2025

Projet QCM

PROJET DE L'APPLICATION QCM EN PYTHON

3RD CYBER SECURITY

Projet de la mini application S.Y.S QCM

Ce projet a était réalisé par les élèves de 3eme année ingénieures en cyber Security :

Membre 1 : Aibeche Yasmine

Membre 2 : Sadaoui Sara Rahma

Membre 3 : Toutou salsabila

Université des sciences et de la technologie de houari Boumediene

USTHB

Faculté d'informatique

Date :

Janvier 2025

Tâches possibles pour l'utilisateur

Pour notre application de QCM l’utilisateur va d’abords devoir se connecter ou créer un compte en entrant son nom d’utilisateur, on lui affichera son id et lui offrira le menu pour l’application qui lui permettra d’effectuer l’une ou plusieurs taches disponible :

- **Démarrer un QCM :** L'utilisateur peut choisir parmi différentes catégories de questions pour tester ses connaissances. Chaque session de QCM est chronométrée et les résultats sont affichés à la fin.

- **Afficher l'historique :** L'utilisateur peut consulter son historique de QCM, y compris les scores obtenus dans chaque session précédente.

- **Changer d'utilisateur :** Si plusieurs utilisateurs utilisent la même application, ils peuvent changer de profil pour enregistrer leurs propres résultats et historique.

- **Initialiser l'historique :** L'utilisateur a la possibilité de réinitialiser son historique, supprimant ainsi toutes les données sauvegardées précédemment.

- **Quitter l'application :** L'utilisateur peut quitter l'application à tout moment via l'option de menu appropriée.

Ce résumé des fonctionnalités utilisateur peut être détaillé davantage dans la section du rapport consacrée à ce que l'utilisateur peut faire.

Difficultés rencontrées :

Problèmes de persistance des données :

Problème : Les modifications de l'historique utilisateur n'étaient pas correctement enregistrées dans le fichier users.json.

Solution : Le fichier JSON a été ouvert en mode écriture après chaque mise à jour, avec un appel à seek(0) pour réinitialiser le pointeur et truncate () pour écraser correctement le fichier.

Gestion des fichiers manquants ou corrompus :

Problème : Le programme plantait si users.json ou le fichier des questions était manquant ou corrompu.

Solution : Utilisation de blocs try-except pour gérer les erreurs FileNotFoundError et JSONDecodeError, avec des structures de données par défaut pour initialiser le programme en cas d'absence de fichiers.

Logique de changement d'utilisateur :

Problème : Des incohérences apparaissaient lors du changement d'utilisateur, comme des mises à jour d'historique appliquées au mauvais utilisateur.

Solution : Utilisation de la variable user_id comme identifiant principal pour chaque utilisateur, avec un débogage minutieux pour assurer la cohérence.

Complexité de la modularisation :

Problème : La séparation du programme en modules multiples a entraîné des problèmes de dépendances, notamment avec les données partagées comme la liste des utilisateurs.

Solution : Standardisation du flux de données entre les modules en passant explicitement les arguments nécessaires aux fonctions, réduisant ainsi les erreurs et améliorant la maintenabilité.

Résultats et réflexions :

Le projet a atteint ses objectifs en créant une application de quiz fonctionnelle et engageante. Les utilisateurs peuvent interagir facilement avec le programme, et la conception modulaire permet d'intégrer de futures améliorations avec peu d'effort. Les défis rencontrés durant le développement ont offert des leçons précieuses en débogage, modularisation et conception de l'expérience utilisateur.