1 Etude qualité de code

Application

1. Qualité sonar way:
Sonar way est un profil de règles de qualité de code par défaut dans SonarQube. Il est conçu pour être pragmatique et non restrictif.

2. Lisibilité du code:
Code élégant et facile à lire.
Les classes et les méthodes doivent être relativement petites (classes < 300 lignes et méthodes < 20).
Les conditions doivent être compréhensibles.
Homogénéité du code : règles de nommage, découpage en package …
Pas de code mort ni de code commenté.
Noms appropriés (doit révéler l'intention, ne pas les tronquer ou le abréger)

3. Réutilisation:
Code factorisé. Code dupliqué à éviter.
Utilisation appropriée de l'héritage et de la composition

4. Configuration:
Les variables susceptibles de changer d'un environnement à l'autre doivent être externalisées et variabilisées par environnement.

5. Cohésion et couplage :
Le code doit être organisé de manière à favoriser la cohésion, c'est-à-dire que les éléments d'une même classe ou module doivent être étroitement liés et collaborer ensemble. Le couplage, quant à lui, doit être maintenu à un niveau minimal, évitant une dépendance excessive entre les différentes parties du code.

6. Gestion des erreurs :
Le code doit être robuste et gérer correctement les erreurs. Les exceptions doivent être capturées et traitées de manière appropriée, en évitant les blocs "catch-all" qui masquent les erreurs et en fournissant des messages d'erreur significatifs.

7. Performances :
Le code doit être optimisé pour des performances efficaces. Cela peut inclure l'utilisation de structures de données appropriées, l'évitement de boucles ou d'opérations coûteuses inutiles, et la minimisation des appels externes coûteux.

8. Sécurité :
Le code doit être conçu avec des bonnes pratiques de sécurité à l'esprit. Cela comprend la validation et l'échappement des données d'entrée, la protection contre les attaques telles

que l'injection SQL ou les attaques de script entre autres, et l'utilisation de mécanismes de gestion des droits d'accès appropriés.

9. Tests :
Le code doit être accompagné de tests unitaires appropriés pour vérifier son bon fonctionnement. Les tests doivent couvrir différents scénarios et cas d'utilisation, et être exécutés de manière régulière pour détecter les éventuelles régressions.

10. Documentation :
Le code doit être correctement documenté. Cela inclut des commentaires clairs et pertinents pour faciliter la compréhension du code, ainsi qu'une documentation supplémentaire pour expliquer les choix de conception, les dépendances externes et tout autre élément pertinent.

Site

Lisibilité du code :
Le code HTML est structuré avec des balises appropriées, et les classes sont utilisées pour appliquer des styles CSS. Cependant, la mise en page de la structure du code pourrait être améliorée avec une indentation plus cohérente et des commentaires pour faciliter la compréhension.

Gestion des erreurs :
Il n'y a pas de code spécifique pour gérer les erreurs. Il est important d'ajouter une gestion adéquate des erreurs pour assurer la robustesse de l'application.

Performances :
Sans une analyse plus approfondie du code, il est difficile de juger de son optimisation pour les performances.

Sécurité :
Il n'y a pas suffisamment d'informations dans le code fourni pour évaluer la sécurité de l'application. Cependant, il est important de prendre en compte les bonnes pratiques de sécurité lors de la conception et du développement d'une application web.

Tests :
Aucun test unitaire n'est présent dans le code fourni. Il est recommandé d'accompagner le code avec des tests unitaires appropriés pour vérifier son bon fonctionnement.

Documentation :
Il n'y a pas de commentaires clairs et pertinents dans le code fourni pour faciliter la compréhension. Une documentation supplémentaire pourrait également être ajoutée pour expliquer les choix de conception et les dépendances externes.