

# Mise en place d'un pipeline CI/CD

## CONFIGURATION DE L'ENVIRONNEMENT

### Utiliser ngrok pour créer une adresse publique

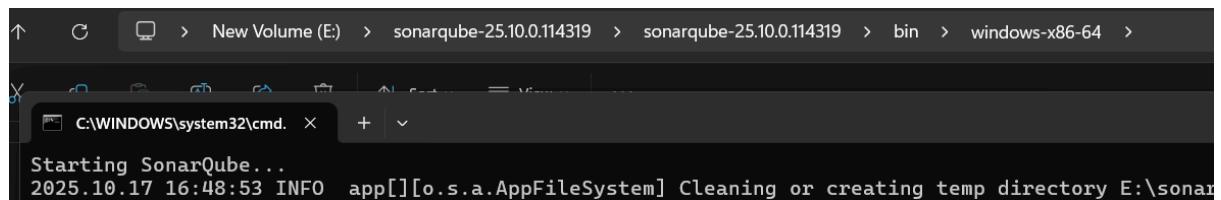
```
◆ Call internal services from your gateway: https://ngrok.com/r/http-request

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



### Génération Token SonarQube

Objectif : Créer un token pour l'authentification Jenkins

Configuration :

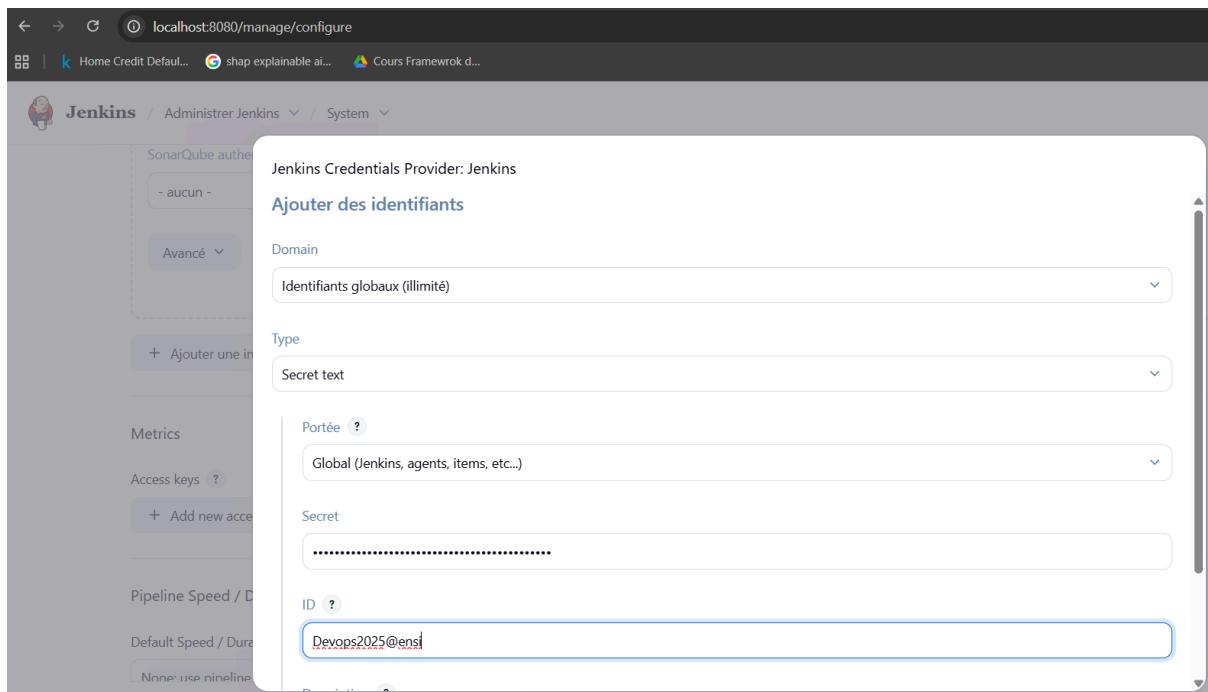
The screenshot shows the SonarQube security interface at the URL `localhost:9000/account/security`. At the top, there is a warning message: "⚠️ Embedded database should be used for evaluation purposes only. It doesn't support scaling, upgrading to a new SonarQube instance is recommended". Below this, the SonarQube logo and navigation links for Projects, Issues, Rules, Quality Profiles, Quality Gates, Administration, and More are visible. A sidebar on the left indicates the user is an Administrator. The main content area has tabs for Profile, Security, Notifications, and Projects, with the Security tab selected. A note below the tabs says: "This will increase the security of your installation by not letting your analysis user's password be stored in plain text in the Jenkins configuration." A section titled "Generate Tokens" contains fields for Name (jenkins), Type (User Token), and Expires in (30 days). A "Generate" button is present.

Nom : jenkins

Type : User Token

## CONFIGURATION JENKINS

### Identifiants SonarQube



Objectif : Stocker sécurisément le token SonarQube

Configuration :

Type : Secret text

Portée : Globale

ID : Devops2025@ensl

**Serveur SonarQube**

The screenshot shows the Jenkins configuration interface for SonarQube servers. At the top, the URL is localhost:8080/manage/configure. The navigation bar includes Home, Credit Default..., shap explainable ai..., and Cours Framewrok d... 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." A checkbox labeled "Environment variables" is checked. Below this, there's a heading "Installations de SonarQube" and a link "Liste des installations de SonarQube". A form is displayed with fields: "Nom" containing "MysonarQubeServer", "URL du serveur" containing "http://localhost:9000", and "Server authentication token" containing "Devops2025@ensi".

## Objectif : Intégrer SonarQube à Jenkins

Configuration :

Nom : MysonarQubeServer

URL : http://localhost:9000

Token : Devops2025@ensi

## Installation Scanner

→ C ⓘ localhost:8080/manage/configureTools/

k Home Credit Default... G shap explainable ai... Cours Framework d...

## Jenkins / Administrer Jenkins / Tools

### Installations SonarQube Scanner

+ Ajouter SonarQube Scanner

#### SonarQube Scanner

Name: MySonarQubeScanner

Install automatically ?

#### Installer depuis Maven Central

Version: SonarQube Scanner 7.2.0.5079

+ Ajouter un installateur

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 \***

**Content type \***

**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, packege in war file and store it in maven repo')
14          {
15              steps {
16                  sh 'mvn clean install'
17              }
18          }
19          post{
20              success {
21                  junit allowEmptyResults: true, testResults: '**/target/sunfire-reports/*.xml'
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 suivi 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



Pipeline

Organise des activités de longue durée qui peuvent s'étendre sur des pipelines (anciennement connues comme workflows) et/ou peut être associé 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/*********`

# EXÉCUTION ET RÉSULTATS

## Syntaxe Pipeline

The screenshot shows the Jenkins Pipeline Syntax Snippet Generator interface. At the top, there's a navigation bar with a Jenkins logo, the text "Jenkins / CI\_Country / Pipeline Syntax". Below the navigation is a horizontal menu bar with several items: "Snippet Generator" (which is highlighted in blue), "Declarative Directive Generator", "Declarative Online Documentation", "Steps Reference", "Global Variables Reference", "Online Documentation", "Examples Reference", and "IntelliJ IDEA GDSL". To the right of the menu, under the heading "Overview", there's a brief description of the snippet generator's purpose: "This **Snippet Generator** will help you learn the Pipeline Syntax. Click the **Generate Pipeline Script**, and you will see a Pipeline Script with all the parameters filled in or pick up just the options you care about. (Most parameters are optional)." Below this is a section titled "Steps" with a "Sample Step" example: "junit: Archive JUnit-formatted test results". Underneath this example, there's a detailed description of the "junit" step: "XML des rapports de test. Une configuration du type Fileset 'includes' qui indique les fichiers à archiver. Le répertoire de base (basedir) du fileset est la racine du dossier de l'archive. Exemple : target/sunfire-reports/\*.xml".

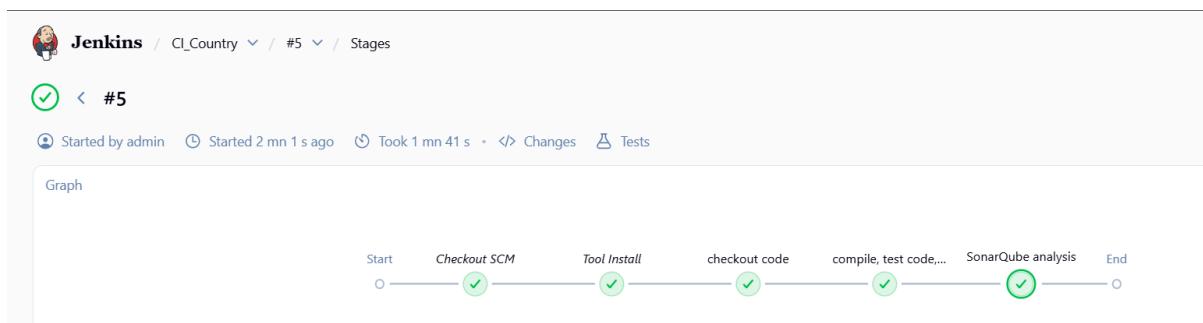
Objectif : Configurer les rapports de tests

Configuration :

Outil : Snippet Generator

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

**Exécution Pipeline**



Objectif : Visualiser l'exécution du build

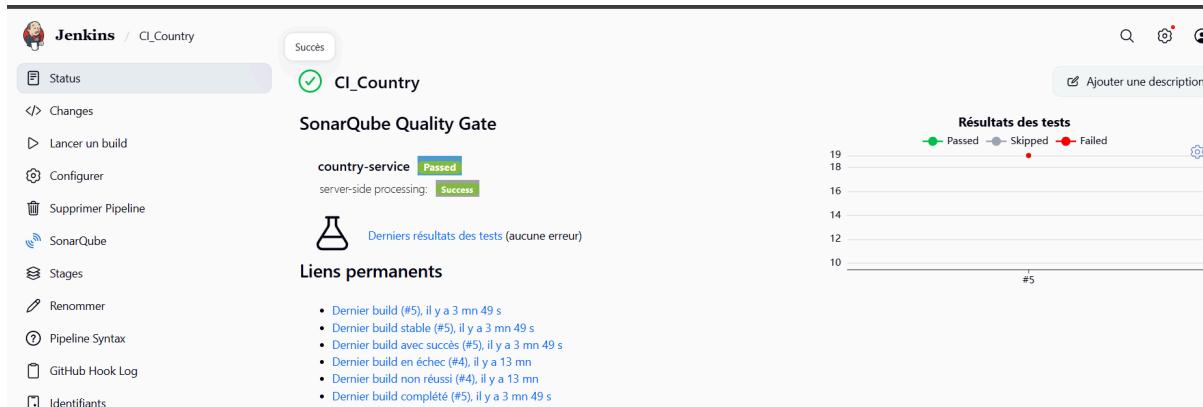
Résultats :

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)

