

Mise en place d'un pipeline CI/CD

CONFIGURATION DE L'ENVIRONNEMENT

Utiliser ngrok pour créer une adresse publique

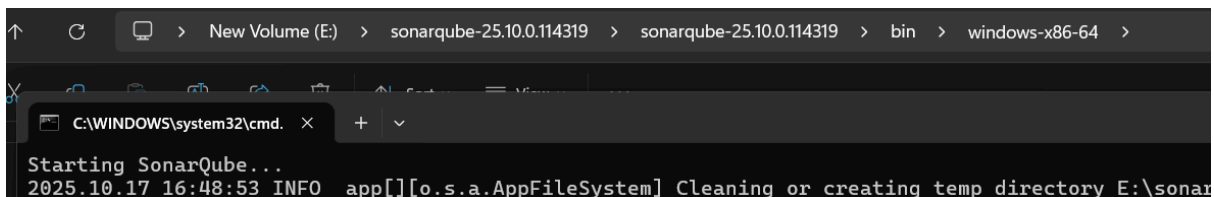
```
♦ Call internal services from your gateway: https://ngrok.com/r/http-request

Session Status      online
Account             ahmedkamel.benzid@ensi-uma.tn (Plan: Free)
Version             3.24.0-msix
Region              Europe (eu)
Latency              905ms
Web Interface        http://127.0.0.1:4040
Forwarding            https://doctrinally-lawlike-otha.ngrok-free.dev -> http://localhost:8080

Connections          ttl    opn    rt1    rt5    p50    p90
                    2      0      0.00   0.00   30.73  31.25
```

Objectif : Rendre Jenkins accessible depuis GitHub

Démarrage SonarQube



The screenshot shows a file explorer window with the path: New Volume (E:) > sonarqube-25.10.0.114319 > sonarqube-25.10.0.114319 > bin > windows-x86-64 >. Below the file explorer, a terminal window is open with the command prompt showing the execution of the SonarQube startup command. The output indicates that the application is starting and cleaning or creating a temporary directory.

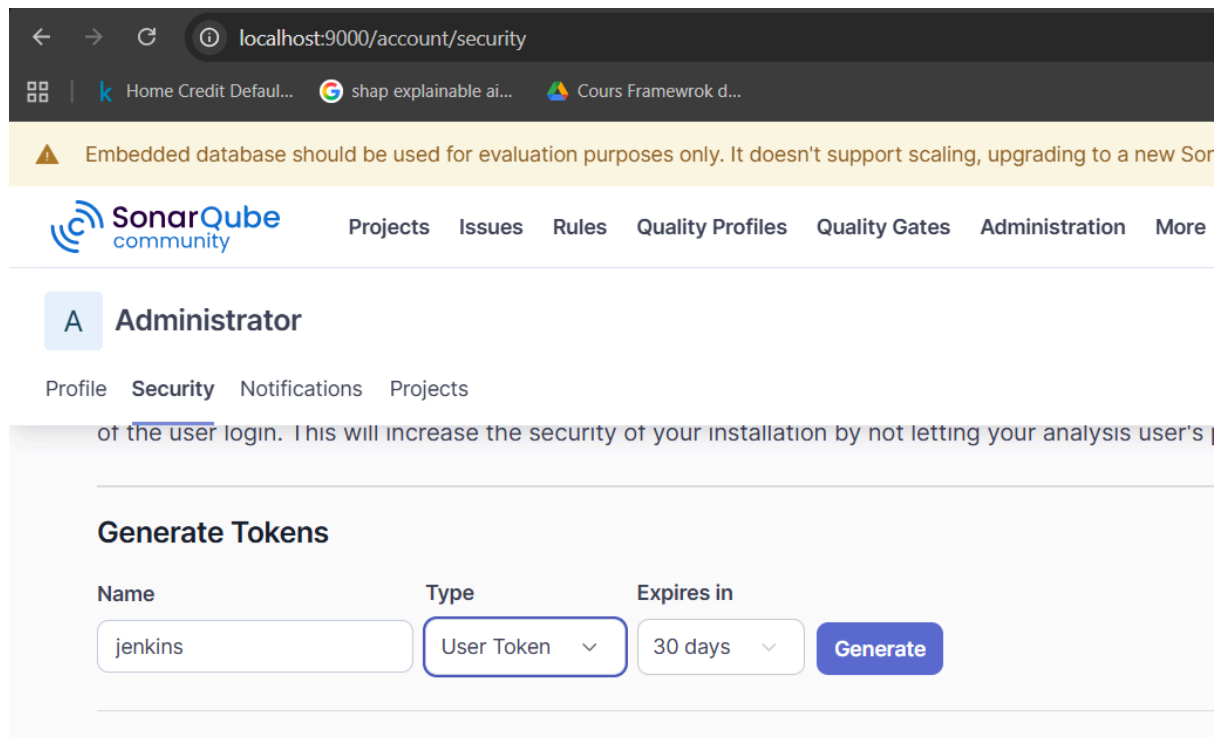
```
C:\WINDOWS\system32\cmd. x + v

Starting SonarQube...
2025.10.17 16:48:53 INFO app[][o.s.a.AppFileSystem] Cleaning or creating temp directory E:\sonar
```

Génération Token SonarQube

Objectif : Créer un token pour l'authentification Jenkins

Configuration :

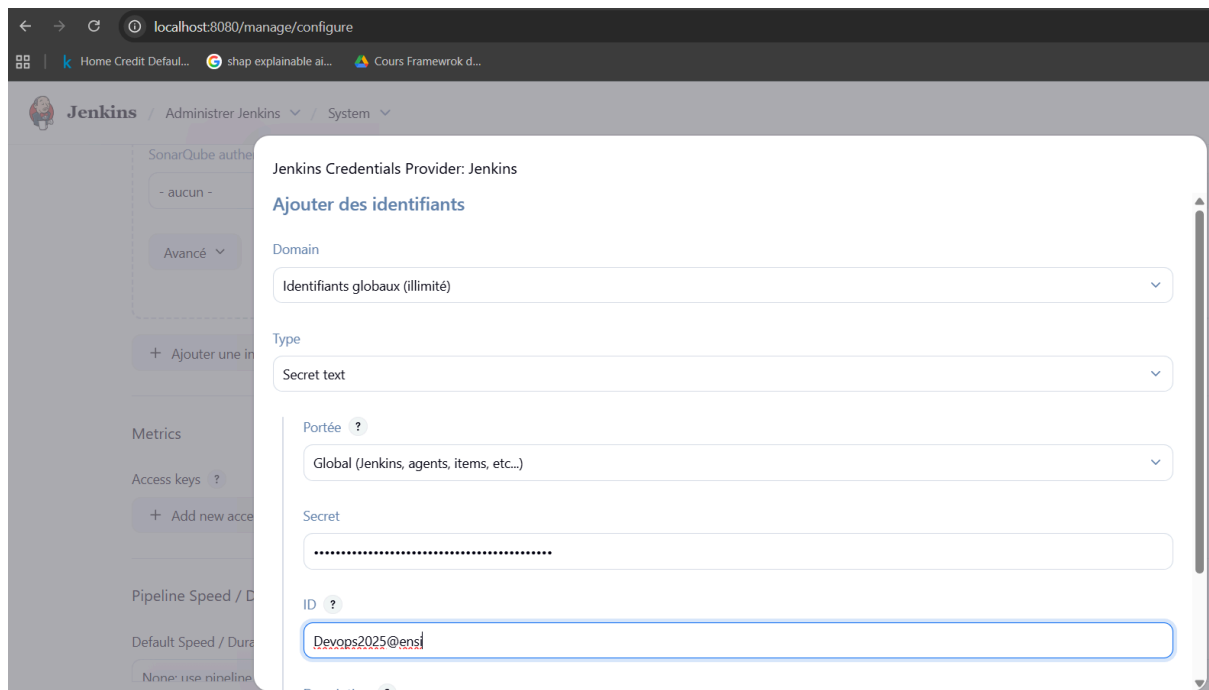


Nom : jenkins

Type : User Token

CONFIGURATION JENKINS

Identifiants SonarQube



Objectif : Stocker sécuriséement le token SonarQube

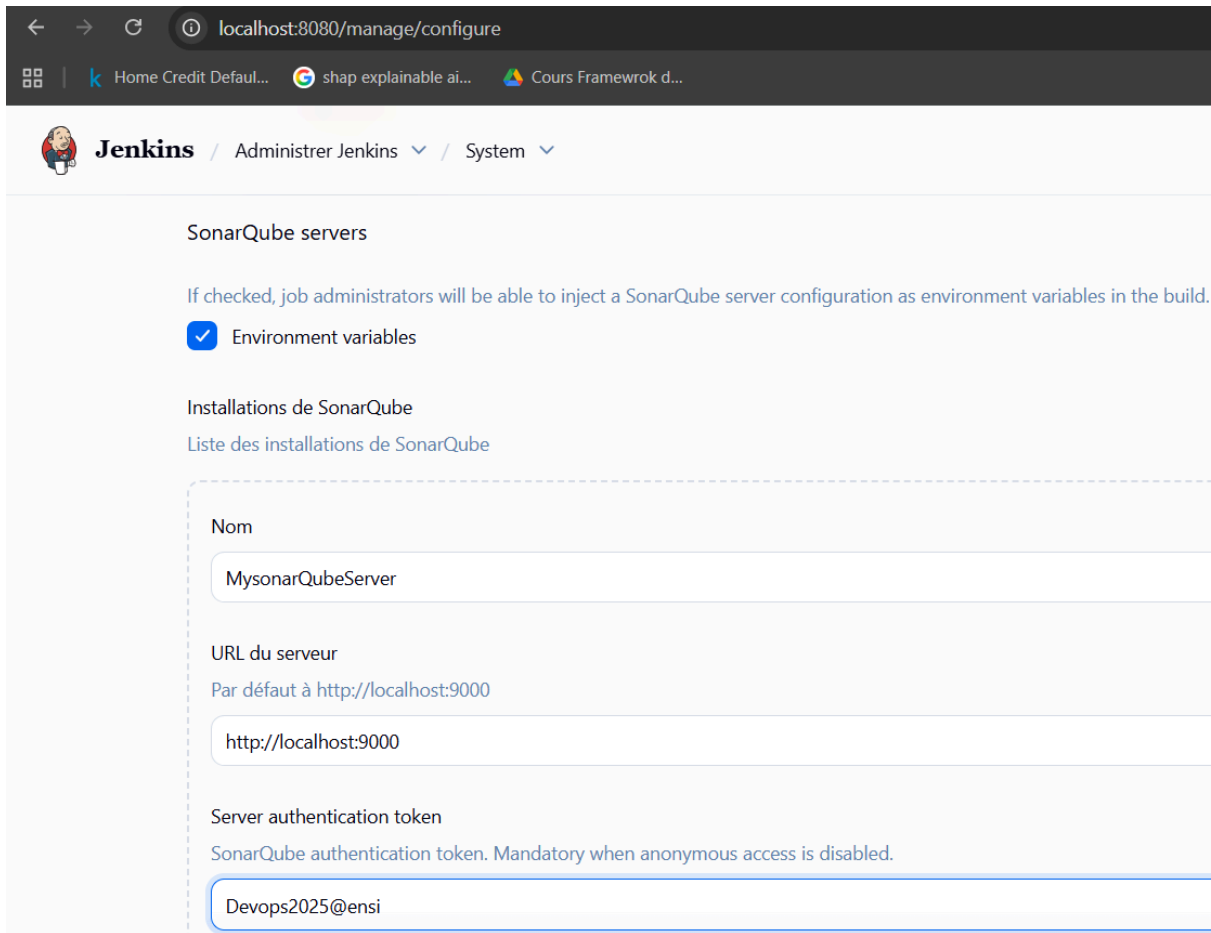
Configuration :

Type : Secret text

Portée : Globale

ID : Devops2025@ensl

Serveur SonarQube



The screenshot shows the Jenkins 'manage/configure' page for SonarQube servers. The browser address bar indicates the URL is localhost:8080/manage/configure. The Jenkins logo and navigation links are at the top. The main section is titled 'SonarQube servers'. A note states: 'If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.' Below this, the 'Environment variables' checkbox is checked. The section 'Installations de SonarQube' contains a table with one entry. The table has three rows: 'Nom' with value 'MysonarQubeServer', 'URL du serveur' with value 'http://localhost:9000', and 'Server authentication token' with value 'Devops2025@ensi'. A note for the token states: 'SonarQube authentication token. Mandatory when anonymous access is disabled.'

Nom	URL du serveur	Server authentication token
MysonarQubeServer	http://localhost:9000	Devops2025@ensi

Objectif : Intégrer SonarQube à Jenkins

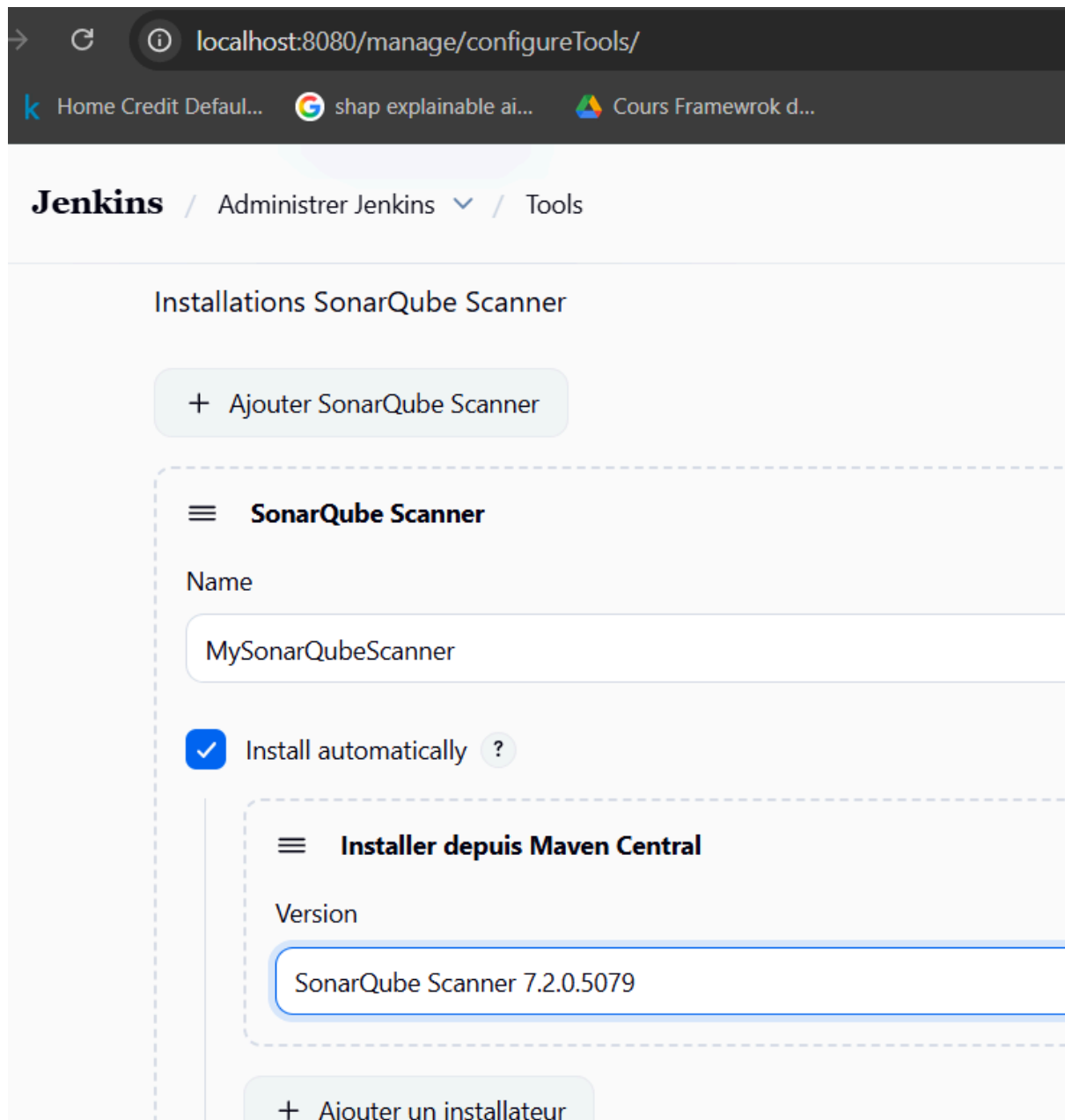
Configuration :

Nom : MysonarQubeServer

URL : http://localhost:9000

Token : Devops2025@ensi

Installation Scanner



Objectif : Installer SonarQube Scanner

Configuration :

Nom : MySonarQubeScanner

Version : 7.2.0.5079

Source : Maven Central

INTÉGRATION GITHUB

Webhook GitHub

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, `x-www-form-urlencoded`, etc). More information can be found in [our developer documentation](#).

Payload URL *

`https://doctrinally-lawlike-otha.ngrok-free.dev/github-webhook`

Content type *

`application/json`

Secret

SSL verification

By default, we verify SSL certificates when delivering payloads.

☒ Enable SSL verification ☐ Disable (not recommended)

Which events would you like to trigger this webhook?

☒ Just the push event.

☐ Send me everything.

☐ Let me select individual events.

Objectif : Déclencher automatiquement les builds

Configuration :

URL : `https://doctrinally-lawlike-otha.ngrok-free.dev/github-webhook`

Content type : JSON

Événements : Push events uniquement

Script Groovy

```

1  pipeline{
2      agent any
3      Tools{
4          maven 'mymaven'
5      }
6      stages{
7          stage('checkout code')
8          {
9              steps {
10                 git branch: 'master', url: 'https://github.com/ahmedkbenzid/TP-Spring-Boot.git'
11             }
12         }
13         stage('compile, test code, package in war file and store it in maven repo')
14         {
15             steps {
16                 sh 'mvn clean install'
17             }
18             post{
19                 success {
20                     junit allowEmptyResults: true, testResults: '**/target/sunfire-reports/*.xml'
21                 }
22             }
23         }
24         stage('SonarQube analysis')
25         {
26             steps {
27                 withSonarQubeEnv(installationName: 'MySonarQubeServer', credentialID: 'Devops2025@ensi'){

```

Objectif : Définir le pipeline d'automatisation

Étapes principales :

Checkout code depuis GitHub

Compilation, tests avec Maven

Analyse SonarQube

Rapports JUnit

CONFIGURATION PROJET

Création Projet

Nouveau Item

Saisissez un nom

CI_Country

Select an item type



Construire un projet free-style

Job legacy polyvalent qui récupère l'état depuis un outil de gestion de code et suit d'étapes post-construction telles que l'archivage d'artefacts.



Construire un projet maven

Construit un projet avec maven. Jenkins utilise directement vos fichiers de configuration. Cette fonctionnalité est encore en bêta mais elle est très puissante.



Pipeline

Organise des activités de longue durée qui peuvent s'étendre sur des pipelines (anciennement connues comme workflows) et/ou passer facilement à des tâches de type libre.

Objectif : Créer le projet Jenkins

Configuration :

Nom : CI_Country

Type : Projet Maven

Triggers GitHub

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- ☐ Construire après le build sur d'autres projets ?
- ☐ Construire périodiquement ?
- ☒ GitHub hook trigger for GITScm polling ?
- ☐ Scrutation de l'outil de gestion de version ?
- ☐ Déclencher les builds à distance (Par exemple, à partir de scripts) ?

Objectif : Activer le déclenchement automatique

Configuration

Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

`https://github.com/ahmedkbenzid/TP-Spring-Boot.git`

Credentials ?

`ahmedkbenzid/*****`



Objectif : Lier le repository Git


Configuration :


URL : `https://github.com/ahmedkbenzid/TP-Spring-Boot.git`


Credentials : `ahmedkbenzid/*****`


Syntaxe Pipeline


 **Jenkins** / CI_Country  / Pipeline Syntax


 Snippet Generator


 Declarative Directive Generator


 Declarative Online Documentation

 Steps Reference

 Global Variables Reference

 Online Documentation

 Examples Reference

 IntelliJ IDEA GDSL


Overview

This **Snippet Generator** will help you learn the Pipeline Syntax, **Generate Pipeline Script**, and you will see a Pipeline Script or pick up just the options you care about. (Most parameters are optional)

Steps

Sample Step

junit: Archive JUnit-formatted test results

junit 

XML des rapports de test

Une configuration du type **Fileset** 'includes' qui indique le répertoire de base (basedir) du fileset est la racine du répertoire de test

target/sunfire-reports/*.xml

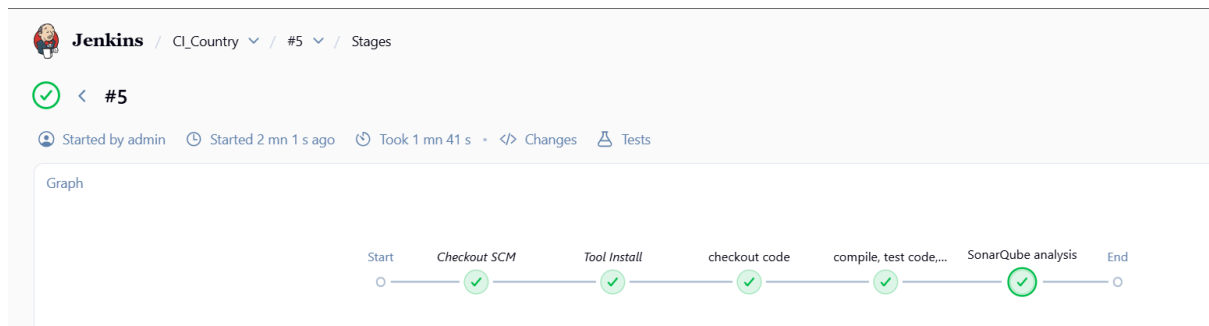
Objectif : Configurer les rapports de tests

Configuration :

Outil : Snippet Generator

Rapports JUnit : target/sunfire-reports/*.xml

Exécution Pipeline



Objectif : Visualiser l'exécution du build

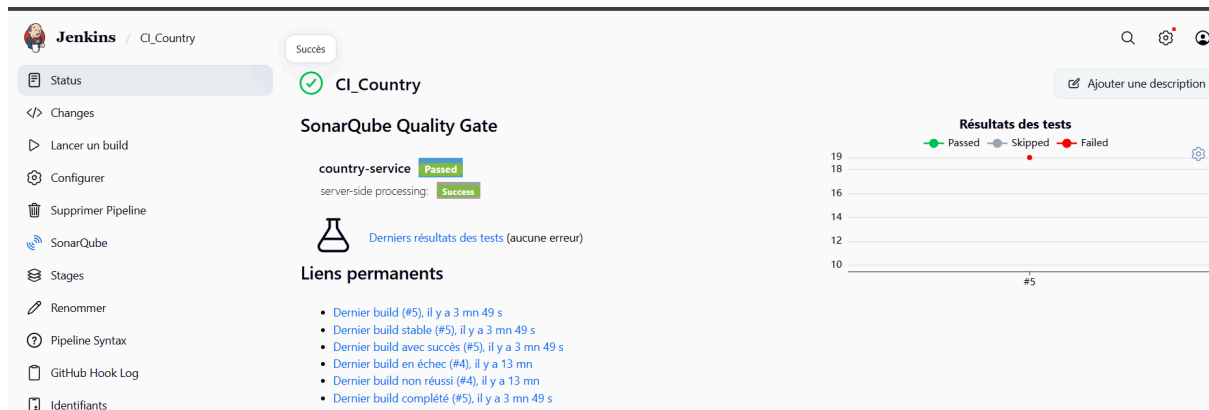
Résultats :

Build #5 réussi

Durée : 1 minute 41 secondes

Étapes : Checkout SCM → Tool Install → Compilation/Tests → SonarQube analysis

Résultats Finaux



Objectif : Analyser la qualité du code

Résultats :

SonarQube Quality Gate passé

Tests : Passed (aucune erreur)

