Qualité Logiciel

Mini-Projet : Analyse de la Qualité du Code et Validation Fonctionnelle d'une Application Logicielle

Contexte et Objectifs :

Dans le cadre de ce module, vous êtes chargé d'évaluer et d'améliorer la qualité d'un projet logiciel développé lors d'un précédent cycle. Le projet comporte une analyse approfondie du code source, des tests logiciels (boîte blanche, boîte noire, et performance), ainsi que l'intégration d'outils et de méthodologies modernes de qualité logicielle. L'objectif est de détecter, analyser, et corriger les erreurs, tout en mettant en place des solutions d'amélioration continue basées sur les principes DevOps.

Livrables attendus :

A. Contexte du projet :

1. Présenter le projet logiciel réalisé l'année dernière, en détaillant son objectif, ses fonctionnalités principales, et les technologies utilisées.
2. Recenser les erreurs logicielles rencontrées lors de son développement ou de son utilisation.
3. Décrire les méthodes employées pour détecter ces erreurs (tests manuels, automatiques, revues de code, etc.).
4. Classer ces erreurs en fonction de leur origine (erreurs de développement, de conception, ou de configuration).

B. Analyse de la qualité avec SonarQube :

1. Installer ou configurer un serveur SonarQube localement ou en ligne.
2. Importer le code source du projet précédent dans SonarQube et analyser sa qualité à l'aide de SonarScanner.
3. Examiner les métriques générées par SonarQube, telles que les bugs, vulnérabilités, duplication de code, complexité cyclomatique et couverture de code.
4. Identifier les points faibles et proposer des améliorations pour résoudre les problèmes détectés.

C. Tests Boîte Blanche :

1. Développer des tests unitaires ciblant les zones critiques du code identifiées par SonarQube.
2. Réévaluer la couverture de code avec SonarQube après chaque itération des tests.
3. Documenter l'efficacité des tests en mettant en avant les erreurs détectées et corrigées.

D. Tests Boîte Noire :

1. Utiliser Selenium pour créer et exécuter des scénarios de tests fonctionnels simulant les comportements réels des utilisateurs.
2. Tester les principales fonctionnalités de l'application en suivant différents cas d'utilisation.

E. Tests de Performance :

1. Configurer un outil de test de performance tel que JMeter, Gatling, ou Locust.
2. Réaliser un **test de stress** en simulant 500 utilisateurs simultanés pour évaluer les limites de l'application.
3. Effectuer un **test de charge**, en doublant puis en triplant le nombre maximal d'utilisateurs pour analyser les variations de performance.

F. Intégration DevOps et Automatisation des Tests (Optionnel) :

1. Mettre en place un pipeline d'intégration continue (Jenkins, GitLab CI/CD, ou GitHub Actions) pour automatiser l'exécution des tests unitaires et fonctionnels.
2. Configurer des alertes pour signaler les échecs de tests et maintenir une surveillance proactive de la qualité logicielle.
3. Automatiser le déploiement continu des versions validées dans des environnements de test ou de production.

G. Livrables :

1. **Rapport écrit détaillé :** Un document expliquant les étapes, outils, résultats, et recommandations. Ce rapport devra inclure des tutoriels pour l'installation et l'utilisation des outils, ainsi que des exemples de scripts de tests.
2. **Présentation orale :** Préparer une présentation PowerPoint synthétisant vos travaux, résultats, et conclusions.
3. **Dépôt des livrables :** Remettre le rapport et la présentation via la plateforme prévue, ainsi qu'une version papier à l'enseignant le jour de l'évaluation.