



DevOp

Chapitre 4 : SonarQube

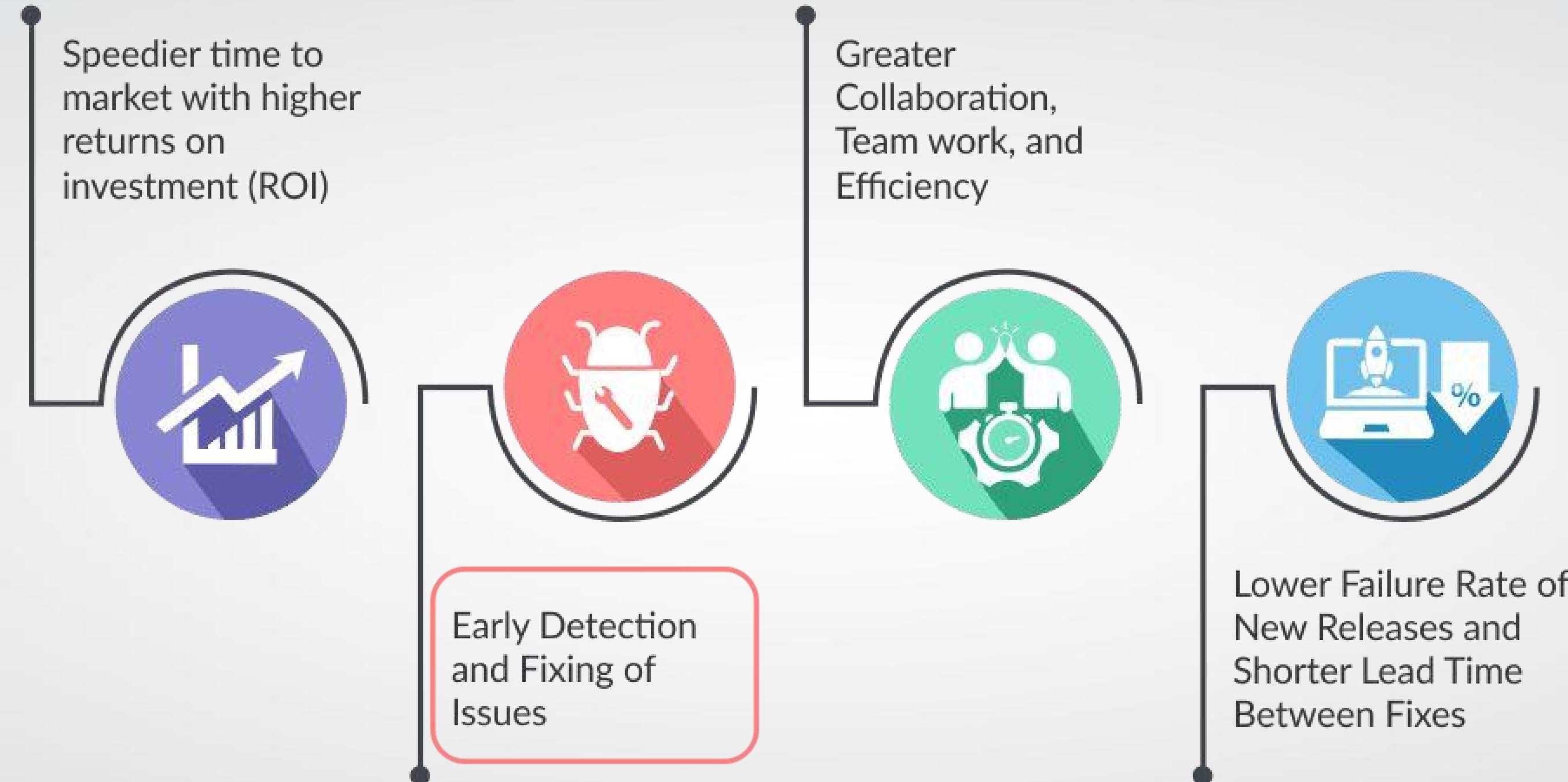
ESPRIT – UP ASI (Architecture des Systèmes d'Information)
Bureau E204

► Plan du cours

- Introduction Tests dynamiques et Tests statiques C'est quoi
- SONARQUBE Caractéristiques de SONARQUBE Installation de
- SONARQUBE (à partir d'une Image Docker) Utilisation de Sonarqube
- Compréhension des résultats d'analyse de code **Travail à faire**
- (Analyser le code d'un projet sur Git avec SONARQUBE Jenkins)
-
-
- en utilisant



► Introduction



► Problématique

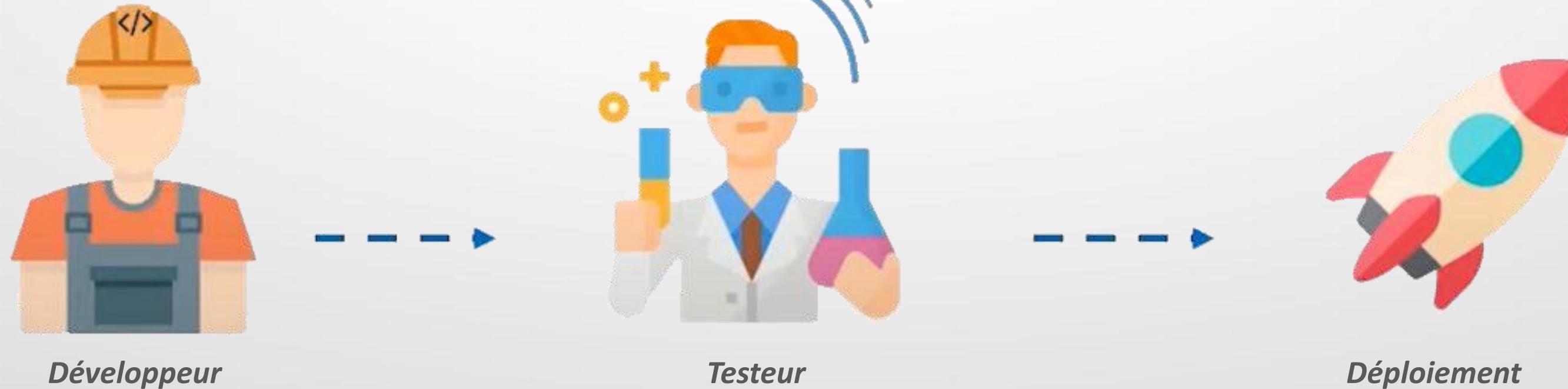


Comment s'assurer que le code des logiciels est de haute qualité dès le début du développement et comment garantir que les applications restent fiables à long terme ?



Tests dynamiques et Tests statiques

- Les **tests** font partie de cycle de vie du développement d'une application donnée.
- Les tests visent à s'assurer que le code qui sera déployé est de **bonne qualité**, **sécurisé** et **exempt de bugs** (environ 30% du temps de développement doit être consacré aux tests).



► Tests dynamiques



- **Les tests dynamiques** consistent à exécuter le code pour repérer les erreurs, les bugs et évaluer les performances à l'aide de différentes méthodes telles que *les tests unitaires, les tests d'intégration, les tests de régression et les tests de charge.*



► Tests statiques



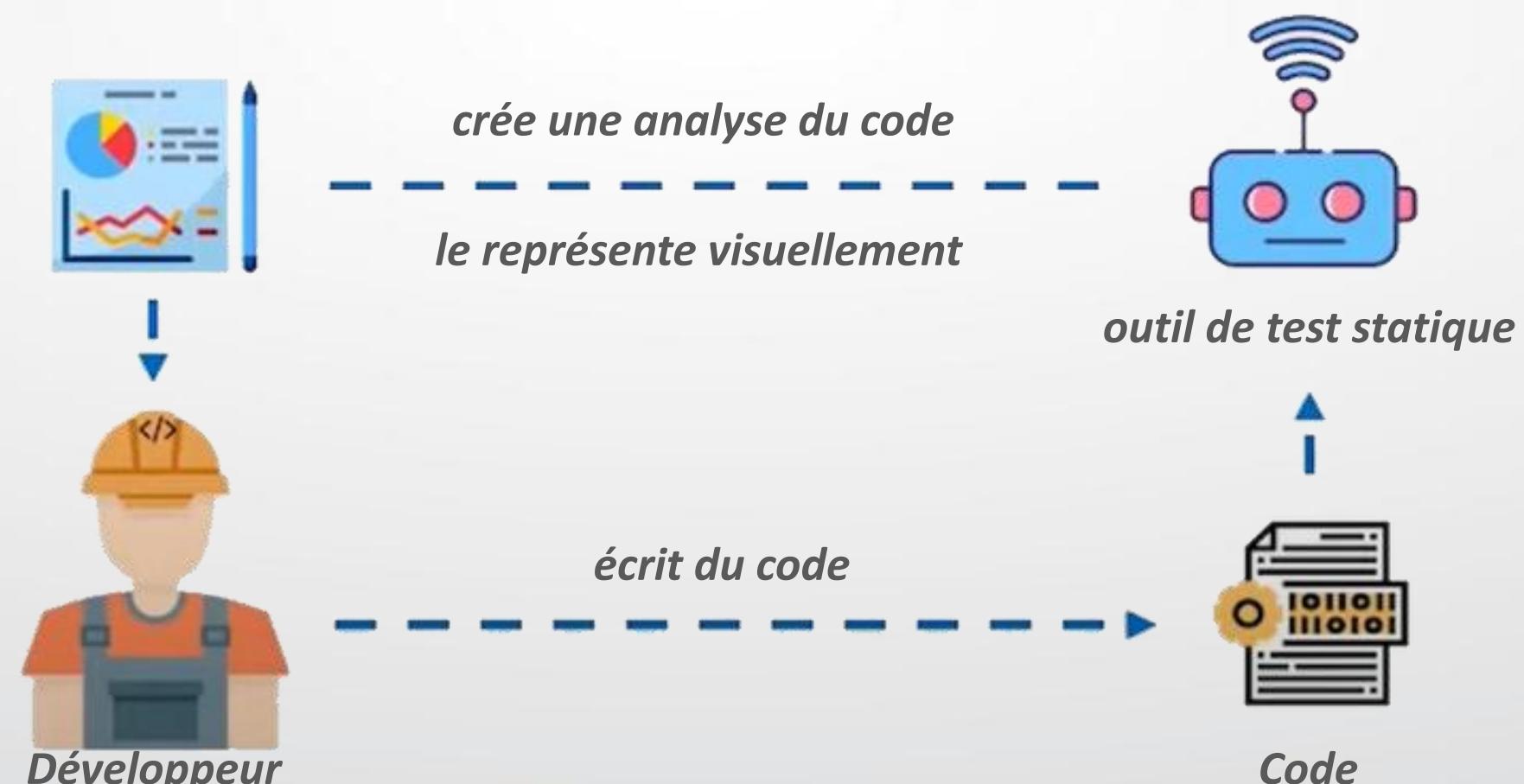
- **Les tests statiques** sont réalisés sans exécuter le code. Ils inspectent le code source, la conception et la documentation pour détecter des problèmes potentiels, comme *les violations de normes de codage, les erreurs de syntaxe, les incohérences de conception*, etc.

→ **SonarQube** est un exemple d'outil d'analyse statique de code utilisé pour effectuer ces tests.





- › SonarQube est un outil de test statique, open-source, qui analyse le code source pour détecter et corriger les problèmes de qualité, tels que les bugs, les vulnérabilités de sécurité et les mauvaises pratiques de codage. Il permet de
- › ***mesurer la qualité du code source en continu*** (revue de code automatique).



SonarQube - Caractéristiques

Les principales fonctionnalités de

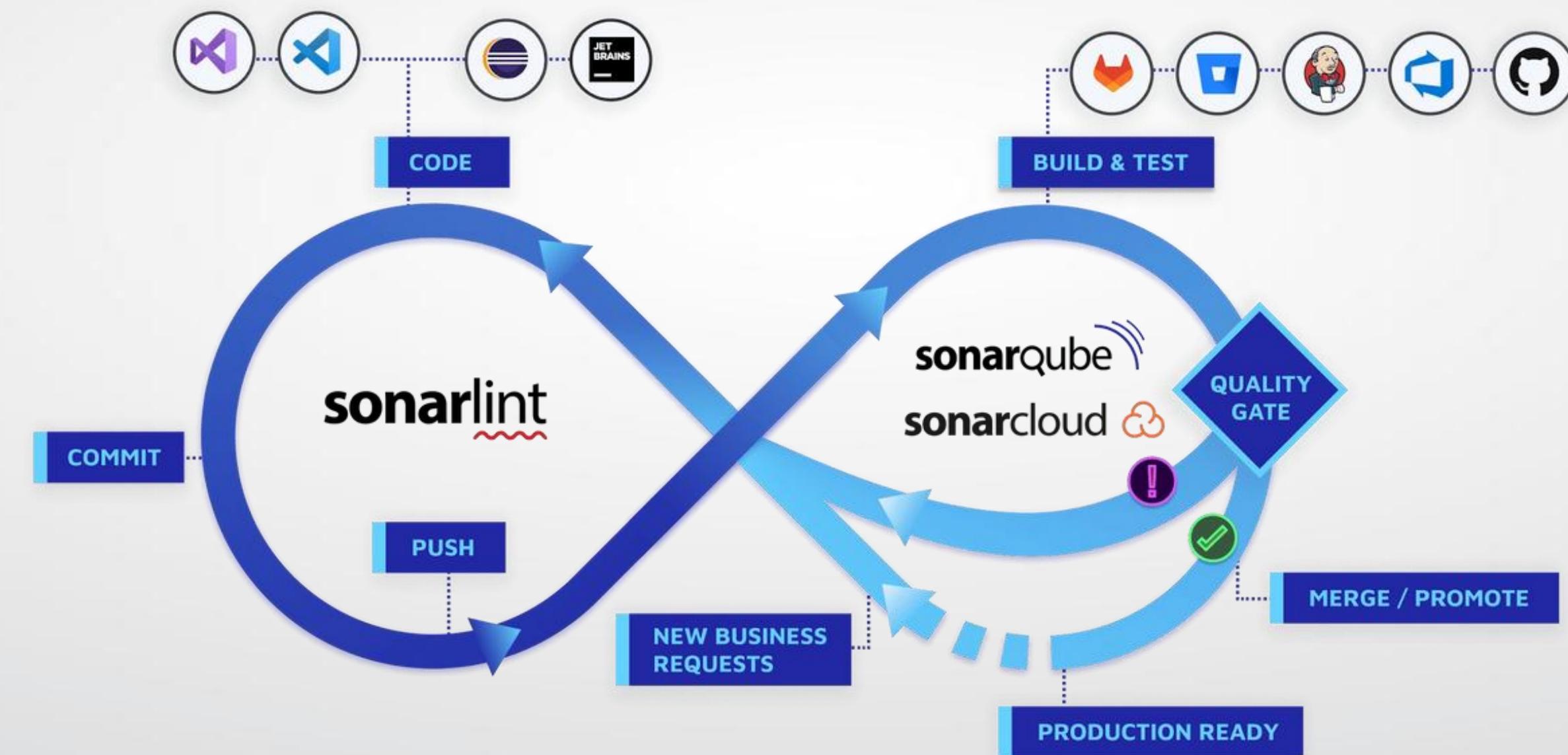


- › Analyse du code source. Prise en charge de nombreux langages (Java, .Net (C#), Python, PHP, JavaScript, **Détection de problèmes dans le code**, comme les bugs...).
- › et les vulnérabilités. Mesure de la qualité du code avec des métriques.
- › Intégration avec des outils de gestion de projets et de développement. Crédit à la création de rapports détaillés sur la qualité du code et son évolution. **Personnalisation**
- › **des règles de qualité.** Support de l'intégration continue (CI/CD).
- ›
- ›



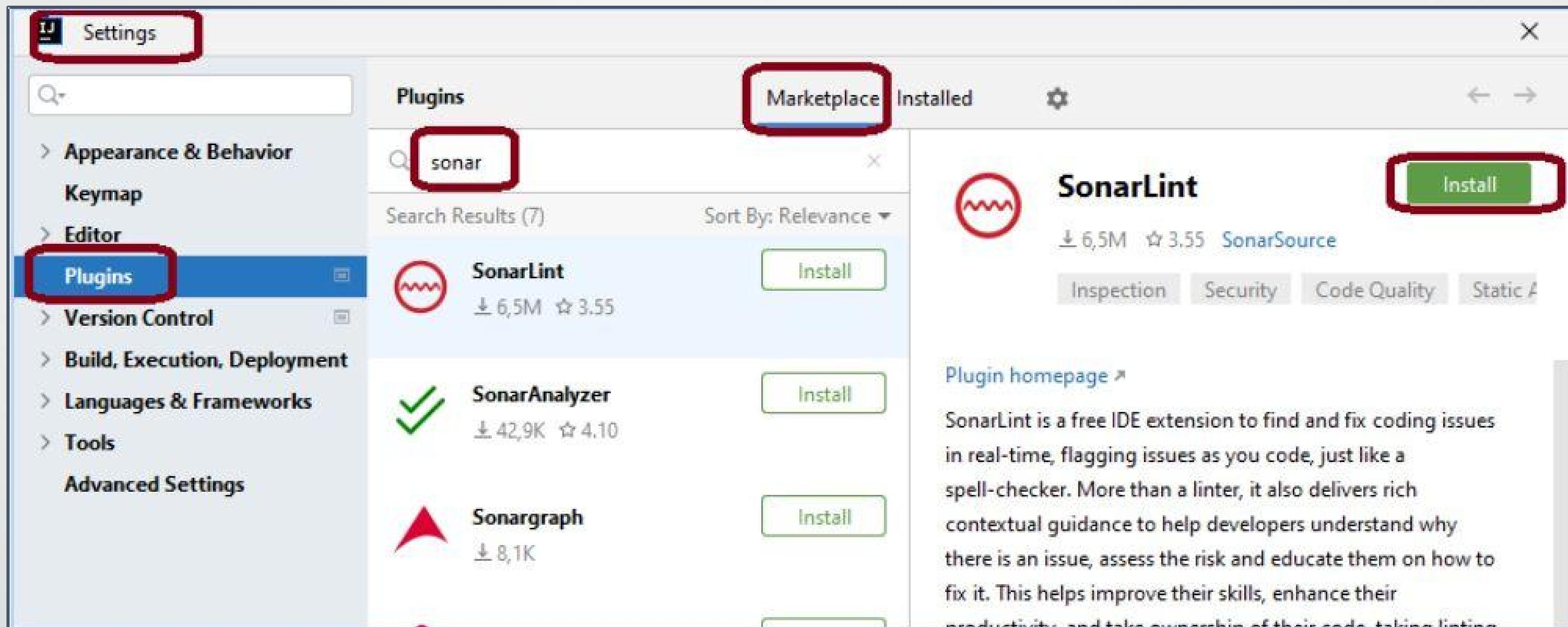
SonarQube - Plugin IDE

- › SonarQube peut être installé de manière indépendante (en mode **standalone**) ou intégré en tant que **plugin** dans un environnement de développement (IDE).



SonarQube - Plugin IntelliJ

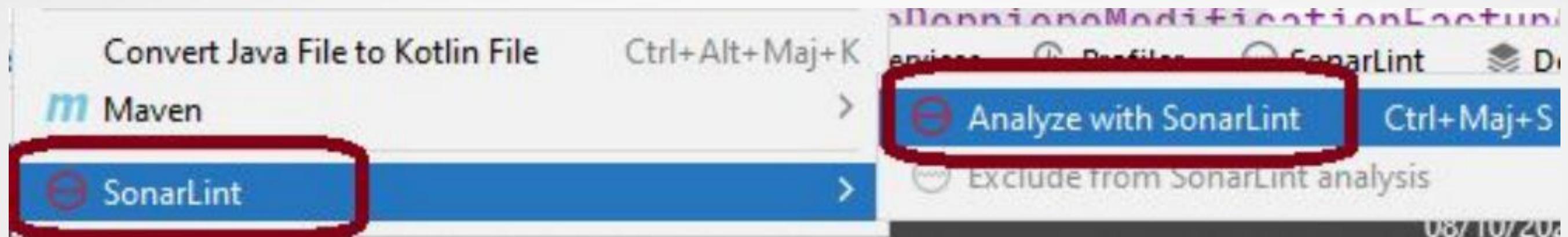
- › Dans IntelliJ, vous pouvez accéder à *Setting->Plugins->Marketplace*, puis recherchez « Sonar », installez le plugin « **SonarLint** », acceptez l'installation et acceptez de redémarrer IntelliJ à la fin du processus d'installation.



SonarQube - Plugin IntelliJ



- › Ouvrez l'un de vos projets dans IntelliJ, faites un clic droit et sélectionnez *SonarLint*
-> *Analyze with SonarLint*, puis examinez les résultats



sonarlint



SonarQube - Plugin IntelliJ

Résultat :

sonarlint

The screenshot shows the IntelliJ IDEA IDE interface with the SonarLint plugin active. The main window displays a Java file, `CategorieProduitController.java`, which contains code for retrieving categories of products. The SonarLint tool has identified several issues in this file, particularly related to local variable usage and return statements.

Issues Found:

- CategorieProduitController.java (5 issues):**
 - (37, 75) Replace this persistent entity with a simple POJO or DTO object.
 - (52, 78) Replace this persistent entity with a simple POJO or DTO object.
 - (42, 4) This block of commented-out lines of code should be removed.
 - (23, 32) Immediately return this expression instead of assigning it to the ter
 - (38, 38) Immediately return this expression instead of assigning it to the ter
- CategorieProduitServiceImpl.java (2 issues)**
- FactureRestController.java (7 issues)**
- FactureServiceImpl.java (7 issues)**

Rule: Local variables should not be declared and then immediately returned.

Intentionality: Intentionality issue | Not Clear

Maintainability: Maintainability (java:S1488)

Why is this an issue?

Declaring a variable only to immediately return or throw it is a bad practice. Some developers argue that the practice improves code readability, because it enables them to explicitly name what is being returned. However, this variable is an internal implementation detail that is not exposed to the callers of the

Analysis of 63 files done 2 minutes ago

Version Control TODO Problems Spring Terminal Endpoints Services Profiler SonarLint Dependencies

23:39 CRLF UTF-8 Tab* ⌂ 13

SonarQube - Installation



1. Téléchargez l'image Docker de SonarQube :

- › Connectez-vous à votre machine virtuelle Ubuntu en utilisant un client SSH. Pour ce faire, démarrez VirtualBox, ouvrez une fenêtre PowerShell et exécutez les commandes vagrant up et vagrant ssh.
- › Assurez-vous d'avoir effectué un chmod au préalable pour éviter les problèmes de droits d'accès. Utilisez la commande docker pull pour télécharger l'image SonarQube depuis le Docker Hub.

A screenshot of a terminal window titled "Sélection vagrant@vagrant: ~". The window shows two commands being run:

```
vagrant@vagrant: $ sudo chmod 666 /var/run/docker.sock
vagrant@vagrant: $ docker pull sonarqube
```

The output of the second command is:

```
Using default tag: latest
latest: Pulling from library/sonarqube
707e32e9fc56: Pull complete
8e560b9ae2a6: Pull complete
15ee7ce1b141: Pull complete
```

SonarQube - Lancement

2. Exécutez le conteneur SonarQube :

› Vous pouvez maintenant lancer un conteneur SonarQube en utilisant la commande docker run .

```
vagrant@vagrant:~$ docker run -d --name sonarqube -p 9000:9000 -p 9092:9092 sonarqube
```

-d : Lance le conteneur en arrière-plan
--name sonarqube : Attribue un nom au conteneur, ici "sonarqube"
-p 9000:9000 -p 9092:9092 : Associe les ports du conteneur aux ports correspondants de l'hôte, permettant ainsi d'accéder à l'interface web et au broker de messagerie.

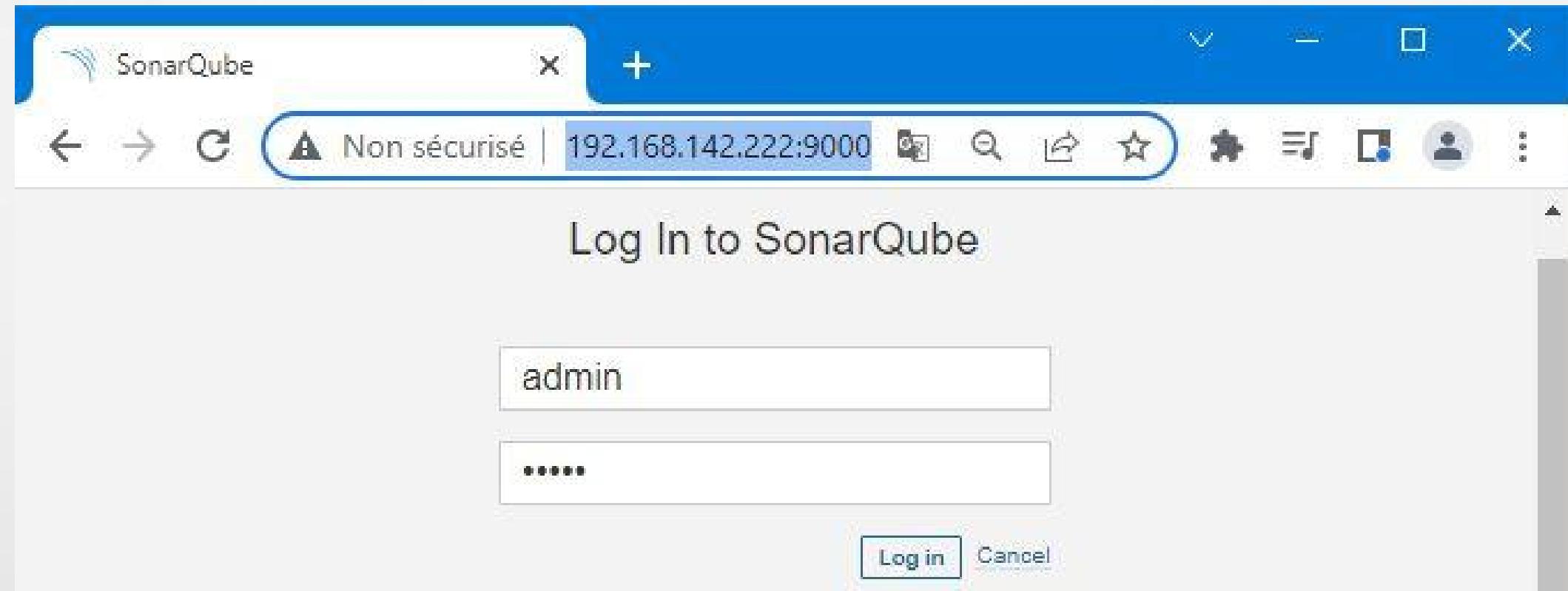


SonarQube - Utilisation



3. Accédez à l'interface web de SonarQube:

› Vous pouvez maintenant accéder à l'interface web de SonarQube en ouvrant votre navigateur et en visitant <http://<ip-vm>:9000>. Vous devrez vous connecter avec les informations d'identification par défaut (admin/admin).



› Vous pouvez changer le mot de passe par sonar par exemple : (admin/sonar).

SonarQube - Utilisation



ATTENTION :

sonarqube

» Si vous arrêtez le conteneur SonarQube (en utilisant CTRL+C ou en arrêtant la VM), évitez de lancer à nouveau docker run sur l'image sonarqube, car cela créerait un nouveau conteneur. Au lieu de cela, utilisez simplement la commande suivante pour redémarrer le conteneur existant : docker start <NAME_CONTAINER ou ID_CONTAINER>

```
vagrant@vagrant:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
80c9d982d123      sonarqube          "/opt/sonarqube/dock..."   11 hours ago       Up 11 hours        0.0.0.0:9000->9000/
tcp, :::9000->9000/tcp, 0.0.0.0:9092->9092/tcp, :::9092->9092/tcp
c15bd5e47fc1      mysql              "docker-entrypoint.s..."  9 days ago        Exited (255) 11 hours ago   3306/tcp, 33060/tcp
mysql-db
c0b0f39329c7      bensalahons/alpine:1.0.0    "/bin/sh -c \"java\""  9 days ago        Exited (1) 9 days ago    objective_galileo
85c915ed2dc8      centos             "sleep 20"            9 days ago        Exited (0) 9 days ago    amazing_hypatiae9be20dalec3  mysql
"doc
ker-entrypoint.s..."  2 weeks ago       Exited (1) 2 weeks ago   niftv bora
vagrant@vagrant:~$ docker start sonarqube
sonarqube
vagrant@vagrant:~$
```

SonarQube - Utilisation avec Jenkins



- › Pour mettre en place l'analyse d'un projet Git avec SONARQUBE en utilisant Jenkins, suivez les étapes suivantes :



- oAccédez à Jenkins via l'URL <http://<ip-vm>:8080>

sonarqube

- oDans le pipeline Jenkins existant, ajoutez un nouveau stage au script Groovy pour récupérer le code de votre projet depuis GitHub (assurez-vous de spécifier l'URL de votre projet)

```
stage ('GIT') {
    steps {
        echo "Getting Project from Git";
        .....
    }
}
```



SonarQube - Utilisation avec Jenkins



oExécutez les commandes Maven **clean** et **compile** pour nettoyer et compiler le code du projet que vous avez récupéré depuis Git (utilisez la commande sh '' correspondante).



```
stage('MVN CLEAN') {  
    steps {  
        sh '.....'  
    }  
}  
  
stage('MVN COMPILE') {  
    steps {  
        .....  
    }  
}
```

sonarqube



SonarQube - Utilisation avec Jenkins



o Lancez la commande Maven **sonar:sonar** pour analyser la qualité du code avec SonarQube et envoyez le rapport au serveur SonarQube (assurez-vous de fournir le mot de passe de votre instance SonarQube).



sonarqube

```
stage('MVN SONARQUBE') {  
    steps {  
        .....  
    }  
}
```

o Lancez le Job via **Jenkins**.



SonarQube - Utilisation avec Jenkins



Tableau de bord > Timesheet-DevOps >

Configurer

Supprimer Pipeline

Full Stage View

GitHub

SonarQube

Open Blue Ocean

Renommer

Pipeline Syntax

Stage View

Average stage times:
(Average full run time: ~2min 17s)

#16
oct. 10
03:51
No Changes

#15
oct. 10
03:36
No Changes



SonarQube - Analyse des résultats

› Accédez à http://<adresse_ip_vm>:9000 et consultez les résultats de l'analyse de votre projet.



- › Sur l'interface, cliquez simplement sur le projet "timesheet-devops" pour accéder aux détails complets de cette analyse. SonarQube détecte automatiquement le **langage utilisé (Java + XML dans notre cas)**. SonarQube vérifie si le code contient du **code dupliqué** (ce qui peut être source d'erreurs). Dans notre cas, le pourcentage de duplications est de 0%.



SonarQube - Analyse des résultats

- › **Bug:** Undéfaut dans le code susceptible de provoquer un comportement indésirable de l'application, tel qu'une exception de pointeur nul, dans notre situation :

The screenshot shows a Java code snippet within a SonarQube analysis interface. The code is as follows:

```
@Override  
public User retrieveUser(String id) {  
    l.info("in retrieveUser id = " + id);  
    //User u = userRepository.findById(Long.parseLong(id)).orElse(null);  
    //int i = 1/0;  
    User u = userRepository.findById(Long.parseLong(id)).get();  
  
    Call "Optional#isPresent()" before accessing the value. Why is this an issue?  
    Bug ▾ Major ▾ Open ▾ Not assigned ▾ 10min effort Comment  
  
    l.info("user returned : " + u);  
    return u;  
}
```

A red vertical bar highlights the line `User u = userRepository.findById(Long.parseLong(id)).get();`. A pink callout box above this line contains the text: "Call 'Optional#isPresent()' before accessing the value. Why is this an issue?". Below the code, there is a status bar with icons for Bug (highlighted), Major, Open, Not assigned, 10min effort, and Comment.

- › **Vulnerability :** Une faiblesse de sécurité dans notre code. **Hotspots Reviewed (Zones à vérifier)** : Parties du code qui nécessitent un examen pour s'assurer qu'elles ne présentent pas de failles de sécurité.

SonarQube - Analyse des résultats



- › SonarQube indique si le code a été soumis à des outils de test, tels que JUnit.



- › SonarQube lui-même n'effectue pas l'analyse, mais se repose sur d'autres outils, tels que JaCoCo.
C'est pourquoi notre projet affiche un taux de couverture (**Coverage**) de 0 %, car JaCoCo n'a pas été intégré.



SonarQube - Analyse des résultats



- › **Code Smells (Mauvaises pratiques de codage)** : Ce n'est pas une anomalie du code (bug), mais plutôt du code qui peut compliquer la tâche de l'équipe de développement ou de support, par exemple en raison de commentaires excessifs ou des imports inutilisés.

The screenshot shows the SonarQube interface displaying two code smell issues:

- This block of commented-out lines of code should be removed. Why is this an issue?**
 -
 - Code Smell ▾
 - Major ▾
 - Open ▾
 - Not assigned ▾
 - 5min effort
 - Comment
- Remove this unused import 'javax.transaction.Transactional'. Why is this an issue?**
 -
 - Code Smell ▾
 - Minor ▾
 - Open ▾
 - Not assigned ▾
 - 2min effort
 - Comment

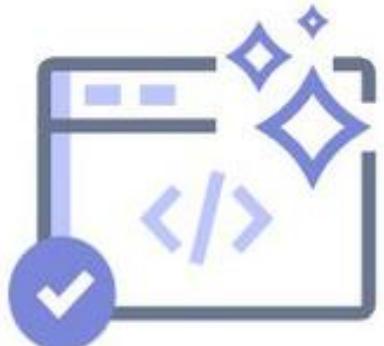


SonarQube - Analyse des résultats



Quality Gate **Passed**

New Code Overall Code

 Enjoy your sparkling clean code!

Category	Count	Grade
Reliability	1 Bugs	C
Security	2 Vulnerabilities	D
Coverage	0.0% Coverage Coverage on 168 Lines to cover	O
Maintainability	30 Code Smells	A
Security Review	0 Security Hotspots ?	A
Duplications	0.0% Duplications Duplications on 547 Lines	O





"Apprendre par le projet, c'est découvrir par l'action, créer par la compréhension, et réussir par la persévérance."



ESPRIT – UP ASI (Architecture des Systèmes d'Information)
Bureau E204

