GUIA PASO 1

En este paso vamos a obtener 4 valores de las variables secretas que vamos a crear en nuestro Git.

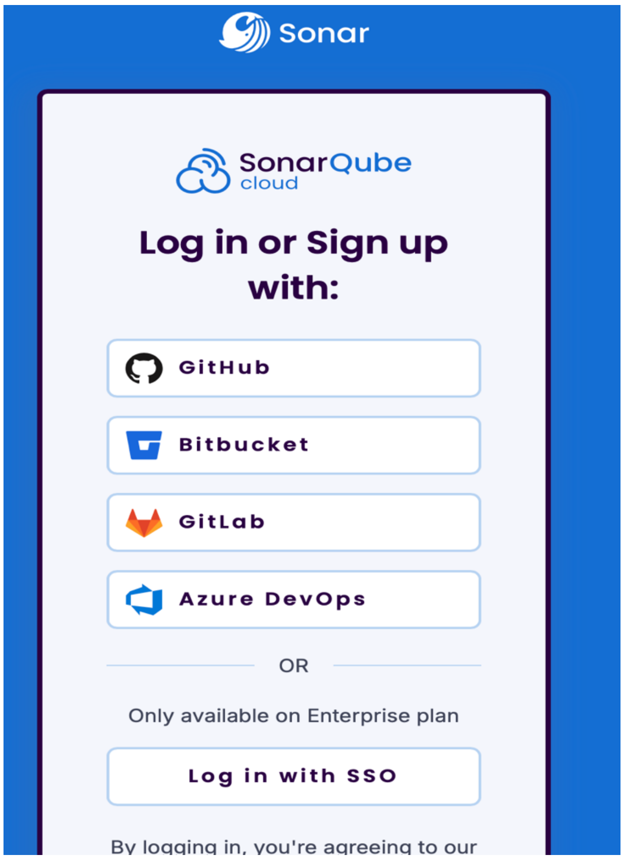
SONAR_HOST_URL

SONAR_ORGANIZATION

SONAR_PROJECT_KEY

SONAR_TOKEN

1. Accedemos al Sonar iniciando sesión desde la cuenta donde vamos a tener nuestro repositorio. Iniciamos sesión con uno de nuestros GIT.

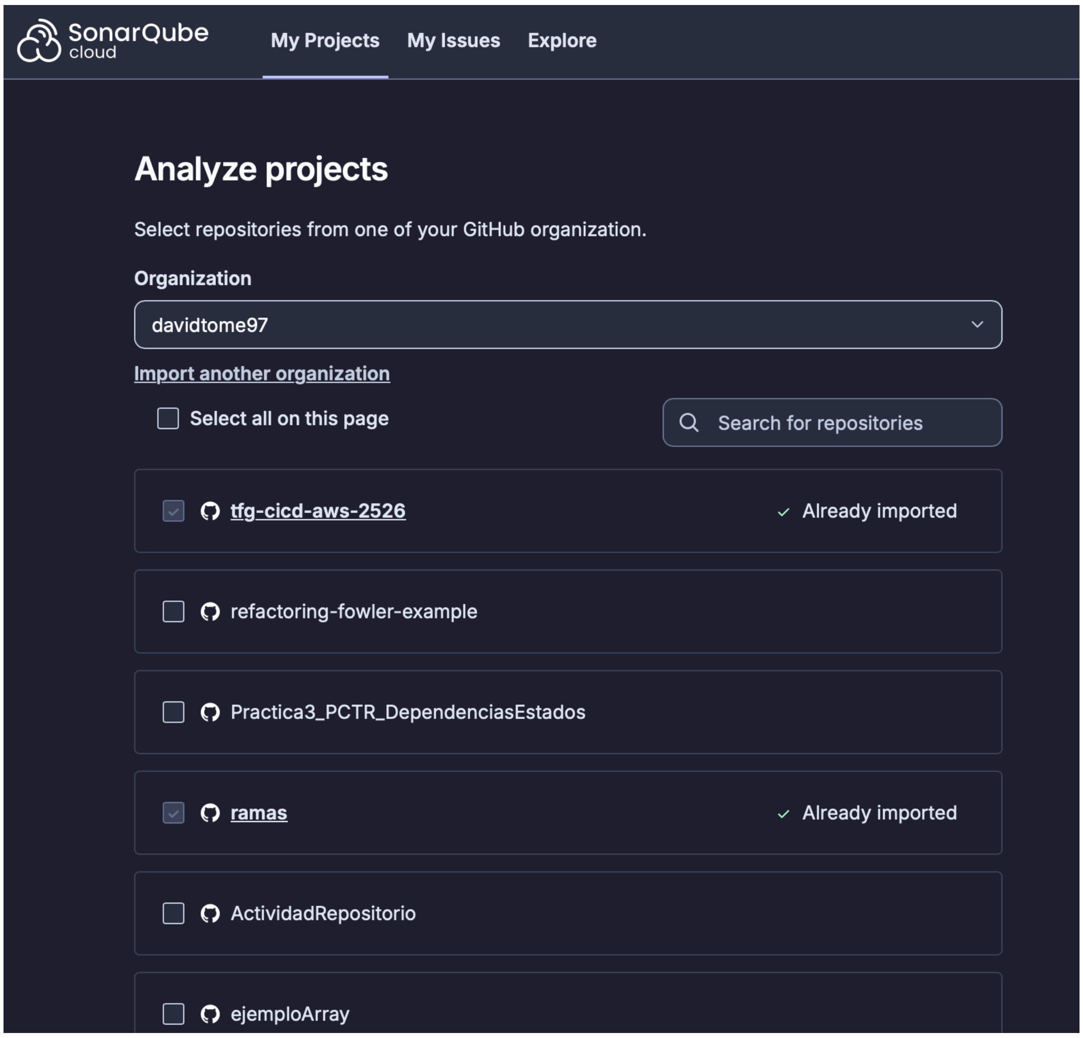


Log in or Sign up with:

Only available on Enterprise plan

By logging in, you're agreeing to our

2. Si no tenemos el proyecto creado todavía, tenemos que dar en el + que nos aparece en la esquina superior derecha y elegimos la opción “analyze new proyect”. Dentro de esa opción elegimos nuestro repositorio que queremos analizar. Aquí vemos ya el valor de nuestra variable “SONAR_ORGANIZATION”.



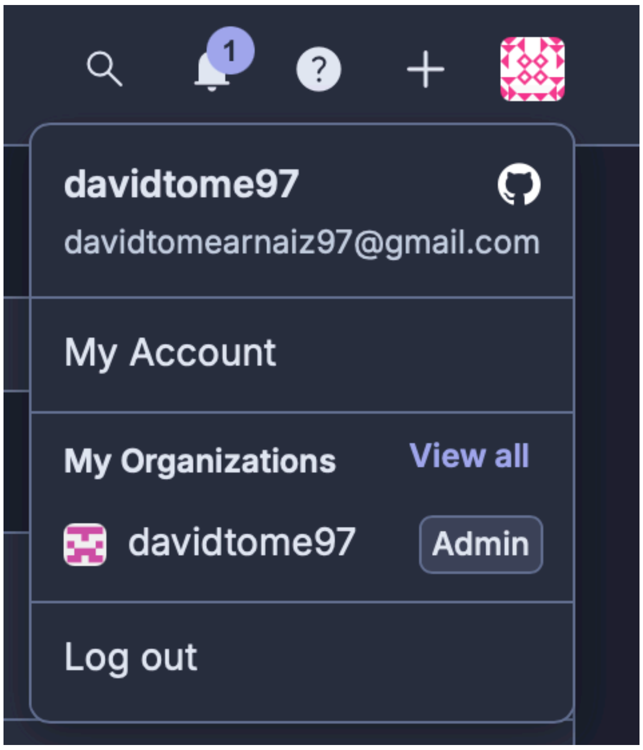
Analyze projects

Select repositories from one of your GitHub organization.

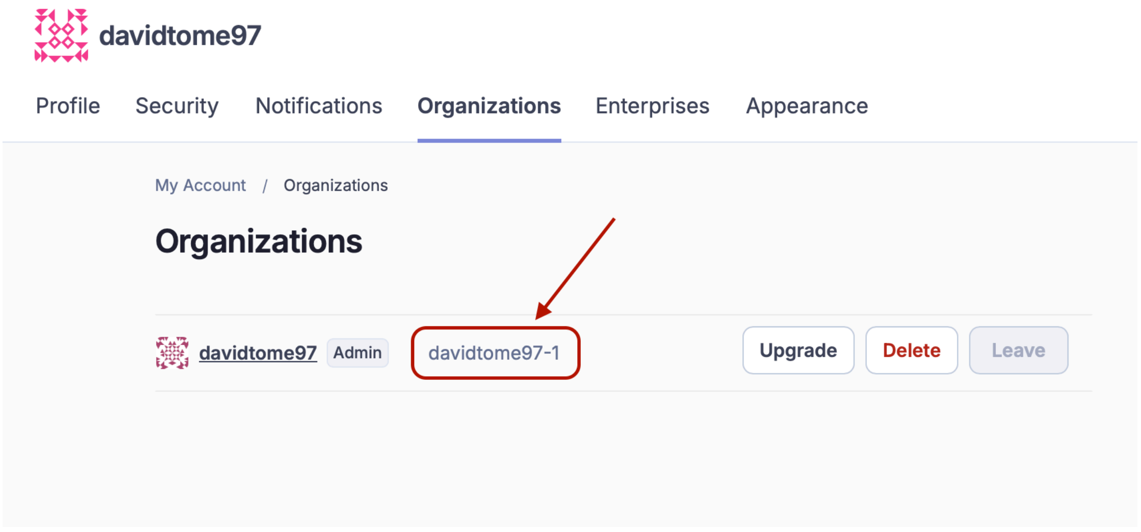
Organization: davidtome97

- [✓] tfg-cicd-aws-2526 — Already imported
- [ ] refactoring-fowler-example
- [ ] Practica3_PCTR_DependenciasEstados
- [✓] ramas — Already imported
- [ ] ActividadRepositorio
- [ ] ejemploArray

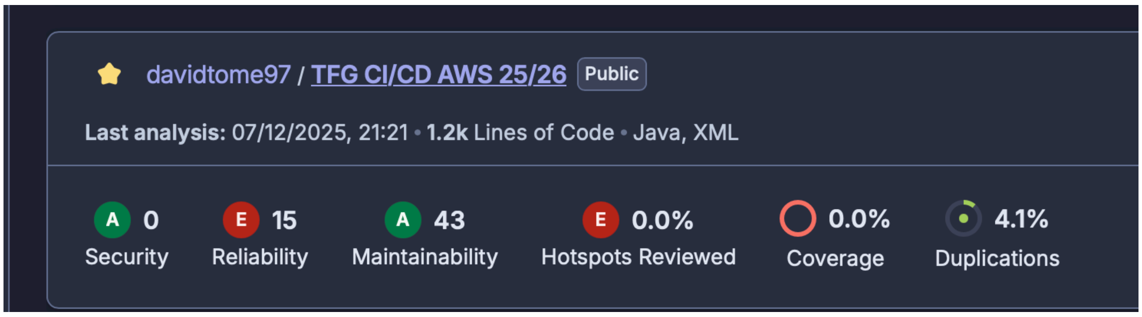
3. En la esquina superior derecha, si damos sobre nuestra foto de perfil, podremos ver una opción que pone: “View all”, pulsamos en el y ahí nos aseguramos de ver nuestro valor de la variable “SONAR_ORGANIZATION”.


davidtome97
davidtomearnaiz97@gmail.com
My Account
My Organizations
View all
davidtome97
Admin
Log out

4. Esa de ahí abajo seria nuestra organización.


davidtome97
Profile
Security
Notifications
Organizations
Enterprises
Appearance
My Account / Organizations
Organizations
davidtome97 Admin davidtome97-1
Upgrade
Delete
Leave

5. Una vez que ya tenemos nuestro proyecto y nuestra organización, vamos a ver el valor de nuestra variable "SONAR_PROJECT_KEY". Para ello damos en nuestro nombre del proyecto, en este caso "TFG CI/CD AWS 25/26".

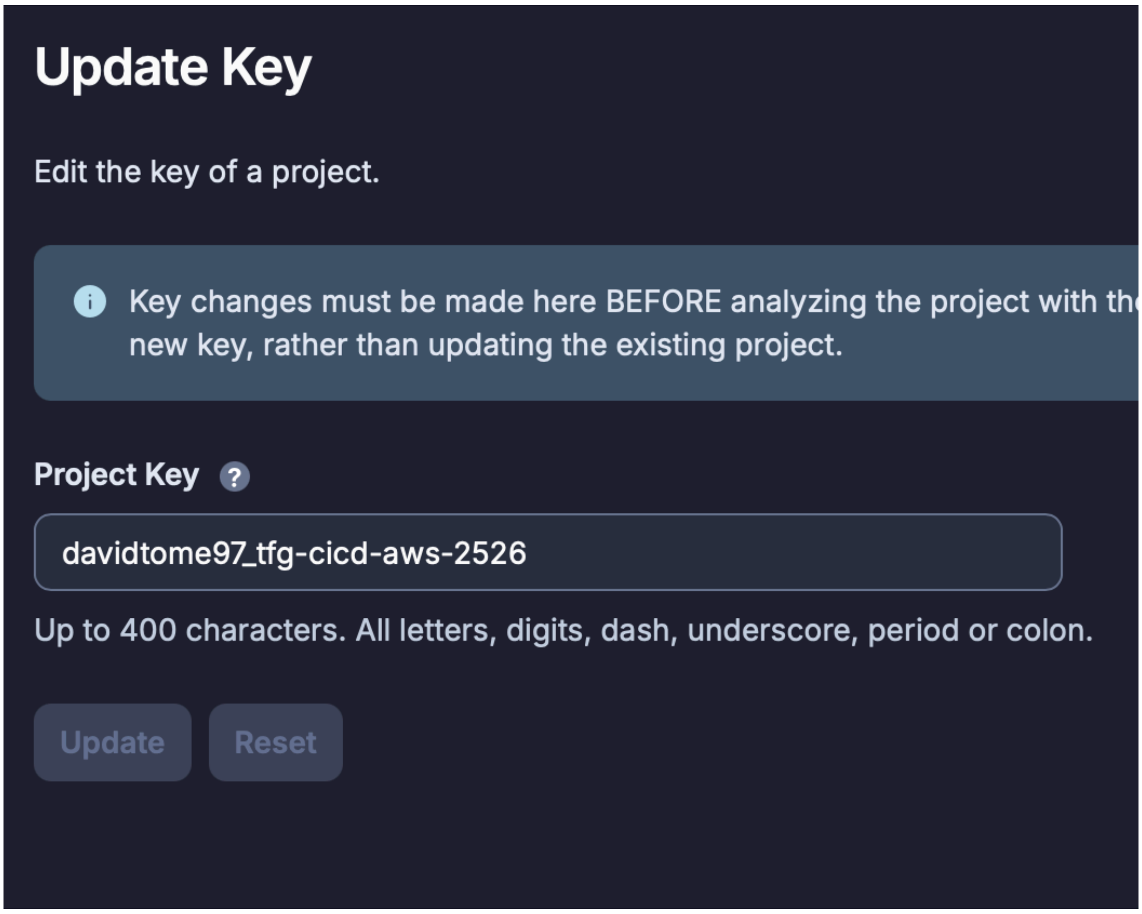


davidtome97 / TFG CI/CD AWS 25/26 Public

Last analysis: 07/12/2025, 21:21 • 1.2k Lines of Code • Java, XML

A 0	E 15	A 43	E 0.0%	0.0%	4.1%
Security	Reliability	Maintainability	Hotspots Reviewed	Coverage	Duplications

6. Dentro de nuestro proyecto, en la esquina inferior izquierda pulsamos en Administration →Update Key. Se nos abrirá una ventana donde aparece nuestro valor de "SONAR_PROJECT_KEY".



Update Key

Edit the key of a project.

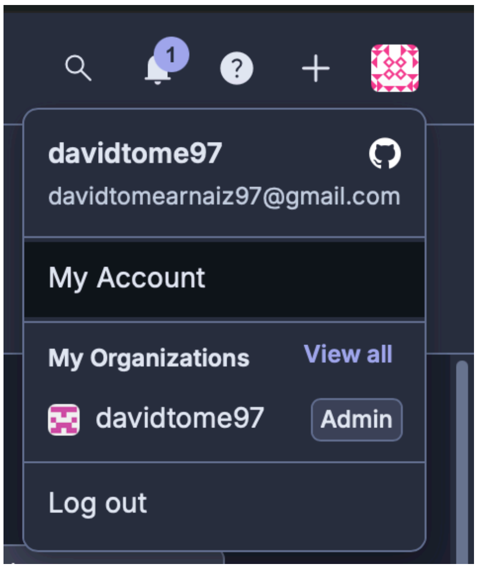
Key changes must be made here BEFORE analyzing the project with th
new key, rather than updating the existing project.

Project Key

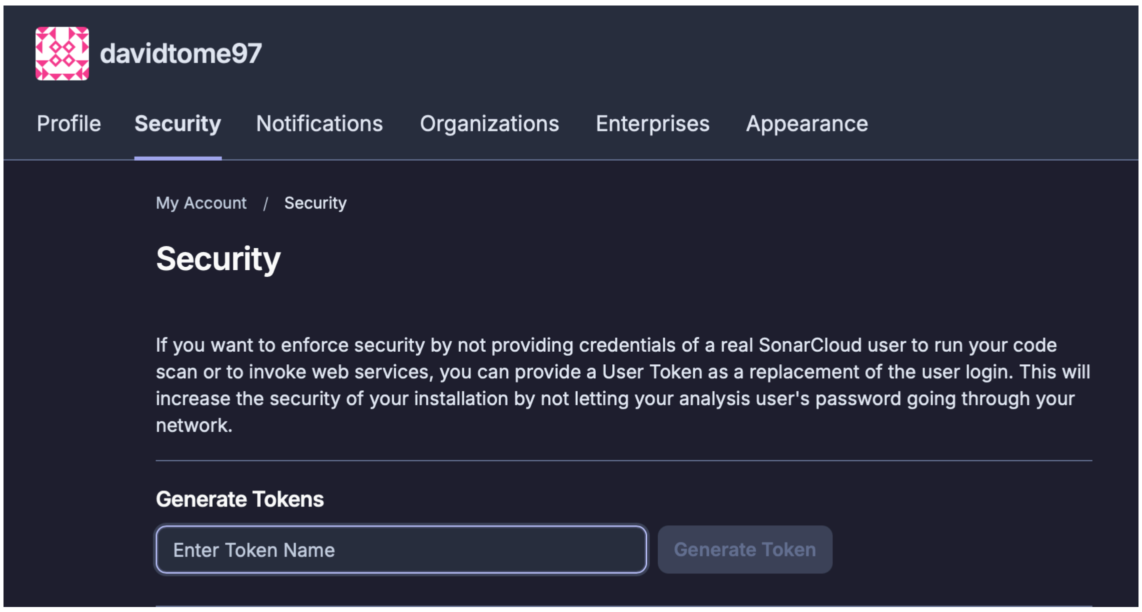
davidtome97_tfg-cicd-aws-2526

Up to 400 characters. All letters, digits, dash, underscore, period or colon.

7. Ya solo nas falta crear el token de Sonar. Para ello, desde el incio de sonar. Si damos a nuestra foto de perfil (parte superior derecha) → My Account



8. Se nos abre una ventana con diferentes pestañas, pulsamos en la de "Security". Escribimos un nombre en el campo "generate tokens" y damos al botón de generar. Copia ese token que te va a dar ¡¡OJO, ESE TOKEN SOLO LO VAS A VER UNA VEZ!! Si se te olvida, tendrás que generar otro nuevo. Aquí ya tendríamos el valor de nuestra variable "SONAR_TOKEN"



Security

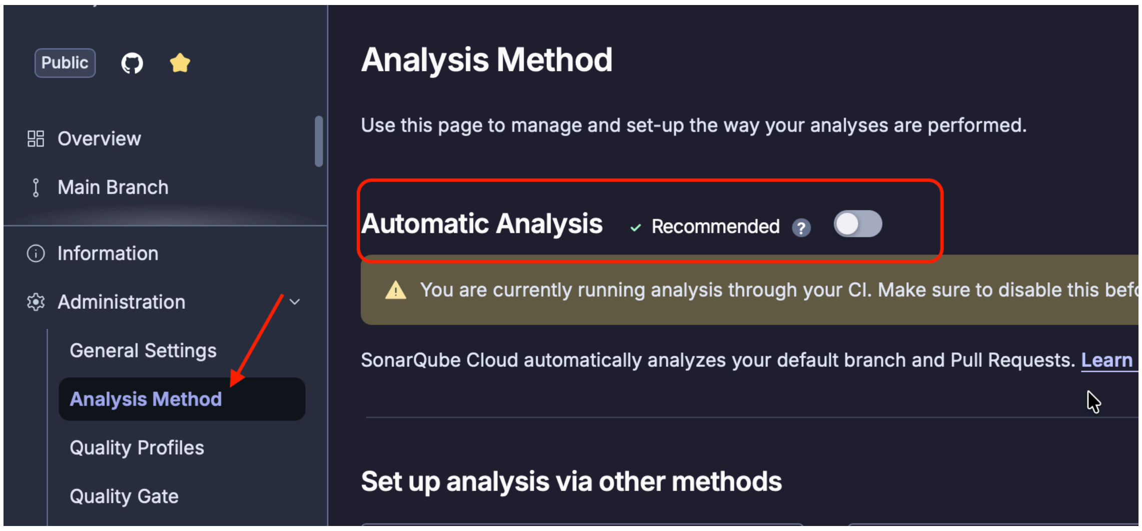
If you want to enforce security by not providing credentials of a real SonarCloud user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

Generate Tokens

9. Por defecto, el valor de nuestra variable "SONAR_HOST_URL" es https://sonarcloud.io

10. Recomiendo desactivar el análisis automáticos para que no de fallo.

En nuestro proyecto



Analysis Method

Use this page to manage and set-up the way your analyses are performed.

Automatic Analysis ✓ Recommended

You are currently running analysis through your CI. Make sure to disable this bef

SonarQube Cloud automatically analyzes your default branch and Pull Requests. Learn

Set up analysis via other methods

Con esto dariamos como finalizado la obtención de nuestros 4 valores que vamos a crear ahora en Git:

SONAR_ORGANIZATION (paso 2-3)

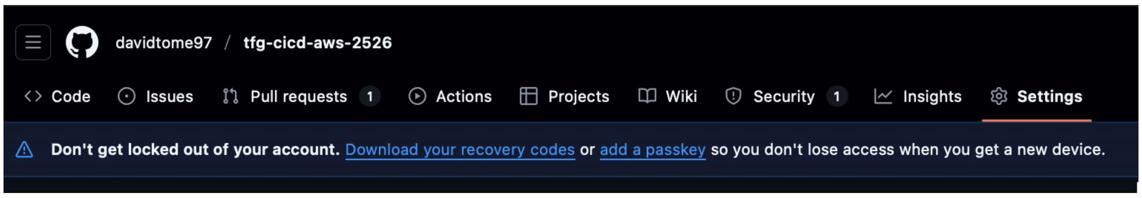
SONAR_PROJECT_KEY (paso 5)

SONAR_TOKEN (paso 7)

SONAR_HOST_URL (paso 8)

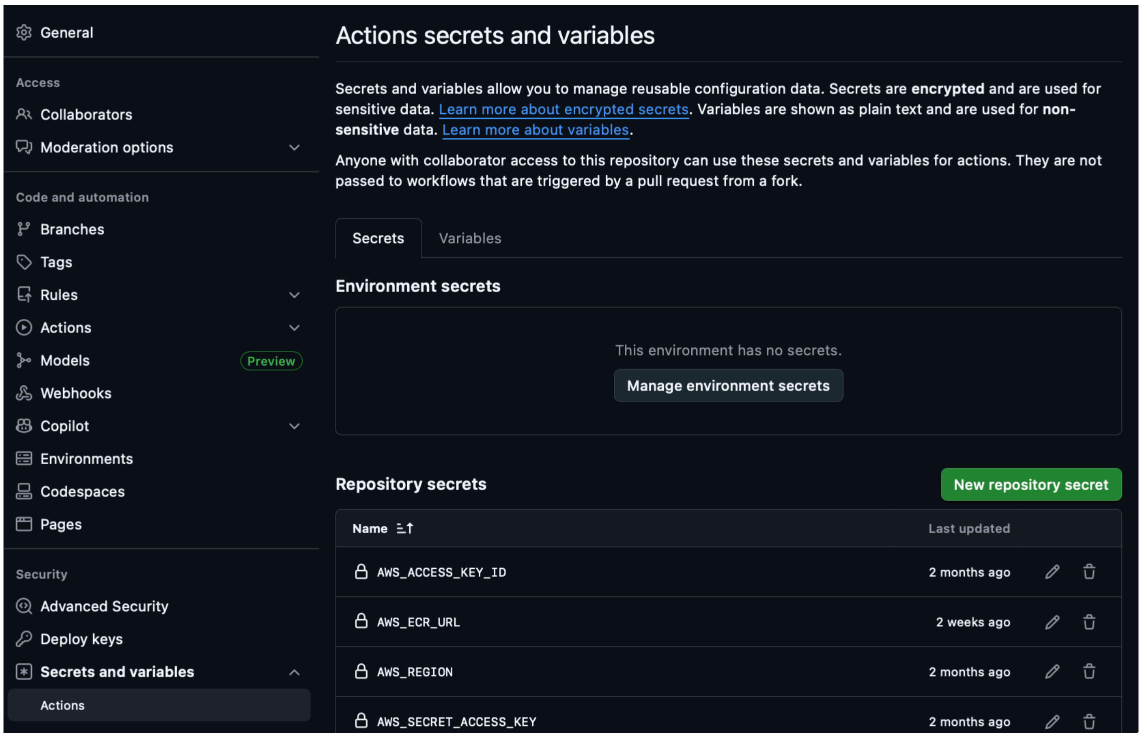
¿Cómo creamos nuestras variables secretas en nuestro github?

1. Iniciamos sesión en nuestro github→ accedemos a nuestro repositorio con el que queremos trabajar →settings.


davidtome97 / tfg-cicd-aws-2526

Don't get locked out of your account. Download your recovery codes or add a passkey so you don't lose access when you get a new device.

2. En el menú lateral, vamos hasta la opción de security → secrets and variables → actions. Ahí en la pestaña de "secrets", damos al botón "New repository secret".



Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. Learn more about encrypted secrets. Variables are shown as plain text and are used for **non-sensitive data**. Learn more about variables.

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

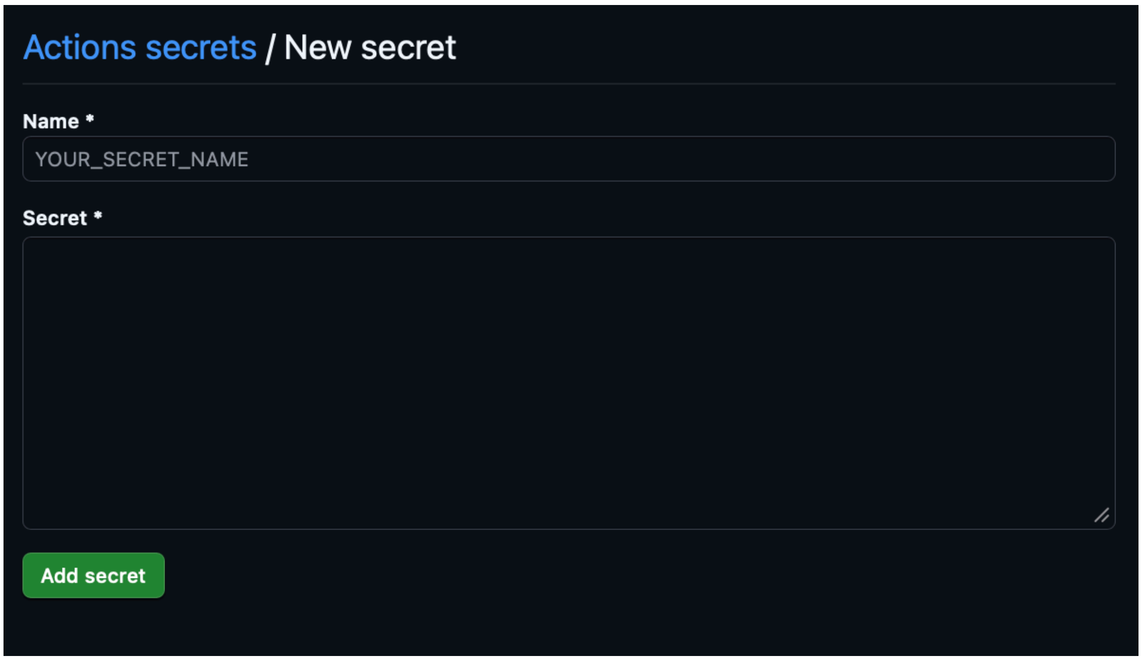
Environment secrets

This environment has no secrets.

Repository secrets

Name	Last updated
AWS_ACCESS_KEY_ID	2 months ago
AWS_ECR_URL	2 weeks ago
AWS_REGION	2 months ago
AWS_SECRET_ACCESS_KEY	2 months ago

3. Escribimos el nombre de las variables que te he proporcionado y el valor de cada una de ellas que hemos obtenido antes y damos a “add secret”. ¡¡MUY IMPORTANTE, EL NOMBRE TIENE QUE SER EL MISMO QUE TE HE PROPORCIONADO AL PRINCIPIO DEL MENÚ!!

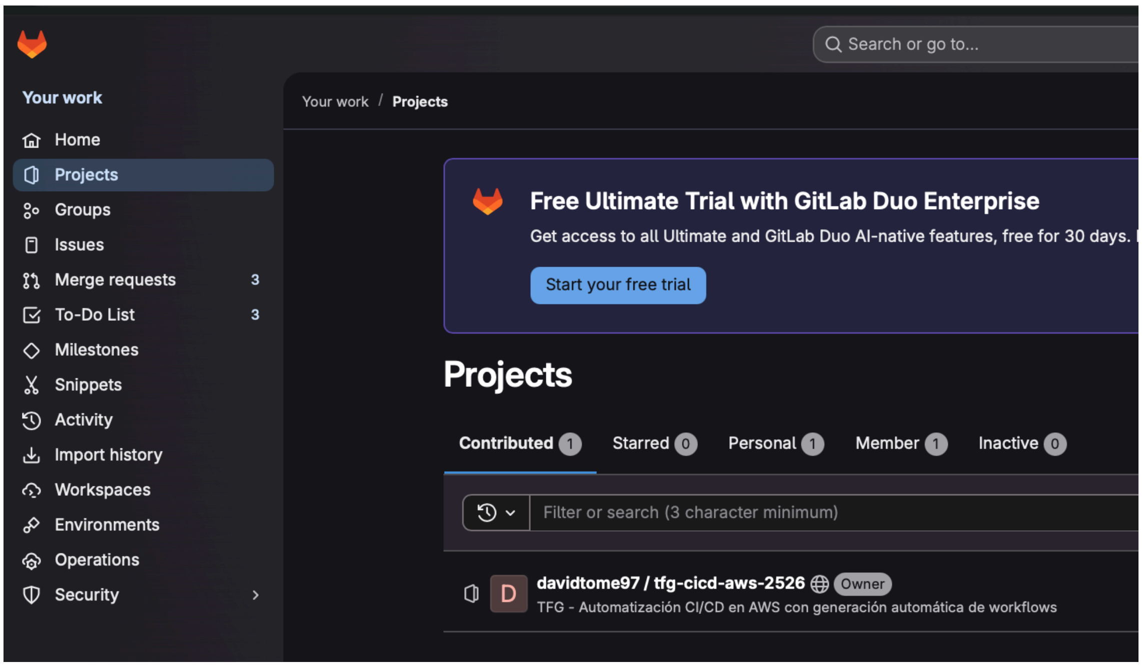


Una vez que creamos todas las variables de este paso hay que hacer un commit para que coja los cambios realizados y no nos falle el siguiente paso:

```
git commit --allow-empty -m "creación de variables”
git push
```

¿Cómo creamos nuestras variables secretas en nuestro gitlab?

1. Iniciamos sesión en nuestro gitlab→ projects→ nuestro proyecto.

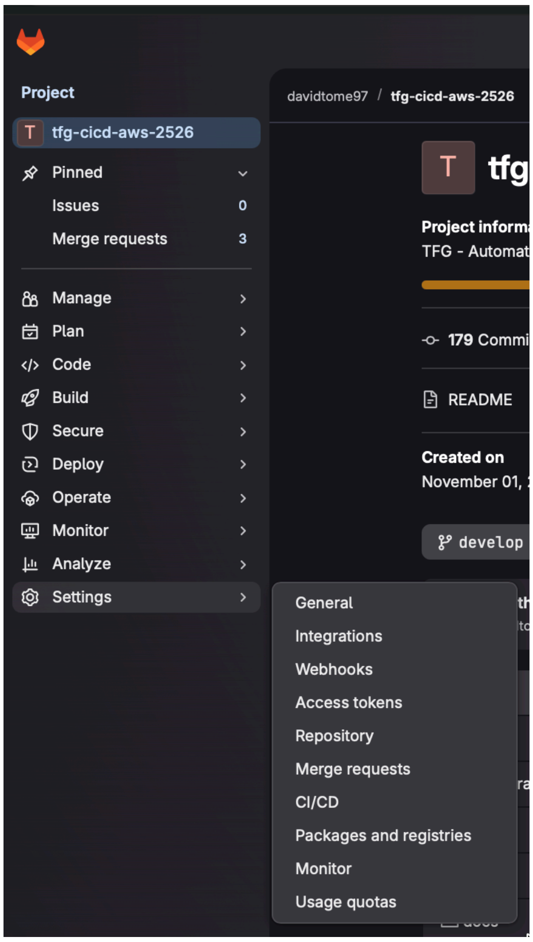


Get access to all Ultimate and GitLab Duo AI-native features, free for 30 days.

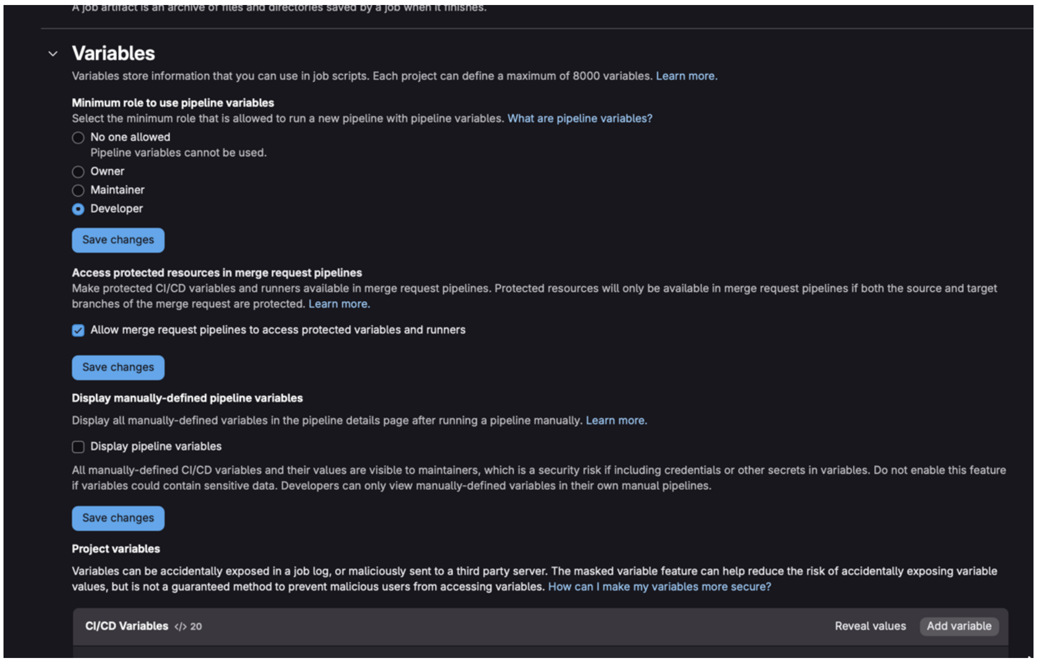
davidtome97 / tfg-cicd-aws-2526

TFG - Automatización CI/CD en AWS con generación automática de workflows

2. En nuestro menú lateral, seleccionamos Settings→CI/CD. Se nos abrirá una nueva ventana.



3. Desplegamos la opción “Variables” y vemos una opción que pone “add variable”.



A job artifact is an archive of files and directories saved by a job when it finishes.

Variables

Variables store information that you can use in job scripts. Each project can define a maximum of 8000 variables. Learn more.

Minimum role to use pipeline variables

Select the minimum role that is allowed to run a new pipeline with pipeline variables. What are pipeline variables?

- No one allowed
  Pipeline variables cannot be used.
- Owner
- Maintainer
- [✓ Developer]

Save changes

Access protected resources in merge request pipelines

Make protected CI/CD variables and runners available in merge request pipelines. Protected resources will only be available in merge request pipelines if both the source and target branches of the merge request are protected. Learn more.

[✓ Allow merge request pipelines to access protected variables and runners]

Save changes

Display manually-defined pipeline variables

Display all manually-defined variables in the pipeline details page after running a pipeline manually. Learn more.

Display pipeline variables

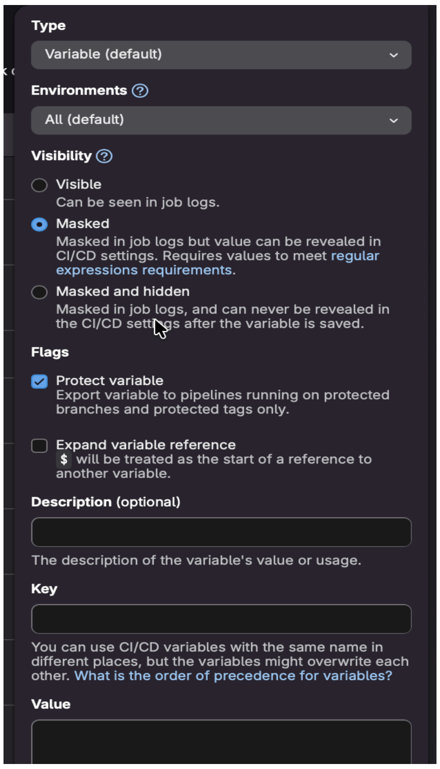
All manually-defined CI/CD variables and their values are visible to maintainers, which is a security risk if including credentials or other secrets in variables. Do not enable this feature if variables could contain sensitive data. Developers can only view manually-defined variables in their own manual pipelines.

Save changes

Project variables

Variables can be accidentally exposed in a job log, or maliciously sent to a third party server. The masked variable feature can help reduce the risk of accidentally exposing variable values, but is not a guaranteed method to prevent malicious users from accessing variables. How can I make my variables more secure?

4. Elegimos el tipo que queremos, la visibility que queremos para esta variable, en “key” ponemos el nombre de la variable y en “value” su valor que hemos obtenido anteriormente.
   ¡¡MUY IMPORTANTE, EL NOMBRE TIENE QUE SER EL MISMO QUE TE HE PROPORCIONADO AL PRINCIPIO DEL MENÚ!!



Type

Variable (default)

Environments

All (default)

Visibility

- Visible
  Can be seen in job logs.
- [✓ Masked]
  Masked in job logs but value can be revealed in CI/CD settings. Requires values to meet regular expressions requirements.
- Masked and hidden
  Masked in job logs, and can never be revealed in the CI/CD settings after the variable is saved.

Flags

- [✓ Protect variable]
  Export variable to pipelines running on protected branches and protected tags only.
- Expand variable reference
  $ will be treated as the start of a reference to another variable.

Description (optional)

The description of the variable's value or usage.

Key

You can use CI/CD variables with the same name in different places, but the variables might overwrite each other. What is the order of precedence for variables?

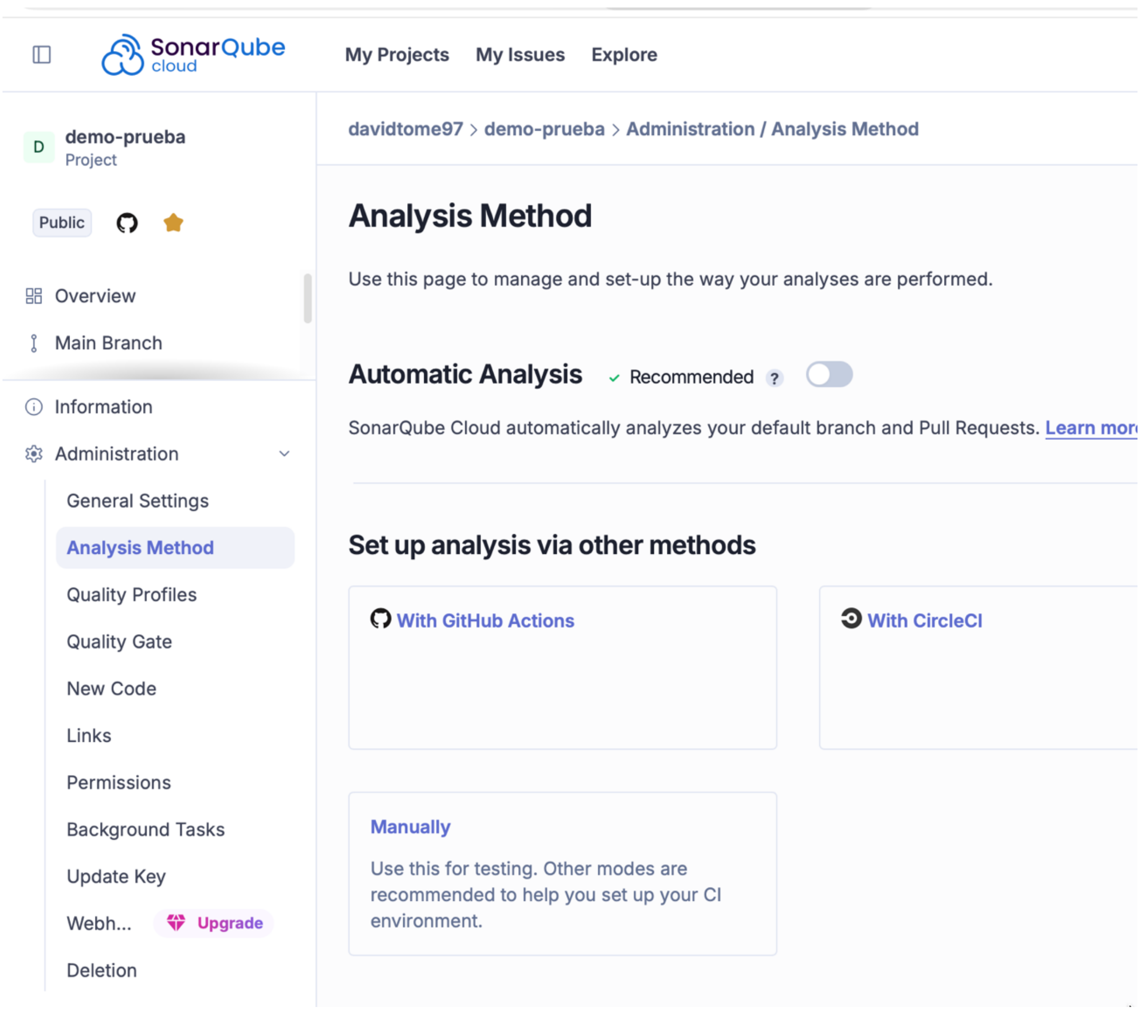
Value

Una vez que creamos todas las variables de este paso hay que hacer un commit para que coja los cambios realizados y no nos falle el siguiente paso:

git commit --allow-empty -m "creación de variables CI/CD"

git push

Si nos da error al pasar nuestro proceso de ci/cd, desactivar los análisis automáticos de sonar. Para ello entramos en nuestro proyecto en sonar→ en el menú lateral→ administration → Analysis Method→ y desactivamos la opción "Automatic Analysis".



Analysis Method

Use this page to manage and set-up the way your analyses are performed.

Automatic Analysis ✓ Recommended

SonarQube Cloud automatically analyzes your default branch and Pull Requests. Learn mor

Set up analysis via other methods

With GitHub Actions

With CircleCI

Manually

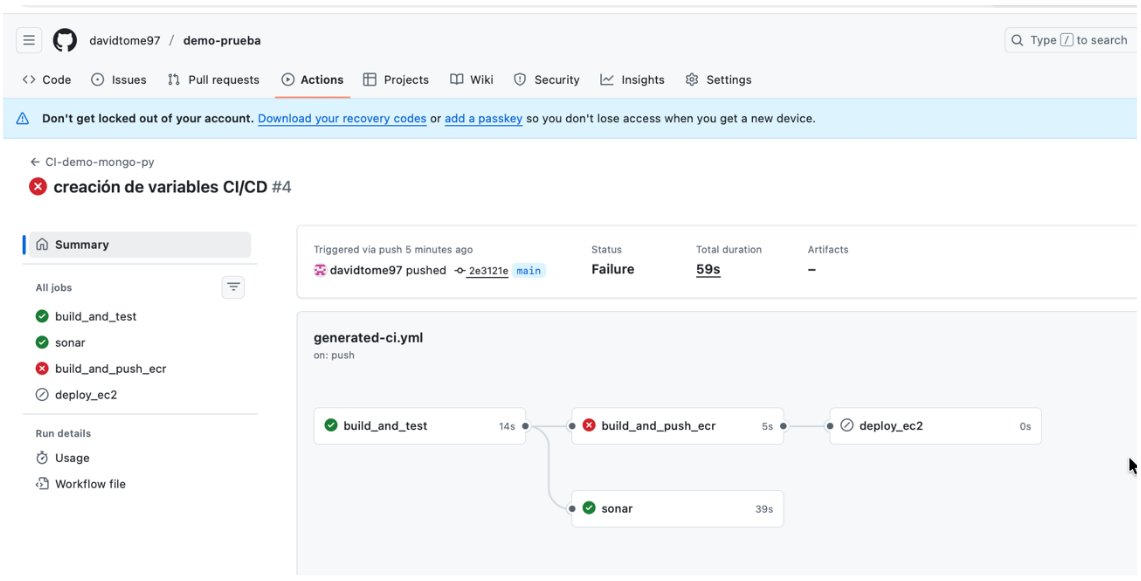
Use this for testing. Other modes are recommended to help you set up your CI environment.

Volvemos a ejecutar el commit por terminal para ver que realmente pasa bien el sonar:

git commit --allow-empty -m "creación de variables CI/CD"

git push

Deberíamos ver algo asi:



davidtome97 / demo-prueba

Type / to search

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

Don't get locked out of your account. Download your recovery codes or add a passkey so you don't lose access when you get a new device.

← CI-demo-mongo-py

creación de variables CI/CD #4

Summary

All jobs
- ✓ build_and_test
- ✓ sonar
- ✗ build_and_push_ecr
- ⊘ deploy_ec2

Run details
- Usage
- Workflow file

Triggered via push 5 minutes ago
davidtome97 pushed 2e3121e main

Status: Failure
Total duration: 59s
Artifacts: –

generated-ci.yml
on: push

✓ build_and_test 14s → ✗ build_and_push_ecr 5s → ⊘ deploy_ec2 0s

✓ build_and_test 14s → ✓ sonar 39s