**Integrantes:**

* **Asi, Milagros**
* **González Toraf, Abril**
* **Moller, Cristian**
* **Silva, Florencia**
* **Silva, Micaela**

Git-github

Tecnologia de la informacion en las organizaciones

tudai - facultad de ciencias exactas - unicen

TRABAJO PRACTICO ESPECIAL

2019

INDICE

[UNA PEQUEÑA INTRODUCCION 1](#_Toc22026232)

[¿Qué es GitHub? 1](#_Toc22026233)

[¿Para qué sirve? 1](#_Toc22026234)

[¿Qué herramientas proporciona? 1](#_Toc22026235)

[¿Qué uso le daremos? 1](#_Toc22026236)

[CREACION DEL REPOSITORIO 2](#_Toc22026237)

[COMANDOS: GIT STATUS, GIT ADD, GIT COMMIT, GIT PUSH 3](#_Toc22026238)

[COMANDO: GIT LOG 4](#_Toc22026239)

[COMANDOS: GIT BRANCH, GIT CHECKOUT 5](#_Toc22026240)

[ISSUES Y MILESTONES 6](#_Toc22026241)

[MERGE PULL REQUEST 7](#_Toc22026242)

[CONCLUSION 8](#_Toc22026243)

[Link repositorio GitHub 8](#_Toc22026244)

# UNA PEQUEÑA INTRODUCCION

### ¿Qué es GitHub?

GitHub es una plataforma de desarrollo colaborativo de software para alojar proyectos utilizando el sistema de control de versiones Git.

### ¿Para qué sirve?

GitHub aloja tu repositorio de código y te brinda herramientas muy útiles para el trabajo en equipo, dentro de un proyecto.

### ¿Qué herramientas proporciona?

Una wiki para el mantenimiento de las distintas versiones de las páginas.

Un sistema de seguimiento de problemas que permiten a los miembros de tu equipo detallar un problema con tu software o una sugerencia que deseen hacer.

Una herramienta de revisión de código, donde se pueden añadir anotaciones en cualquier punto de un fichero y debatir sobre determinados cambios realizados en un commit específico.

Un visor de ramas donde se pueden comparar los progresos realizados en las distintas ramas de nuestro repositorio.

### ¿Qué uso le daremos?

Crearemos una cuenta gratuita y comenzaremos a crear repositorios de código, para lograr que el grupo trabaje colaborativamente. Intentando fortalecer nuestros vínculos y comunicación.

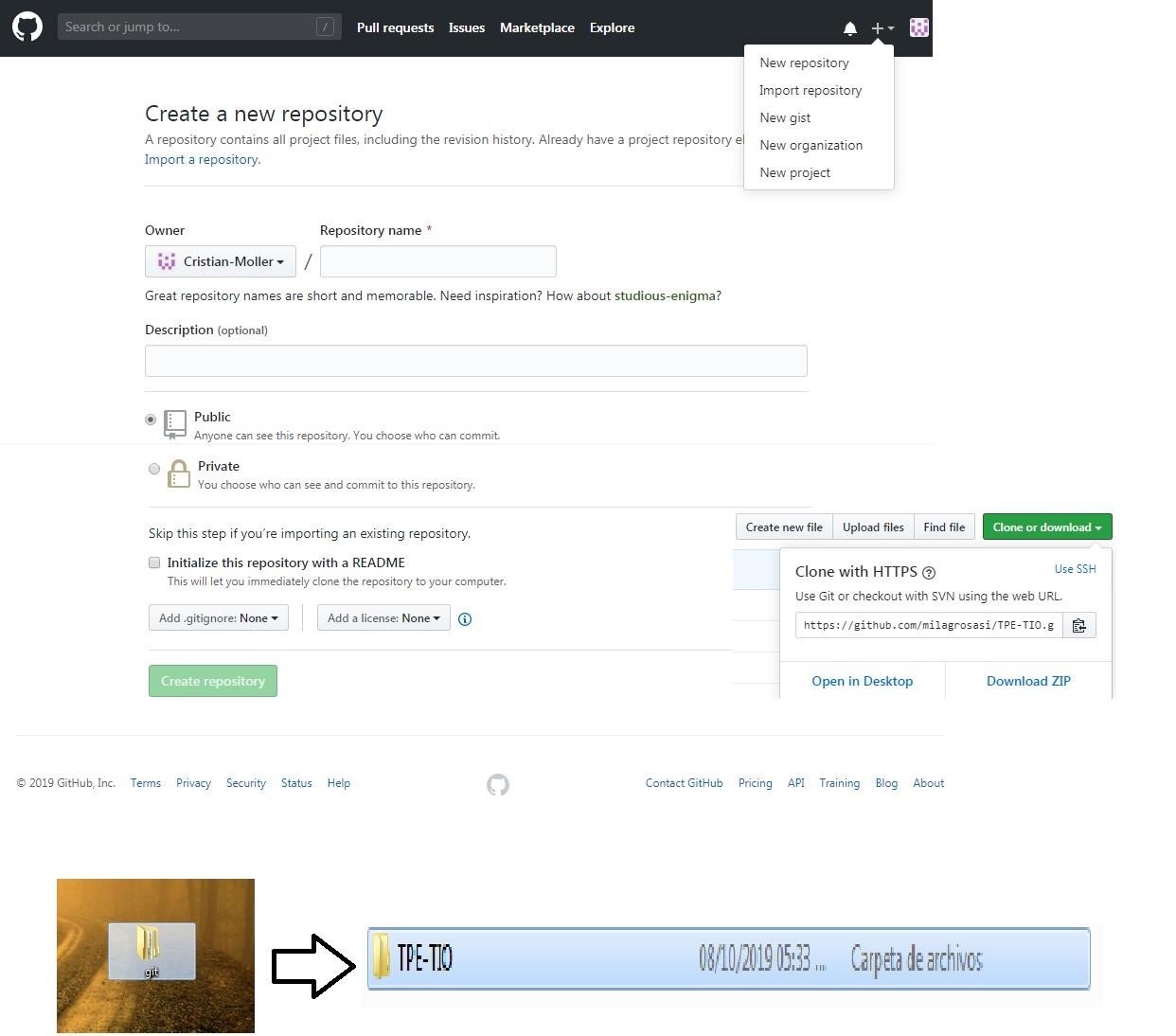
## CREACION DEL REPOSITORIO

1º Iniciamos el TP con la creación del Repositorio Nuevo en la plataforma GITHUB (una vez que teníamos creados nuestros respectivos usuarios).

Le introducimos nombre al repositorio, la creación fue pública, sin extensión y sin licencia.

2º En nuestro escritorio o donde quisiéramos clonar el repositorio, creamos la Carpeta GIT, abrimos la consola de GIT, puede ser por ejemplo Git Bash o powershell en Windows 10.

3º Luego copiamos el enlace para Clonar dicho repositorio. Con el comando git clone \*URL del repositorio\*, en la consola, clonamos el repositorio dentro de dicha carpeta, para poder comenzar a trabajar en la misma. Este comando además de cumplir la función de “copiarnos” el repositorio en nuestra pc también copia las ramas que estén creadas para poder trabajar en ellas en caso de que sea necesario.



## COMANDOS: GIT STATUS, GIT ADD, GIT COMMIT, GIT PUSH

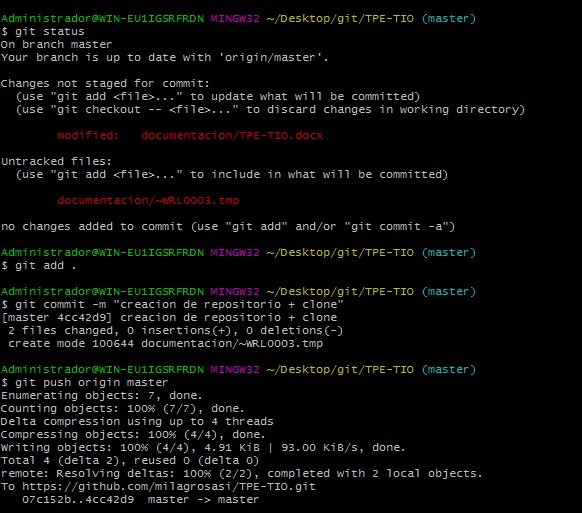
Al iniciar la documentación de los procedimientos, se creó la carpeta documentación, por lo que se debió hacer en GIT BASH:

1º Introducir el comando git status: el cual nos indica en rojo las carpetas modificadas.

2º Con git add . : agregamos los archivos.

3º Con git commit -m “mensaje” confirmamos los cambios con el detalle entre comillas.

4º Por último con git push origin \*nombre\_rama\* (en este caso es master): subimos los cambios realizados a nuestro repositorio remoto.

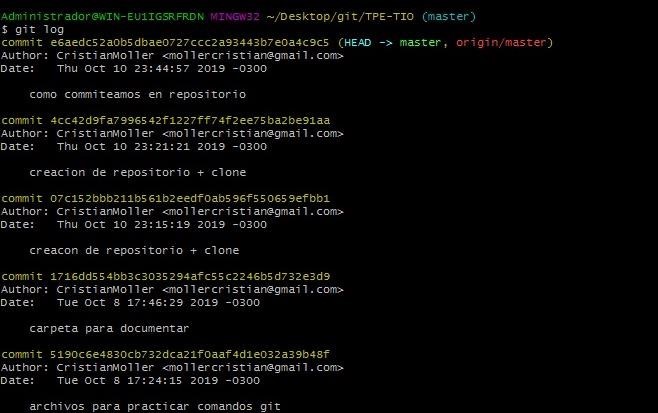


## COMANDO: GIT LOG

El mismo nos lista los diferentes commits realizados sobre el repositorio creado.

Cada commit realizado tendrá un id, un Autor y una fecha de modificación.

Estos datos identifican el commit y nos permitirán volver un cambio en el caso de ser necesario.



## COMANDOS: GIT BRANCH, GIT CHECKOUT

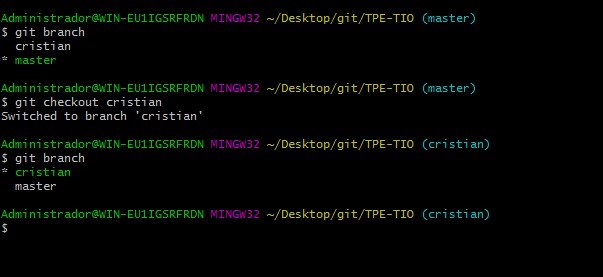
El comando git checkout -b “nombre\_rama” creará una nueva rama y nos posicionará directamente en ella para poder trabajar en párelo a la rama **master**, para dejar esta última como versión estable en el proyecto.

Para poder cambiar de rama utilizaremos el comando git checkout “nombre\_rama”.

Luego se puede utilizar el comando git branch para verificar el correcto cambio de rama, este comando listará las ramas del proyecto y mostrará en verde y con un asterisco en la rama que estamos ubicados.

Una vez realizados los cambios guardaremos y confirmaremos los cambios realizados, luego los subiremos al proyecto desde esta nueva rama.

Las ramas son muy útiles para trabajo en equipo y simultaneo, donde cada usuario tendrá su propia rama y podrá trabajar sin riesgo de entorpecer el trabajo de sus compañeros.

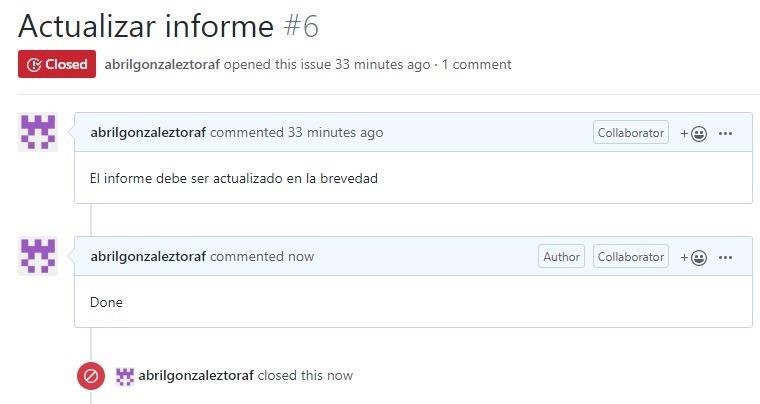


## ISSUES Y MILESTONES



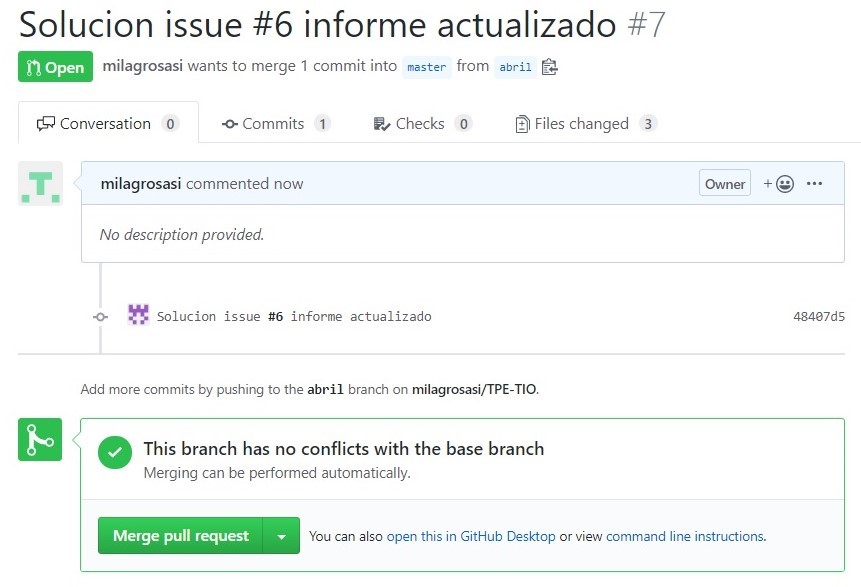
Las issues son “publicaciones” que se hacen dentro del repositorio para mostrar problemas o solicitar que los miembros del equipo ayuden con a resolverlas. Estas se crean dentro de la web y quedan abiertas hasta que se encuentra una solución, luego la issue se puede cerrar.

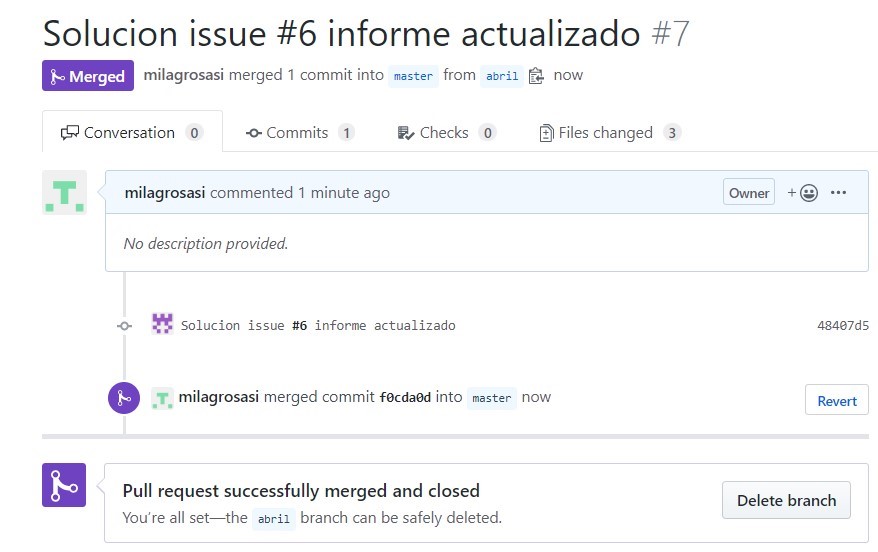
Las Milestone son “cajas” contenedoras de issues donde se pueden poner las issue que estén relacionadas al mismo tema.



## MERGE PULL REQUEST

Para hacer el Merge fuimos a la página con la rama ya creada y commiteada y desde la misma hicimos el pull request. Una vez que git verifica que el código a mergear no genera conflictos con el código que hay en master, te deja hacer el merge y el código queda todo unido en master.





## CONCLUSION

Este trabajo nos sirvió para poder conocer la herramienta Git - GitHub y aprender a manejarla, crear repositorios o trabajar con repositorios ya existentes. La herramienta es muy útil a la hora de continuar con el proyecto, ya que por cuestiones de tiempo o espacio no siempre es posible estar todos reunidos en un mismo lugar físico; y uno puede seguir trabajando en forma colaborativa, simplemente bajando las actualizaciones, de los avances ya realizados por nuestros compañeros.

## Link repositorio GitHub

<https://github.com/milagrosasi/TPE-TIO/>