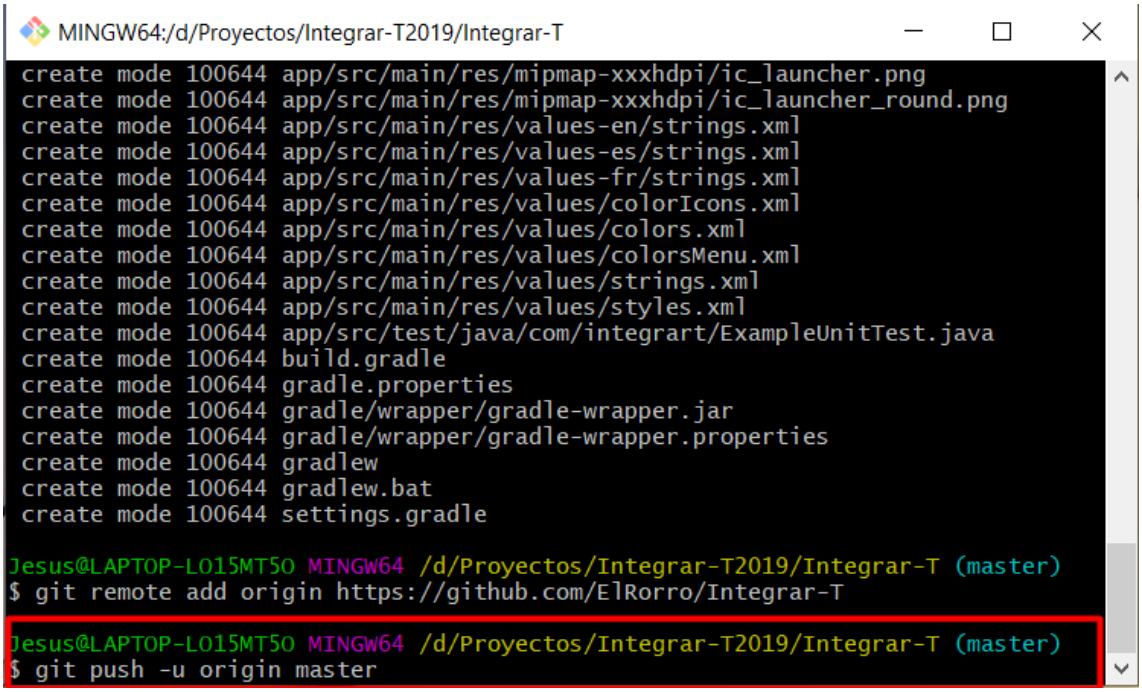
Ahora realizamos el commit añadiendo siempre un comentario para identificar lo que hemos subido. El comando es **git commit – m 'mensaje a especificar'**.



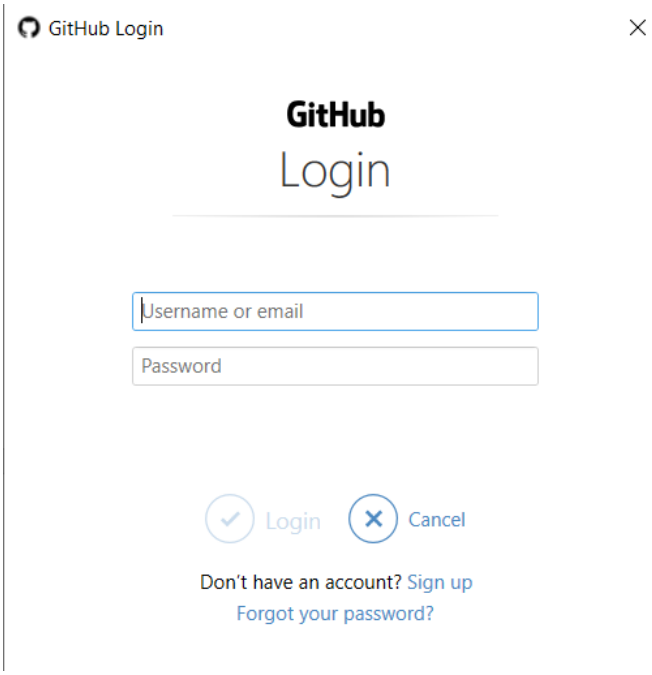
Ya hemos generado el commit. Ahora debemos asociar el proyecto con la ruta del repositorio que tenemos en github, es decir, al repositorio dónde queremos subir el código, para ello ejecutamos el siguiente comando. git remote add origin <ruta de nuestro repositorio> en mi caso quedaría de la siguiente manera **git remote add origin https://github.com/ElRorro/Integrar-T**



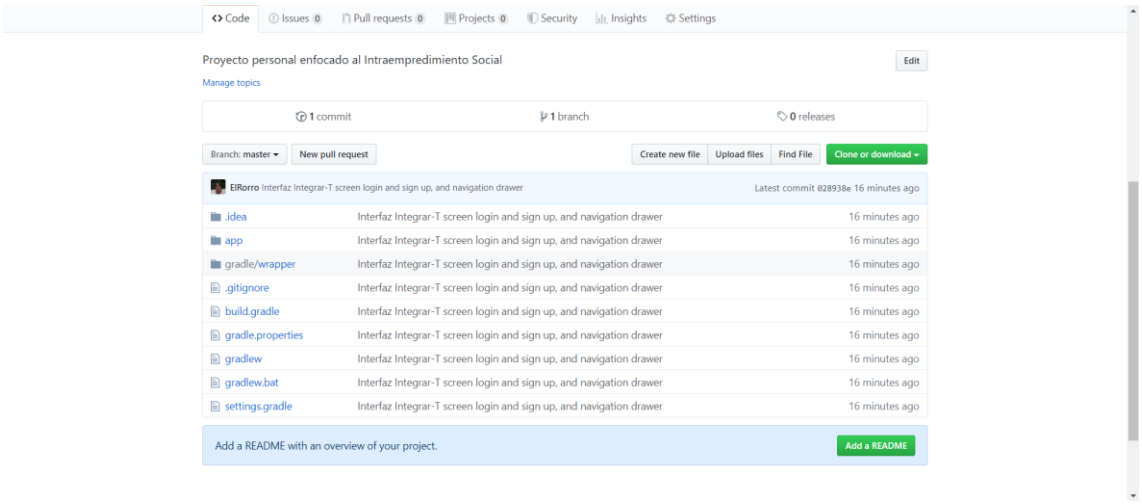
Ahora procedemos a subir el proyecto a nuestro repositorio que hemos asociado, con el siguiente comando **git push -u origin master**, estamos indicando que lo suba a la rama **master**, en caso de tener otra rama, podriamos indicar esa rama y se subiría ahí.



Nos aparecerá la siguiente pantalla, para que metamos nuestras credenciales de git, para verificar que somos nosotros los que estamos realizando la subida.

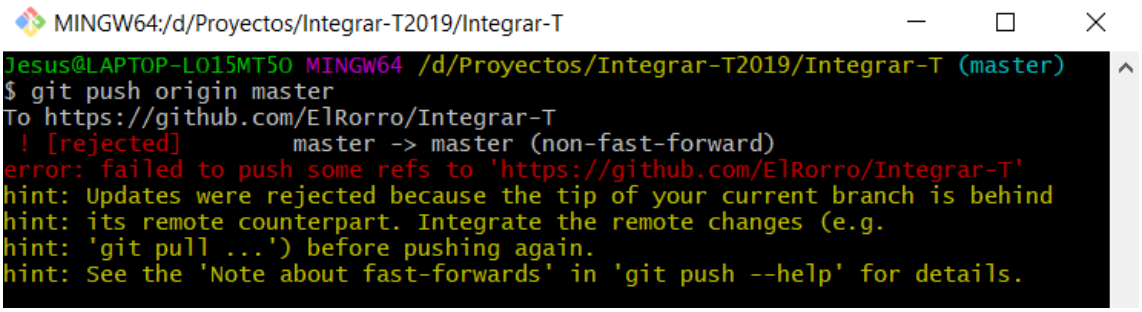


Una vez que hemos realizado el comando anterior nos vamos a nuestro repositorio de git y veremos como se han subido los ficheros que tenemos en local.



¡! IMPORTANTE ¡!
Debemos tener en cuenta que, si nos pide primeramente realizar un pull del repositorio, debemos de ejecutar el comando git pull o git pull –rebase.

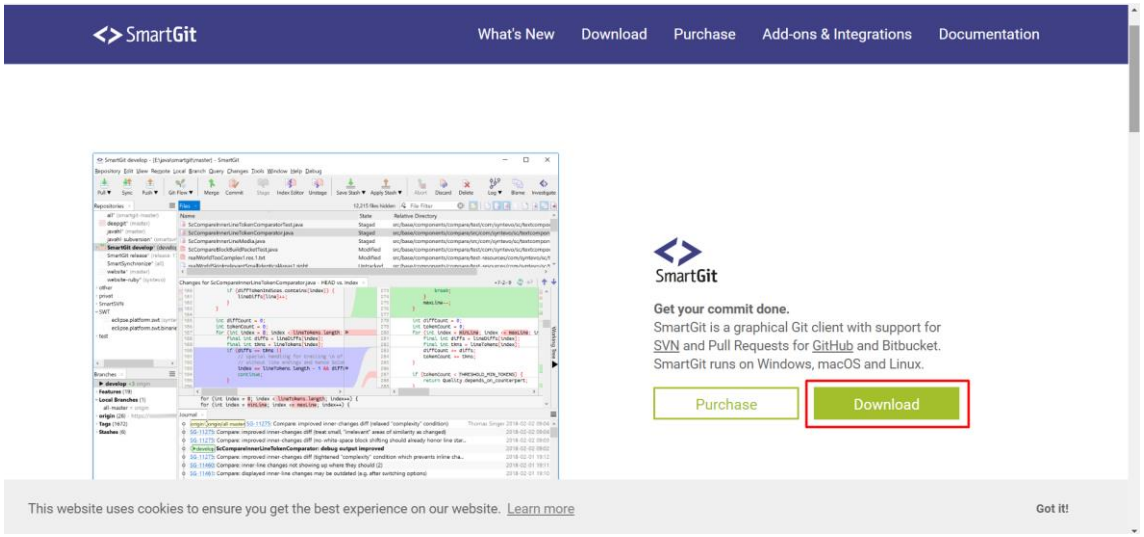
Además, uno de los errores comunes que puede aparecer es lo siguiente:

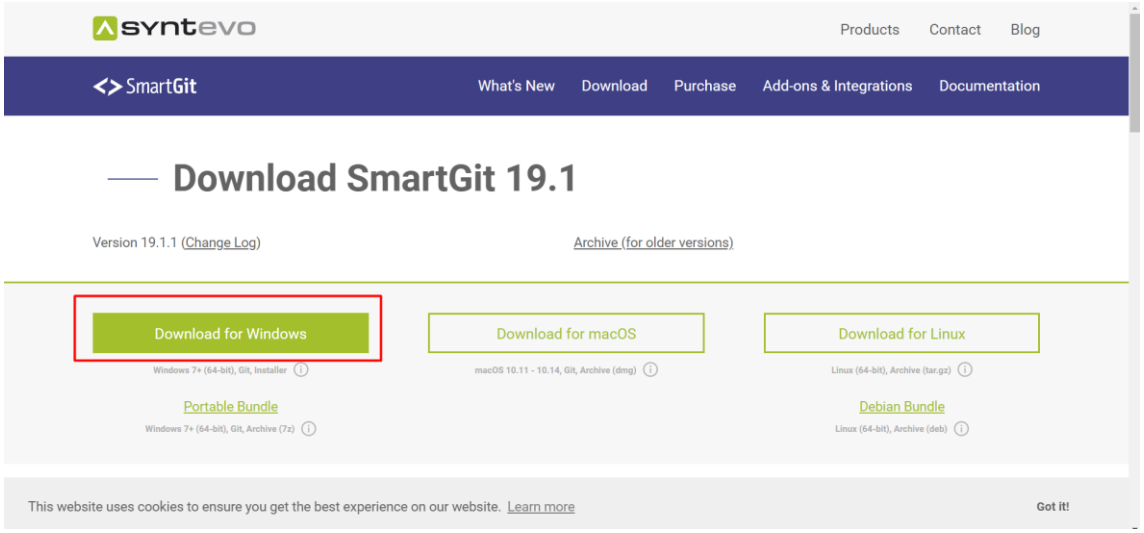


Para solucionarlo debemos ejecutar el siguiente comando git push -f origin master.

2. Instalación SmartGit

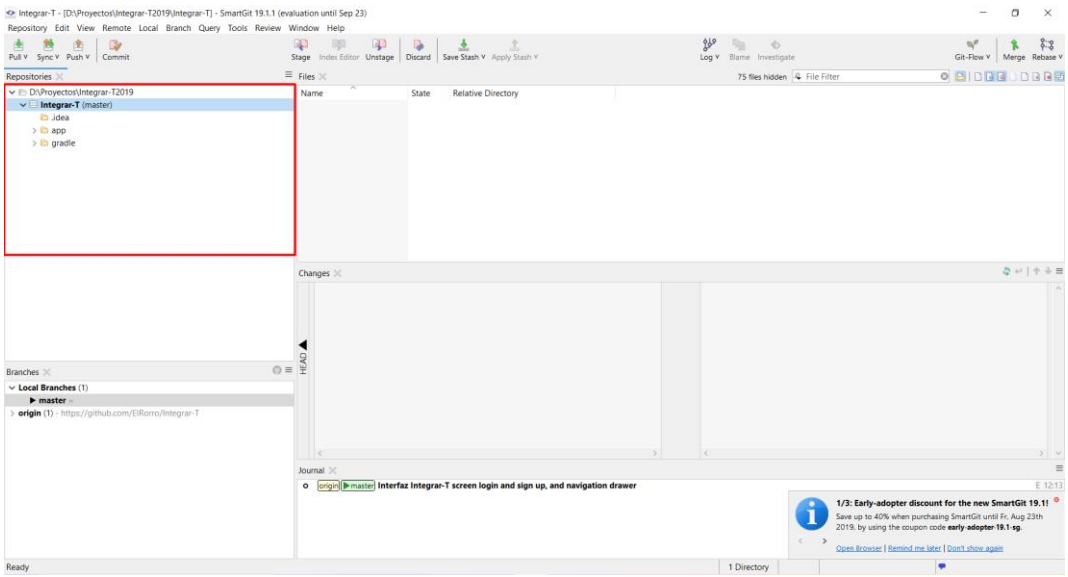
Accedemos a la página de SmartGit para descargar la herramienta
https://www.syntevo.com/smartgit/





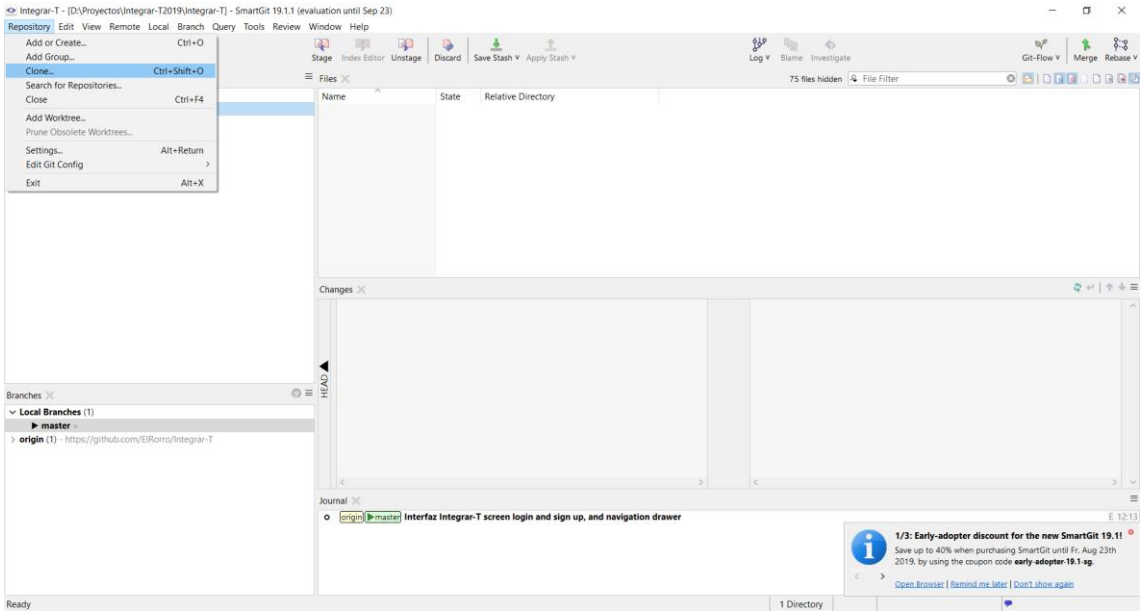
Una vez que lo hemos descargado y hemos realizado la instalación, accedemos a la interfaz de la herramienta. Es la siguiente:

Como ya tenemos nuestro proyecto sincronizado al repositorio de git nos va a aparecer en la herramienta de SmartGit como veremos a continuación:

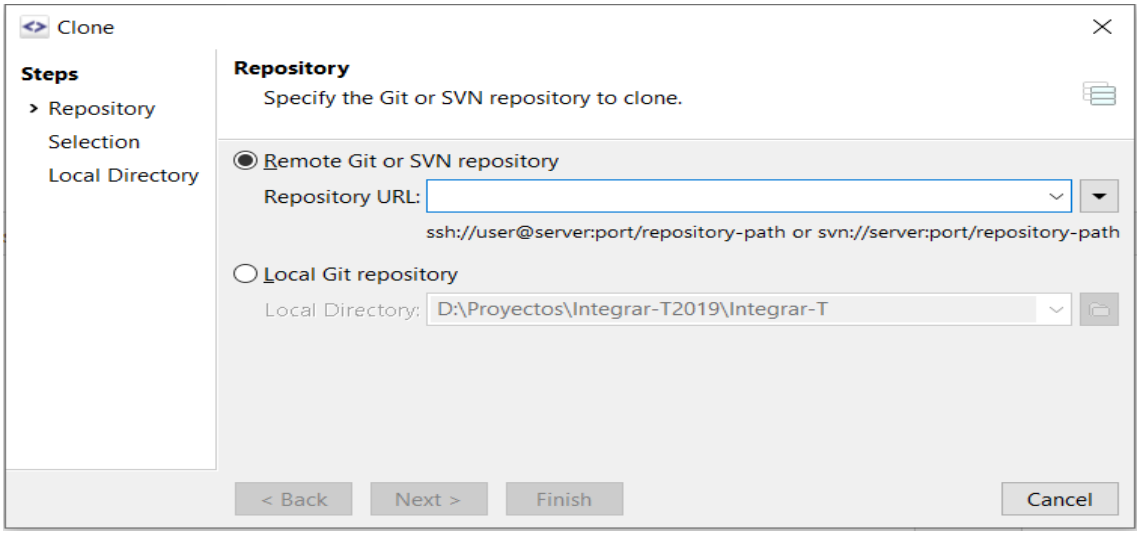


También podemos realizar desde esta herramienta los pasos que hemos realizado con la herramienta de git-scm, a continuación os voy a explicar como se hace.

Pulsamos en la pestaña repository→clone



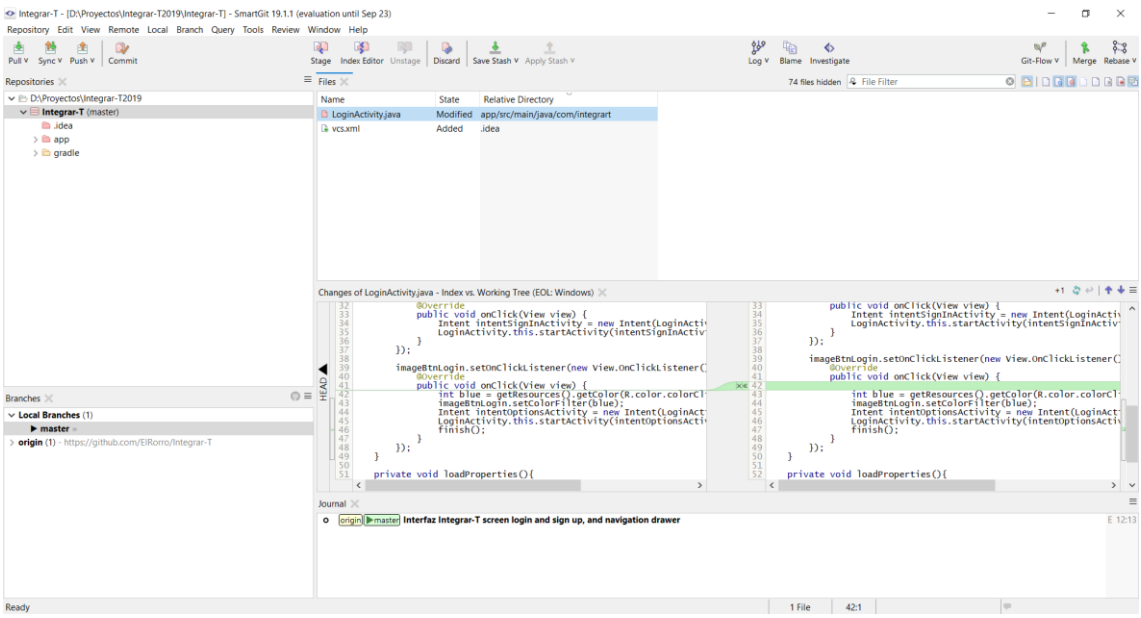
Nos aparece la siguiente ventana, dónde indicaremos si nuestro repositorio a clonar es **.SVN** o **.GIT**. Además, podemos añadir un proyecto que ya tengamos en local como ha pasado en nuestro caso con el proyecto que ya teníamos clonado anteriormente con la herramienta git-scm.



En este caso, para ahorrarnos los pasos realizados con la herramienta **git-scm** sería un repositorio en remoto dónde indicaríamos la url de nuestro repositorio. Una vez que hemos puesto la url, pulsaremos en siguiente, y ya lo clonamos y nos aparecerá en la parte izquierda de la interfaz como podemos ver en fotos anteriores.

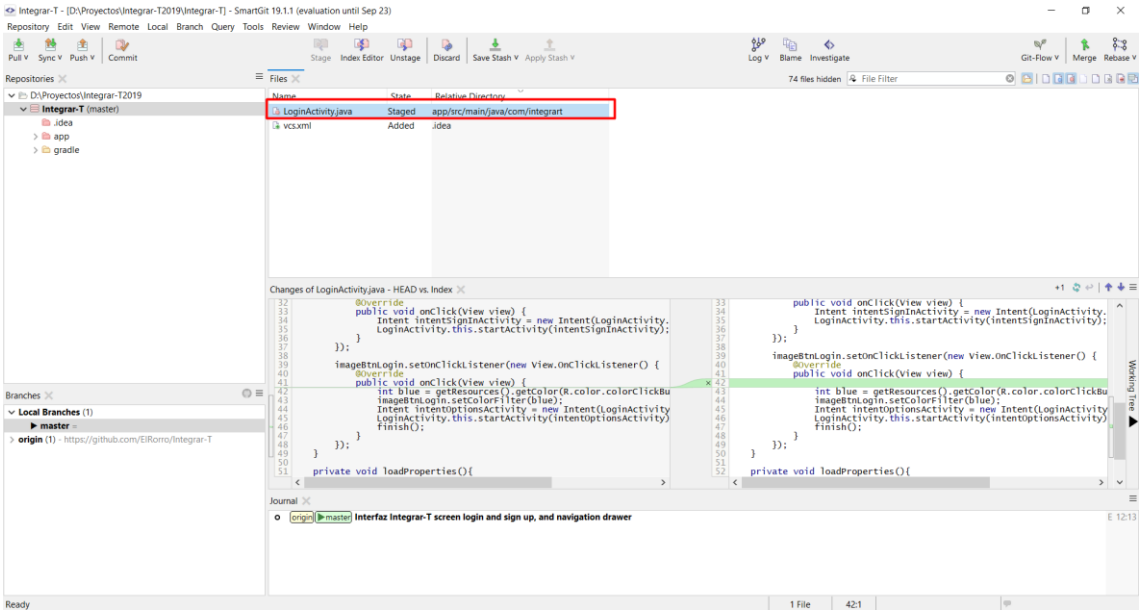
Cuando realicemos cambios en algún fichero, nos aparecerá de la siguiente manera en la interfaz, con un color rojo el nombre del fichero que hemos modificado.

Podemos ver que el archivo seleccionado es el que hemos modificado, la parte de la izquierda es el código que tenemos en el repositorio y el de la derecha el que tenemos en local, es decir, el que hemos modificado. Además se puede ver la línea marcada en verde que es la modificada.

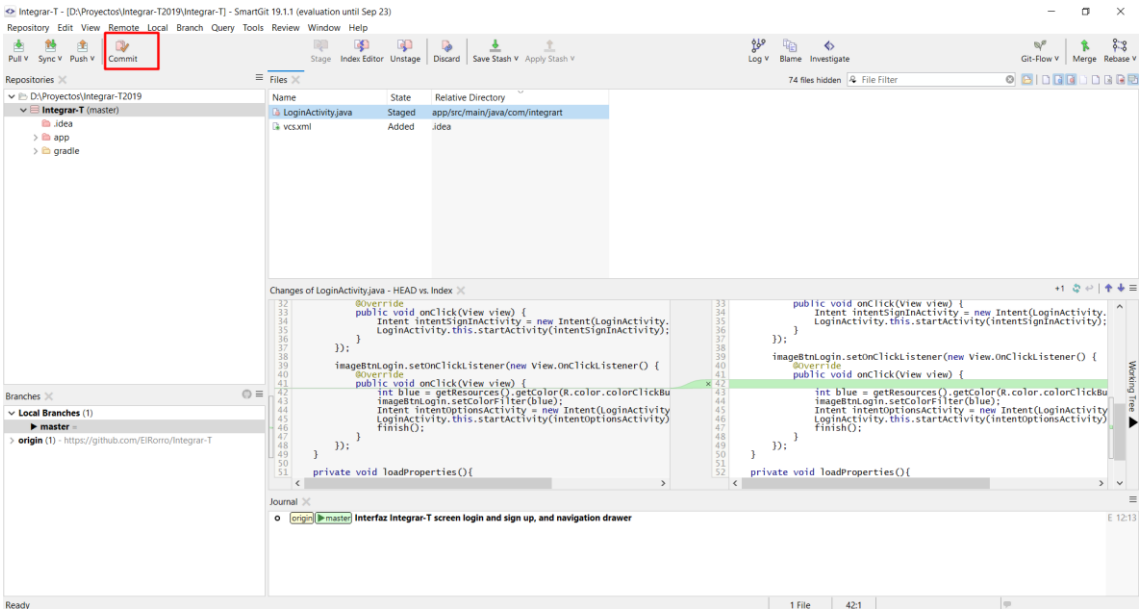


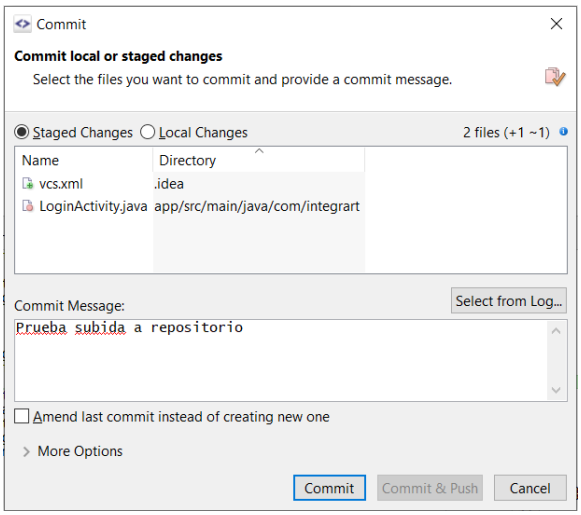
Para subir el fichero o los ficheros modificados debemos de realizar lo siguiente:

1. Seleccionamos los ficheros que queremos subir.
2. Pulsamos el botón **stage**. Cambiará de color rojo el fichero a fichero blanco con un punto rojo, como veremos a continuación.

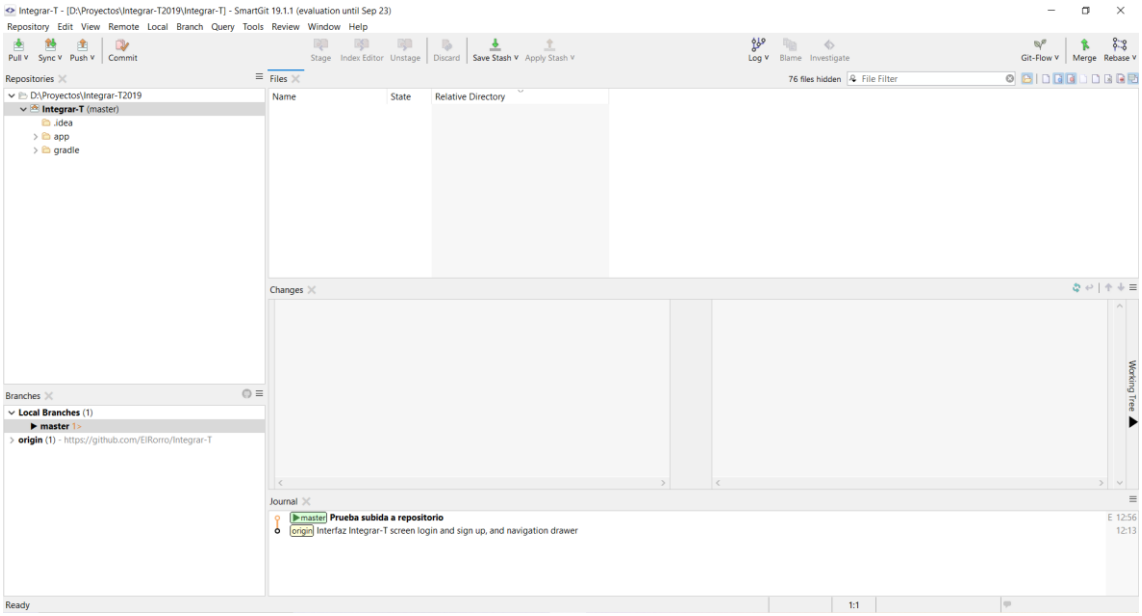


3. Pulsamos el botón de commit, para preparar el fichero de la subida y siempre especificamos un mensaje para identificar que es lo que estamos subiendo.



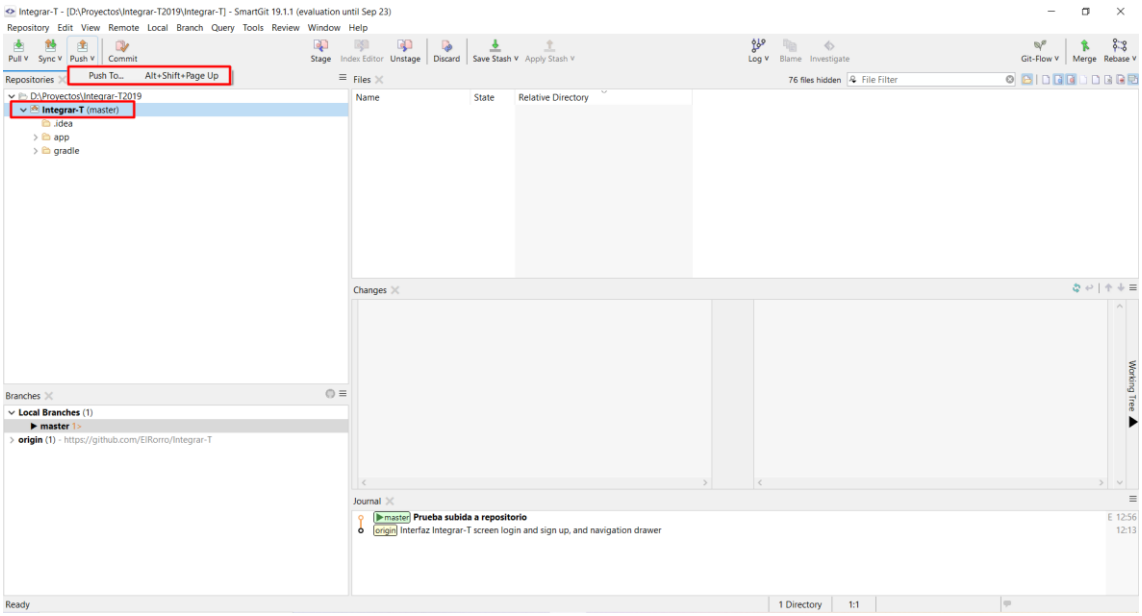


Pulsamos en **commit** y veremos como desaparecen los ficheros, debido a que ya sido asignados para ser subidos.

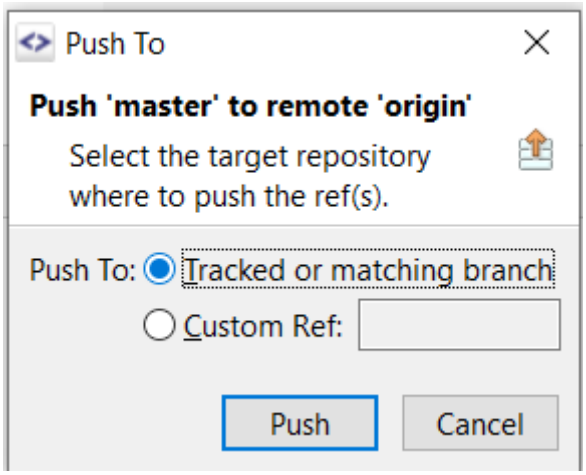


4. Por último, pulsamos el botón push, para subirlos a la rama que le especifiquemos en nuestro caso solo existe la **master** por lo que lo subiremos a esa.

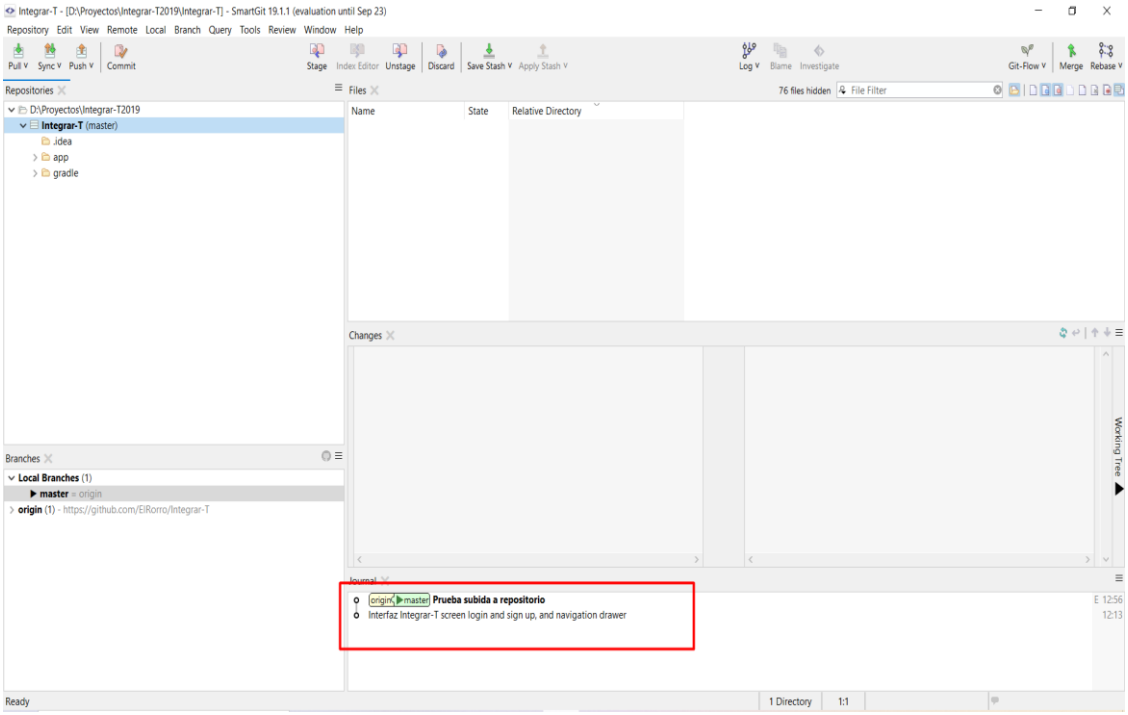
   Mantenemos pulsado el botón, para que nos salga la opción de push to… en caso de que no nos salga una flecha. Además, podemos ver como en el propio proyecto nos sale una flecha naranja que nos indica que tenemos ficheros para subir. IMPORTANTE: Debemos tener cuidado a la hora de estar trabajando con compañeros, ya que si ellos han realizado una subida nosotros en local no estaremos actualizados para ello debemos de realizar un pull y además la propia herramienta de SmartGit nos lo indicara con una flecha verde en la carpeta del proyecto.



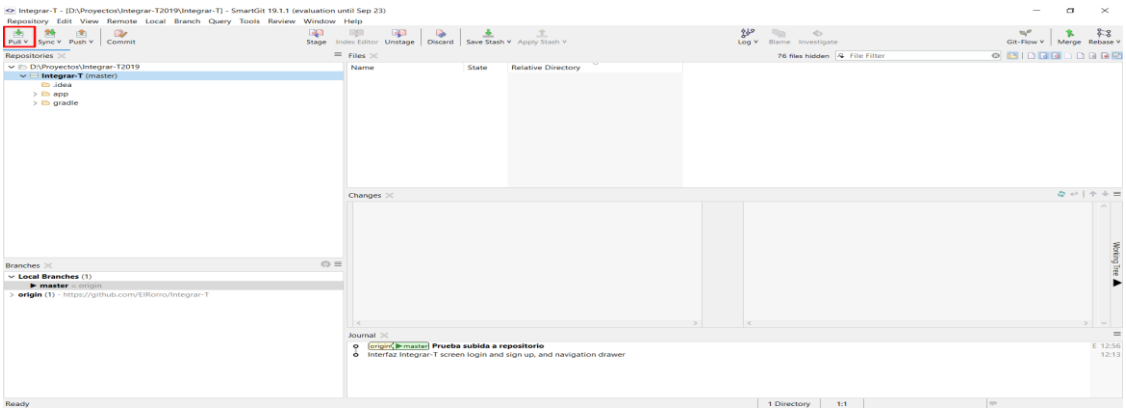
En esta pestaña indicaríamos la referencia de la rama a la que queremos subir, pero como solo tenemos la **master**, seleccionamos la opción que hay seleccionada.



Cuando hemos realizado la subida podemos ver en la herramienta de SmartGit en que commit nos encontramos y como se ha realizado la subida. Podemos ver que nos encontramos en **prueba subida a repositorio**.

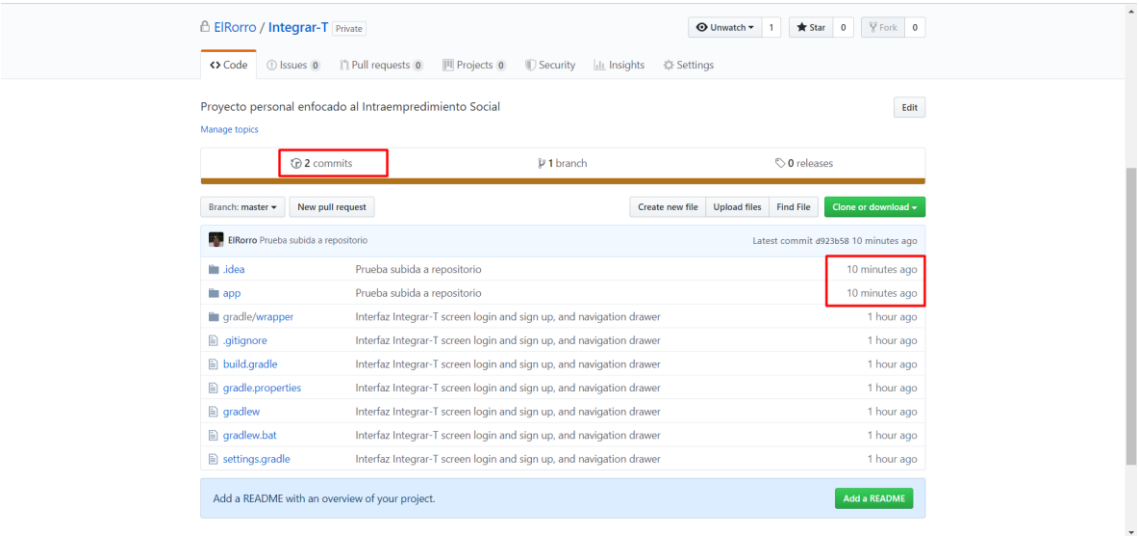


5. Podemos hacer una actualización de nuestro local respecto a lo que hay en el repositorio si lo tenemos desactualizado por que se ha realizado alguna subida, debemos realizar un **pull**.



Por último, nos vamos a nuestro repositorio de git para ver si se ha realizado la subida correctamente.

Vemos que se ha realizado un cambio hace 10 minutos, pero para comprobarlo con mas precisión, pulsamos en **commits** y vemos que último **commit** se ha realizado.



Y vemos que el último **commit** y es el que hemos realizado por lo que se ha hecho correctamente.