**Metodología de la Investigación**

**Actividad:**

**1- Realizar cuestionario en el campus de 21hs, a 23 hs.**

**2- En base a lo visto realizar las actividades hechas en la clase de hoy**

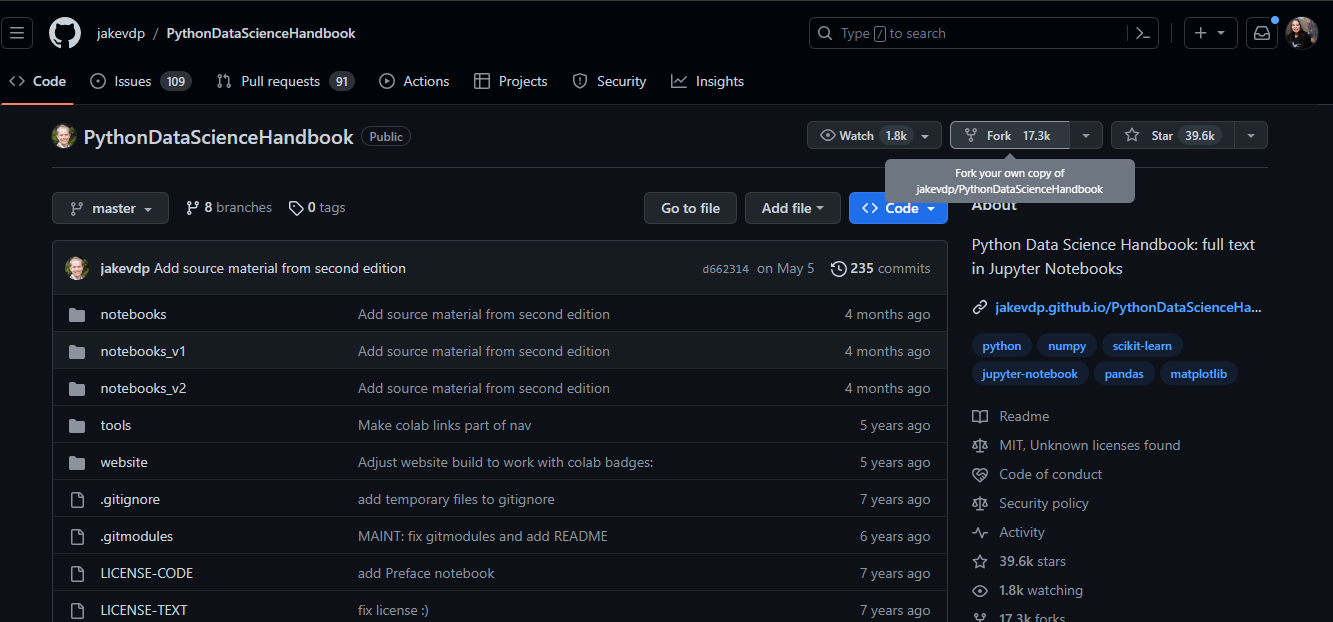
**3- Enviar capturas por grupo (1 una por cada punto) con nombre del grupo y alumnos que participaron.**

**Fechas de entrega:19/09**

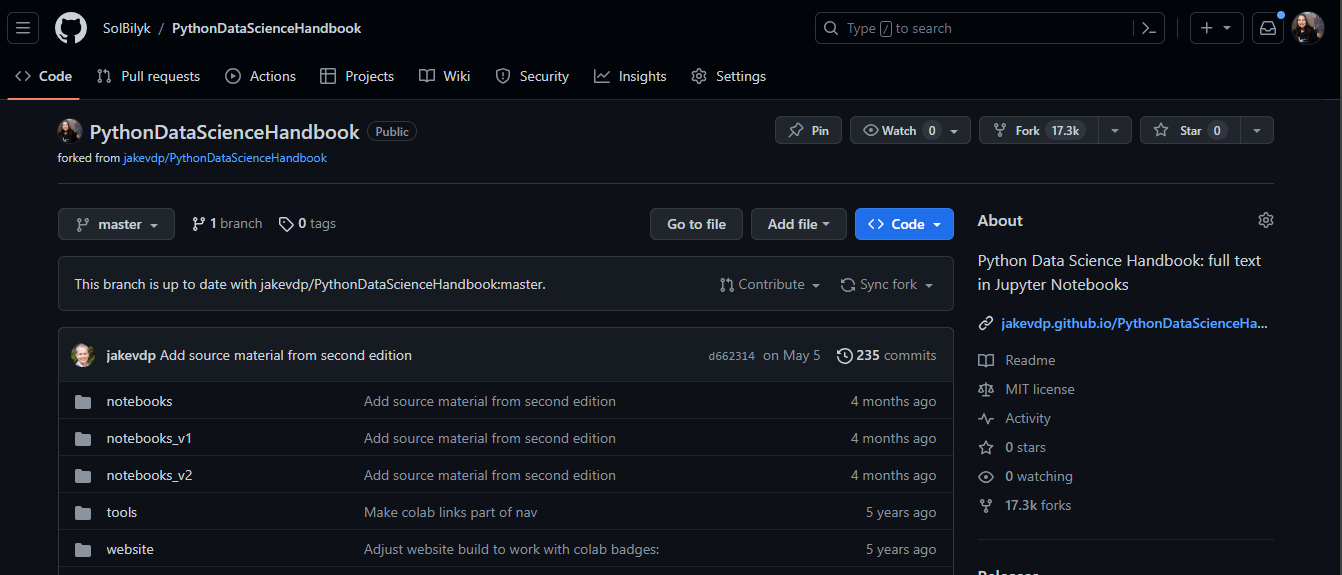
**FORK**

Cuando hablamos de fork nos estamos refiriendo a realizar una copia de un repositorio público y que podemos copiar y trabajar en nuestra cuenta de github.

Seleccionamos un archivo abierto y hacemos click en el botón Fork para copiar todos los archivos a nuestra cuenta de github.

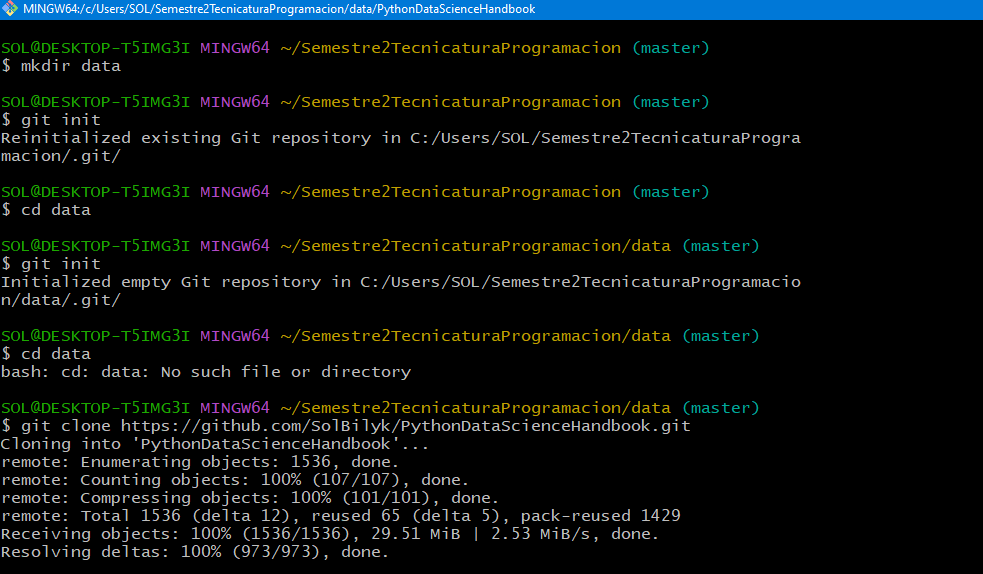


Copia del proyecto en mi cuenta.

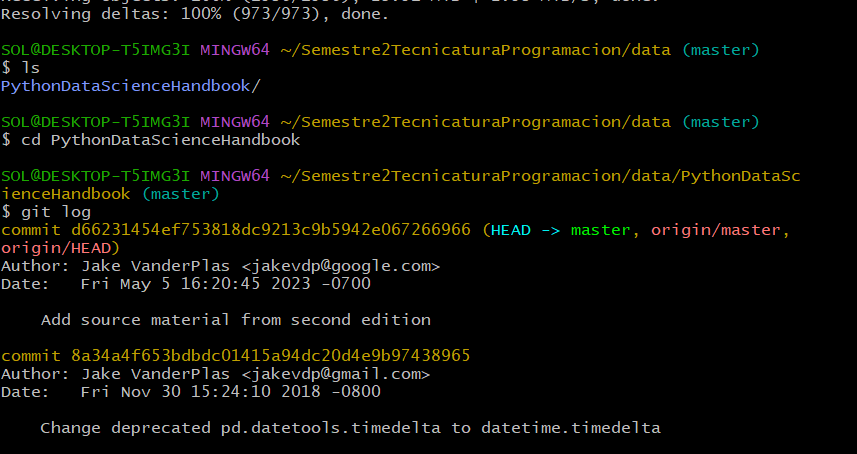


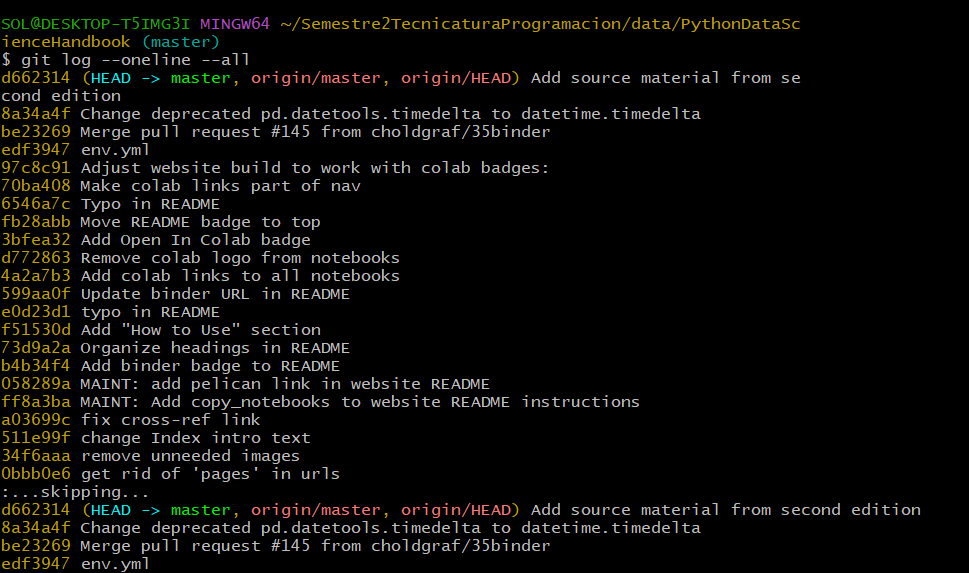
Descargamos el repositorio copiado a nuestra cuenta local.

De forma local, creamos e inicializamos un repositorio nuevo llamado data, ingresamos al repositorio y con el comando git clone + dirección https clonamos el repositorio nuevo a nuestro repositorio local.

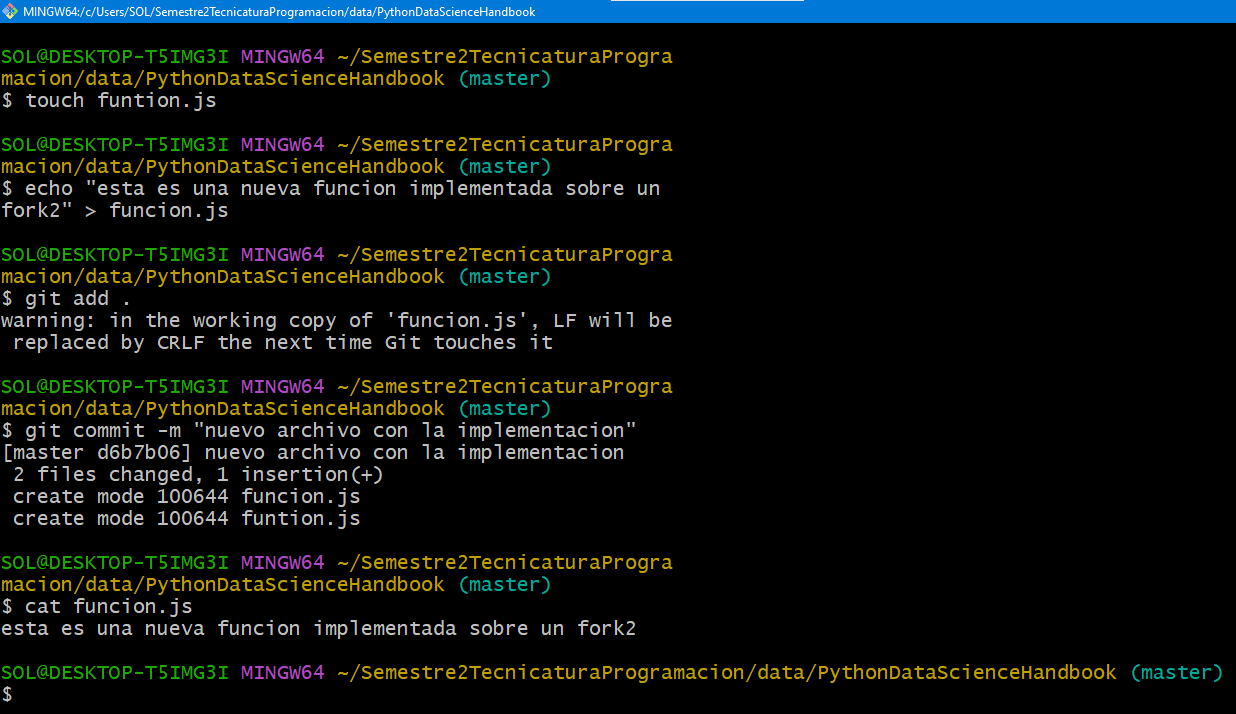


Ingresamos al repositorio y con el comando git log –oneline –all podemos ver todos los commits que tiene.

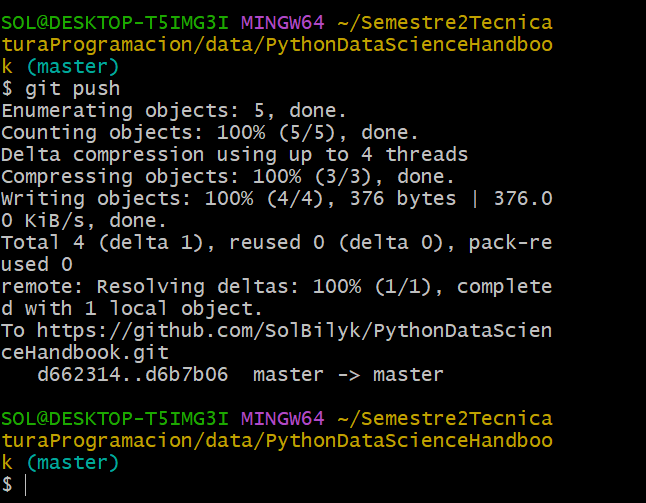




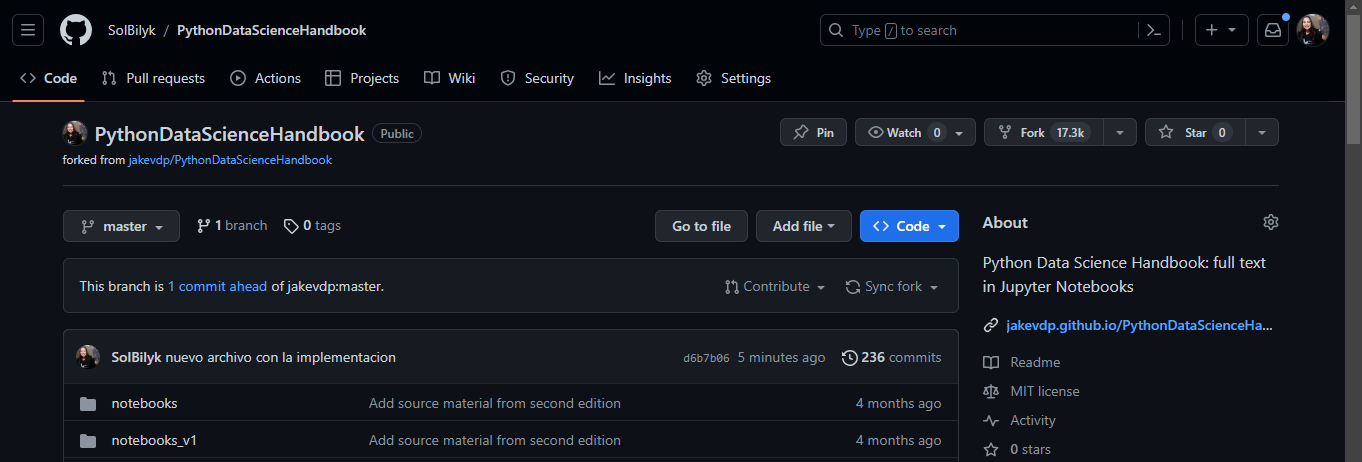
Agregamos un archivo nuevo con el touch “nombres”.txt, Luego comiteamos.



Luego utilizamos el comando git push



Actualizamos la pagina de github y aparecen los cambios realizados



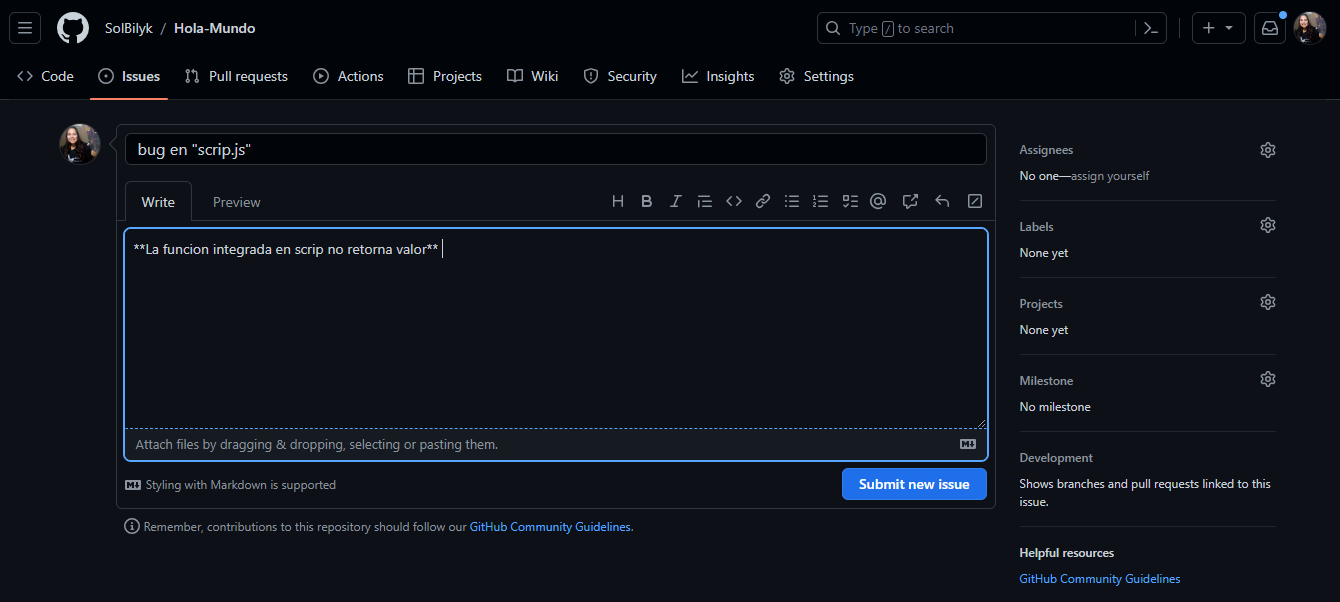
**ISSUES:**

Se utiliza para rastrear ideas, retroalimentación, tareas o errores para trabajar. Este trabajo por lo general lo realizan los tester, que son los encargados de que todo funcione bien.

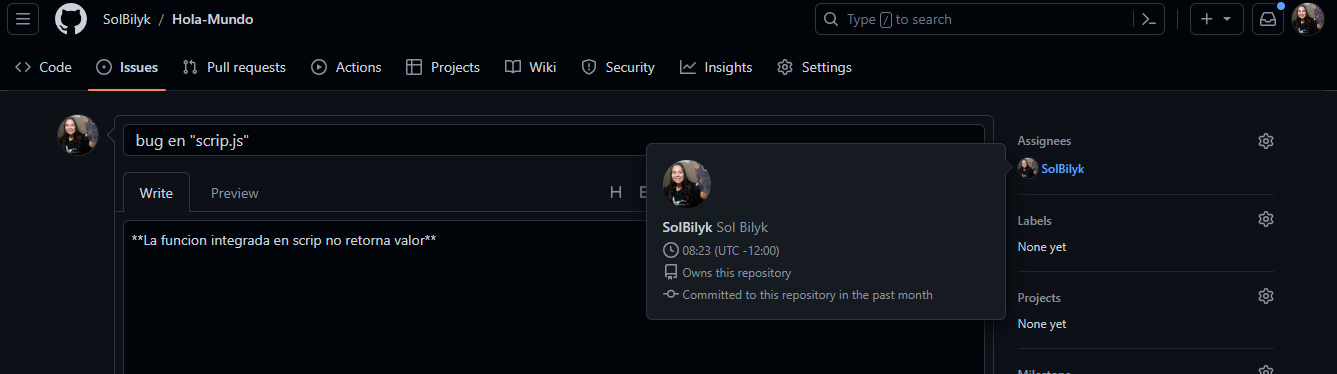
También va a servir para alertar al desarrollador los errores y poder corregirlos.

Entonces GithUB tiene issues que permite colocar los errores de un proyecto o código.

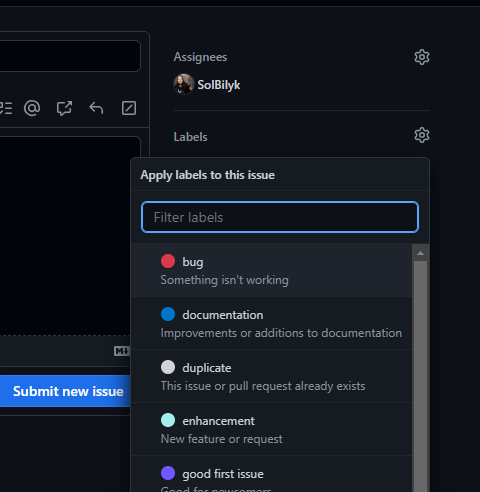
Añadiremos a modo ejemplo una issues con bug en “scrip.js” de nombre y una descripción.



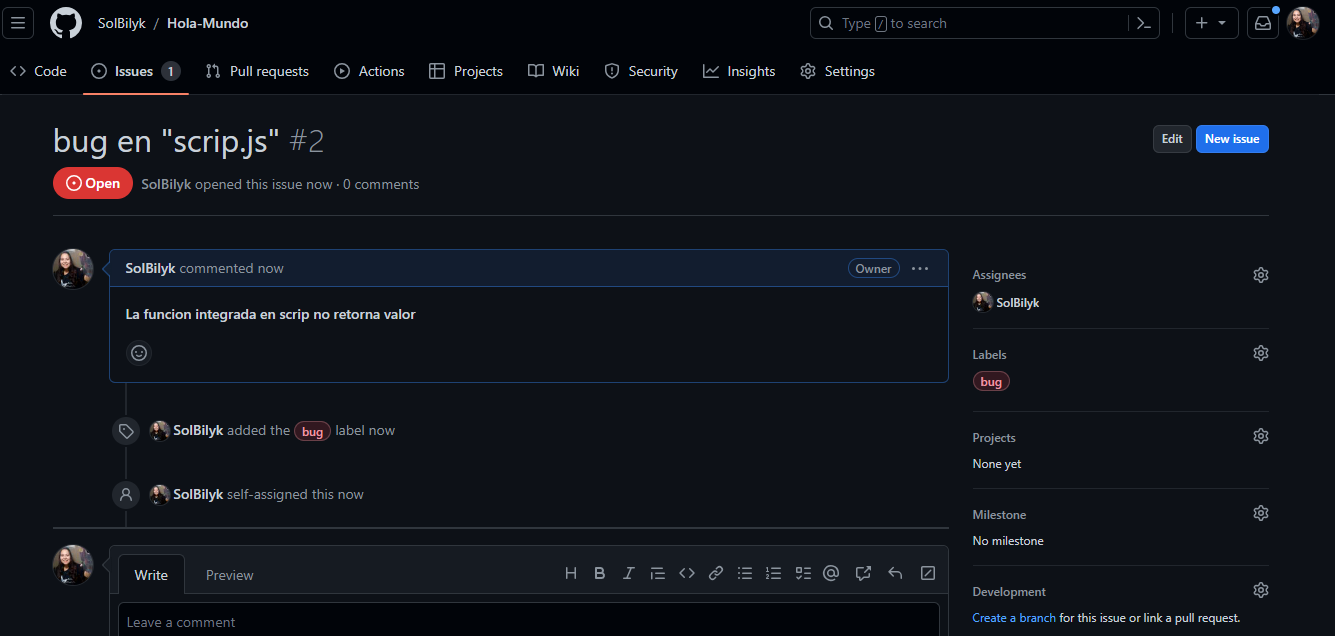
Me asigne a mi misma para las correcciones



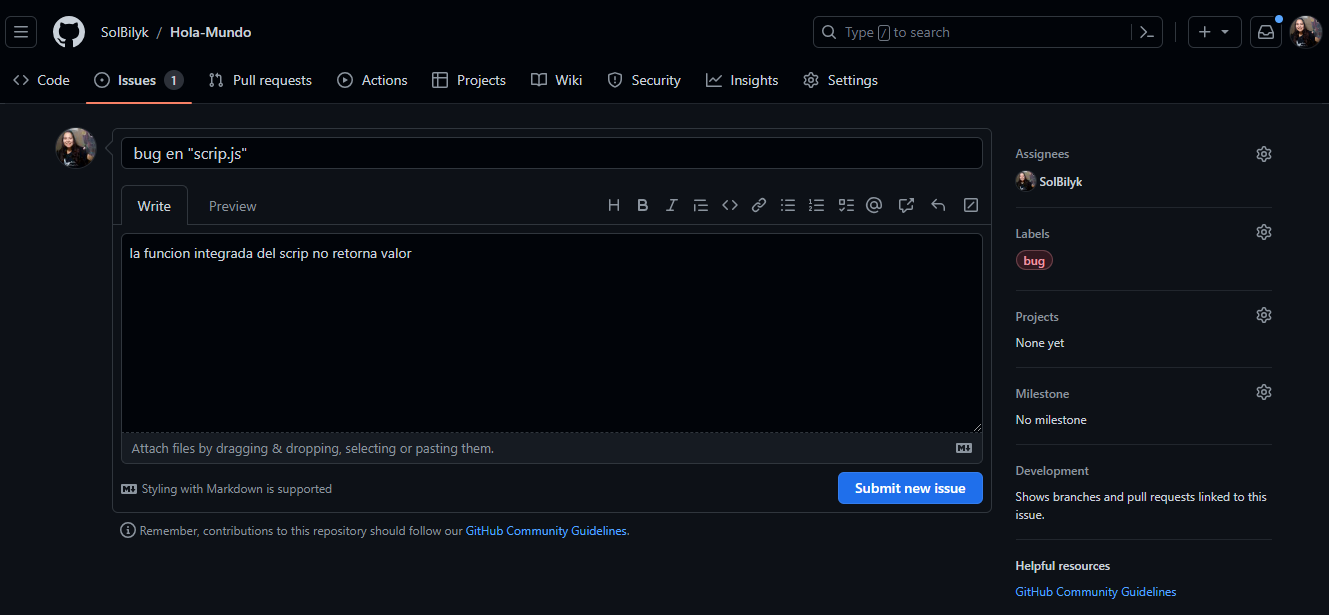
Etiqueta: bug



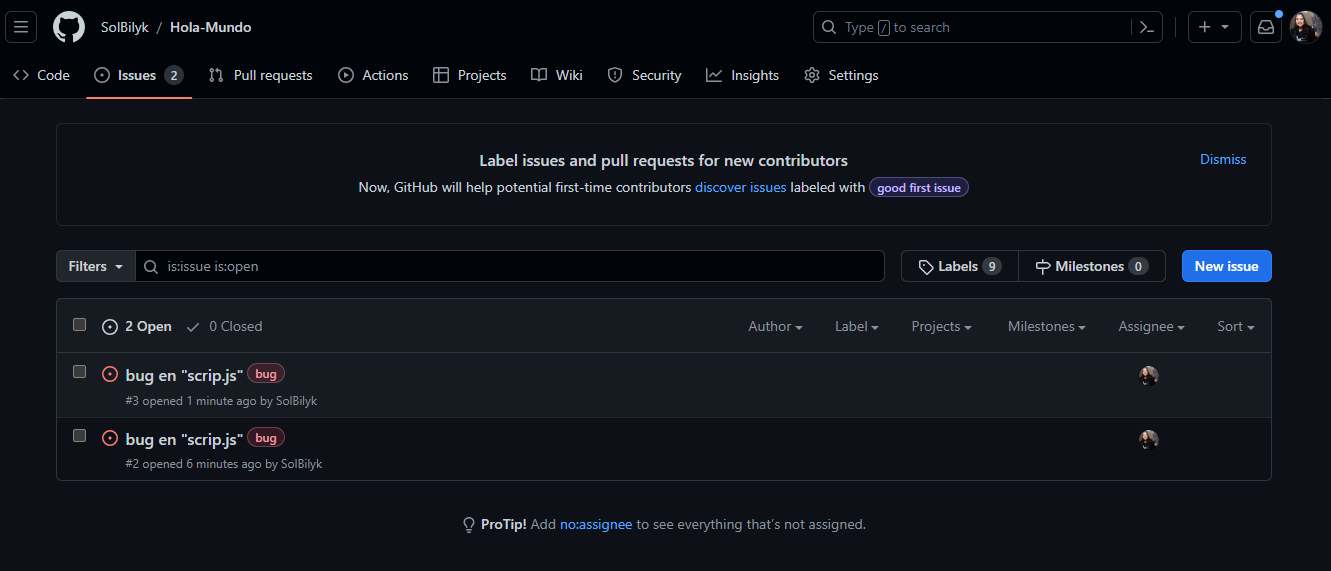
Luego hacemos click en comment.



Seleccionamos submit new issue



Podemos visualizar la issue creada:

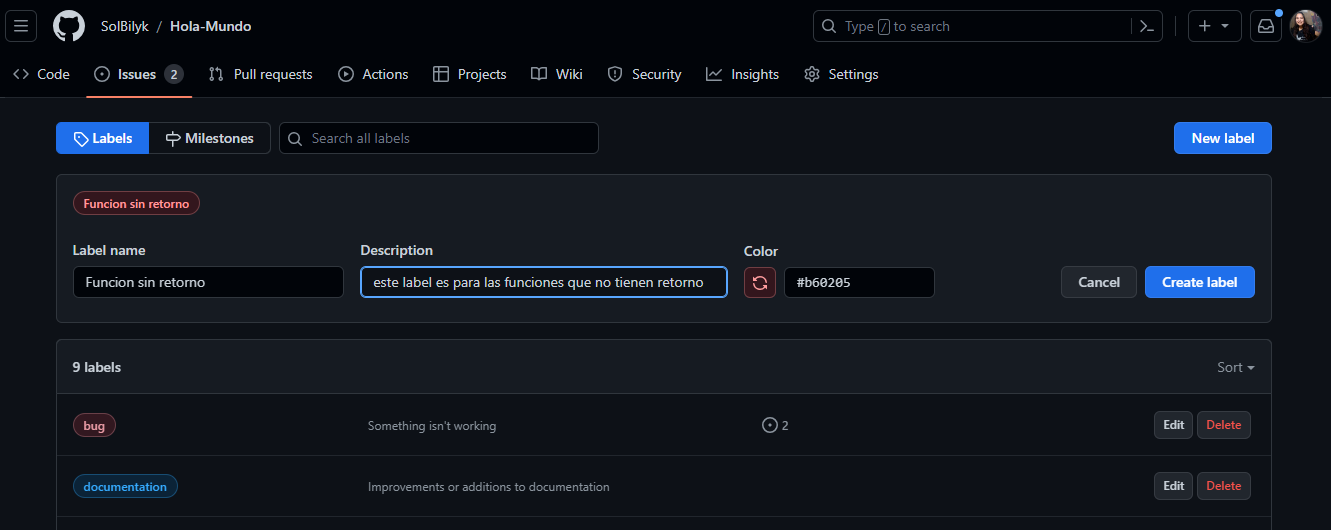


También tenemos la opción de crear nuestras label. Colocaremos un nombre

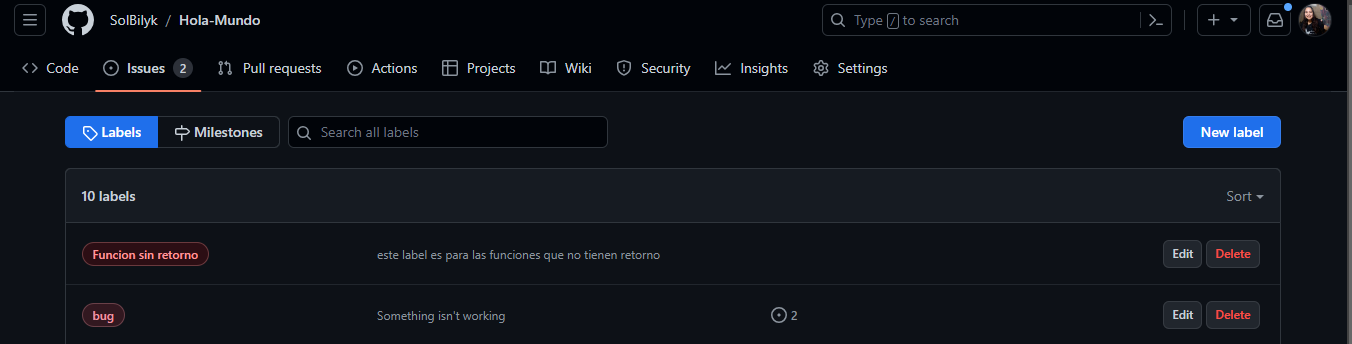
por ej: función sin retorno y una descripción

por ej: este label es para funciones que no tienen retorno,

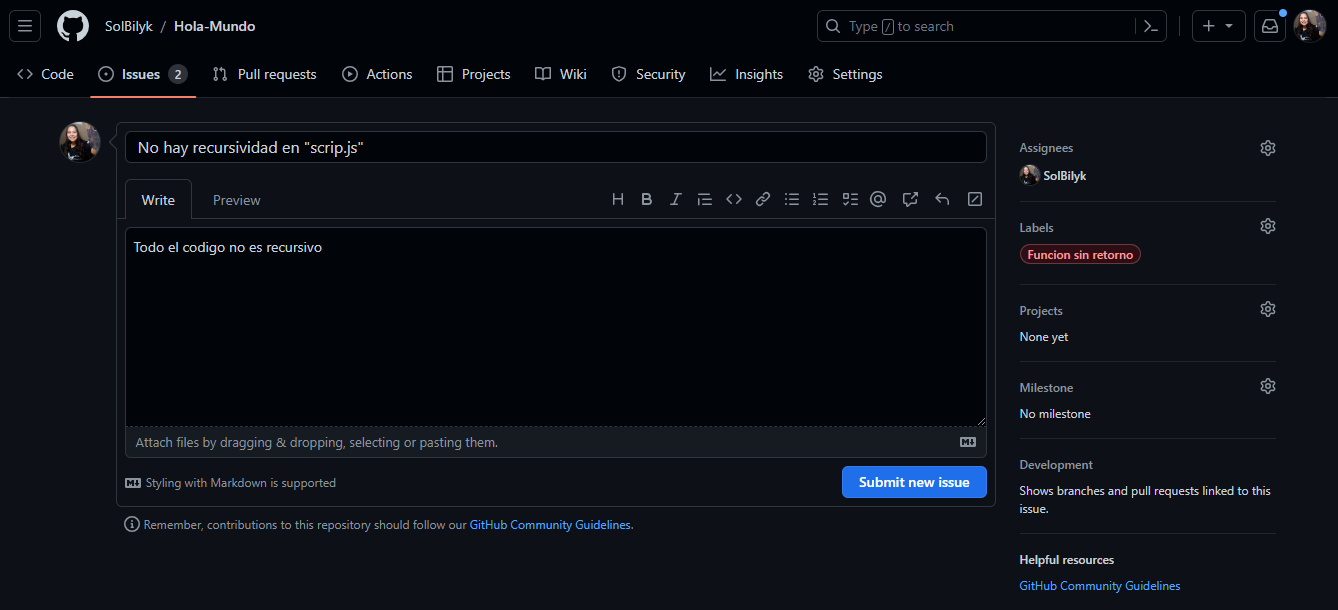
seleccionamos color y hacemos click en crear label



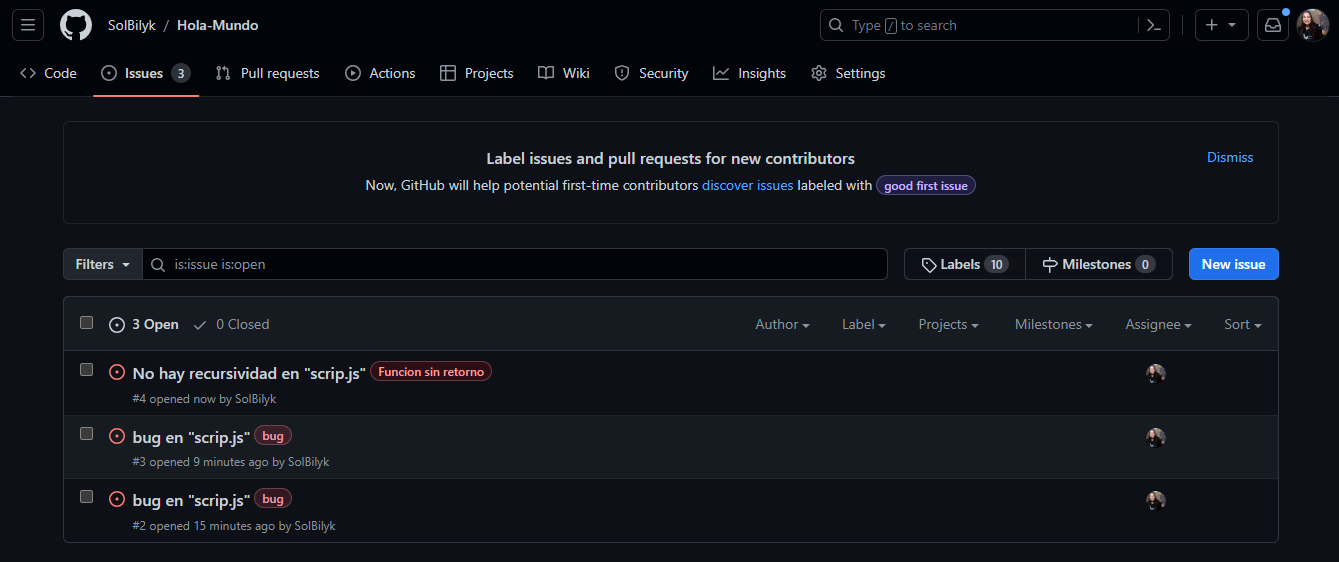
Vemos la label creada



Crearemos otra issues, con otro nombre por ej: No hay recursividad en “scrip.js” y en la descripción colocaremos por ej: Todo el código no es recursivo. pero esta vez seleccionaremos la labels creada.



Observamos la nueva issue creada:

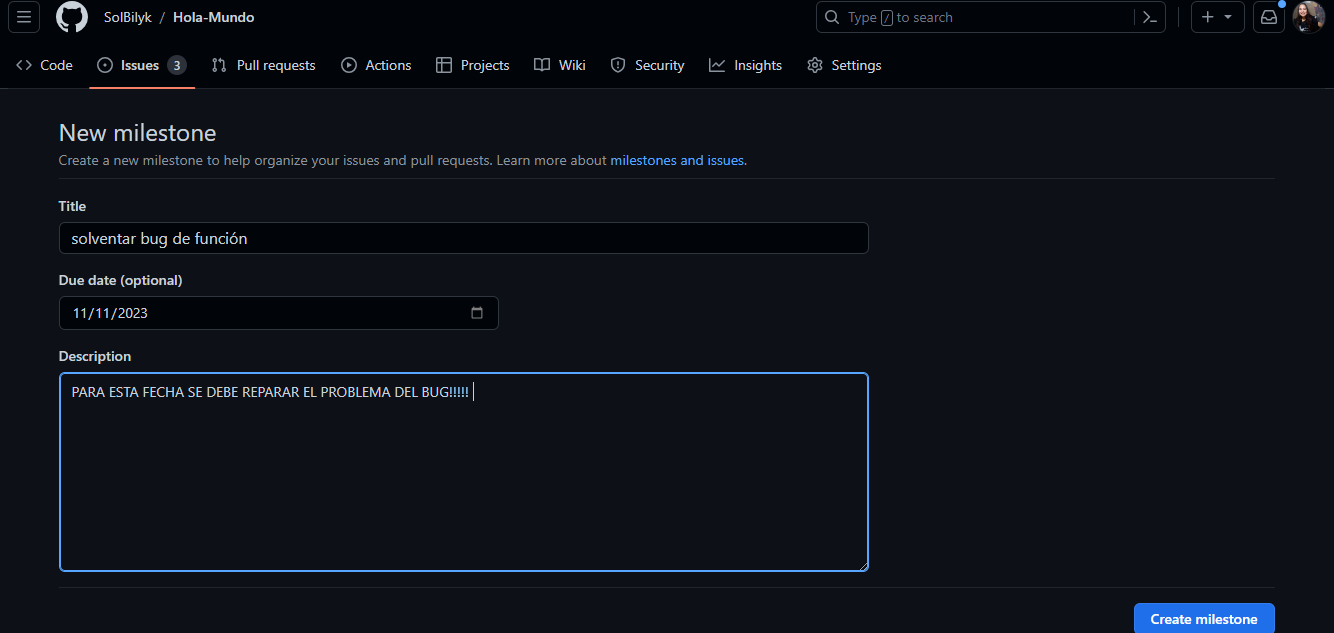


**Milestone**

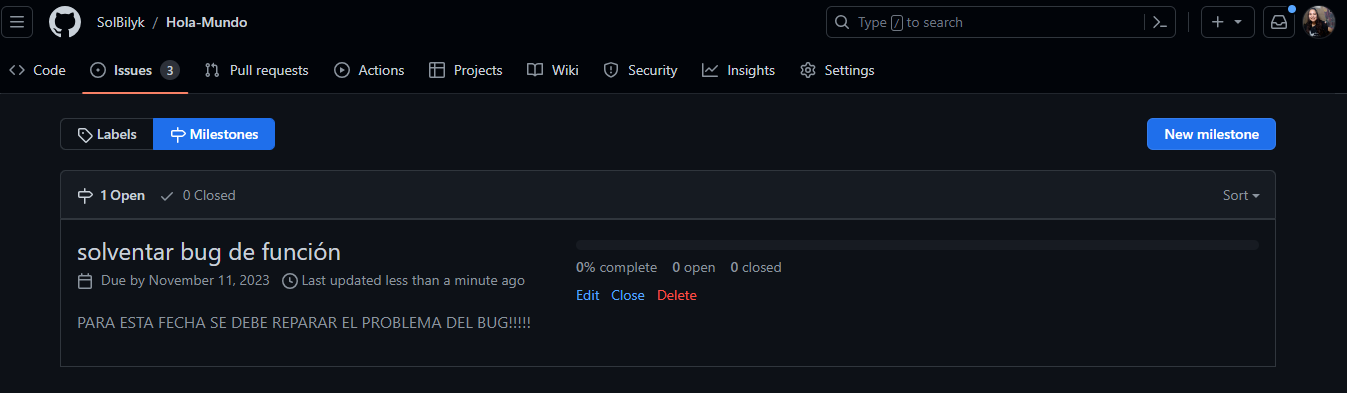
Es una funcionalidad de github que me permite establecer una fecha para la resolución de un problema.

El milestone me va a permitir establecer un tiempo determinado para solucionar un problema, cuando el problema se resuelva, se elimina el milestone. Creamos un nuevo milestone.

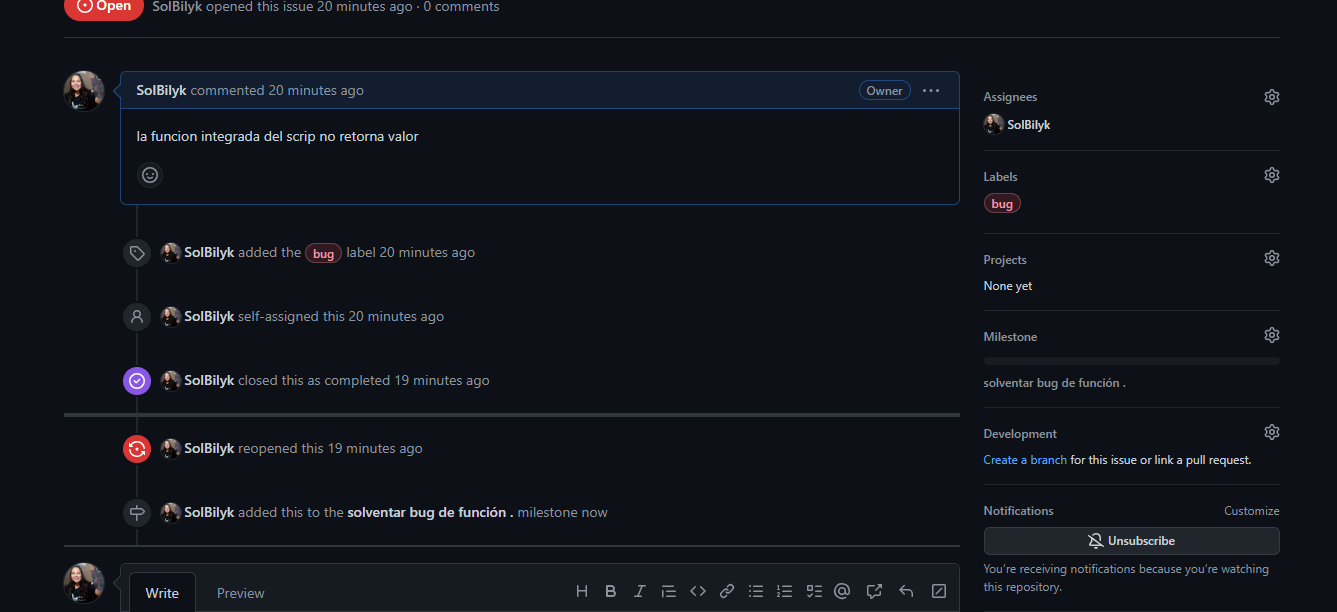
Colocaremos un título por ej: solventar bug de función seleccionar un tiempo que nos puede llevar el arreglo y por último agregar una descripción de lo que se va a realizar

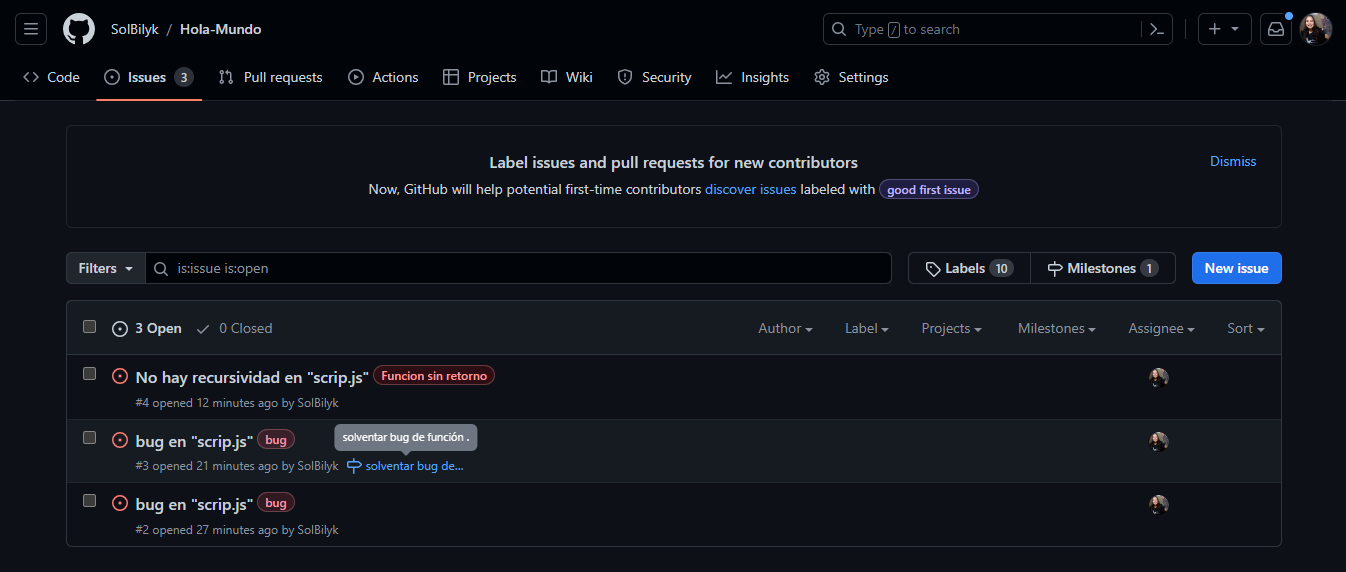


Milestone creado:

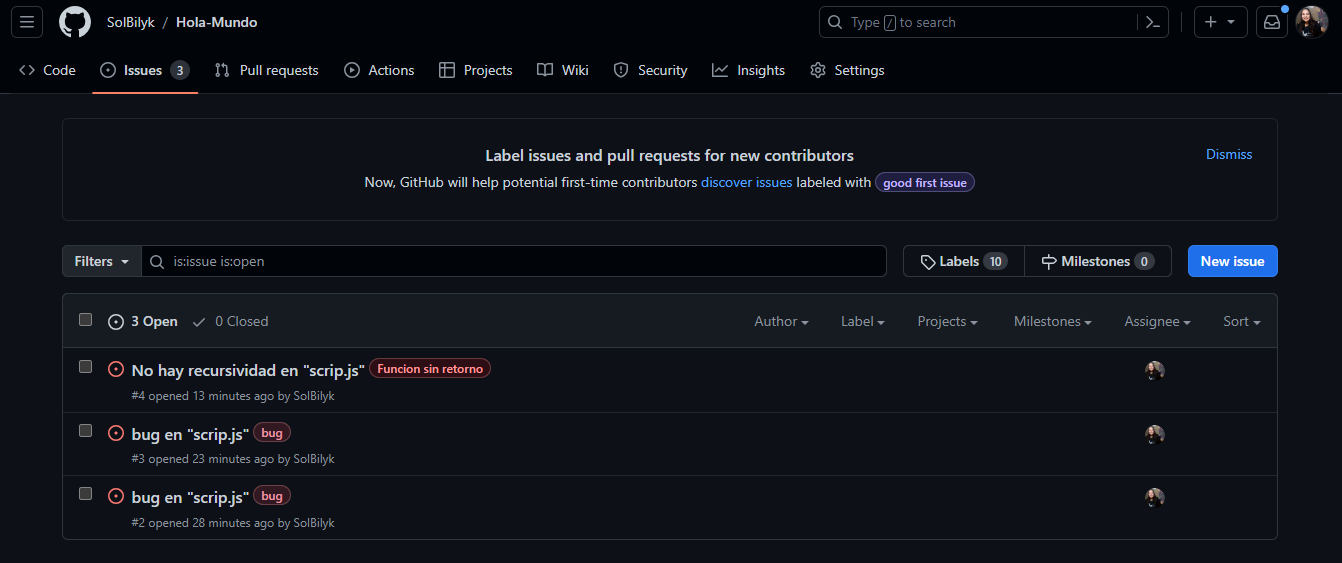


Agregamos el milestone a uno de los issues





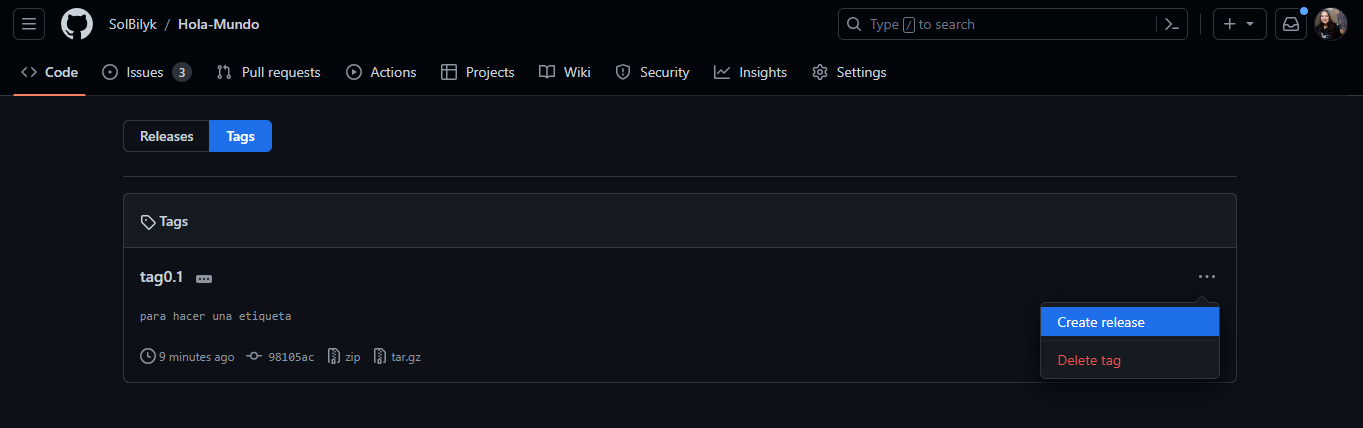
Una vez cumplido el tiempo y realizada la corrección del error, eliminaremos el milestone y ya no aparece en el issue.



**RELEASES**

Es una versión que se lanza del proyecto, para eso seleccionaremos una tag creada de las clases anteriores.

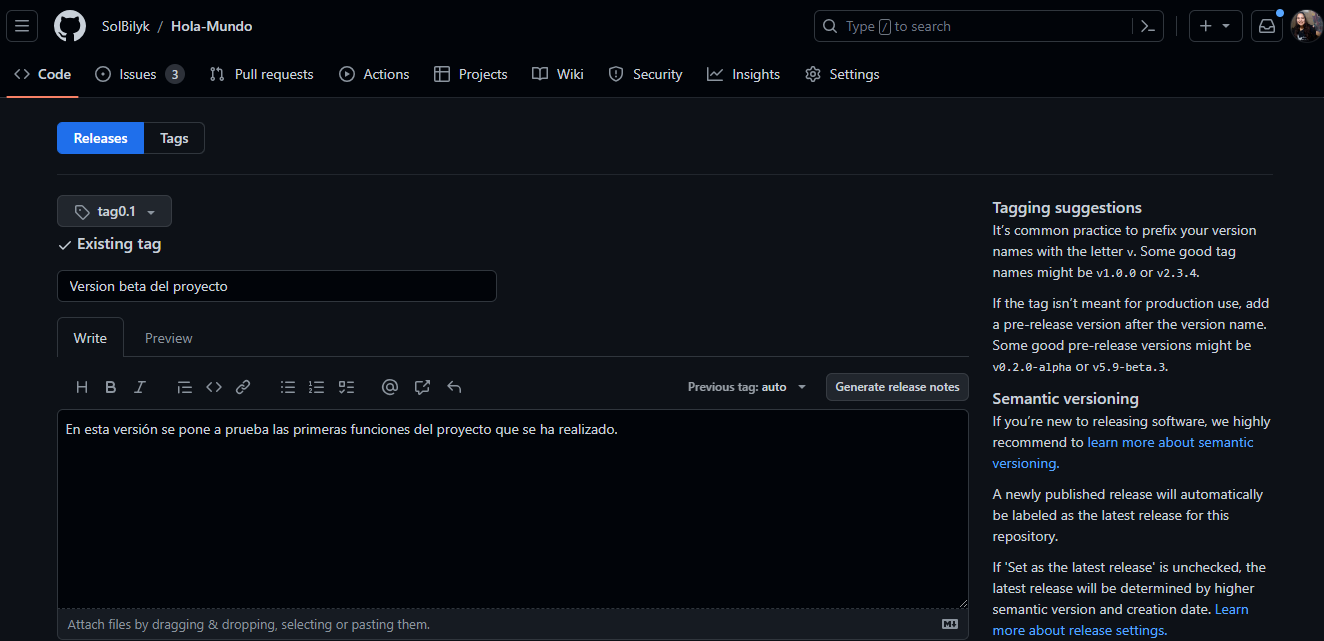
Seleccionamos una tag creada



Creamos un nuevo release

Muchas veces al proyecto lo haremos por etapas, colocaremos un título por ej:

versión Beta del proyecto y luego añadiremos una descripción por ej: En esta versión se pone a prueba las primeras funciones del proyecto que se ha realizado.



Primer release del proyecto

