

Subir proyecto a Sonarqube desde GitLab

1.- Crear un proyecto, indicando el nombre del proyecto

Clave del proyecto * ?

 ✓

Hasta 400 caracteres. Los caracteres permitidos son alfanuméricos, '-' (guión), '_' (guion bajo), '.' (punto) y ':' (dos puntos), con al menos uno que no sea un dígito.

Nombre para mostrar * ?

 ✓

Hasta 255 caracteres

[Configurar](#)

2.- Ingresar un token, se puede generar o tener uno existente.

1 Provide a token

prueba: 4d96405adbf0cb08647929fdef9a9d6606998552 


The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your [user account](#).


[Continue](#)


3.- Elegimos la forma en la se va a subir el proyecto


How do you want to analyze your repository?


Do you want to integrate with your favorite CI? Choose one of the following tutorials.



With Jenkins


With GitHub Actions



With Bitbucket Pipelines


With GitLab CI


With Azure Pipelines


Other CI

Are you just testing or have an advanced use-case? Analyze your project locally.


Locally

4.- Elegir el tipo de tecnología que se va a implementar

1 Set your project key

1 What option best describes your build?

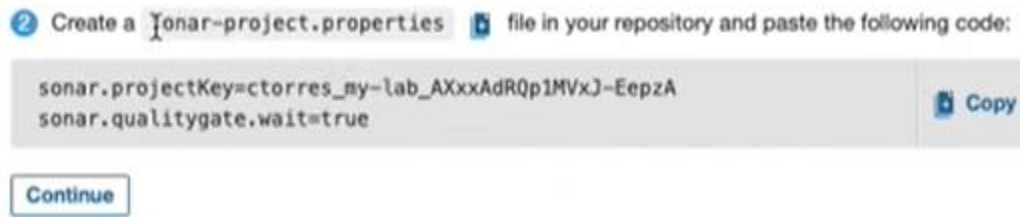
Maven

Gradle

.NET

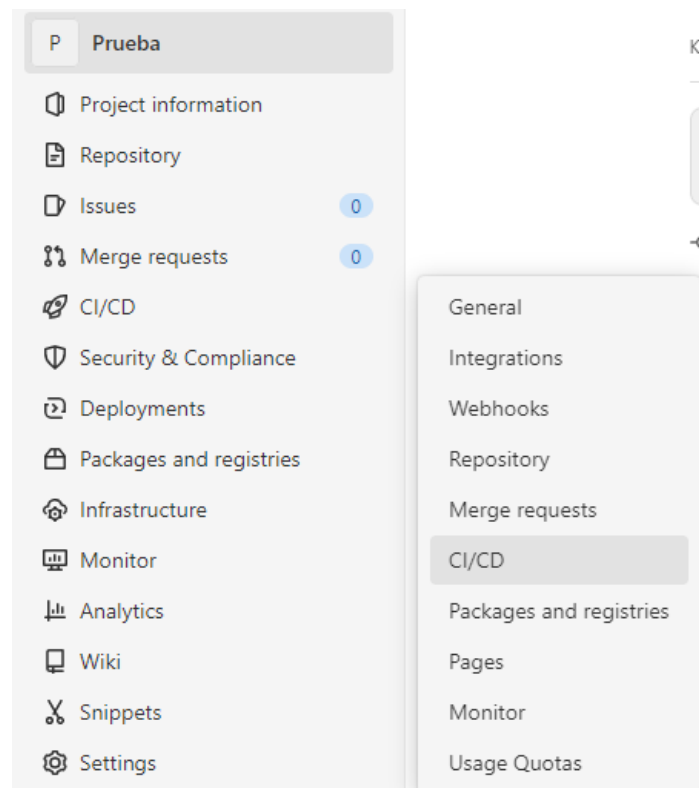
Other (for JS, TS, Go, Python, PHP, ...)

5.- Se debe de crear un archivo en la carpeta del proyecto con el contenido que se indique. El nombre del archivo debe de ser **sonar-project.properties**

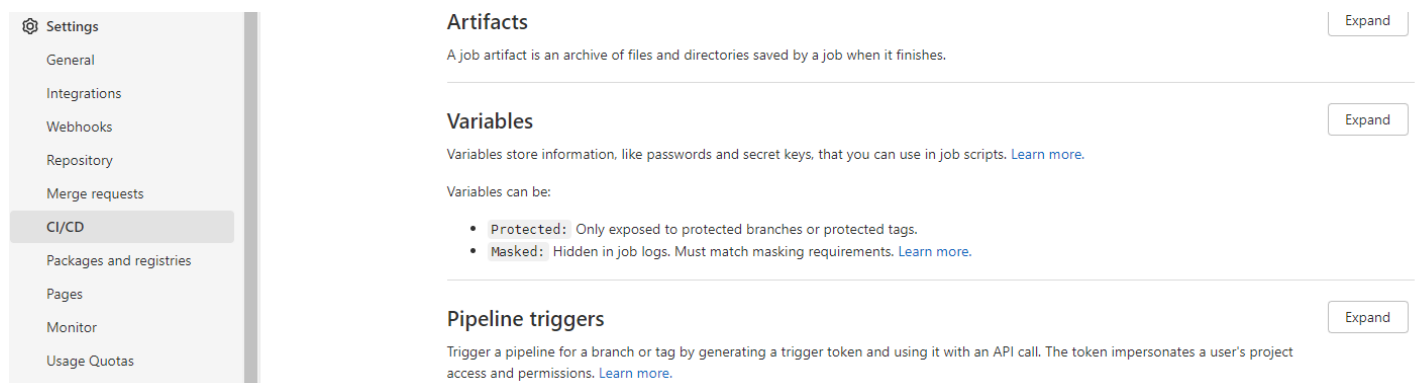


6.- Agregar variables de entorno

En Gitlab ir a la configuración del proyecto y CI/ CD



Ir a las variables y le damos al botón de **Expand**.



En esta parte podemos agregar una variable

Variables

Collapse

Variables store information, like passwords and secret keys, that you can use in job scripts. [Learn more](#).

Variables can be:

- **Protected:** Only exposed to protected branches or protected tags.
- **Masked:** Hidden in job logs. Must match masking requirements. [Learn more](#).

Environment variables are configured by your administrator to be **protected** by default.

Type	↑ Key	Value	Protected	Masked	Environments
There are no variables yet.					

Add variable

Agregamos las dos variables que nos piden, SONAR_TOKEN y SONAR_HOST_URL, estos dos son la *key*, en valor le ponemos el token que nos genera sonarqube.

Add variable

Key

Value

Type

Variable

Environment scope

All (default)

Flags

☒ Protect variable

Export variable to pipelines running on protected branches and tags only.

☐ Mask variable

Variable will be masked in job logs. Requires values to meet regular expression requirements. [More information](#)

Cancel

Add variable

Generate a token

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your [user account](#).

Analyze "my-lab" 2

Generate

Continue

Set your project key

Add environment variables

a. Define the SonarQube Token environment variable

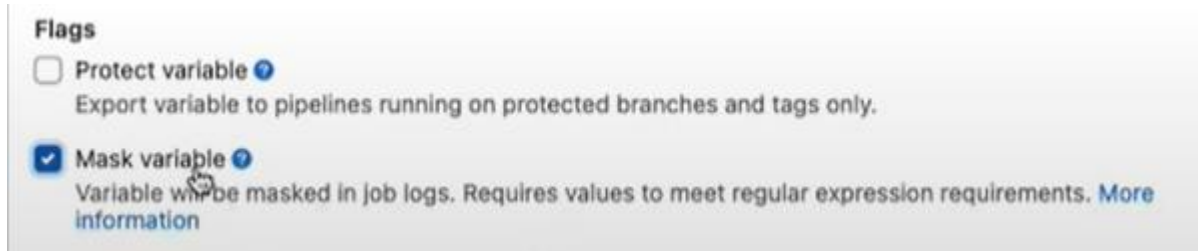
In GitLab, go to **Settings > CI/CD > Variables** to add

1 In the **Key** field, enter `SONAR_TOKEN`

2 In the **Value** field, enter an existing token, or a newly generated one:

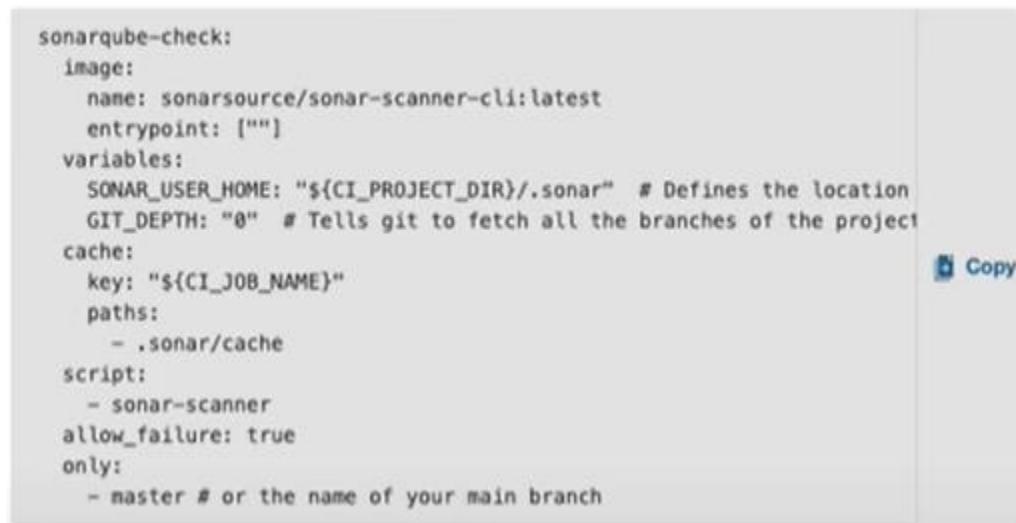
Generate a token

En *flags*, quitamos la opción de *Protect variable* y elegimos la opción de **Mask variable**.

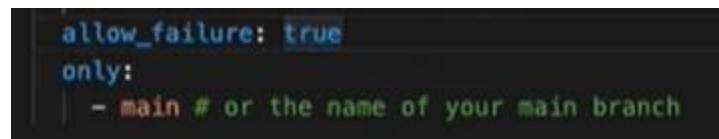


7.- Se debe de crear/actualizar el archivo `.gitlab-ci.yml` con el contenido que se nos muestra.

Create or update your `.gitlab-ci.yml` file with the following content.

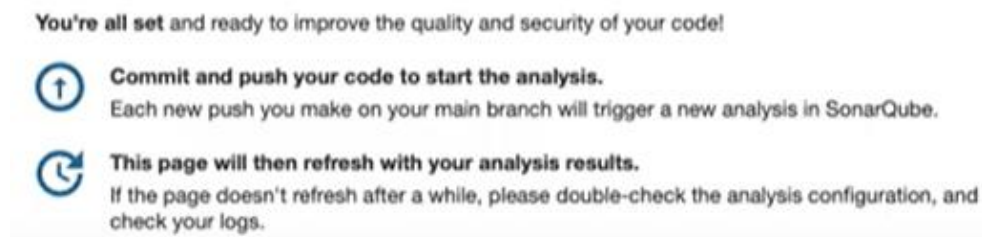


De este archivo modificamos la última línea, ya que nuestra rama es **main**.



8.- Terminamos el tutorial de SonarQube

SonarQube espera que Gitlab corra el pipeline para hacer la conexión del análisis. Debemos de hacer un *commit* y *push* de las modificaciones que se hicieron en Gitlab.



```
Enumerando objetos: 5, listo.
Contando objetos: 100% (5/5), listo.
Compresión delta usando hasta 12 hilos
Comprimiendo objetos: 100% (3/3), listo.
Escribiendo objetos: 100% (3/3), 374 bytes | 374.00 KiB/s, listo.
Total 3 (delta 1), reusado 0 (delta 0), pack-reusado 0
To http://gitlab.mylab.local/ctorres/my-lab.git
9b7d9bf..0b9e641 main -> main
```

9.- En Gitlab vamos a CI/CD-pipelines, vemos el pipeline se ejecuta automáticamente



El pipeline está en estado pendiente, porque no hay ningún runner que pueda ejecutar la acción.

10.- Configurar el runner

En el video, utilizan un servidor que se encarga de correr los runners con *vmware*,

Para instalar el runner, debemos de instalar primero Docker, ya que es un runner tipo Docker, de: <https://docs.docker.com/desktop/install/windows-install/>

Un requisito del sistema para poder instalar Docker es tener instalado el servidor WSL 2.

Requisitos del sistema

Su máquina con Windows debe cumplir con los siguientes requisitos para instalar correctamente Docker Desktop.

Servidor WSL 2

[Backend de Hyper-V y contenedores de Windows](#)

Servidor WSL 2

- Windows 11 de 64 bits: Home o Pro versión 21H2 o superior, o Enterprise o Education versión 21H2 o superior.
- Windows 10 de 64 bits: Home o Pro 21H1 (compilación 19043) o superior, o Enterprise o Education 20H2 (compilación 19042) o superior.
- Habilite la función WSL 2 en Windows. Para obtener instrucciones detalladas, consulte la [documentación de Microsoft](#).
- Se requieren los siguientes requisitos previos de hardware para ejecutar correctamente WSL 2 en Windows 10 o Windows 11:
 - Procesador de 64 bits con [traducción de direcciones de segundo nivel \(SLAT\)](#)
 - RAM del sistema de 4GB
 - El soporte de virtualización de hardware a nivel de BIOS debe estar habilitado en la configuración del BIOS. Para obtener más información, consulte [Virtualización](#).
- Descargue e instale el paquete de actualización del kernel de Linux.

Pasos para instalar Docker:

Instalar de forma interactiva

1. Haga doble clic en **Docker Desktop Installer.exe** para ejecutar el instalador.

Si aún no ha descargado el instalador (**Docker Desktop Installer.exe**), puede obtenerlo de [Docker Hub](#) . Por lo general, se descarga en su **Downloads** carpeta, o puede ejecutarlo desde la barra de descargas recientes en la parte inferior de su navegador web.

2. Cuando se le solicite, asegúrese de que la opción **Usar WSL 2 en lugar de Hyper-V** en la página Configuración esté seleccionada o no, según su elección de back-end.

Si su sistema solo admite una de las dos opciones, no podrá seleccionar qué backend usar.

3. Siga las instrucciones del asistente de instalación para autorizar al instalador y continuar con la instalación.

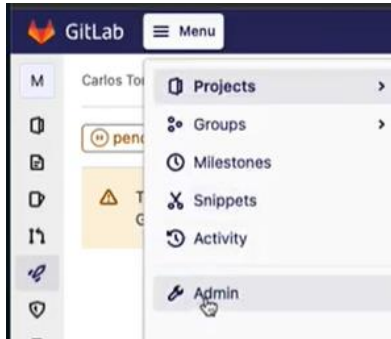
4. Cuando la instalación sea exitosa, haga clic en **Cerrar** para completar el proceso de instalación.

5. Si su cuenta de administrador es diferente a su cuenta de usuario, debe agregar el usuario al grupo **de usuarios de docker** . Ejecute **Administración de equipos** como **administrador** y vaya a **Usuarios y grupos locales > Grupos > usuarios de docker** . Haga clic derecho para agregar el usuario al grupo. Cierre sesión y vuelva a iniciarla para que los cambios surtan efecto.

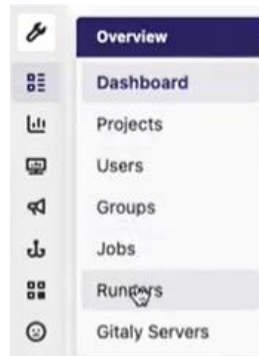
Para iniciar Docker, debemos de iniciar la aplicación y aceptar los términos. Al terminar la instalación de Docker, podemos instalar el runner de: <https://docs.gitlab.com/runner/install/windows.html> En la misma página vienen los pasos a seguir para la instalación.

Para la configuración del runner, se debe de registrar el runner. Hay 3 tipos de runners, compartidos, en grupo o para un proyecto específico. En este caso se hará compartido.
<https://docs.gitlab.com/runner/register/index.html#windows> En la misma página vienen los pasos a seguir para registrar el runner.

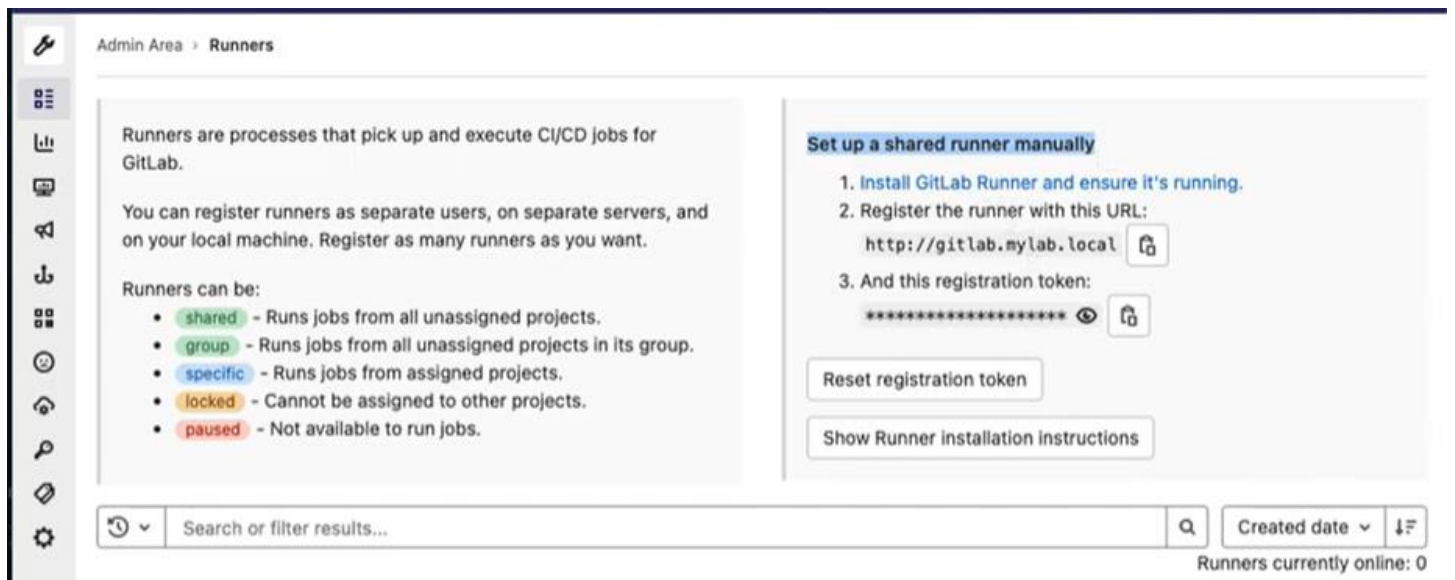
Nota: Para obtener el URL de la instancia de GitLab ir a menú/admin



De ahí a overview/runners



En esta parte está la información que se nos va a pedir durante el registro.



Cuando se termine el registro del runner, en la misma ventana (actualizar) deberá de aparecer el runner:

Admin Area > Runners

Runners are processes that pick up and execute CI/CD jobs for GitLab.

You can register runners as separate users, on separate servers, and on your local machine. Register as many runners as you want.

Runners can be:

- **shared** - Runs jobs from all unassigned projects.
- **group** - Runs jobs from all unassigned projects in its group.
- **specific** - Runs jobs from assigned projects.
- **locked** - Cannot be assigned to other projects.
- **paused** - Not available to run jobs.

Set up a shared runner manually

1. Install GitLab Runner and ensure it's running.
2. Register the runner with this URL:
`http://gitlab.mylab.local`
3. And this registration token:

Reset registration token

Show Runner installation instructions

Search or filter results... Created date

Runners currently online: 1

Type/State	Runner	Version	IP Address	Projects/jobs	Tags	Last contact
shared	#1 (QNs7Fsss) pro-runner-docker	14.3.2	172.16.1.5	n/a 0	pro-runner-docker	14 seconds ago

11.- Modificar archivo `.gitlab-ci.yml`

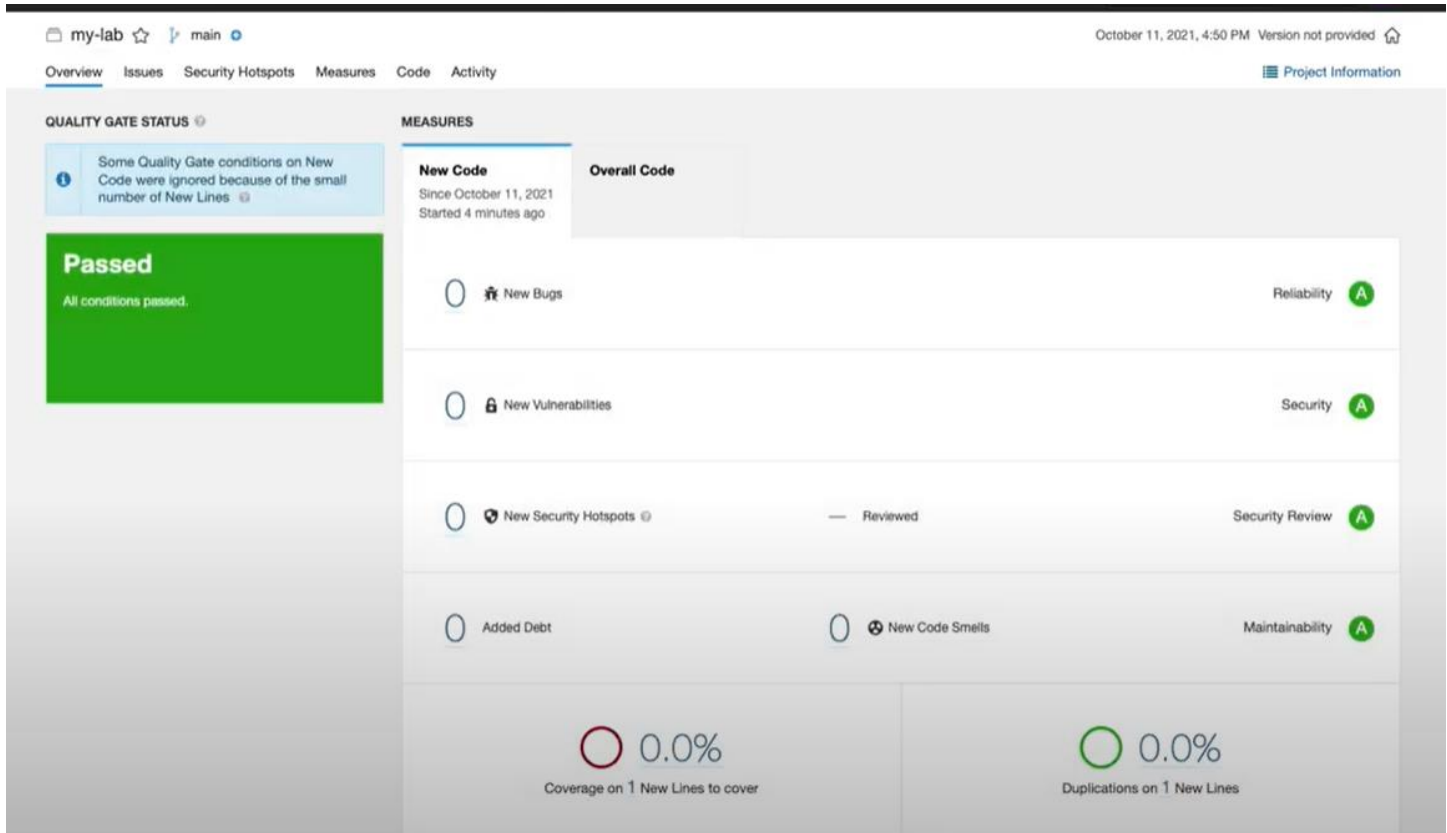
Se agregan dos líneas en el que se va a indicar el nombre del runner, en este caso fue *pro-runner-docker*.

```
15   only:
16     - main # or the name of your main branch
17   tags:
18     - pro-runner-docker
19
```

Se suben los cambios con un commit y push.

Cuando vamos al pipeline, debe de aparecer un nuevo pipeline, que va a tener estado pendiente, después de un tiempo va a empezar a correr hasta que, si no hay ningún error, debe de aparecer el estatus como pasado.

Con esto ya podemos ir a sonarqube, y ver el análisis del proyecto.



Cada que le hagamos un cambio al código, debemos de subir el archivo actualizado a GitLab con un commit y push, cuando se realice esto, debe de aparecer un nuevo pipeline. De igual forma, va a tener estado pendiente, después de un tiempo va a empezar a correr hasta que, si no hay ningún error, debe de aparecer el estatus como pasado, con esto podemos ver el nuevo análisis del proyecto.