Rapport TP : Qualité, Sécurité et Maintenabilité du Code avec SonarQube

# Étape 1 : Identification manuelle des problèmes

Fichiers analysés :  
- app.js  
- auth.js  
- utils.js  
- userController.js

|  |  |  |
| --- | --- | --- |
| Fichier | Type de problème | Description |
| userController.js | Bug | `if (password = user.password)` : erreur d’affectation au lieu de comparaison |
| userController.js | Sécurité | Mot de passe en clair dans le code |
| userController.js | Sécurité | Aucune validation/sanitation des entrées utilisateur |
| userController.js | Faille XSS | Injection HTML possible via username dans welcomeMessage |
| utils.js | Code smell | Fonction `sanitizeInputAgain` est redondante et inutile |

# Étape 2 : Correction du code

Corrections apportées :  
- Remplacement de `=` par `===` dans la condition de mot de passe.  
- Suppression de `sanitizeInputAgain` (duplicat inutile).  
- Sanitation de l’input `username` avec `sanitizeInput`.  
- Ajout de validation simple des champs `username` et `password`.  
- Mot de passe de l’utilisateur maintenant hashé avec bcrypt.  
  
Installation des packages nécessaires :  
- bcrypt : `npm install bcrypt`

# Étape 3 : Mise en place de SonarQube

Installation :  
- Java 11 : `sudo apt install default-jdk`  
- Téléchargement de SonarQube depuis le site officiel  
- Lancement via `./bin/linux-x86-64/sonar.sh start`  
  
Analyse :  
- Création du projet `legacy-app` sur l’interface SonarQube  
- Fichier `sonar-project.properties` ajouté  
- Lancement via `sonar-scanner`  
  
Fichier de configuration :  
sonar.projectKey=legacy-app  
sonar.projectName=Legacy App  
sonar.projectVersion=1.0  
sonar.sources=.  
sonar.language=js  
sonar.sourceEncoding=UTF-8

# Étape 4 (Bonus) : Intégration Git + CI

Intégration avec GitHub Actions :  
- Repo lié à SonarCloud  
- Création du fichier ".github/workflows/sonar.yml"