PolyQuiz

Plan de tests logiciels

Version 1.0

Historique des révisions

Date	Version	Description	Auteur
2024-11-26	1.0	- Identification des stratégies de tests (fonctionnels, non-fonctionnels, interface, sécurité, performance). - Détail des techniques de test (tests manuels, outils comme ThunderClient, Chrome DevTools). - Élaboration de la liste des exigences fonctionnelles et non-fonctionnelles à tester, avec leur association aux types de tests.	Aymen Ghodbane
2024-11-28	2.0	Ajout de l'introduction, les ressources et les jalons	Aymen Ghodbane

Plan de tests logiciels

1. Introduction

Ce document décrit le Plan de tests logiciels du projet PolyQuiz, visant à garantir la qualité, la fiabilité et la conformité du système aux exigences définies. Il couvre les fonctionnalités à tester, les stratégies de validation, les types de tests (fonctionnels, performance, sécurité, etc.), les outils et ressources nécessaires, ainsi que les jalons du projet. Organisé pour assurer une traçabilité des exigences et une coordination efficace des activités, ce plan encadre chaque étape du processus, depuis la définition des tests jusqu'à leur exécution et validation finale. Il servira de référence évolutive pour s'assurer que le système répond aux attentes des utilisateurs tout en minimisant les anomalies.

2. Exigences à tester

Type de test	Couverture	Exigences associés
Tests de fonction	Valider les fonctionnalités principales comme le clavardage, la gestion des comptes et les parties.	**Fonctionnelles:** 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 3.10, 3.11, 3.14, 3.15
Tests d'interface	Vérifier la navigation intuitive et l'affichage des interfaces utilisateur dans l'application.	**Non-fonctionnelles**: 4.1
Tests d'intégrité de données	Garantir la cohérence des données dans la base (comptes, parties, clavardage).	**Fonctionnelles:** 3.2, 3.3, 3.5
Tests de performance	Mesurer les temps de réponse des fonctionnalités critiques.	**Non-fonctionnelles** : 4.3
Tests de sécurité	Vérifier l'authentification, le contrôle d'accès et la gestion sécurisée des données sensibles.	**Non-fonctionnelles** : 4.2, 4.6

3. Stratégie de test

3.1. Types de test

3.1.1. Tests de fonction

Objectif de test:	Vérifier que les fonctionnalités implémentées respectent les exigences fonctionnelles définies dans le SRS
Technique:	Identification des entrées attendues et vérification des sorties.
Critère de complétion:	Tous les cas d'utilisation doivent être validés avec succès. Couverture fonctionnelle complète de toutes les exigences critiques.
Considérations spéciales:	Inclure les cas aux limites pour les entrées utilisateur (valeurs minimales/maximales, données invalides).

3.1.2. Tests d’interface usager

Objectif de test:	Valider que l'interface utilisateur est intuitive, répond aux attentes de l'utilisateur, et donne accès à toutes les fonctionnalités prévues. Vérifier que les interactions avec l'interface sont fluides, cohérentes, et accessibles.
Technique:	Tests manuels avec des scénarios utilisateur définis. Tests d’utilisabilité pour identifier des problèmes d’ergonomie.
Critère de complétion:	L'utilisateur doit pouvoir accéder à toutes les fonctionnalités principales sans difficulté.

	Navigation sans erreur sur toutes les pages.
Considérations spéciales:	Vérifier que la mise en page reste lisible et fonctionnelle dans les deux langues, notamment pour les contenus longs ou les textes traduits.

3.1.3. Tests d’intégrité des données

Objectif de test:	Vérifier l'intégrité, la cohérence, et la fiabilité des données stockées dans la base de données. Assurer que les règles de validation et de contrainte sont correctement appliquées
Technique:	Envoi de requêtes API (POST, PUT, DELETE) via ThunderClient avec des données valides, invalides ou aux limites pour vérifier les réponses du backend.
Critère de complétion:	Toutes les règles de validation (ex. : formats, valeurs minimales/maximales) sont respectées avant envoi à la base de données. Les erreurs retournées par le backend sont cohérentes et explicites
Considérations spéciales:	

3.1.4. Tests de performance

Objectif de test:	Valider que les fonctionnalités essentielles, telles que le chargement de l'historique des messages, des avatars et des listes d'amis, s'exécutent dans un délai raisonnable pour garantir une expérience utilisateur fluide. Vérifier que les temps de réponse respectent les exigences de performance définies.
Technique:	Utilisation d'outils de mesure des temps de réponse comme ThunderClient ou Chrome DevTools pour évaluer les performances côté client et backend. Analyse des temps de réponse API pour des requêtes spécifiques (ex. : chargement d'une liste d'amis ou d'un historique de messages).
Critère de complétion:	Les éléments critiques (ex. : listes, données utilisateurs, images) doivent se charger dans des délais raisonnables, adaptés à l'expérience utilisateur.
Considérations spéciales:	Tester dans des environnements réseau variés, incluant des connexions lentes ou instables, pour garantir une performance acceptable.

3.1.5. Tests de sécurité et de contrôle d'accès

Objectif de test:	Vérifier que les restrictions d'accès sont correctement appliquées, comme rendre un quiz invisible ou verrouiller une partie, afin que seuls les utilisateurs autorisés puissent accéder aux ressources.
Technique:	Simulation de scénarios utilisateur avec des rôles et droits différents (ex. : créateur d'un quiz, organisateur d'une partie, joueur).
Critère de complétion:	Les ressources restreintes par un utilisateur (ex. : rendre un quiz invisible ou verrouiller l'accès à une partie) ne doivent plus être accessibles par d'autres utilisateurs non autorisés.

Considérations spéciales:	Tester les actions concurrentes, par exemple : tenter de créer une partie avec un quiz que le créateur a rendu invisible entre-temps, et s'assurer que l'action échoue avec un message d'erreur clair.

3.2. Outils

Les outils suivants seront utilisés au sein de la discipline de test:

Type de test	Outil
Tests de fonction	Tablette, PC windows 10 +
Tests d'interface usager	Tablette, PC windows 10 +, utilisateur externe
Tests d'intégrité des données	ThunderClient
Tests de performance	ThunderClient
Tests de sécurité et de contrôle d'accès	Tablette, PC windows 10 +

4. Ressources

4.1. Équipe de test

Rôle	Membre de l'équipe	Responsabilités
Concepteur des tests pour le client lourd	Damaris	Concevoir les scénarios et cas de test pour le client lourd, en tenant compte des fonctionnalités clés.
Exécutant des tests pour le client lourd	Motassem	Exécuter les tests définis pour le client lourd, documenter les résultats et signaler les anomalies.
Concepteur des tests pour le client léger	Lina	Élaborer les cas de test pour le client léger, y compris les scénarios d'interface et d'expérience utilisateur.
Exécutant des tests pour le client léger	Manel	Réaliser les tests sur le client léger, valider les fonctionnalités, et rapporter les erreurs détectées.
Responsables des tests non fonctionnels	Aymen, Simon	Concevoir et exécuter des tests non fonctionnels (performance, contrôle d'accès), et analyser les résultats.
Utilisateur externe	Frère de Simon	Tester l'application dans des conditions réelles en jouant le rôle d'un utilisateur final. Il apporte un retour objectif sur l'ergonomie, la facilité d'utilisation, et identifie les éventuelles lacunes qui n'auraient pas été détectées en interne.

4.2. Système

Les tests nécessitent un PC sous Windows 10+ et une tablette Android 13.0. Ces appareils couvriront les scénarios sur desktop et mobile. Le backend est déployé sur AWS, tandis que les données sont stockées sur MongoDB Atlas dans un environnement isolé dédié aux tests. Des comptes de test seront créés sur Auth0.

5. Jalons du projet

Jalon	Effort	Date de début	Date de fin
Définition des exigences de test	2	28 septembre	5 octobre
Mise en place de l'environnement de test	3	5 octobre	15 octobre
Développement des cas de test	5	16 octobre	15 novembre
Exécution des tests initiaux	3	20 octobre	22 novembre
Fixer les anomalies	5	20 octobre	25 novembre
Finalisation et validation finale	2	26 novembre	29 novembre