Université Bretagne Sud
IUT de Vannes
Département Informatique

BUT 3
R6.A.06. Maintenance applicative

Mini-projet de maintenance applicative

Cloner le projet se trouvant sur le dépôt Git suivant :
https://github.com/ShellOnYou/SOY_Monolith
Suivre les instructions dans le README du dépôt Github pour démarrer l'application.

Semaines 1 et 2 :

1. Construire un diagramme de classes UML de la base de données et du code + un diagramme de séquence pour un scénario de requête HTTP ;
2. Identifier un cas d'erreur non traité (dans le scénario précédent, ou un autre), le reporter dans un rapport de bug, selon la structure-type vue en cours. Utiliser un outil (Jira ou Github/Gitlab Issues) pour le documenter ;
3. Le cas d'erreur doit être corrigé par votre binôme, qui doit tester sa correction et la documenter. Utiliser par exemple une branche Git, pull/merge requests ;
4. Dans le code source, identifier 3 défauts de types différents :
    - Code dupliqué
    - Mauvaise lisibilité (noms de variables peu explicites, méthodes trop longues)
    - Complexité cyclomatique élevée
    - ...
5. Utiliser SonarQube pour analyser le projet et reporter 3 défauts (issues et security hotspots) différents de ceux de la question précédente. Est-ce que l'outil a identifié les premiers défauts ? Si non, expliquer pourquoi.
   Pour lancer SonarQube, utiliser le ficher docker-compose.yml fourni sur Moodle. Modifier ce fichier pour indiquer le chemin du dossier où se trouve le code source sur votre machine.
   Démarrer SonarQube avec la commande : docker compose up -d
   Cette orchestration démarre 3 services : SonarQube, sa base de données Postgres et Sonar Scanner CLI.
   Ouvrez sur votre navigateur la page : http://localhost:9000
   Les credentials par défaut sont : admin et admin
   Créer un projet avec le nom SoYMono et générer un token à utiliser dans la commande ci-dessous.
   Sur un terminal, lancer la commande suivante (en remplaçant le token par le votre):
   docker exec -it sonar_scanner sonar-scanner -Dsonar.projectKey= SoYMono -Dsonar.sources=. -Dsonar.host.url=http://localhost:9000 -Dsonar.token=sqp_806254e33f55f6b547587a8e998155599fb8e9b4
6. Reporter dans votre rapport la dette technique relevée par SonarQube. Expliquer quelques mesures ;
7. Prioriser 3 défauts à corriger et expliquer pourquoi ;

8. Renseigner des tickets pour corriger ces défauts. L'un des étudiants du binôme, qui a renseigné le ticket de bug de la question 2 en documentera un et l'autre étudiant en fera deux;
9. Proposer des solutions aux 3 défauts de types différents relevés par SonarQube ;
10. Implémenter l'une des solutions et mesurer le temps (en vous appuyant sur des outils comme Github -durée entre ouverture du ticket et sa fermeture- ou Jira. Est-ce que la mesure de la dette technique correspond ? Dire pourquoi ;

Semaine 3 :

11. Est-ce qu'un LLM peut aider dans la maintenance d'un projet d'application ? Expliquer comment ;
12. Proposer une chaîne de CI permettant de garantir que l'application soit dans un état de maintenance propre ;
13. Comparer les résultats de qualité SonarQube de l'application SoyMonolith avec ceux de la même application ayant une architecture à microservices : https://github.com/ShellOnYou/SOY_MS_HTTP

Livrables attendus à la fin : chaque fin de semaine, rendre l'état actuel des livrables

- Rapport détaillant :
  - Le rapport de bug, sa correction et les résultats des tests
  - Les quelques défauts détectés manuellement et par SonarQube
  - La dette technique
  - Les priorités et le plan d'action proposé
  - Le processus de maintenance basé sur un LLM
  - La documentation de la chaîne de CI
- Diagrammes UML
- Code source modifié avec une explication des corrections apportées.
- Rapport généré par l'outil d'analyse (exemple : fichier PDF de SonarQube).
- Documentation mise à jour (README et commentaires).