[Título del documento]

[Subtítulo del documento]

[Escuela]

[Título del curso]

INDICE

[Introducción: 2](#_Toc157599074)

[Funcionamiento de la aplicación: 2](#_Toc157599075)

[phpUnit: 3](#_Toc157599076)

[Instalación 3](#_Toc157599077)

[Funcionamiento del test: 4](#_Toc157599078)

[Conexión a github: 6](#_Toc157599079)

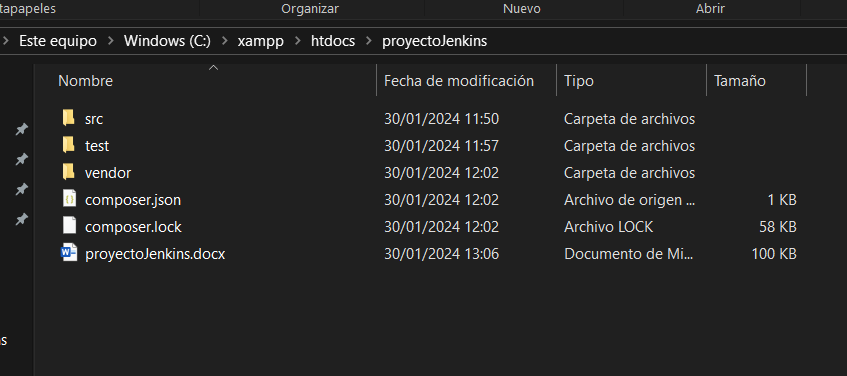
# Introducción:

Este proyecto consiste en la creación de una aplicación PHP para hacer una serie de pruebas usando PhpUnit y luego conectarlo a Jenkins. Antes de nada, presentamos la raíz del proyecto:

“src” se compone de StringUtils.php y functions.php que conforma el proyecto base.

“test” se compone para StringUtilsTest para realizar las pruebas de la carpeta src.

El resto son implementaciones necesarias para trabajar realizar los test y para documentar el proyecto.



# Funcionamiento de la aplicación:

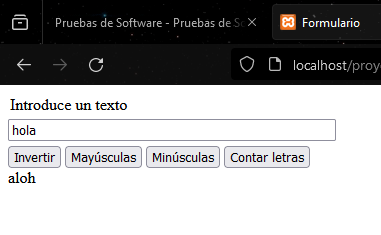
Se crea una aplicación PHP llamada StringUtils, que contenga un formulario con un campo de texto. Al introducir el usuario una cadena en el campo y pulsar el botón correspondiente, realiza una llamada AJAX a una función php. Las funciones disponibles son:

-Darle la vuelta al string

-Pasarlo a mayúsculas

-Pasarlo a minúsculas

-Contar las letras



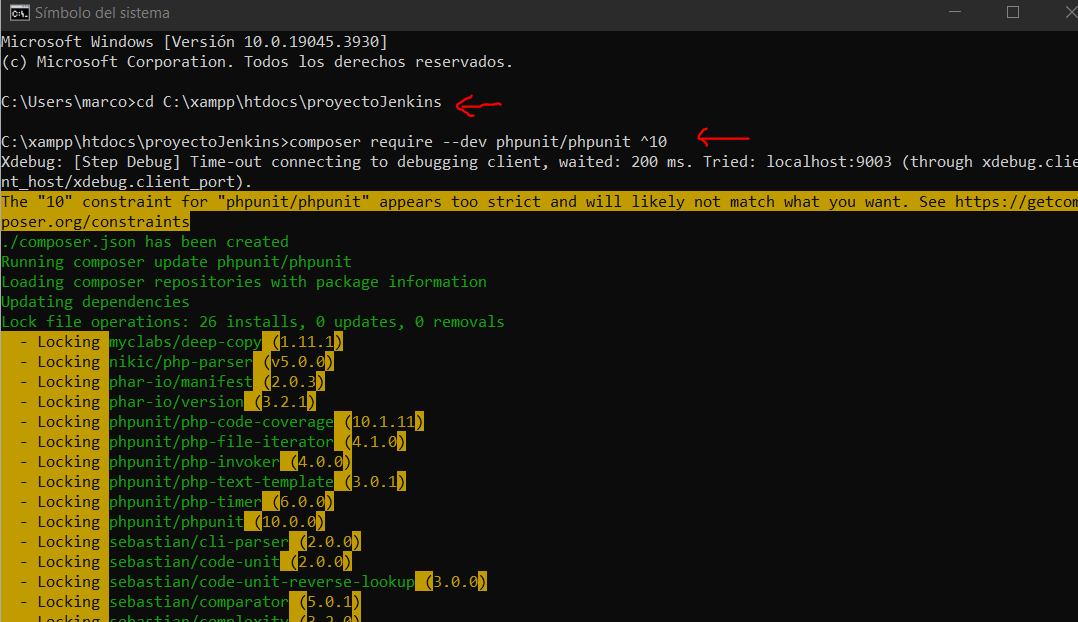
# phpUnit:

PhpUnit es un entorno para realizar pruebas unitarias en el lenguaje de programación PHP.

## Instalación

Abrimos la consola del sistema y hacemos un “cd” hasta la raíz de nuestro proyecto, luego aplicamos el siguiente comando:

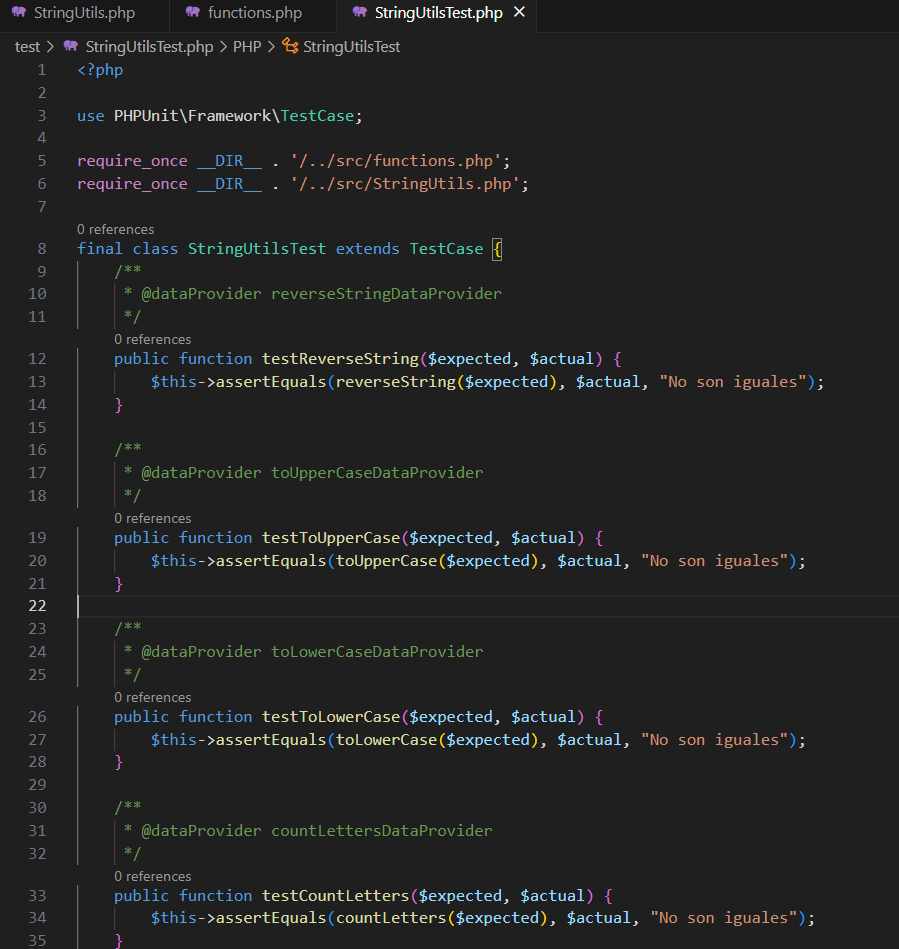
require –phpunit/phpunit ^10



## Funcionamiento del test:

Por convención los nombres de la clase deben terminar en \*\*\*Test y los métodos deben comenzar por test\*\*\*. Además por defecto phpunit los escanea en la carpeta test, que debemos crear en nuestro proyecto.

En este test, pasamos como parámetro la palabra “hola” y se comprobará en cada una de las funciones del programa (4) si recibe el parámetro adecuado ya transformado.

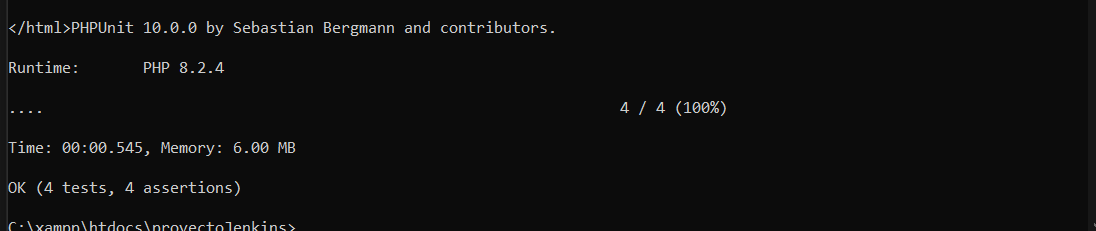




Para ejecutar el test tenemos que abrir la consola y dirigirnos a la raíz de nuestro proyecto con un CD. Luego ejecutamos el siguiente comando cada vez que queramos hacer un test:

“.\vendor\bin\phpunit test”

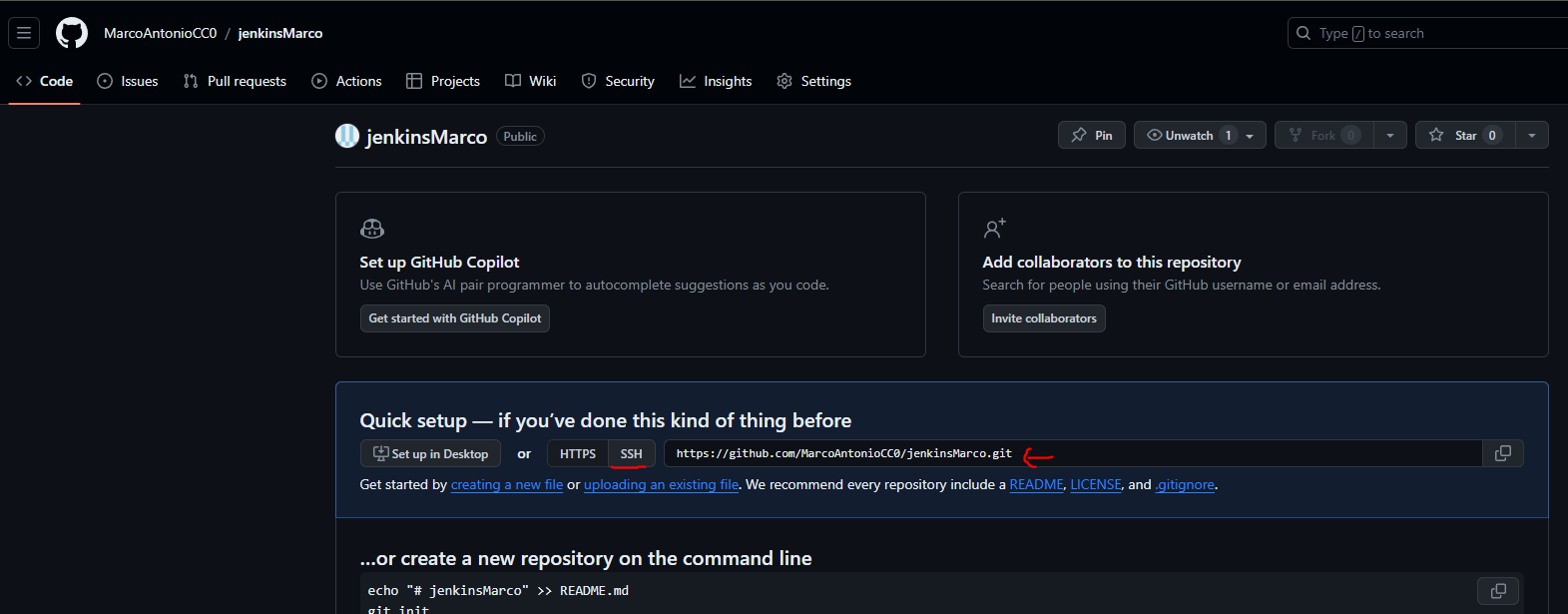
Ejemplo del test realizado con éxito:



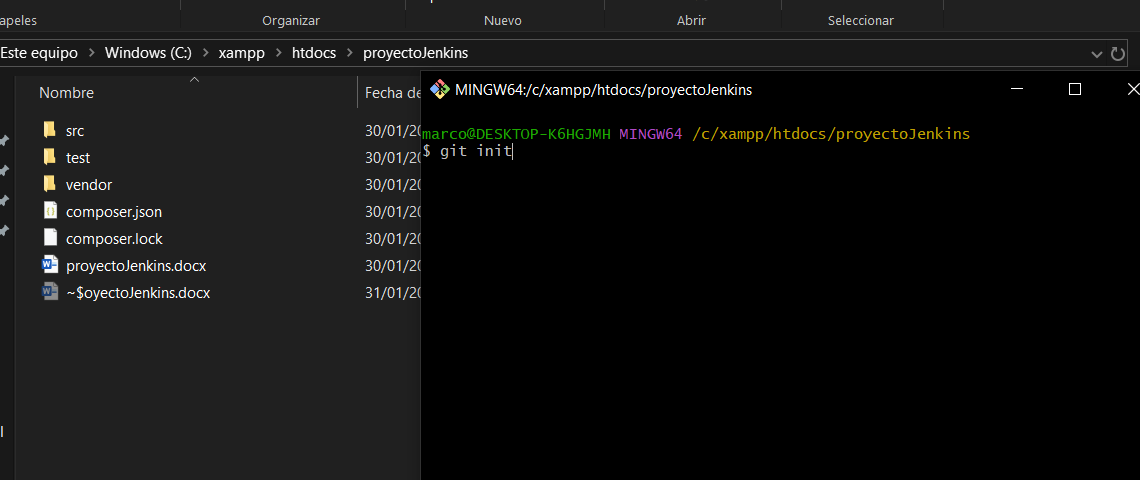
## Conexión a github:

Con el proyecto ya listo, lo subimos a github.

Primero creamos un repositorio en nuestra cuenta de github y una vez creado copiamos el SSH generado:



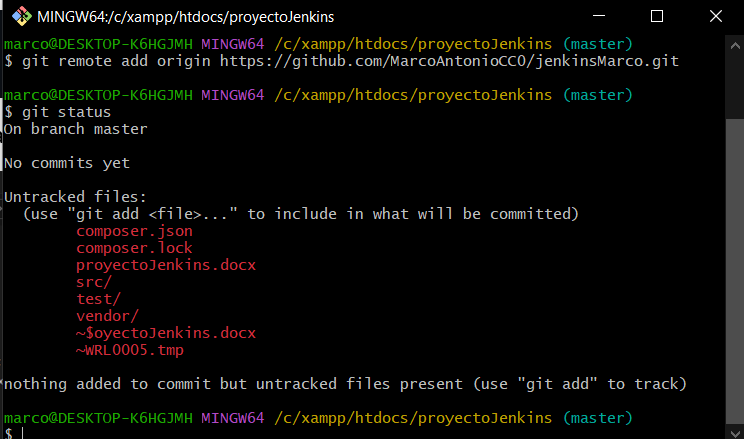
Luego, en la raíz de nuestro proyecto, abrimos gitbush y escribimos: git init:



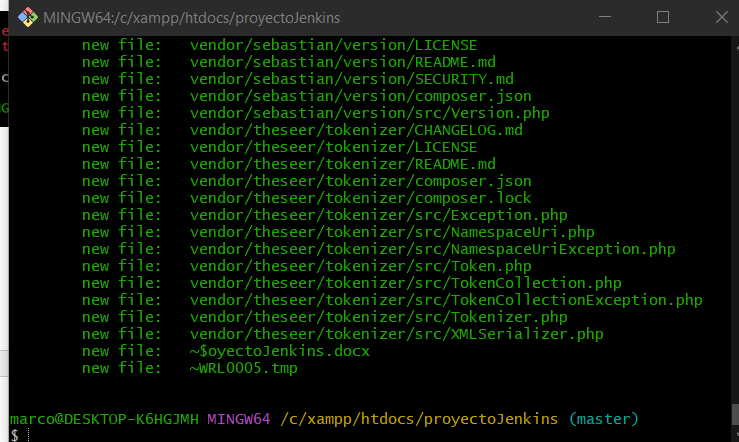
Y lo agregamos al repositorio con el siguiente comando:

git remote add origin https://github.com/MarcoAntonioCC0/jenkinsMarco.git

Ya tenemos el repositorio conectado, solo faltaría subirlo ya que si hacemos un git status comprobamos que no se ha guardado los cambios (rojo)

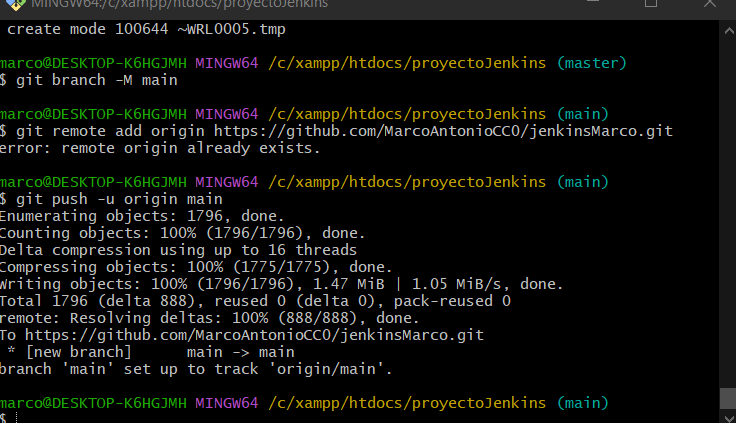


Para ello hacemos : git add . y hacemos un commit para capturar nuestra primera versión del proyecto : git commit -m “comentario de ejemplo”



Para más comodidad se crea una rama donde haré un push : git branch -M \*nombre\*

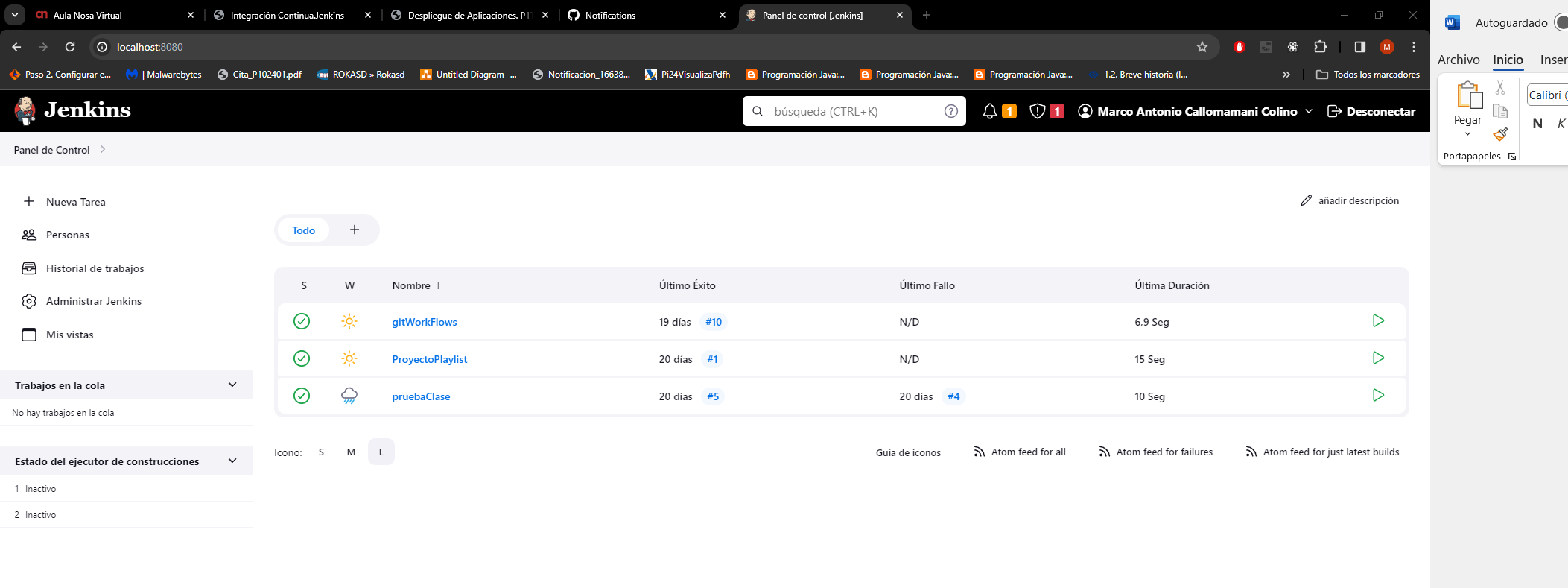
Y luego subimos al repositorio todos los cambios con un push: git push -u origin \*nombre rama\*



El proyecto ya está conectado y por cada cambio relevante se hará un commit y push para actualizar el repositorio.

# Jenkins:

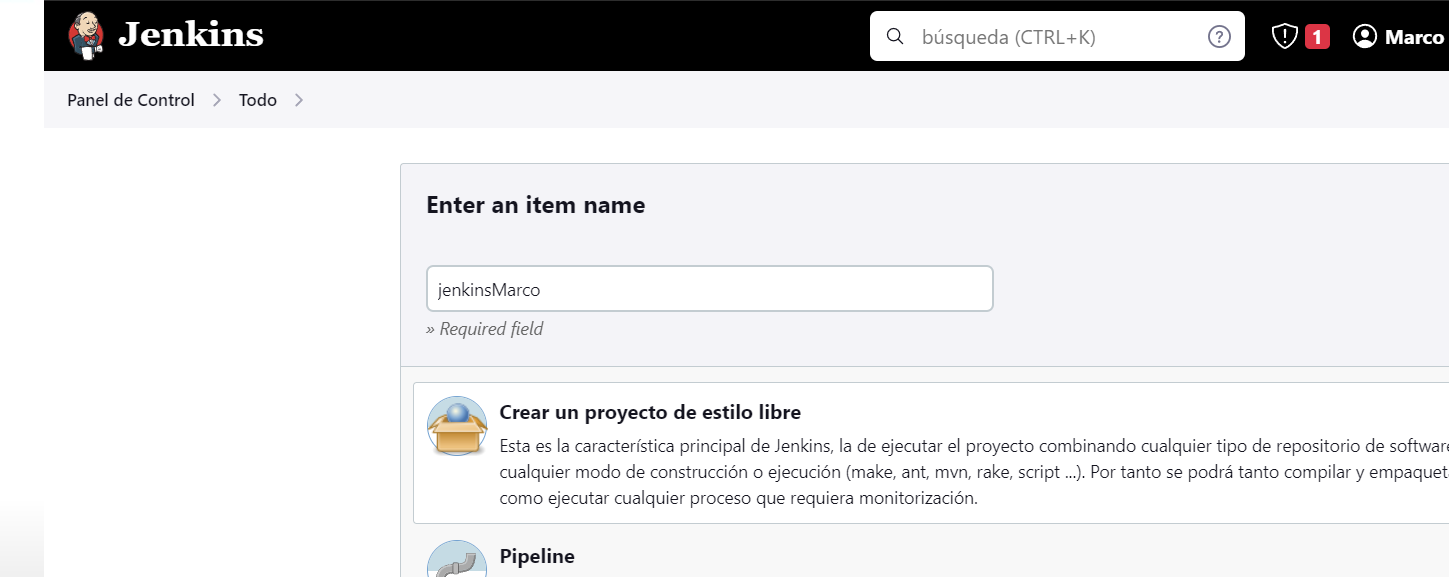
Preparamos el entorno de Jenkins antes de conectar con el proyecto



## Conectar el proyecto en github

En el apartado de “panel de control” arriba al a izquierda le damos a “Nueva Tarea”.

Ahí seleccionamos crear un proyecto de estilo libre y le aplicamos un nombre:

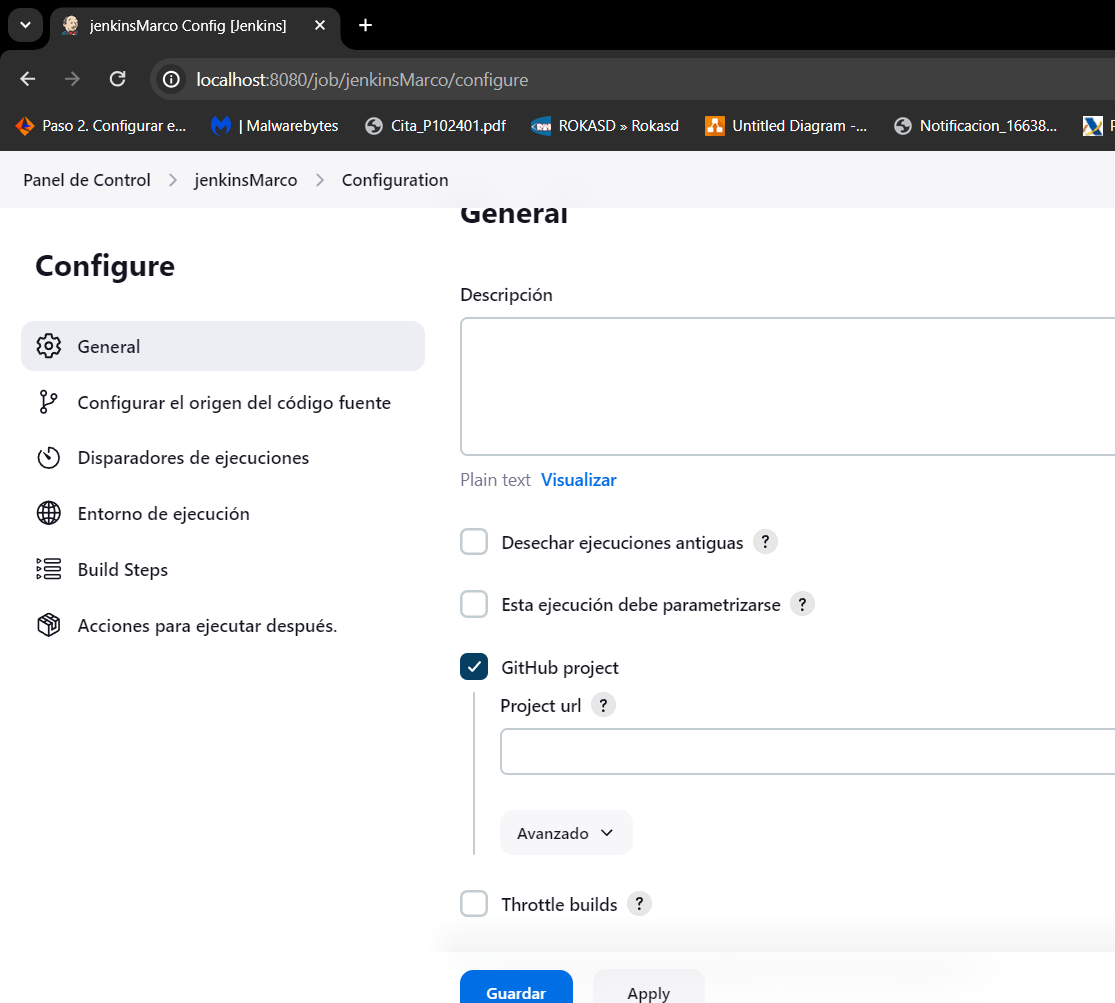


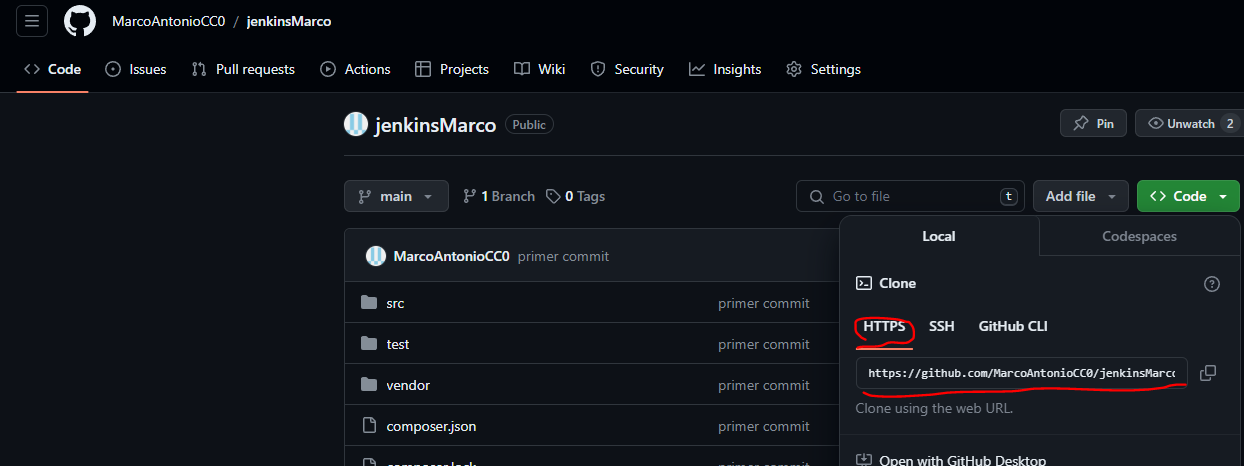
## Configuración de proyecto

Una vez creado, nos pedirá una serie de modificaciones el cual solo nos interesa 2 en concreto:

### 1º Conexión a github:

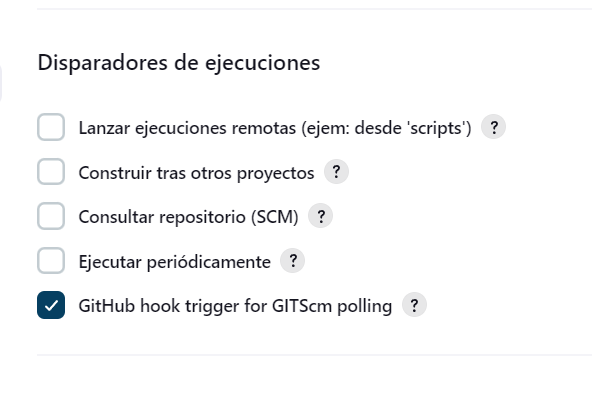
Marcamos github Project y copiamos ahí el enlace HTTPS que está disponible en nuestro repositorio github:



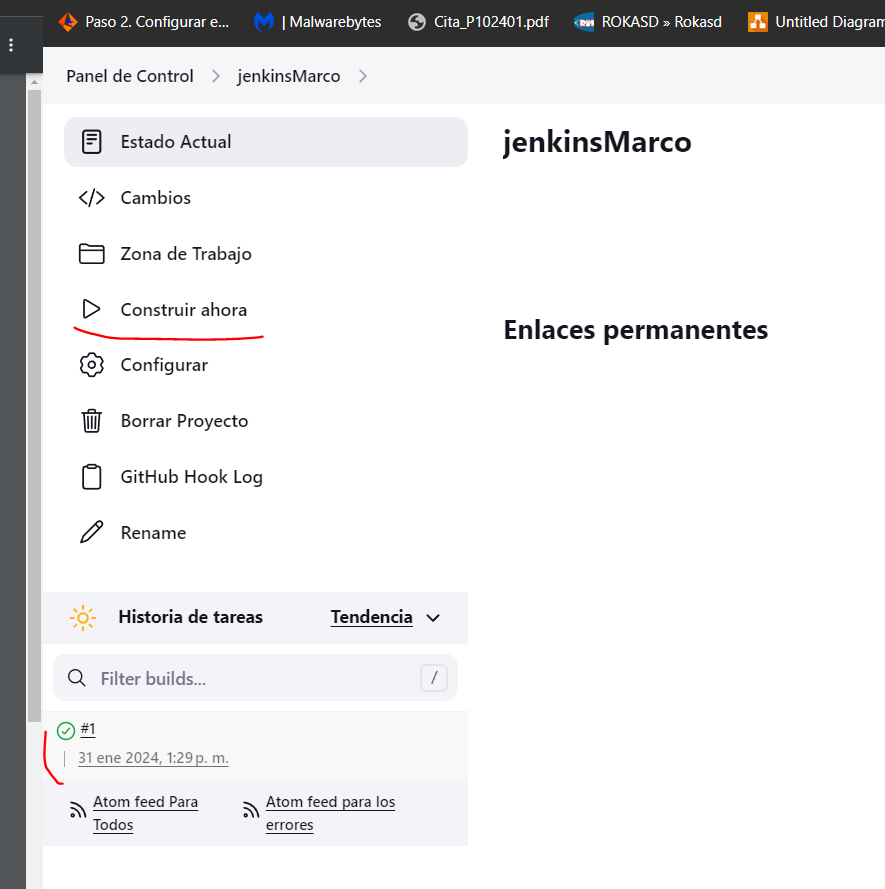


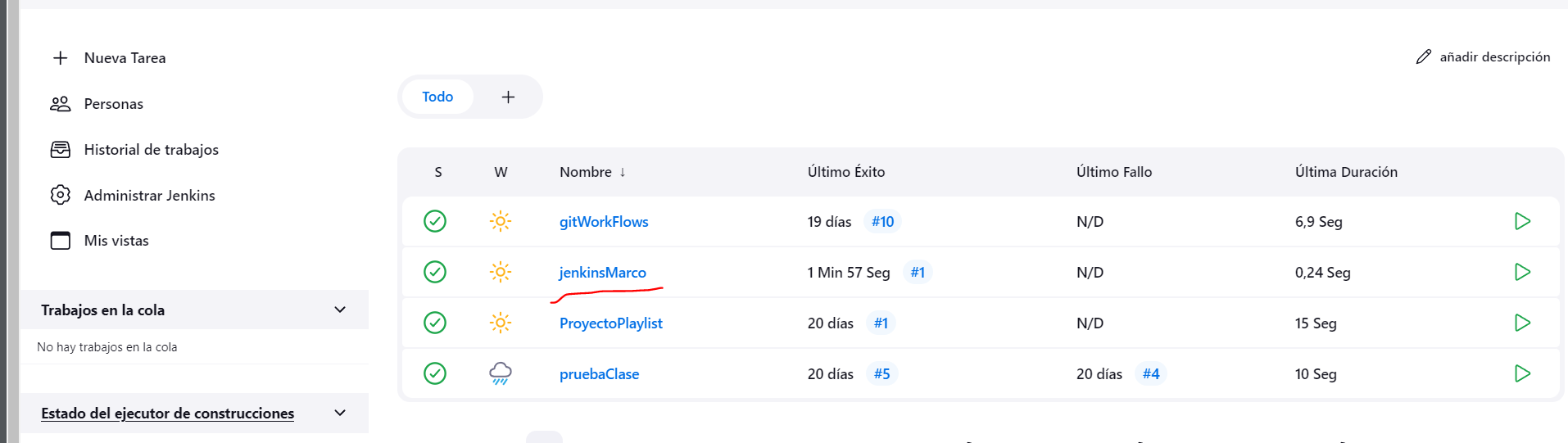
## 2º Avisos de los cambios hechos a github

En la opción de Disparadores de ejecuciones marcamos la casilla “Github Hook trigger for GITScm polling” para que en cada push que se haga desde git se notifique.



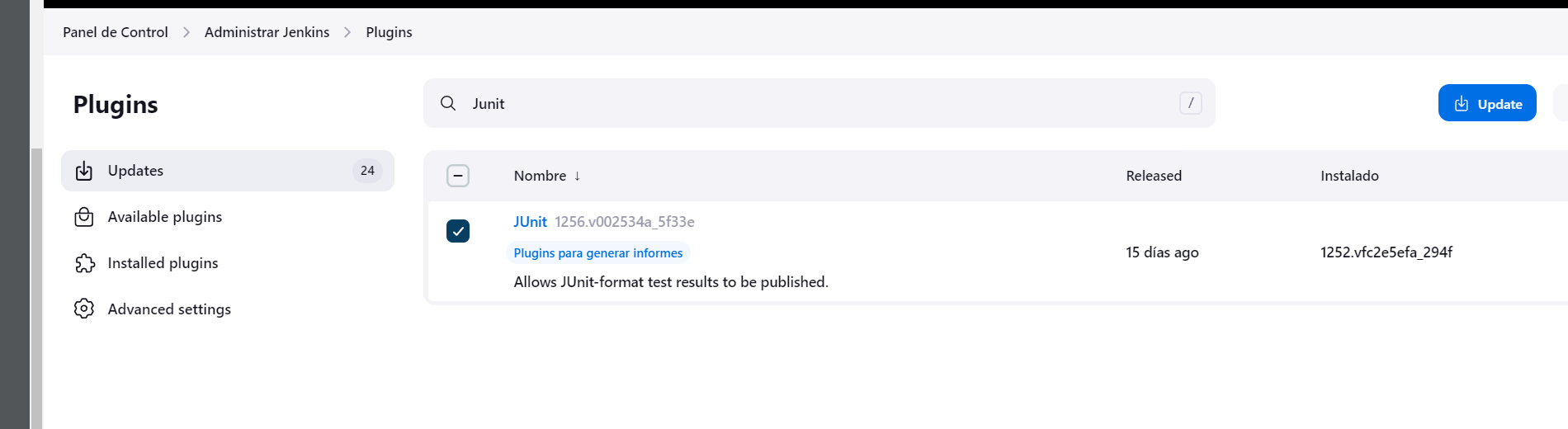
Una vez terminado ya tenemos nuestro proyecto. Por defecto nos dejará en el apartado de Estado actual, si probamos a darle a botón “Construir ahora” hará una pequeña comprobación del estado del archivo y se guardará como historial las últimas comprobaciones:





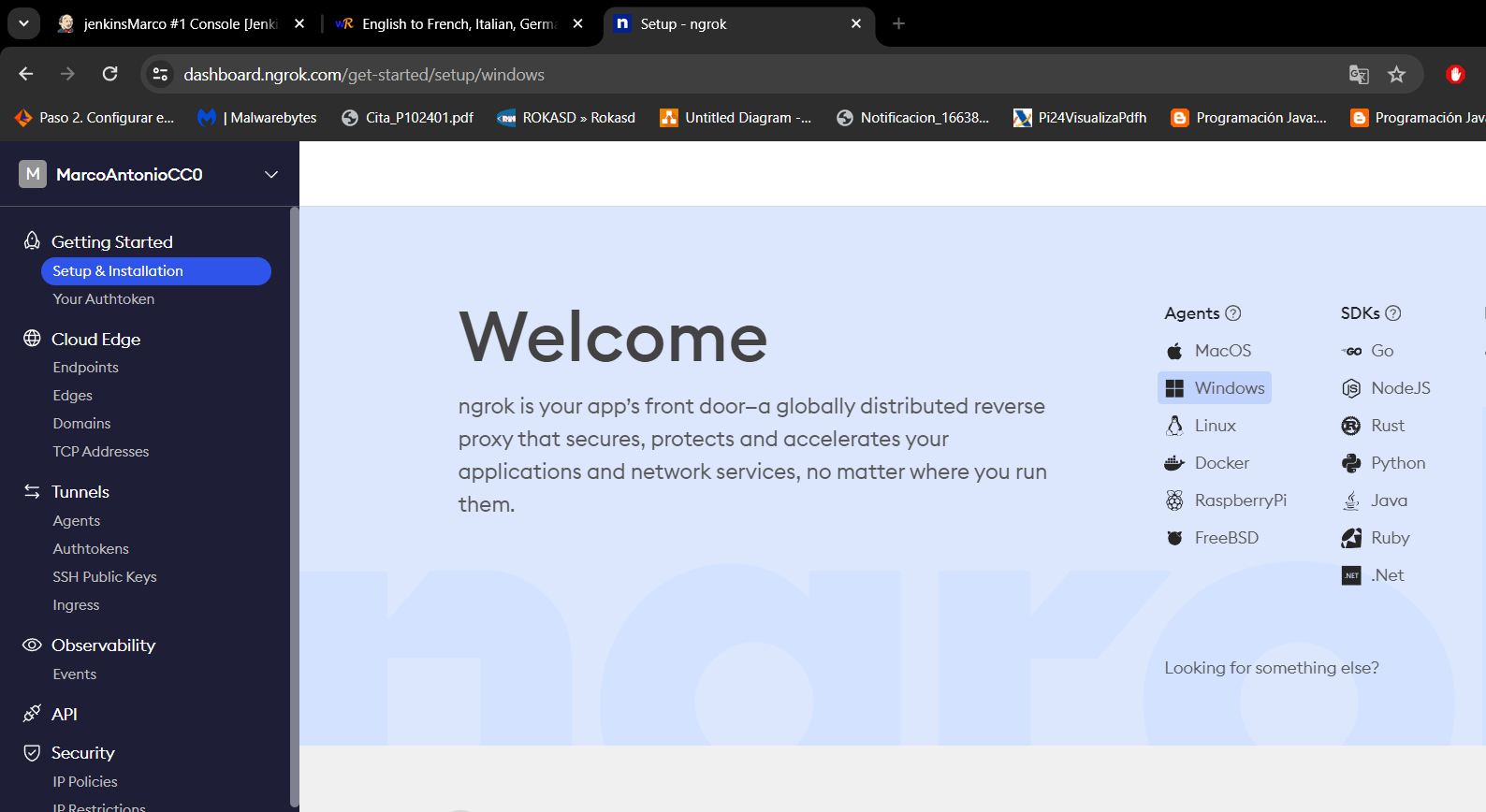
## Plugin Junit:

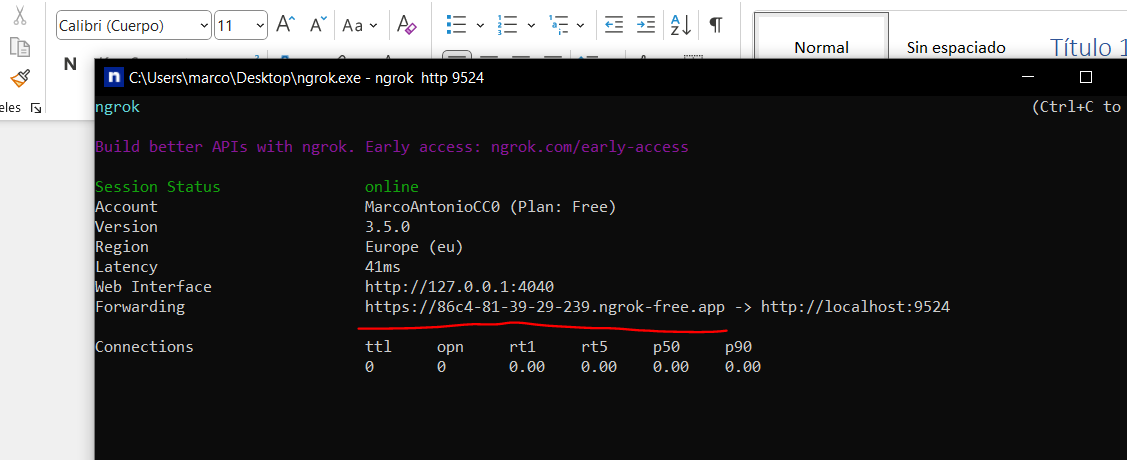
Instalamos en Jenkins el plugin Junit para generar informes de pruebas. Por defecto ya está instalado, pero en caso contrario lo encontrarás en el apartado Administrar Jenkins > plugins:

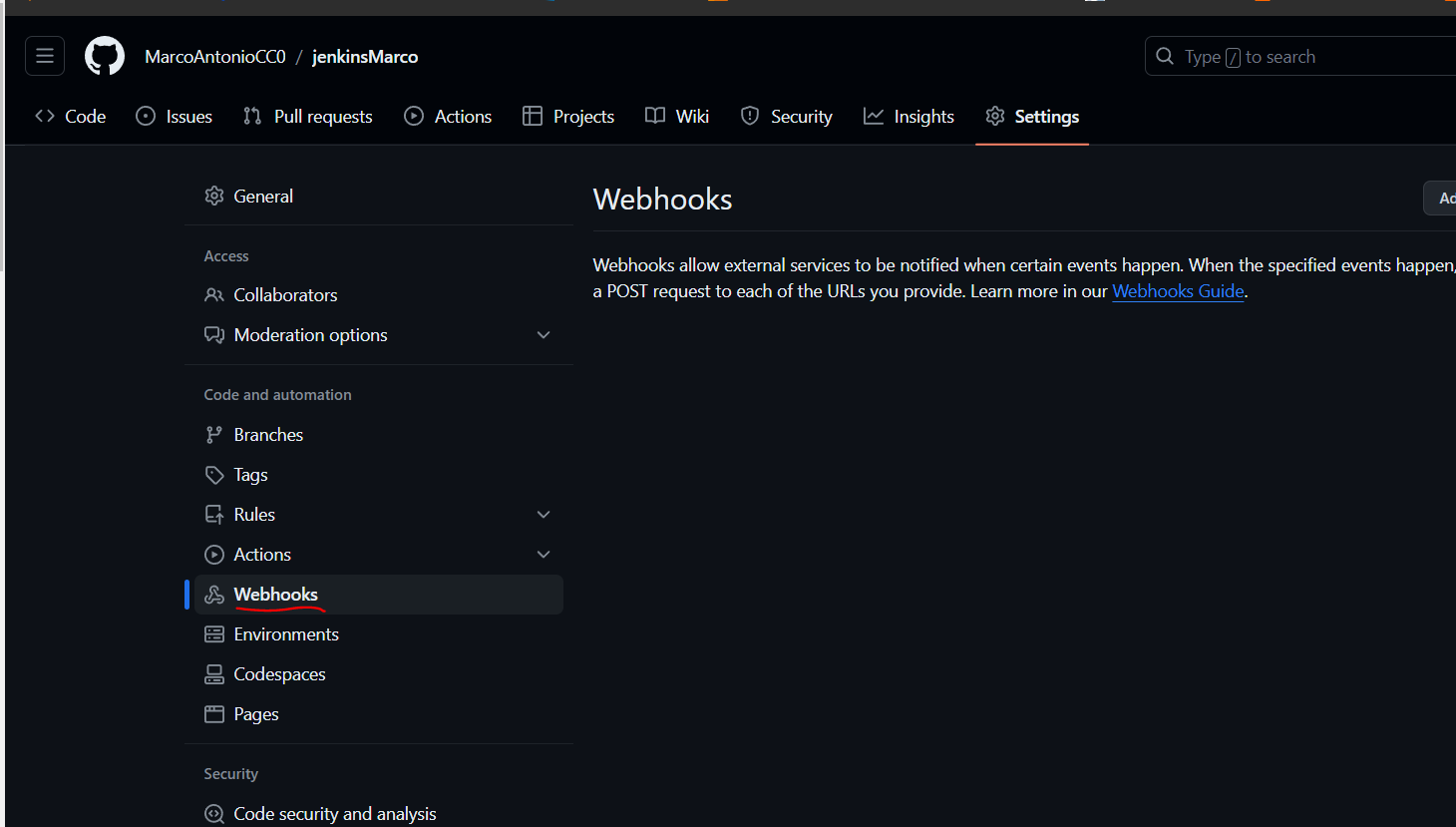


## Ngrok:

Para conectar un proyecto de Jenkins con un repositorio github es necesario instalar e implementar ngrok en Windows. Es recomendable crear una cuenta con github.







Añadimos el código + githubwebhook:

<https://86c4-81-39-29-239.ngrok-free.app/github-webhook/>

Nos avisa por cada push que hagamos

