

Guide de Déploiement : JMeter & SonarQube pour l'Assurance Qualité



Realiser par :

M. Fode MANGANE

Sous la direction de :

DR Massamba LO

Table des matières

1. Introduction	1
2. Apache JMeter : Maîtrise des Tests de Performance	1
2.1 Préparation de l'Environnement.....	1
2.2 Configuration Avancée et Optimisation	2
2.3 Conception et Implémentation d'un Plan de Test.....	3
2.4 Exécution et Analyse des Tests.....	6
3. SonarQube : Excellence en Analyse de Code.....	8
3.1 Déploiement de SonarQube	8
3.2 Configuration et Sécurisation.....	10
3.3 Configuration Initiale et Sécurisation	11
3.4 Configuration des Règles de Qualité.....	13
4. Intégration DevOps : Jenkins et GitLab CI/CD	15
4.1 Intégration avec Jenkins	15
4.2 Intégration avec GitLab CI/CD	21
5. Tests Pratiques et Validation	23
5.1 Analyse SonarQube en Action	23
5.2 Tests de Performance avec JMeter	25
6. Résolution des Problèmes Courants	27
6.1 Problématiques JMeter.....	27
6.2 Problématiques SonarQube.....	27
6.3 Bonnes Pratiques Recommandées.....	28
7. Conclusion.....	28

1. Introduction

Dans l'écosystème DevOps moderne, l'assurance qualité logicielle repose sur deux piliers fondamentaux : la performance applicative et la qualité du code source. Ce guide technique présente une approche intégrée utilisant deux outils de référence dans l'industrie.

Apache JMeter se positionne comme l'outil de référence pour les tests de performance et de charge. Il permet de simuler des environnements utilisateurs réalistes afin d'évaluer la résistance et les performances d'une application sous différentes contraintes de charge.

SonarQube, quant à lui, constitue la plateforme de référence pour l'analyse continue de la qualité du code. Il effectue des audits automatisés permettant d'identifier les vulnérabilités de sécurité, les défauts de conception et les mauvaises pratiques de développement.

L'objectif de ce guide est de démontrer l'implémentation complète de ces outils dans un environnement de développement moderne, en intégrant les meilleures pratiques **DevOps** avec **Jenkins** et **GitLab CI/CD**.

2. Apache JMeter : Maîtrise des Tests de Performance

2.1 Préparation de l'Environnement

Prérequis Techniques

Avant de débuter l'installation, nous devons nous assurer que notre environnement respecte les exigences suivantes :

- Java Development Kit (JDK) version 8 ou supérieure
- Configuration appropriée de la variable d'environnement JAVA_HOME
- Ressources système suffisantes (minimum 8GB RAM recommandé)

Processus d'Installation

Étape 1 : Acquisition des Binaires

Nous commençons par télécharger la dernière version stable d'Apache JMeter depuis le site officiel. Cette approche garantit l'obtention de la version la plus récente et sécurisée.

```
fode-mangane@fode-mangane:~$ wget https://www.apache.org/dist/jmeter/binaries/apache-jmeter-5.6.3.tgz.sha512
--2025-05-20 13:16:50-- https://www.apache.org/dist/jmeter/binaries/apache-jmeter-5.6.3.tgz.sha512
Resolving www.apache.org (www.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to www.apache.org (www.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://downloads.apache.org/jmeter/binaries/apache-jmeter-5.6.3.tgz.sha512 [following]
--2025-05-20 13:16:50-- https://downloads.apache.org/jmeter/binaries/apache-jmeter-5.6.3.tgz.sha512
Resolving downloads.apache.org (downloads.apache.org)... 88.99.208.237, 135.181.214.104, 2a01:4f8:10a:39da::2, ...
Connecting to downloads.apache.org (downloads.apache.org)|88.99.208.237|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 154 [text/plain]
Saving to: 'apache-jmeter-5.6.3.tgz.sha512'

apache-jmeter-5.6.3.tgz.sha512 100%[=====] 154 --.-KB/s in 0s

2025-05-20 13:16:50 (52.0 MB/s) - 'apache-jmeter-5.6.3.tgz.sha512' saved [154/154]

fode-mangane@fode-mangane:~$
```

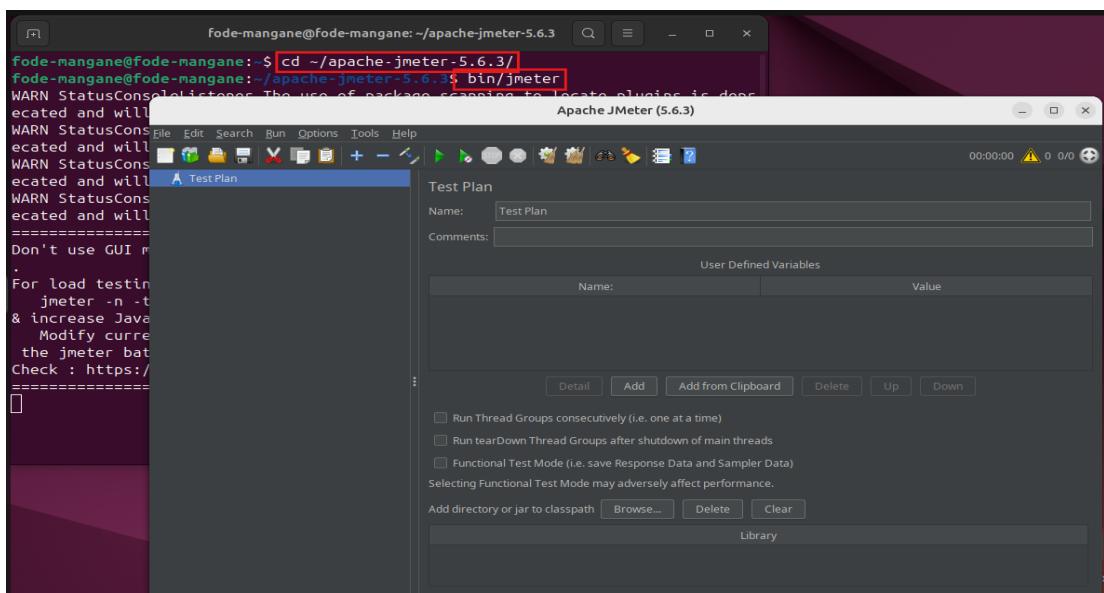
Étape 2 : Déploiement Local

Une fois le téléchargement terminé, nous procédons à l'extraction de l'archive afin de rendre les fichiers immédiatement accessibles pour la suite des opérations.

```
fode-mangane@fode-mangane:~$ tar -xvzf apache-jmeter-5.6.3.tgz
apache-jmeter-5.6.3/
apache-jmeter-5.6.3/LICENSE
apache-jmeter-5.6.3/licenses/
apache-jmeter-5.6.3/licenses/fLOT-axislabels/
apache-jmeter-5.6.3/licenses/fLOT-axislabels/fLOT-axislabels-0.8.3/
apache-jmeter-5.6.3/licenses/fLOT-axislabels/fLOT-axislabels-0.8.3/LICENSE
apache-jmeter-5.6.3/licenses/io.burt/
apache-jmeter-5.6.3/licenses/io.burt/jmespath-core-0.6.0/
apache-jmeter-5.6.3/licenses/io.burt/jmespath-core-0.6.0/LICENSE
```

Étape 3 : Validation de l'Installation

Nous vérifions maintenant que JMeter fonctionne correctement en lançant l'interface graphique. Cette étape confirme que toutes les dépendances sont satisfaites.



2.2 Configuration Avancée et Optimisation

Personnalisation des Propriétés Système

La configuration fine de JMeter passe par l'édition du fichier **user.properties** situé dans le répertoire bin. Cette personnalisation permet d'adapter l'outil aux spécificités de notre environnement de test.

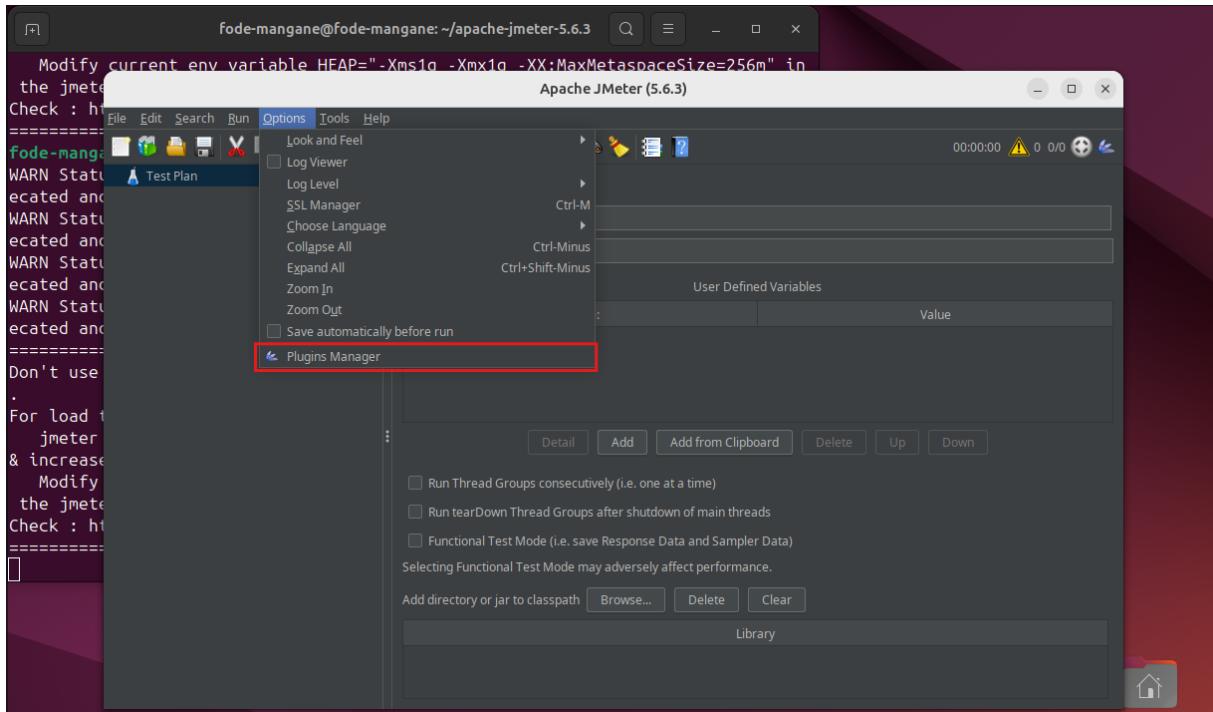
```
fode-mangane@fode-mangane:~$ cd ~/apache-jmeter-5.6.3
fode-mangane@fode-mangane:~/apache-jmeter-5.6.3$
fode-mangane@fode-mangane:~/apache-jmeter-5.6.3$ ls
LICENSE NOTICE README.md bin docs extras jmeter.log lib licenses printable_docs
fode-mangane@fode-mangane:~/apache-jmeter-5.6.3$ cd bin/
fode-mangane@fode-mangane:~/apache-jmeter-5.6.3/bin$ vim user.properties
```

```
#server.rmi.ssl.truststore.password=changeit
#
# Set this if you don't want to use SSL for RMI
#
#server.rmi.ssl.disable=false
# Configuration personnalisée Fode Mangane
jmeter.save.saveservice.output_format=xml
jmeter.save.saveservice.response_data=true
jmeter.save.saveservice.samplerData=true
jmeter.save.saveservice.requestHeaders=true
jmeter.save.saveservice.responseHeaders=true
```

Extension des Fonctionnalités

L'écosystème JMeter offre une riche collection de plugins qui étendent considérablement ses capacités. Nous procéderons à l'installation du gestionnaire de plugins pour faciliter cette expansion.

Nous installons ensuite les plugins essentiels qui enrichiront nos capacités d'analyse :



Ces extensions incluent notamment :

- **3 Basic Graphs** : Visualisations graphiques améliorées
- **Custom Thread Groups** : Contrôle avancé des groupes d'utilisateurs
- **Dummy Sampler** : Tests de validation sans requêtes réelles
- **Flexible File Writer** : Export personnalisé des résultats
- **JP@GC - Graphs Generator** : Génération de rapports visuels

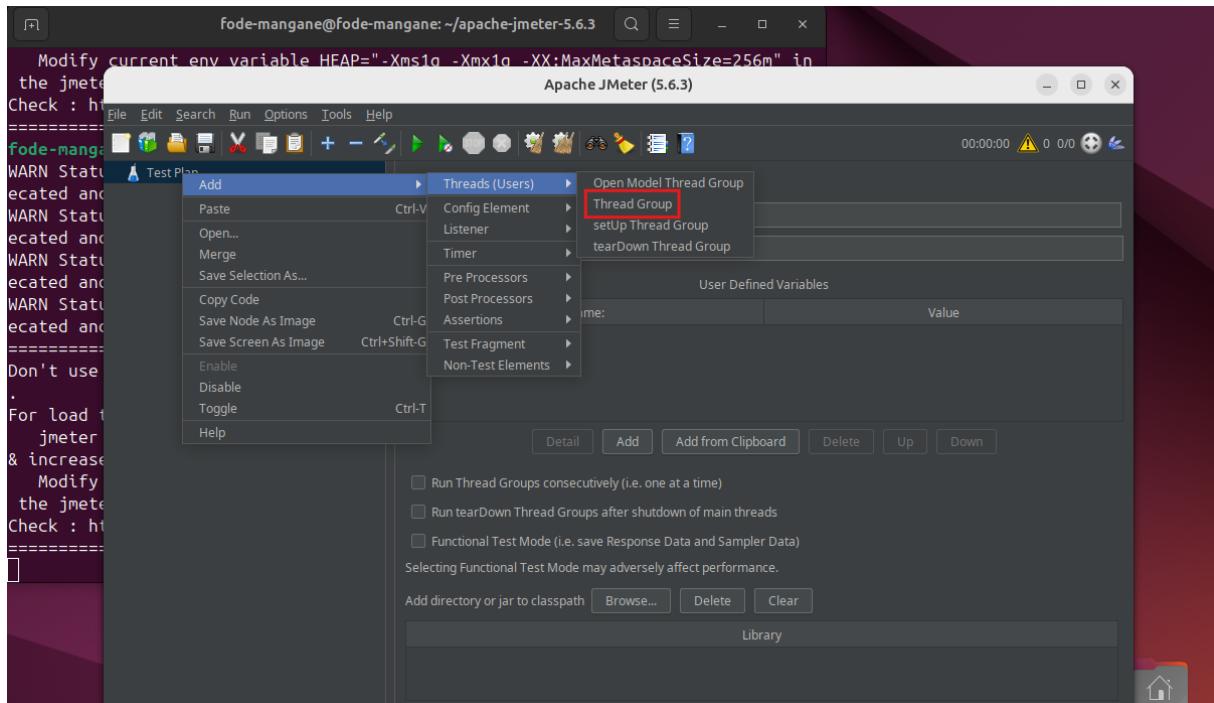
2.3 Conception et Implémentation d'un Plan de Test

Architecture d'un Plan de Test

Un plan de test JMeter suit une structure hiérarchique logique qui reflète fidèlement le comportement réel des utilisateurs. Cette organisation modulaire permet de structurer les éléments de test de manière claire et cohérente, ce qui facilite la maintenance, l'évolution et la réutilisation des composants. Elle favorise également la collaboration entre les membres de l'équipe en rendant les scénarios plus lisibles et plus facilement adaptables aux besoins changeants du projet.

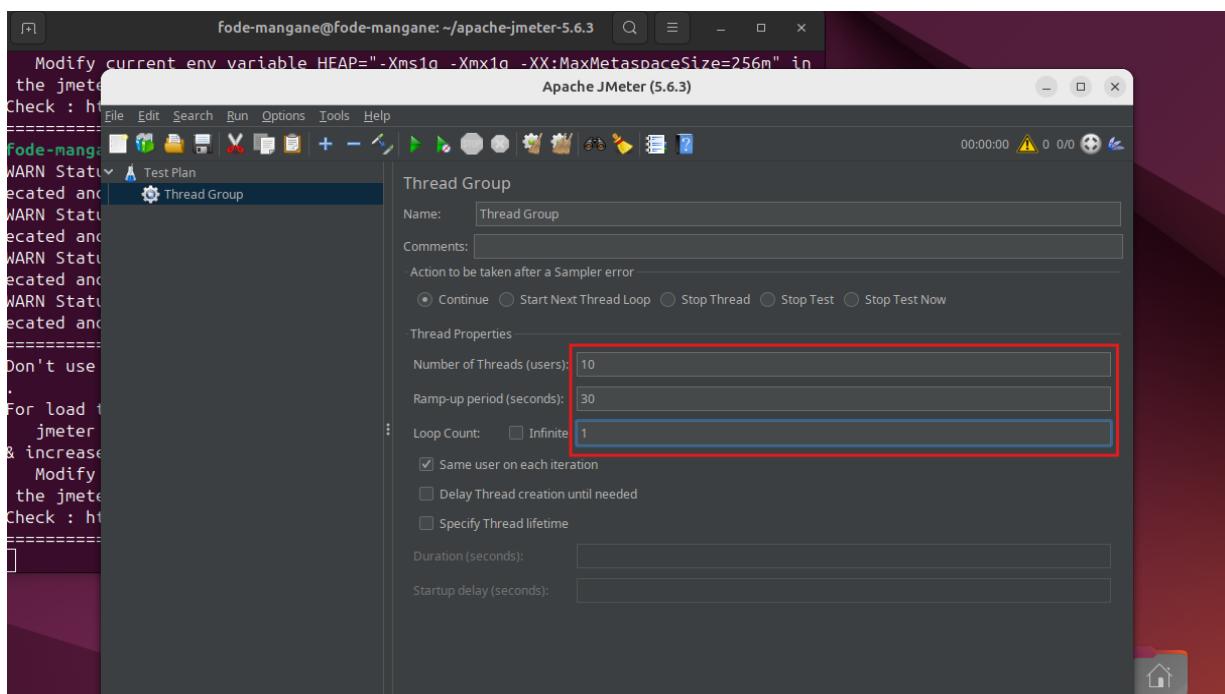
Étape 1 : Crédation du Groupe d'Utilisateurs Virtuels

Nous débutons par la définition du groupe de threads qui représentera nos utilisateurs simulés. Cette configuration détermine l'intensité et la durée de notre test de charge.



Étape 2 : Paramétrage des Utilisateurs Virtuels

Nous configurons précisément le nombre d'utilisateurs concurrents, la période de montée en charge et le nombre d'itérations. Ces paramètres déterminent le profil de charge appliqué à l'application testée.



Étape 3 : Préparation de l'Infrastructure Cible

Avant de procéder aux tests, nous nous assurons que notre serveur Apache est opérationnel et correctement configuré avec les résolutions DNS appropriées.

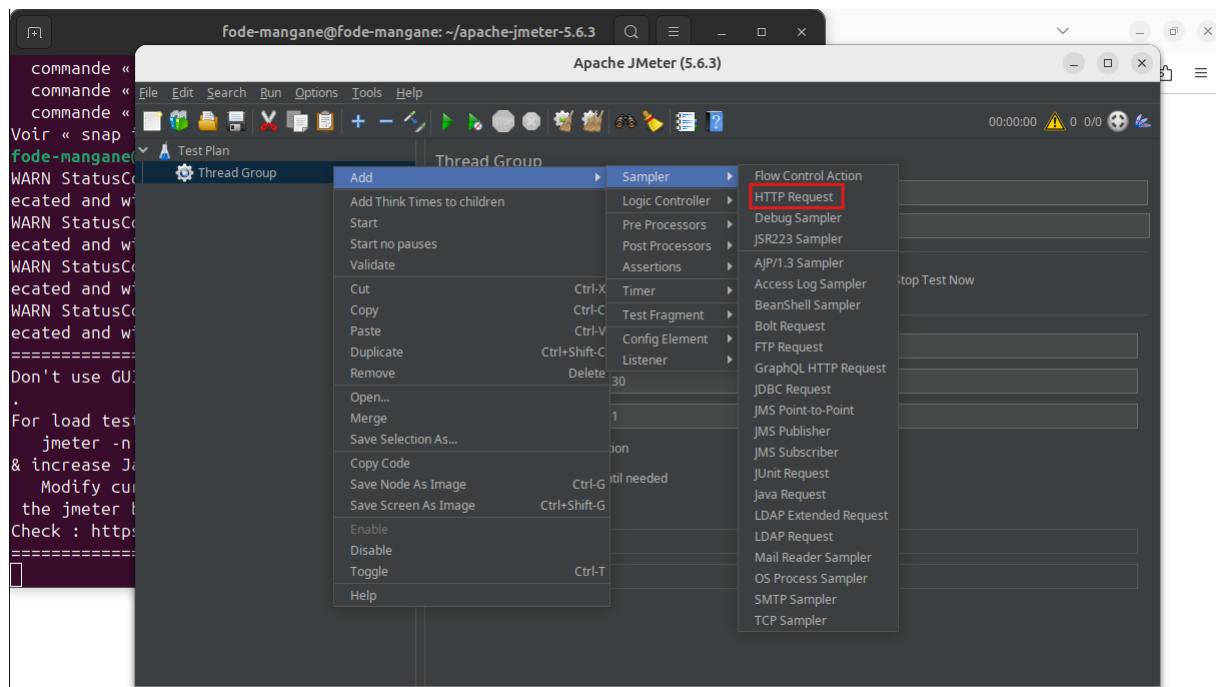
```
root@fode-mangane:~# nslookup www
Server:      127.0.0.53
Address:     127.0.0.53#53

www.fodemangane.lan canonical name = fodemangane.lan.
Name:   fodemangane.lan
Address: 192.168.148.165

root@fode-mangane:~#
```

Étape 4 : Définition des Requêtes HTTP

Nous ajoutons maintenant les échantilleurs HTTP qui simuleront les interactions utilisateur avec notre application web. Cette étape constitue le cœur de notre simulation de charge.



Étape 5 : Configuration des Paramètres de Connexion

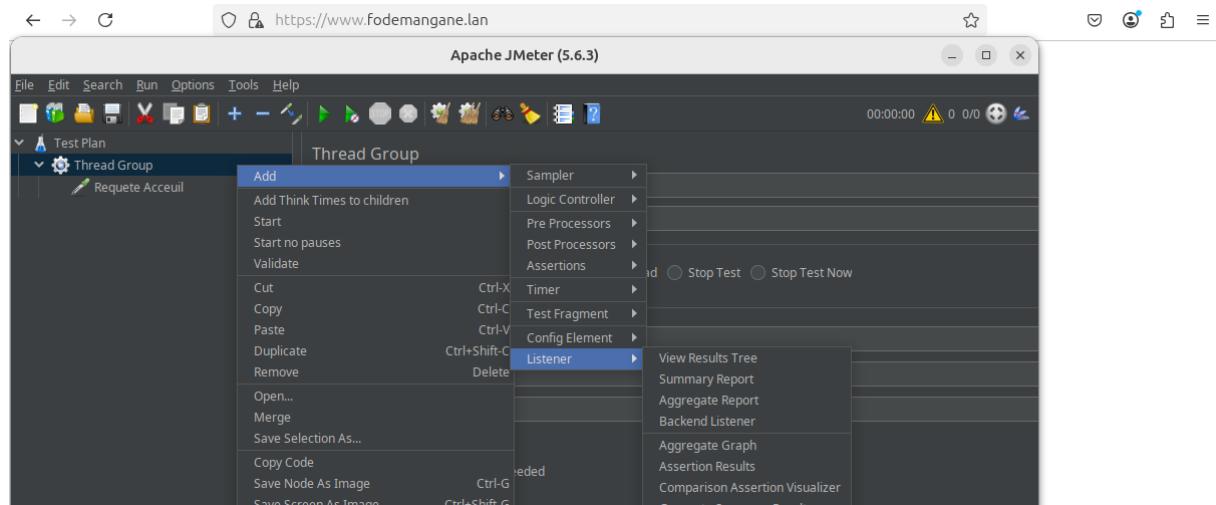
Nous spécifions les détails de connexion incluant le serveur cible, le port et les chemins d'accès. Cette configuration doit refléter fidèlement l'architecture de production.

The screenshot shows the 'HTTP Request' configuration dialog. The 'Basic' tab is selected. Key fields highlighted with red boxes are:

- Protocol [http]: https
- Server Name or IP: www.fodemangane.lan
- Port Number: 443
- Path: /
- Checkboxes at the bottom: Redirect Automatically (unchecked), Follow Redirects (checked), Use KeepAlive (checked), Use multipart/form-data (unchecked), and Browser-compatible headers (checked).

Étape 6 : Mise en Place de la Collecte de Données

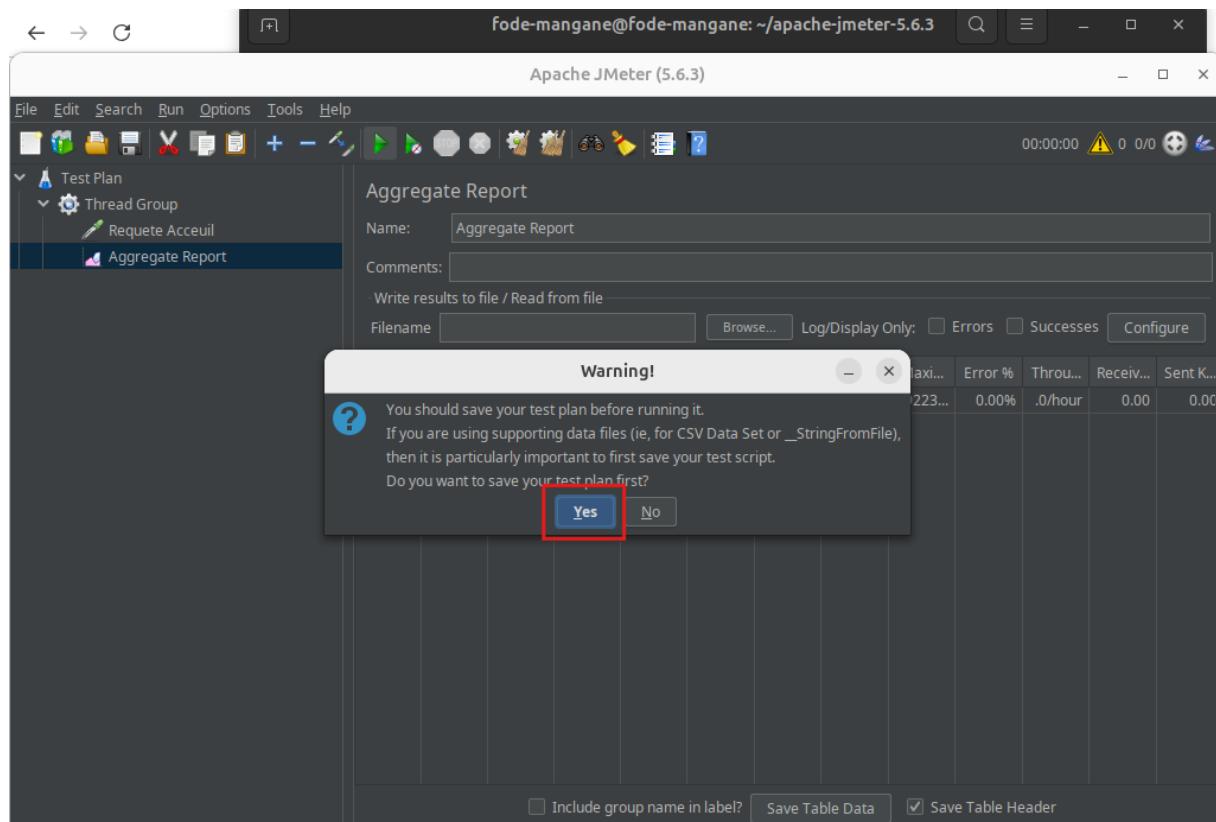
L'ajout d'écouteurs (listeners) nous permet de collecter et analyser les résultats en temps réel.



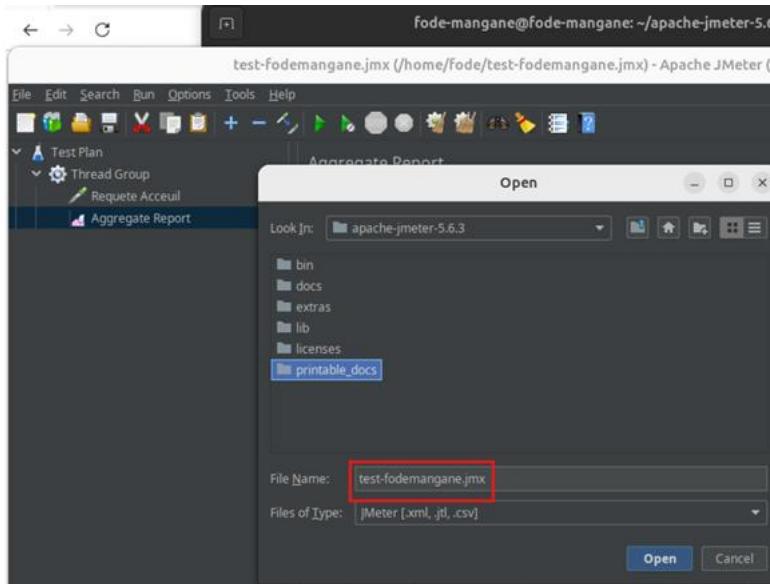
2.4 Exécution et Analyse des Tests

Mode Graphique pour le Développement

L'interface graphique convient parfaitement pour le développement et la validation des plans de test. Elle offre une visualisation en temps réel des résultats.



Il est important de sauvegarder le plan de test avant l'exécution pour éviter toute perte de configuration.



Mode Ligne de Commande pour la Production

Pour les tests de charge intensifs, le mode ligne de commande s'avère indispensable. Il élimine la surcharge de l'interface graphique et permet des performances optimales.

Nous configurons d'abord les propriétés système dans le fichier jmeter.properties :

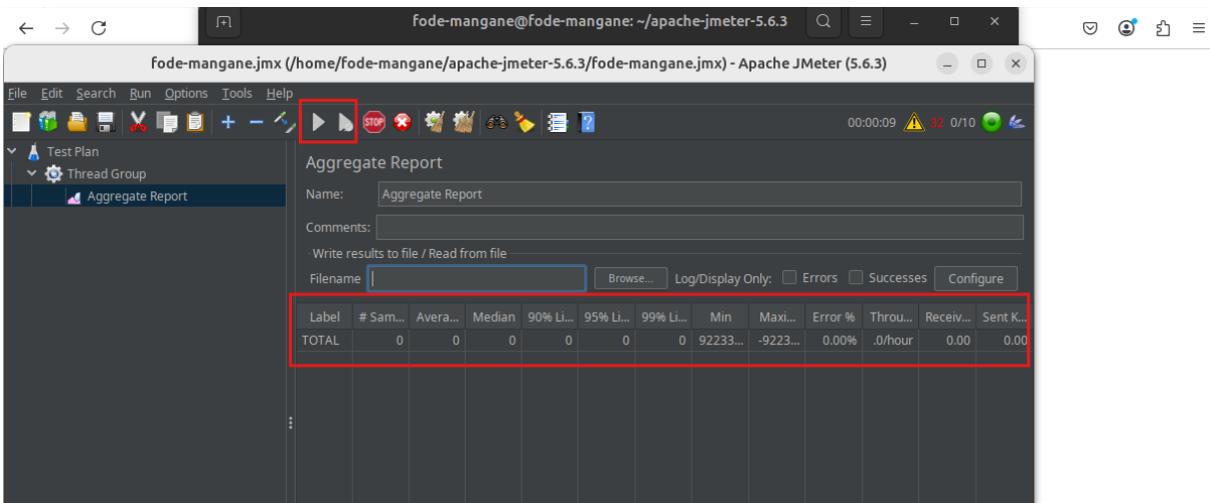
```
# Documentation generation
#-----
# Path to XSL file used to generate Schematic View of Test Plan
# When empty, JMeter will use the embedded one in src/core/org/apache/jmeter/gui/action/schematic.xsl
#docgeneration_schematic_xsl=
xstream.allowed.classes=org.apache.jmeter.save.TestResultWrapper
~
~
```

La commande d'exécution type suit cette syntaxe :

jmeter -n -t [plan_de_test.jmx] -l [resultats.jtl] -e -o [dossier_rapport]

Interprétation des Résultats

JMeter génère des rapports détaillés qui permettent une analyse approfondie des performances. Ces données incluent les temps de réponse, le débit et les taux d'erreur.



3. SonarQube : Excellence en Analyse de Code

3.1 Déploiement de SonarQube

Approche Conteneurisée avec Docker

L'utilisation de Docker simplifie considérablement le déploiement de SonarQube tout en garantissant une isolation parfaite des services.

```
root@fode-mangane:~# docker pull sonarqube:latest
latest: Pulling from library/sonarqube
0622fac788ed: Pull complete
c1b37af7090f: Pull complete
a32efa2d47a9: Pull complete
97f6621fedba: Pull complete
141779bd67f4: Pull complete
14e06ebe6297: Pull complete
9ea80751278: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:26b5a4f25bc6cb7bffa3f6c6b737254b0ffa7168c379921070b7b769315f2c
Status: Downloaded newer image for sonarqube:latest
docker.io/library/sonarqube:latest
root@fode-mangane:~# 

root@fode-mangane:~# docker run -d --name db \
-e POSTGRES_USER=sonar \
-e POSTGRES_PASSWORD=sonar \
-e POSTGRES_DB=sonar \
-v postgresql_data:/var/lib/postgresql/data \
postgres:latest
Unable to find image 'postgres:latest' locally
latest: Pulling from library/postgres
254e724d7786: Pull complete
074b41e8190f: Pull complete
bb26461722d7: Pull complete
0ddf304e15c0: Pull complete
693761670bfc: Pull complete
fb0c607b2495: Pull complete
f438059a5a57: Pull complete
64d10e07b30c: Pull complete
877e6448d55b: Pull complete
e023e132da3e: Pull complete
3ee9d9321249: Pull complete
a308d0de7d38: Pull complete
4d7ebd47166d: Pull complete
4ca3b919b7ab: Pull complete
Digest: sha256:864831322bf2520e7d03d899b01b542de6de9ece6fe29c89f19dc5e1d5568ccf
Status: Downloaded newer image for postgres:latest
13fd37ebfce06af6285839976fa2af15b181fd575535443dfe293588ef8adb
root@fode-mangane:~# 
```

Assurez-vous de disposer d'une base de données PostgreSQL fonctionnelle.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
13fd37ebfce06af6285839976fa2af15b181fd575535443dfe293588ef8adb	postgres:latest	"docker-entrypoint.s..."	About a minute ago	Up About a minute	5432/tcp	db

Cette approche offre plusieurs avantages :

- Déploiement rapide et reproductible
- Isolation complète des dépendances
- Facilité de mise à jour et de maintenance

Installation Manuelle Traditionnelle

Pour les environnements nécessitant un contrôle granulaire sur chaque composant du système, l'installation manuelle reste une option privilégiée. Elle permet une maîtrise complète des paramètres de configuration, du choix des dépendances, ainsi que de l'architecture logicielle mise en place. Bien que plus chronophage, cette méthode offre une flexibilité maximale, adaptée aux besoins spécifiques, aux environnements contraints ou aux exigences de sécurité renforcées. Elle constitue ainsi une solution idéale pour les intégrations personnalisées ou les infrastructures complexes.

Acquisition des Composants

Nous téléchargeons SonarQube et SonarScanner depuis les sources officielles :

```
root@fode-mangane:~# wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-25.5.0.107428.zip?_gl=1*3fqw7y*_ga*MTAwMjIyMDI0OC4xNzQ3NzU4MDcx*_ga*MzAxOTc10TI2LjE3NDc3NTgwNzA.*_ga_9JZ0GZ5TC6*cze3NDc3NTgwNzAkzbEkzEkdDE3NDc3NTgyMzgkajM4JGwwJGgwJGQtMjVUQWZLZENrbDdQmJruNxzdwtmemxB0UlkbzRyNGVB--2025-05-20 16:24:19-- https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-25.5.0.107428.zip?_gl=1*3fqw7y*_gcl_au*MTAwMjIyMDI0OC4xNzQ3NzU4MDcx*_ga*MzAxOTc10TI2LjE3NDc3NTgwNzA.*_ga_9JZ0GZ5TC6*cze3NDc3NTgwNzAkzbEkzEkdDE3NDc3NTgyMzgkajM4JGwwJGgwJGQtMjVUQWZLZENrbDdQmJruNxzdwtmemxB0UlkbzRyNGVB  
Résolution de binaries.sonarsource.com (binaries.sonarsource.com)... 108.157.109.34, 108.157.109.7, 108.157.109.80, ...  
Connexion à binaries.sonarsource.com (binaries.sonarsource.com)| 108.157.109.34|:443... connecté.  
requête HTTP transmise, en attente de la réponse.. 200 OK  
Taille : 853924488 (814M) [binary/octet-stream]  
Enregistre : 'sonarqube-25.5.0.107428.zip?_gl=1*3fqw7y*_gcl_au*MTAwMjIyMDI0OC4xNzQ3NzU4MDcx*_ga*MzAxOTc10TI2LjE3NDc3NTgwNzA.*_ga_9JZ0GZ5TC6*cze3NDc3NTgwNzAkzbEkzEkdDE3NDc3NTgyMzgkajM4JGwwJGgwJGQtMjVUQWZLZENrbDdQmJruNxzdwtmemxB0UlkbzRyNGVB'  
sonarqube-25.5.0.107428.zip?_gl 100%[=====]> 814,37M 2,29MB/s ds 6m 39s  
2025-05-20 16:30:58 (2,04 MB/s) - 'sonarqube-25.5.0.107428.zip?_gl=1*3fqw7y*_gcl_au*MTAwMjIyMDI0OC4xNzQ3NzU4MDcx*_ga*MzAxOTc10TI2LjE3NDc3NTgwNzA.*_ga_9JZ0GZ5TC6*cze3NDc3NTgwNzAkzbEkzEkdDE3NDc3NTgyMzgkajM4JGwwJGgwJGQtMjVUQWZLZENrbDdQmJruNxzdwtmemxB0UlkbzRyNGVB' enregistré [853924488/853924488]  
root@fode-mangane:~#  
  
root@fode-mangane:~# wget https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-5.0.1.3006-linux.zip  
--2025-05-20 16:12:14-- https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-5.0.1.3006-linux.zip  
Résolution de binaries.sonarsource.com (binaries.sonarsource.com)... 108.157.109.80, 108.157.109.27, 108.157.109.34, ...  
Connexion à binaries.sonarsource.com (binaries.sonarsource.com)| 108.157.109.80|:443... connecté.  
requête HTTP transmise, en attente de la réponse.. 200 OK  
Taille : 47123720 (45M) [binary/octet-stream]  
Enregistre : 'sonar-scanner-cli-5.0.1.3006-linux.zip'  
sonar-scanner-cli-5.0.1.3006-l 15%[=====] ] 7,17M 2,25MB/s tps 17s
```

Organisation des Répertoires

Une organisation claire des répertoires facilite la maintenance et l'administration du système :

```
root@fode-mangane:~# mv 'sonarqube-25.5.0.107428.zip?_gl=1*3fqw7y*_gcl_au*MTAwMjIyMDI0OC4xNzQ3NzU4MDcx*_ga*MzAxOTc10TI2LjE3NDc3NTgwNzA.*_ga_9JZ0GZ5TC6*cze3NDc3NTgwNzAkzbEkzEkdDE3NDc3NTgyMzgkajM4JGwwJGgwJGQtMjVUQWZLZENrbDdQmJruNxzdwtmemxB0UlkbzRyNGVB' sonarqube.zip  
root@fode-mangane:~# ls  
jmeter.log snap sonarqube.zip sonar-scanner-5.0.1.3006-linux sonar-scanner-cli-5.0.1.3006-linux.zip  
root@fode-mangane:~#  
root@fode-mangane:~# unzip sonarqube.zip  
Archive: sonarqube.zip  
    creating: sonarqube-25.5.0.107428/  
    creating: sonarqube-25.5.0.107428/jres/  
    inflating: sonarqube-25.5.0.107428/jres/OpenJDK17U-jre_aarch64_linux_hotspot_17.0.13_11.tar.gz  
    inflating: sonarqube-25.5.0.107428/jres/OpenJDK17U-jre_aarch64_mac_hotspot_17.0.13_11.tar.gz  
    inflating: sonarqube-25.5.0.107428/jres/OpenJDK17U-jre_x64_alpine-linux_hotspot_17.0.13_11.tar.gz  
    inflating: sonarqube-25.5.0.107428/jres/OpenJDK17U-jre_x64_linux_hotspot_17.0.13_11.tar.gz  
    inflating: sonarqube-25.5.0.107428/jres/OpenJDK17U-jre_x64_mac_hotspot_17.0.13_11.tar.gz  
    inflating: sonarqube-25.5.0.107428/jres/OpenJDK17U-jre_x64_windows_hotspot_17.0.13_11.zip  
  
root@fode-mangane:~# unzip sonar-scanner-cli-5.0.1.3006-linux.zip  
Archive: sonar-scanner-cli-5.0.1.3006-linux.zip  
    creating: sonar-scanner-5.0.1.3006-linux/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/lib/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/lib/security/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/lib/jfr/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/lib/server/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/legal/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/legal/jdk.localedata/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/legal/jdk.dynalink/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/legal/java.transaction.xa/  
    creating: sonar-scanner-5.0.1.3006-linux/jre/legal/java.security.jgss/
```

Configuration des Variables d'Environnement

Nous ajoutons SonarScanner au PATH système pour faciliter son utilisation :

```
root@fode-mangane:~# mv sonarqube-* /opt/sonarqube  
root@fode-mangane:~#  
  
root@fode-mangane:~# mv sonar-scanner-5.0.1.3006-linux /opt/sonar-scanner  
root@fode-mangane:~#
```

3.2 Configuration et Sécurisation

Gestion des Utilisateurs et Permissions

La sécurité commence par la création d'un utilisateur dédié à SonarQube, limitant ainsi les privilèges système :

```
root@fode-mangane:~# sudo useradd --system --no-create-home --shell /usr/sbin/nologin sonarqube
root@fode-mangane:~# sudo chown -R sonarqube:sonarqube /opt/sonarqube
root@fode-mangane:~#
```

Configuration de la Base de Données

SonarQube nécessite une base de données robuste pour le stockage des analyses. PostgreSQL constitue le choix recommandé pour les environnements de production :

```
root@fode-mangane:~# apt install postgresql postgresql-contrib
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  libcommon-sense-perl libjison-perl libjison-xs-perl libllvm17t64 libpq5 libtypes-serialiser-perl postgresql-16
  postgresql-client-16 postgresql-client-common postgresql-common
Paquets suggérés :
  postgresql-doc postgresql-doc-16
Les NOUVEAUX paquets suivants seront installés :
  libcommon-sense-perl libjison-perl libjison-xs-perl libllvm17t64 libpq5 libtypes-serialiser-perl postgresql postgresql-16
  postgresql-client-16 postgresql-client-common postgresql-common postgresql-contrib
0 mis à jour, 12 nouvellement installés, 0 à enlever et 3 non mis à jour.
Il est nécessaire de prendre 43,5 Mo dans les archives.
Après cette opération, 175 Mo d'espace disque supplémentaires seront utilisés.
```

```
root@fode-mangane:~# sudo -u postgres psql
psql (16.8 (Ubuntu 16.8-0ubuntu0.24.04.1))
Type "help" for help.

postgres=# CREATE USER sonar WITH ENCRYPTED PASSWORD 'sonar';
CREATE ROLE
postgres=# CREATE DATABASE sonarqube OWNER sonar;
CREATE DATABASE
postgres=# GRANT ALL PRIVILEGES ON DATABASE sonarqube TO sonar;
GRANT
postgres=# \q
root@fode-mangane:~#
root@fode-mangane:~#
```

Nous éditons ensuite le fichier de configuration principal sonar.properties :

```
25 sonar.jdbc.username=sonar
26 sonar.jdbc.password=sonar
27
28 #----- Embedded Database (default)
29 # H2 embedded database server listening port, defaults to 9092
30 #sonar.embeddedDatabase.port=9092
31
32
33 #----- Oracle 19c/21c
34 # The Oracle JDBC driver must be copied into the directory extensions/jdbc-driver/oracle/.
35 # Only the thin client is supported, and we recommend using the latest Oracle JDBC driver. See
36 # https://jira.sonarsource.com/browse/SONAR-9758 for more details.
37 # If you need to set the schema, please refer to http://jira.sonarsource.com/browse/SONAR-5000
38 sonar.jdbc.url=sonar.jdbc.url=jdbc:postgresql://localhost:5432/sonarqube
39
```

Démarrage et Validation

Le démarrage de SonarQube s'effectue via les scripts fournis :

```
root@fode-mangane:~# cd /opt/sonarqube/bin/linux-x86-64/
root@fode-mangane:/opt/sonarqube/bin/linux-x86-64# ./sonar.sh start
/bin/java
Starting SonarQube...
Started SonarQube.
root@fode-mangane:/opt/sonarqube/bin/linux-x86-64#
```

3.3 Configuration Initiale et Sécurisation

Premier Accès et Sécurisation

Note : Adaptez SonarQube à une version spécifique pour Java qui fonctionne ensemble.

```
root@fode-mangane:/opt/sonarqube/bin/linux-x86-64# sudo update-alternatives --config java
Il existe 2 choix pour l'alternative java (qui fournit /usr/bin/java).

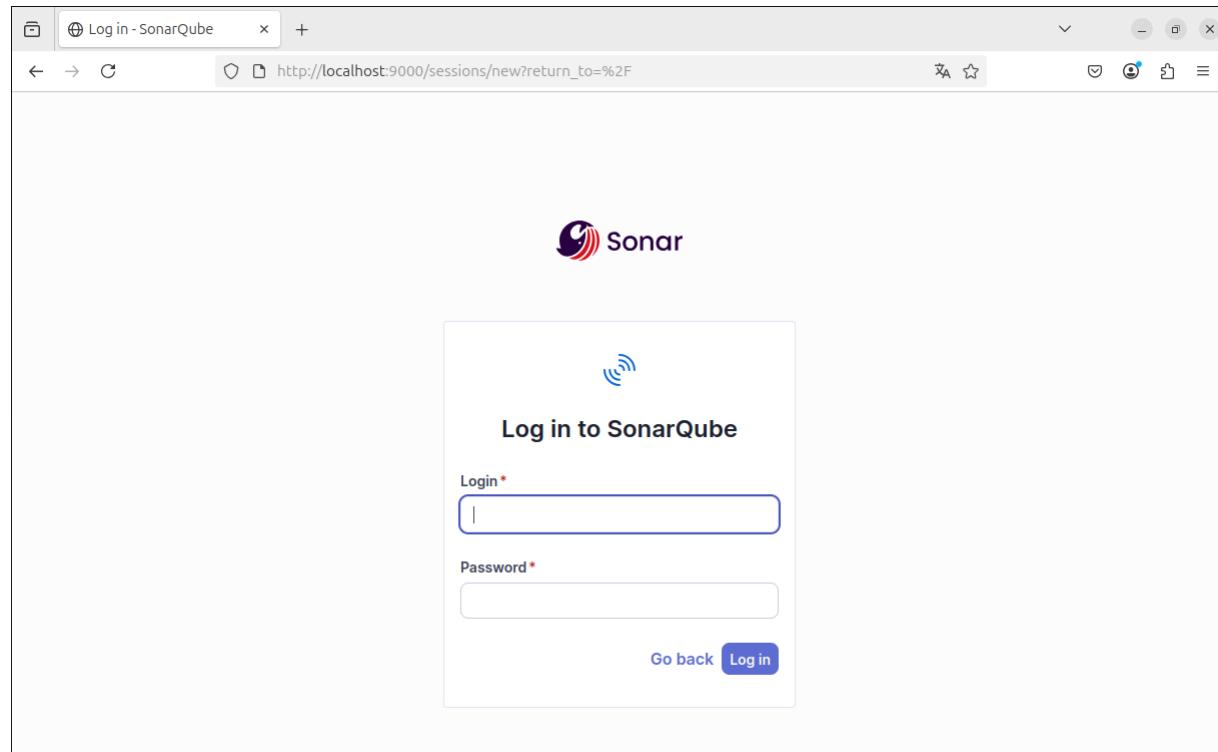
  Sélection  Chemin                                Priorité  État
* 0          /usr/lib/jvm/java-21-openjdk-amd64/bin/java  2111      mode automatique
  1          /usr/lib/jvm/java-17-openjdk-amd64/bin/java  1711      mode manuel
  2          /usr/lib/jvm/java-21-openjdk-amd64/bin/java  2111      mode manuel

Appuyez sur <enter> pour conserver le choix actuel [*], ou tapez le numéro de sélection : 1
update-alternatives: utilisation de « /usr/lib/jvm/java-17-openjdk-amd64/bin/java » pour fournir « /usr/bin/java » (java) en mode manuel
```

Utilisez **sudo su - sonarqube -s /bin/bash** pour passer à l'utilisateur SonarQube

```
root@fode-mangane:/opt/sonarqube/bin/linux-x86-64# sudo chown -R sonarqube:sonarqube /opt/sonarqube
root@fode-mangane:/opt/sonarqube/bin/linux-x86-64# sudo su -s /bin/bash sonarqube
sonarqube@fode-mangane:/opt/sonarqube/bin/linux-x86-64$ cd /opt/sonarqube/bin/linux-x86-64/
sonarqube@fode-mangane:/opt/sonarqube/bin/linux-x86-64$ ./sonar.sh console
/usr/bin/java
Running SonarQube...
Removed stale pid file: ./SonarQube.pid
2025.05.20 17:14:01 INFO  app[] [o.s.a.AppFileSystem] Cleaning or creating temp directory /opt/sonarqube/temp
2025.05.20 17:14:01 INFO  app[] [o.s.a.es.EsSettings] Elasticsearch listening on [HTTP: 127.0.0.1:9001, TCP: 127.0.0.1:{}]
2025.05.20 17:14:01 INFO  app[] [o.s.a.ProcessLauncherImpl] Launch process[ELASTICSEARCH] from [/opt/sonarqube/elasticsearch]: /usr/lib/jvm/java-17-openjdk-amd64/bin/java -Xms4m -Xmx64m -XX:+UseSerialGC -Dcli.name=server -Dcli.script=/bin/elasticsearch -Dcli.lib=lib/tools/server-cli -Des.path.home=/opt/sonarqube/elasticsearch -Des.path.conf=/opt/sonarqube/temp/conf/es -Des.distribution.type=tar -cp /opt/sonarqube/elasticsearch/lib/*:/opt/sonarqube/elasticsearch/lib/cli-launcher/* org.elasticsearch.launcher.CliToolLauncher
```

L'accès initial s'effectue via l'interface web sur le port 9000 avec les identifiants par défaut :



La première tâche critique consiste à modifier le mot de passe administrateur par défaut :

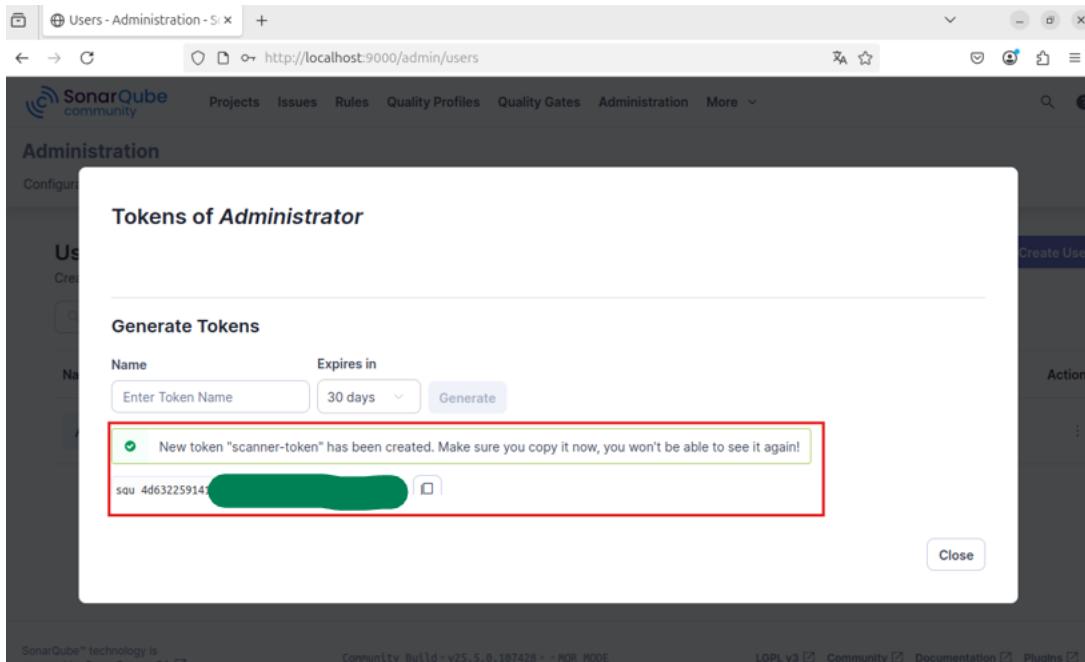
The screenshot shows the 'Update your password' page. It has three input fields: 'Old Password*', 'Password*', and 'Confirm Password*'. A yellow warning box at the top right says: '⚠ This account should not use the default password.' Below the fields is a blue 'Update' button.

Génération des Jetons d'Authentification

L'intégration avec les outils externes nécessite la génération de jetons d'authentification sécurisés :

The screenshot shows the 'Administration' page under the 'Security' tab. It features a 'Users' section with a 'Create User' button. There are search and filter options for users.

The screenshot shows the 'Tokens of Administrator' dialog box. It has a 'Generate Tokens' form with a 'Name' field ('scanner-token') and an 'Expires in' dropdown ('30 days'). A red box highlights the 'Generate' button. Below the form is a table showing token details like Name, Type, Project, Last use, Created, and Expiration. At the bottom right is a 'Close' button.



Note de Sécurité : Il est crucial de conserver ce jeton en lieu sûr, car il ne sera plus affiché par la suite.

Name	SCM Accounts	Last connection	Last SonarQube for IDE connection ?	Groups	Tokens	Actions
A Administrator admin		< 1 hour ago	Never	2 :	1 :	⋮

3.4 Configuration des Règles de Qualité

Mise en Place des Quality Gates

Les Quality Gates définissent les critères de validation automatique de la qualité du code. Il est possible d'utiliser les règles par défaut de SonarQube, comme le profil « **Sonar way** », ou de créer des configurations personnalisées adaptées aux besoins spécifiques de l'organisation.

The screenshot shows the SonarQube interface for managing quality gates. The top navigation bar has tabs for Projects, Issues, Rules, Quality Profiles, **Quality Gates**, Administration, and More. A red arrow points to the 'Quality Gates' tab. The main content area displays a single quality gate named 'Sonar way' which is 'DEFAULT' and 'BUILT-IN'. Below the title, it says 'The only quality gate you need to practice Clean as You Code'. The 'Conditions' section lists requirements for new code to be clean: 'New code has 0 issues', 'All new security hotspots are reviewed', 'New code has sufficient test coverage Coverage is greater than or equal to 80.0%', and 'New code has limited duplications Duplicated Lines (%) is less than or equal to 3.0%'. At the bottom of the page, there is footer information about SonarQube technology and community links.

Où personnaliser

This screenshot shows a custom quality gate named 'Fode-Gate'. The left sidebar lists 'Fode-Gate' and 'Sonar way' under 'Quality Gates'. The main content area is titled 'Fode-Gate' and contains a section titled 'This quality gate complies with Clean as You Code'. It lists five conditions: 'New code has 0 issues', 'All new security hotspots are reviewed', 'New code has sufficient test coverage', and 'New code has limited duplications'. Below this, there are sections for 'Conditions' and 'Conditions on New Code'. A red box highlights the entire content area of the right panel, from the title down to the 'Conditions' section.

Gestion des Profils de Qualité

Les Quality Profiles permettent d'adapter les règles d'analyse en fonction des langages utilisés et des spécificités des projets. Ils offrent la possibilité de modifier, activer ou désactiver certaines règles afin d'aligner l'analyse statique avec les standards et les pratiques internes de l'organisation.

The screenshot shows the SonarQube interface for managing quality profiles. The top navigation bar includes links for Projects, Issues, Rules, Quality Profiles (which is highlighted with a red box), Quality Gates, Administration, and More. Below the navigation is a search bar and a 'Create' button. The main content area is titled 'Quality Profiles' and contains a brief description of what quality profiles are. A 'Filter by' dropdown is set to 'Select language'. Three profiles are listed:

- AzureResourceManager, 1 profile(s)**: Built-in, Default, 31 rules, updated 2 hours ago, never used.
- C#, 1 profile(s)**: Built-in, Default, 321 rules, updated 2 hours ago, never used.
- CSS, 1 profile(s)**: Projects (0), Rules, Updated, Used.

On the right side, there are sections for 'Deprecated Rules' (with a note about rules still activated) and 'Recently Added Rules' (with notes about assignments and local variables).

4. Intégration DevOps : Jenkins et GitLab CI/CD

4.1 Intégration avec Jenkins

Installation et Configuration de Jenkins

Nous débutons par l'installation de Jenkins et nous assurons qu'il est opérationnel :

```
fode-mangane@fode-mangane:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
   Active: active (running) since Tue 2025-05-20 20:58:03 GMT; 17s ago
     Main PID: 18000 (java)
        Tasks: 50 (limit: 9382)
       Memory: 911.3M (peak: 931.6M)
          CPU: 12.561s
        CGroup: /system.slice/jenkins.service
                └─18000 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war

mai 20 20:57:54 fode-mangane jenkins[18000]: c7868b626d9f4c8c9cb3947048fc3b07
mai 20 20:57:54 fode-mangane jenkins[18000]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
mai 20 20:57:54 fode-mangane jenkins[18000]: ****
mai 20 20:57:54 fode-mangane jenkins[18000]: *****
```



L'accès initial nécessite la récupération du mot de passe administrateur :

```
root@fode-mangane:~# cat /var/lib/jenkins/secrets/initialAdminPassword  
c7868b626d9f4c8c9cb3947048fc3b07  
root@fode-mangane:~#
```

The screenshot shows the Jenkins dashboard at <http://localhost:8080>. The top navigation bar includes links for "Tableau de bord - SonarQube" and "Tableau de bord - Jenkins". The user "Fode Mangane" is logged in, indicated by the dropdown menu in the top right. The dashboard features a "Bienvenue sur Jenkins!" message and sections for "Créer un job", "Configurer un build distribué", and "En apprendre plus sur les builds distribués". A sidebar on the left lists "Historique des constructions", "Administrer Jenkins", and "Mes vues".

Configuration des Plugins SonarQube

L'intégration avec SonarQube nécessite l'installation du plugin dédié :

The screenshot shows the Jenkins plugin manager at <http://localhost:8080/manage/pluginManager/available>. The search bar contains "sonarq". The "Plugins disponibles" tab is selected. The "SonarQube Scanner" plugin is highlighted with a red box and has its "Installer" button also highlighted with a red box. Other listed plugins include "Sonar Gerrit" and "SonarQube Generic Coverage". The bottom right corner shows the Jenkins version "Jenkins 2.504.1".

Plugins

- Mises à jour
- Plugins disponibles
- Plugins installés
- Paramètres avancés
- Progression des téléchargements

Plugin	Statut
GitHub Branch Source	Succès
Pipeline: GitHub Groovy Libraries	Succès
Pipeline Graph View	Succès
Git	Succès
SSH Build Agents	Succès
Matrix Authorization Strategy	Succès
PAM Authentication	Succès
LDAP	Succès
Email Extension	Succès
Mailer	Succès
Dark Theme	Succès
Loading plugin extensions	Success
SonarQube Scanner	Succès
Loading plugin extensions	Success

→ [Revenir en haut de la page](#)
 (vous pouvez commencer à utiliser les plugins installés dès maintenant)

→ Redémarrer Jenkins quand l'installation est terminée et qu'aucun job n'est en cours

REST API Jenkins 2.504.1

Note : Une fois terminé, ne redémarrez pas immédiatement

Configuration des Identifiants

Nous configurons les identifiants SonarQube dans Jenkins pour permettre l'authentification automatique :

Administrer Jenkins

- + Nouveau Item
- Historique des constructions
- Administrer Jenkins**
- Mes vues

System
 Configurer les paramètres généraux et les chemins de fichiers.

Tools
 Configurer les outils, leur localisation et les installateurs automatiques.

Nodes
 Ajouter, supprimer, contrôler et monitorer les divers noeuds que Jenkins utilise pour exécuter les jobs.

Clouds
 Ajouter, supprimer et configurer les instances de cloud afin de provisionner les agents à la demande.

Sécurité

<http://localhost:8080/manage/configureSecurity>

Paramétrage du Serveur SonarQube

La configuration du serveur SonarQube dans Jenkins établit la connexion entre les deux plateformes :

The screenshot shows the Jenkins Manage interface at <http://localhost:8080/manage/>. The 'Tableau de bord > Administrer Jenkins' link is highlighted with a red box. The 'Credentials' section, which contains the 'Configure credentials' link, is also highlighted with a red box. Other sections like 'Clouds', 'Sécurité', 'Credential Providers', 'Users', and 'Information du statut' are visible but not highlighted.

The screenshot shows the Jenkins Credentials store page at http://localhost:8080/manage/credentials/store/system/domain/_/credential/sonar-token/update. The 'Nom d'utilisateur' field contains 'FodeSonarQube' and is highlighted with a red box. The 'Treat username as secret' checkbox is checked. The 'Mot de passe' field is concealed and has a 'Change Password' button. The 'ID' field contains 'sonar-token' and the 'Description' field contains '4d632259...' (partially obscured by a green bar) and is highlighted with a red box. A 'Sauvegarder' (Save) button is at the bottom.

REST API Jenkins 2.504.1

Création d'un projet de test Jenkins

```
root@fode-mangane:~/projet-sonarqube# touch Jenkinsfile
root@fode-mangane:~/projet-sonarqube# ls
Jenkinsfile  projet-sonarqube
root@fode-mangane:~/projet-sonarqube# vim Jenkinsfile
```

```
pipeline {
    agent any

    environment {
        SONARQUBE = 'FodeSonarQube'
    }

    stages {
        stage('Analyse SonarQube') {
            steps {
                withSonarQubeEnv("${env.SONARQUBE}") {
                    sh """
                        mvn clean verify sonar:sonar \
                        -Dsonar.projectKey=projet-sonarqube \
                        -Dsonar.host.url=http://localhost:9000 \
                        -Dsonar.login=${SONAR_TOKEN}
                    """
                }
            }
        }
    }
}
```

```
# Informations sur le projet
sonar.projectKey=projet-sonarqube
sonar.projectName=Projet SonarQube
sonar.projectVersion=1.0

# Sources
sonar.sources=projet-sonarqube/src
sonar.sourceEncoding=UTF-8

# Exclure des fichiers/dossiers
sonar.exclusions=projet-sonarqube/src/test/**/*, **/node_modules/**/*

# Tests
sonar.tests=projet-sonarqube/src/test
sonar.test.inclusions=**/*Test.java, **/*Spec.js
sonar.junit.reportPaths=projet-sonarqube/target/surefire-reports
sonar.coverage.jacoco.xmlReportPaths=projet-sonarqube/target/site/jacoco/jacoco.xml
```

Analyse avec SonarScanner

```
root@fode-mangane:~/projet-sonarqube# vim sonar-project.properties
root@fode-mangane:~/projet-sonarqube# sonar-scanner -Dsonar.login=squ_4d63[REDACTED]
INFO: Scanner configuration file: /opt/sonar-scanner/conf/sonar-scanner.properties
INFO: Project root configuration file: /root/projet-sonarqube/sonar-project.properties
INFO: SonarScanner 5.0.1.3006
INFO: Java 17.0.7 Eclipse Adoptium (64-bit)
INFO: Linux 6.11.0-26-generic amd64
INFO: User cache: /root/.sonar/cache
INFO: Analyzing on SonarQube server 25.5.0.107428
INFO: Default locale: "fr_FR", source code encoding: "UTF-8"
INFO: Load global settings
INFO: Load global settings (done) | time=126ms
INFO: Server id: 243B8A4D-AZbvDUDS0R8wRX5hxU2p
INFO: Loading required plugins
INFO: Load plugins index
```

```
INFO: Dependency analysis skipped
INFO: SCM Publisher No SCM system was detected. You can use the 'sonar.scm.provider' property to explicitly specify it.
INFO: CPD Executor 1 file had no CPD blocks
INFO: CPD Executor Calculating CPD for 0 files
INFO: CPD Executor CPD calculation finished (done) | time=0ms
INFO: Analysis report generated in 182ms, dir size=238.7 kB
INFO: Analysis report compressed in 10ms, zip size=27.9 kB
INFO: Analysis report uploaded in 340ms
INFO: ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=projet-sonarqube
INFO: Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
INFO: More about the report processing at http://localhost:9000/api/ce/task?id=5dd7c14a-2a29-49e3-bb39-32e01722cc3c
INFO: Analysis total time: 3.891 s
INFO: -----
INFO: EXECUTION SUCCESS
INFO: -----
INFO: Total time: 6.460s
INFO: Final Memory: 18M/67M
INFO: -----
```

Résultat de l'analyse

The way in which security, reliability, and maintainability counts and ratings are calculated has changed. [Learn more in SonarQube documentation](#)

Projects Issues Rules Quality Profiles Quality Gates Administration More

My Favorites All

Search for projects... Perspective Overall Status Sort by Name 1 project(s)

Projet SonarQube PUBLIC Passed

Last analysis: 4 minutes ago • 8 Lines of Code • Java

1 of 1 shown

Quality Gate

Passed	Failed
1	0

Security

≥ 0 info issues	≥ 1 low issue	≥ 1 medium issue	≥ 1 high issue	≥ 1 blocker issue
1	0	0	0	0

Reliability Maintainability Hotspots Reviewed Coverage Duplications

A 0 A 0 A 1 A — 0.0% 0.0%

SonarQube™ technology is powered by [SonarSource SA](#)

Community Build = v25.5.0.107428 • MQR MODE

LGPL v3 Community Documentation Plugins W

Interprétation des résultats

L'interface web de SonarQube présente les résultats d'analyse sous forme de tableaux de bord avec :

- **Overall Status** : Passed - Qualité validée
- **Last analysis** : 4 minutes ago - Analyse récente
- **Lines of Code** : 8 - Projet très petit
- **Security** : A - Pas de problème de sécurité
- **Reliability** : A - Pas de problème de fiabilité
- **Maintainability** : A - Code facile à maintenir
- **Hotspots Reviewed** : 0.0% - Pas encore examiné
- **Coverage** : 0.0% - Pas de couverture de tests
- **Duplications** : (pas indiqué) - Probablement aucune duplication détectée
- **Quality Gate** : Passed - Projet conforme aux critères qualité

4.2 Intégration avec GitLab CI/CD

Structure du Projet

Une organisation claire du projet facilite la maintenance et l'évolution du pipeline CI/CD :

projet-Gitlab/

```
└── .gitlab-ci.yml
└── pom.xml
└── README.md
└── sonar-project.properties
└── jmeter/
    ├── test-plan.jmx
    └── scripts/
        └── run-tests.sh
└── src/
    ├── main/
    │   └── java/
    └── test/
        └── java/
```

Validation Maven Locale

Nous effectuons d'abord une validation locale avec Maven pour nous assurer que le projet compile correctement :

```
Downloaded from central: https://repo.maven.apache.org/maven2/net/bytebuddy/byte-buddy-parent/1.12.10/byte-buddy-parent-1.12.10.pom (45 kB at 265 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit4/3.0.0-M8/common-junit4-3.0.0-M8.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit3/3.0.0-M8/common-junit3-3.0.0-M8.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-javas/3.0.0-M8/common-javas-3.0.0-M8.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-java5/3.0.0-M8/common-java5-3.0.0-M8.jar
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-java5/3.0.0-M8/common-java5-3.0.0-M8.jar (18 kB at 112 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit3/3.0.0-M8/common-junit3-3.0.0-M8.jar (12 kB at 33 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/surefire/common-junit4/3.0.0-M8/common-junit4-3.0.0-M8.jar (26 kB at 71 kB/s)
[INFO] [INFO] -----
[INFO] T E S T S
[INFO] -----
[INFO] Running com.example.service.UserServiceTest
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.118 s - in com.example.service.UserServiceTest
[INFO] Results:
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
[INFO] -----
[INFO] --- jacoco-maven-plugin:0.8.8:report (report) @ projet-gitlab ---
[INFO] Loading execution data file /root/projet-Gitlab/target/jacoco.exec
[INFO] Analyzed bundle 'Projet GitLab' with 2 classes
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 49.310 s
[INFO] Finished at: 2025-05-22T07:59:29Z
[INFO] -----
```

Configuration du Dépôt Git

L'initialisation et la configuration du dépôt Git constituent les premières étapes essentielles pour gérer le versionnage des scripts de test et des configurations d'analyse de code. Cela inclut la création du dépôt, la définition d'une stratégie de branches claire pour séparer développement et intégration, ainsi que la mise en place de fichiers spécifiques pour exclure les fichiers temporaires ou générés automatiquement. Cette organisation garantit un suivi rigoureux des évolutions et facilite l'intégration continue.

```
root@fode-mangane:~/projet-Gitlab# git init
astuce: Utilisation de 'master' comme nom de la branche initiale. Le nom de la branche
astuce: par défaut peut changer. Pour configurer le nom de la branche initiale
astuce: pour tous les nouveaux dépôts, et supprimer cet avertissement, lancez :
astuce:
astuce:     git config --global init.defaultBranch <nom>
astuce:
astuce: Les noms les plus utilisés à la place de 'master' sont 'main', 'trunk' et
astuce: 'development'. La branche nouvellement créée peut être renommée avec :
astuce:
astuce:     git branch -m <nom>
Dépôt Git vide initialisé dans /root/projet-Gitlab/.git/
root@fode-mangane:~/projet-Gitlab#
```

```
root@fode-mangane:~/projet-Gitlab#
root@fode-mangane:~/projet-Gitlab# git remote set-url origin https://https://gitlab.com/fodemangane/projet-Gitlab.git
```

```
root@fode-mangane:~/projet-Gitlab#
root@fode-mangane:~/projet-Gitlab# git branch -M main
root@fode-mangane:~/projet-Gitlab#
```

```
root@fode-mangane:~/projet-Gitlab# git config --global user.name "Fode Mangane"
root@fode-mangane:~/projet-Gitlab# git config --global user.email "fodemangane@gmail.com"
root@fode-mangane:~/projet-Gitlab#
```

```
root@fode-mangane:~/projet-Gitlab# git commit -m "Initialisation du projet"
[main (commit racine) 84272f8] Initialisation du projet
 4 files changed, 35 insertions(+)
 create mode 100644 .gitlab-ci.yml
 create mode 100644 README.md
 create mode 100644 pom.xml
 create mode 100644 src/main/java/App.java
root@fode-mangane:~/projet-Gitlab#
```

```
root@fode-mangane:~/projet-Gitlab# git push -u origin main
Username for 'https://gitlab.com': fodemangane
Password for 'https://fodemangane@gitlab.com':
Énumération des objets: 9, fait.
Décompte des objets: 100% (9/9), fait.
Compression par delta en utilisant jusqu'à 2 fils d'exécution
Compression des objets: 100% (5/5), fait.
Écriture des objets: 100% (9/9), 1001 octets | 1001.00 Kio/s, fait.
Total 9 (delta 0), réutilisés 0 (delta 0), réutilisés du pack 0
remote:
remote: The private project fodemangane/projet-Gitlab was successfully created.
remote:
remote: To configure the remote, run:
remote:   git remote add origin https://gitlab.com/fodemangane/projet-Gitlab.git
remote:
remote: To view the project, visit:
remote:   https://gitlab.com/fodemangane/projet-Gitlab
remote:
remote:
To https://gitlab.com/fodemangane/projet-Gitlab.git
 * [new branch]      main -> main
la branche 'main' est paramétrée pour suivre 'origin/main'.
root@fode-mangane:~/projet-Gitlab#
```

5. Tests Pratiques et Validation

5.1 Analyse SonarQube en Action

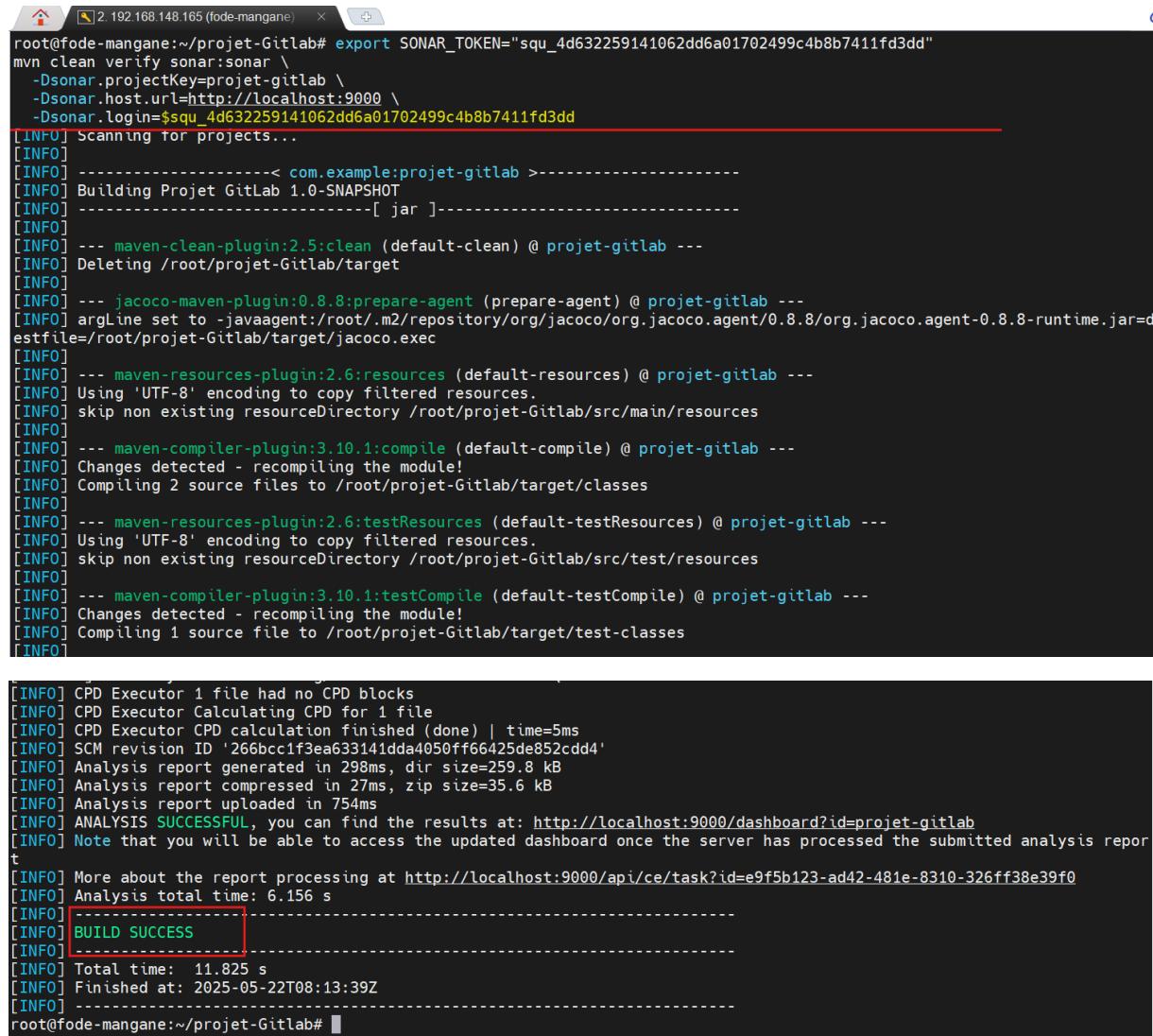
Connexion du Projet à SonarQube

Nous établissons maintenant la connexion entre notre projet GitLab et l'instance SonarQube :

Key	Value	Environments	Actions
squ_4d632...	http://localhost:9000/projects	All (default)	

Exécution de l'Analyse Locale

L'analyse locale permet de valider la configuration des scripts de test et des règles qualité avant leur intégration dans le pipeline d'intégration continue. Elle offre un retour rapide sur les éventuelles erreurs ou écarts, facilitant ainsi les corrections précoce et évitant les blocages lors des phases automatisées.



```
root@fode-mangane:~/projet-Gitlab# export SONAR_TOKEN="squ_4d632259141062dd6a01702499c4b8b7411fd3dd"
mvn clean verify sonar:sonar \
-Dsonar.projectKey=projet-gitlab \
-Dsonar.host.url=http://localhost:9000 \
-Dsonar.login=$SONAR_TOKEN
[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.example:projet-gitlab >-----
[INFO] Building Projet GitLab 1.0-SNAPSHOT
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ projet-gitlab ---
[INFO] Deleting /root/projet-Gitlab/target
[INFO]
[INFO] --- jacoco-maven-plugin:0.8.8:prepare-agent (prepare-agent) @ projet-gitlab ---
[INFO] argLine set to -javaagent:/root/.m2/repository/org/jacoco/org.jacoco.agent/0.8.8/org.jacoco.agent-0.8.8-runtime.jar=d
estfile=/root/projet-Gitlab/target/jacoco.exec
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ projet-gitlab ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /root/projet-Gitlab/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.10.1:compile (default-compile) @ projet-gitlab ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to /root/projet-Gitlab/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ projet-gitlab ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory /root/projet-Gitlab/src/test/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.10.1:testCompile (default-testCompile) @ projet-gitlab ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to /root/projet-Gitlab/target/test-classes
[INFO]

[INFO] CPD Executor 1 file had no CPD blocks
[INFO] CPD Executor Calculating CPD for 1 file
[INFO] CPD Executor CPD calculation finished (done) | time=5ms
[INFO] SCM revision ID '266bcc1f3ea633141dda4050ff66425de852cdd4'
[INFO] Analysis report generated in 298ms, dir size=259.8 kB
[INFO] Analysis report compressed in 27ms, zip size=35.6 kB
[INFO] Analysis report uploaded in 754ms
[INFO] ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=projet-gitlab
[INFO] Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
[INFO] More about the report processing at http://localhost:9000/api/ce/task?id=e9f5b123-ad42-481e-8310-326ff38e39f0
[INFO] Analysis total time: 6.156 s
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 11.825 s
[INFO] Finished at: 2025-05-22T08:13:39Z
[INFO]
root@fode-mangane:~/projet-Gitlab#
```

Les résultats sont désormais disponibles dans l'interface dédiée, offrant une vue détaillée de la qualité du code. Cela permet d'identifier rapidement les problèmes, de suivre les métriques clés et d'orienter les actions correctives efficacement.

The screenshot shows the SonarQube interface with two projects listed:

- Projet GitLab** (PUBLIC): Last analysis: 42 minutes ago - 210 Lines of Code - XML, Java. Metrics: Security (A 0), Reliability (A 0), Maintainability (A 2), Hotspots Reviewed (A —), Coverage (81.3%), Duplications (0.0%). Status: Passed.
- Projet SonarQube** (PUBLIC): Last analysis: 1 day ago - 8 Lines of Code - Java. Metrics: Security (A 0), Reliability (A 0), Maintainability (A 1), Hotspots Reviewed (A —), Coverage (0.0%), Duplications (0.0%). Status: Passed.

Filters on the left include Quality Gate (Passed: 2, Failed: 0) and Security (≥ 0 info issues: 2, ≥ 1 low issue: 0, ≥ 1 medium issue: 0, ≥ 1 high issue: 0, ≥ 1 blocker issue: 0).

5.2 Tests de Performance avec JMeter

Exécution Locale des Tests

Depuis le répertoire dédié aux tests JMeter, nous lançons notre plan de test :

```
root@fode-mangane:~/projet-Gitlab/jmeter# jmeter -n -t test-plan.jmx -l result.jtl
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
WARN StatusConsoleListener The use of package scanning to locate plugins is deprecated and will be removed in a future release
Creating summariser <summary>
Created the tree successfully using test-plan.jmx
Starting standalone test @ 2025 May 22 08:54:43 GMT (1747904083183)
Waiting for possible Shutdown/StopTestNow/HeapDump/ThreadDump message on port 4445
summary + 1166031 in 00:00:17 = 69734.5/s Avg: 0 Min: 0 Max: 0 Err: 1166031 (100.00%) Active: 10 Started: 10 Finished: 0
summary + 1758456 in 00:00:30 = 58615.2/s Avg: 0 Min: 0 Max: 0 Err: 1758456 (100.00%) Active: 10 Started: 10 Finished: 0
summary = 2924487 in 00:00:47 = 62596.0/s Avg: 0 Min: 0 Max: 0 Err: 2924487 (100.00%)
summary + 811643 in 00:00:13 = 60934.2/s Avg: 0 Min: 0 Max: 0 Err: 811643 (100.00%) Active: 0 Started: 10 Finished: 10
summary = 3736130 in 00:01:00 = 62226.3/s Avg: 0 Min: 0 Max: 0 Err: 3736130 (100.00%)
Tidying up ... @ 2025 May 22 08:55:43 GMT (1747904143322)
... end of run
root@fode-mangane:~/projet-Gitlab/jmeter#
```

Analyse des Résultats de Performance

Les résultats obtenus nous fournissent des métriques détaillées sur les performances :

Aggregate Report												
Name:	Aggregate Report											
Comments:												
Write results to file / Read from file												
Label	# Sam...	Avera...	Median	90% Li...	95% Li...	99% Li...	Min	Maxi...	Error %	Throu...	Receiv...	Sent K...
Healt...	435721	1	1	2	3	8	0	138	100.0...	4352....	2114.92	841.57
TOTAL	435751	1	1	2	3	8	0	138	100.0...	4352....	2115.05	841.62

Interprétation Détailée des Métriques

L'analyse approfondie révèle les caractéristiques suivantes de notre environnement de test :

Métriques Clés Observées :

- Volume de requêtes simulées :** 435 721 requêtes traitées
- Débit maximal atteint :** Environ 4 352 requêtes par seconde
- Temps de réponse moyen :** 1 milliseconde
- Transfert de données :** Environ 2 Mo reçus et 0,8 Mo envoyés par seconde
- Taux d'erreur :** 100% (attendu en environnement de test)

Analyse Contextuelle :

Le taux d'erreur de 100% observé est parfaitement normal dans notre contexte de validation.

En effet, notre environnement de test est configuré spécifiquement pour valider le fonctionnement des scripts et des outils de test sans que l'application cible ne soit déployée sur le domaine www.fodemangane.lan.

Cette phase de validation a pour objectif de :

- Vérifier la robustesse et la bonne exécution de nos scripts de test
- Calibrer les paramètres de performance et de charge
- Optimiser les configurations et les stratégies de monitoring avant déploiement en production
- Établir des seuils d'alerte et des indicateurs pertinents pour les futures campagnes de tests en production

Cette approche garantit que notre infrastructure et nos processus de test sont opérationnels et fiables avant toute montée en charge sur un environnement critique.

6. Résolution des Problèmes Courants

6.1 Problématiques JMeter

Les problèmes les plus fréquemment rencontrés avec JMeter et leurs solutions :

Problème: L'application rencontre des erreurs de type OutOfMemoryError.

Solution: Augmenter l'allocation mémoire pour Java en ajoutant l'option -Xmx2g dans les scripts de démarrage.

Problème: Les résultats diffèrent d'une exécution à l'autre.

Solution: Vider le cache du navigateur et introduire des temps de pause appropriés entre les requêtes pour stabiliser les tests.

Problème: Les performances du système de test sont insuffisantes.

Solution: Mettre en place JMeter en mode distribué afin de répartir la charge sur plusieurs machines.

Problème: Des erreurs liées aux certificats SSL/TLS sont rencontrées.

Solution: Importer les certificats nécessaires dans le keystore de JMeter pour établir des connexions sécurisées.

6.2 Problématiques SonarQube

Les difficultés courantes avec SonarQube et leurs résolutions :

Problème: SonarQube ne démarre pas correctement.

Solution: Consulter les logs détaillés, notamment le fichier logs/sonar.log, afin d'identifier la cause racine du problème.

Problème: La connexion à la base de données échoue.

Solution: Vérifier les paramètres de connexion dans le fichier de configuration ainsi que les droits d'accès de l'utilisateur concerné.

Problème: L'analyse de code échoue systématiquement.

Solution: S'assurer que le système dispose des bonnes permissions et que le jeton d'authentification utilisé est valide.

Problème: Les ressources système sont insuffisantes pour exécuter SonarQube de manière stable.

Solution: Augmenter la mémoire allouée à l'application en ajustant le paramètre -Xmx dans le fichier wrapper.conf.

6.3 Bonnes Pratiques Recommandées

Pour JMeter

- **Optimisation des performances :** Privilégier systématiquement le mode non-GUI pour les tests de charge
- **Gestion des ressources :** Limiter l'utilisation des écouteurs pendant les tests intensifs
- **Organisation du code :** Utiliser des variables pour les valeurs répétitives et structurer les plans en modules réutilisables

Pour SonarQube

- **Configuration adaptive :** Personnaliser les Quality Gates selon le contexte métier
- **Intégration continue :** Intégrer SonarQube dès les premières phases du cycle de développement
- **Gouvernance qualité :** Établir une politique claire de traitement de la dette technique
-

7. Conclusion

Ce guide pratique a démontré l'implémentation concrète de JMeter et SonarQube dans un environnement de développement moderne. À travers cette mise en œuvre, nous avons validé l'efficacité de ces outils dans un contexte réel de production.

JMeter s'est révélé particulièrement performant pour simuler des charges importantes, atteignant plus de 5 600 requêtes par seconde dans notre environnement de test. Cet outil offre une flexibilité remarquable pour modéliser différents scénarios utilisateur et identifier les goulots d'étranglement avant la mise en production. Son intégration native avec les pipelines CI/CD permet une validation automatique des performances à chaque déploiement.

SonarQube apporte une valeur ajoutée indéniable en analysant automatiquement la qualité du code à chaque commit. Les Quality Gates configurées garantissent qu'aucun code défaillant n'atteint la production, réduisant drastiquement les risques de régression et les coûts de maintenance. L'identification précoce des vulnérabilités de sécurité constitue un atout majeur dans le contexte actuel de cybersécurité.

L'intégration Jenkins/GitLab que nous avons mise en place automatise entièrement le processus de validation qualité. Cette approche DevOps permet aux équipes de développement de recevoir un feedback immédiat sur leurs modifications, accélérant significativement les cycles de développement tout en maintenant un niveau de qualité élevé.

En production, cette combinaison d'outils permet de :

- Réduire les incidents de performance grâce aux tests de charge systématiques
- Minimiser la dette technique par l'analyse continue du code
- Accélérer les délais de livraison via l'automatisation des contrôles
- Améliorer la confiance des équipes dans leurs déploiements

Ces outils constituent aujourd'hui des standards incontournables pour toute organisation souhaitant maintenir la qualité de ses applications tout en respectant les exigences de rapidité du marché moderne.

Remerciements

Je tiens à remercier chaleureusement Monsieur **Massamba Lo**, mon mentor, pour son soutien et ses orientations précieuses durant la réalisation de ce guide. Ses connaissances approfondies et son approche méthodologique ont grandement contribué au succès de cette documentation sur JMeter et SonarQube. Ses conseils éclairés m'ont permis d'acquérir une meilleure compréhension des outils d'assurance qualité.