**Projet 3**

Plan de tests logiciels

Version 2.0

Historique des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Auteur** |
| 2021-04-12 | 1.0 | Rédaction initiale du plan de tests logiciels | Équipe 107 |
| 2021-04-18 | 2.0 | Modifications finales pour la remise du produit final | Équipe 107 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table des matières

[1. Introduction 1](#_Toc69755659)

[2. Exigences à tester 1](#_Toc69755660)

[3. Stratégie de test 1](#_Toc69755661)

[3.1. Types de test 2](#_Toc69755662)

[3.1.1. Tests de fonction 2](#_Toc69755663)

[3.1.2. Tests d’interface usager 2](#_Toc69755664)

[3.1.3. Tests d’intégrité des données 2](#_Toc69755665)

[3.1.4. Tests de performance 2](#_Toc69755666)

[3.1.5. Tests de charge 3](#_Toc69755667)

[3.1.6. Tests de stress 3](#_Toc69755668)

[3.1.7. Tests de volume 3](#_Toc69755669)

[3.1.9. Tests d’échec/récupération 3](#_Toc69755670)

[3.2. Outils 4](#_Toc69755671)

[4. Ressources 4](#_Toc69755672)

[4.1. Équipe de test 4](#_Toc69755673)

[4.2. Système 5](#_Toc69755674)

[5. Jalons du projet 5](#_Toc69755675)

**Plan de tests logiciels**

# 

# 1. Introduction

Le but du plan de tests est de définir différents tests pour assurer le bon fonctionnement de notre application. Dans ce document, nous allons décrire les exigences à tester, expliquer notre stratégie de test en décrivant les types de tests et les outils utilisés, parler des ressources et finalement conclure avec les jalons du projet.

# 2. Exigences à tester

|  |  |
| --- | --- |
| Exigences | Tests |
| Clavardage - Intégration | Tests de fonction, tests d’interface-usager |
| Clavardage - Canaux de discussion | Tests de fonction |
| Profil utilisateur et historique | Tests de fonction, test d'interface-usager, tests d’intégrité de données, tests de charge |
| Modes de jeu | Tests de fonction, tests d’interface-usager |
| Création d’une paire mot-image | Tests de fonction, tests d’interface-usage, tests de volume, test d’intégrité de données |
| Personnalité des joueurs virtuels | Tests de fonction |
| Effets visuels et sonores | Tests de fonction |
| Tutoriel | Tests de fonction, tests d’interface-usager |
| Langues | Tests de fonction, tests d’interface-usager |
| Thèmes de couleurs différents | Tests de fonction, tests d’interface-usager |
| Classement | Tests d’interface-usager |
| Système de points et niveaux | Tests de fonction |
| Album de dessins | Tests de fonction, tests d’interface-usager, tests de charge |
| Système de notifications | Tests de fonction, tests d’interface-usager |
| Expérience utilisateur | Tests d'interface usager, tests d'échec/récupération, tests de stress, tests de performance |

# 3. Stratégie de test

## 3.1. Types de test

### 3.1.1. Tests de fonction

|  |  |
| --- | --- |
| Objectif de test: | S’assurer du bon fonctionnement d’une fonctionnalité et que son résultat respecte les attentes du cas d’utilisation. |
| Technique: | Exécuter la fonctionnalité et comparer les résultats avec les requis du SRS et des cas d’utilisation. |
| Critère de complétion: | Les résultats obtenus sont conformes au SRS. |
| Considérations spéciales: |  |

### 3.1.2. Tests d’interface usager

|  |  |
| --- | --- |
| Objectif de test: | Vérifier si l’utilisateur a bien accès à toutes les différentes interfaces disponibles et que la navigation se fait de manière intuitive et sans erreurs.  Vérifier le bon fonctionnement des différents boutons, modales et entrée de données. |
| Technique: | Le testeur doit vérifier qu’il peut naviguer sur toutes les différentes interfaces qui lui sont disponibles et tester les différents boutons. Il doit également entrer des données dans les champs de texte pour vérifier la validation de ces dernières. |
| Critère de complétion: | La navigation est conforme aux attentes du SRS, les différentes interfaces sont disponibles en fonction des accès attendus de l’utilisateur, les boutons sont fonctionnels et les champs de texte n’acceptent pas les données invalides. |
| Considérations spéciales: |  |

### 3.1.3. Tests d’intégrité des données

|  |  |
| --- | --- |
| Objectif de test: | Vérifier qu’une vérification est effectuée avant l’insertion dans la base de données, empêchant ainsi l’insertion de données invalides. |
| Technique: | Envoyer des requêtes HTTP au serveur contenant des données invalides et des données valides. |
| Critère de complétion: | Seules les données valides envoyées se retrouvent dans le serveur à la suite des différentes requêtes. |
| Considérations spéciales: |  |

### 3.1.4. Tests de performance

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que l’application peut supporter la connexion de plusieurs utilisateurs en même temps, ainsi que plusieurs parties se déroulant en même temps. |
| Technique: | Tous les membres de l’équipe se connectent avec plusieurs utilisateurs et commencent différentes parties pour tester la performance de l’application. |
| Critère de complétion: | Le logiciel n’a pas de délai excessif et l’expérience utilisateur n’est pas affectée lorsqu’il y a beaucoup de trafic sur le serveur. |
| Considérations spéciales: |  |

### 3.1.5. Tests de charge

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que l’application est capable de charger les dessins de l’album de dessins d’un utilisateur sans délai excessif. |
| Technique: | Enregistrer le nombre maximum de dessins pour un utilisateur et accéder à l’album de dessins. |
| Critère de complétion: | Les dessins sont chargés et affichés sans délai trop excessif. |
| Considérations spéciales: |  |

### 3.1.6. Tests de stress

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que le serveur peut supporter plusieurs parties en même temps. |
| Technique: | Démarrer un grand nombre de parties simultanément pour tester la performance de l’application. |
| Critère de complétion: | La performance de l’application n’est pas considérablement affectée malgré les parties simultanées. |
| Considérations spéciales: |  |

### 3.1.7. Tests de volume

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que le serveur peut supporter de grosses quantités de données envoyées et que la performance de l’application n’est pas affectée. |
| Technique: | Sauvegarder de très gros dessins contenant plusieurs traits et points, envoyer plusieurs messages dans différentes salles de clavardages par plusieurs utilisateurs en même temps. |
| Critère de complétion: | Les dessins sont enregistrés correctement sur la base de données et leur affichage est exécuté sans délai excessif. |
| Considérations spéciales: |  |

### 3.1.9. Tests d’échec/récupération

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que les cas de défaillances sont gérés correctement pour ne pas affecter l’expérience utilisateur. |
| Technique: | Faire déconnecter des utilisateurs lors d’une partie. |
| Critère de complétion: | Les autres utilisateurs sont notifiés lorsqu’un autre membre de la partie est soudainement déconnecté et la partie s’arrête sans faire arrêter le serveur. |
| Considérations spéciales: |  |

## 3.2. Outils

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

|  |  |
| --- | --- |
| **Type de test** | **Outil** |
| Tests de fonction | Client électron, émulateur android |
| Tests d'interface usager | Client électron, émulateur android |
| Test d'intégrité des données | Postman, MongoDB |
| Tests de performance | Client électron, émulateur android |
| Tests de charge | Client électron |
| Tests de stress | Client électron, émulateur android |
| Tests de volume | Client électron, émulateur android |
| Tests d'échec/récupération | Client électron, émulateur android |

# 4. Ressources

## 4.1. Équipe de test

|  |  |  |
| --- | --- | --- |
| **Rôle** | **Membre de l’équipe** | **Responsabilités** |
| Rédiger les cas de tests client lourd | **Alice Gong** | Rédaction des cas de tests pour le client lourd |
| Testeur client léger | **Oliver Jean** | Réalisation des tests pour le client léger  Noter le résultat des tests |
| Testeur client lourd et léger | **Charles Jiang** | Réalisation des tests pour le client léger et lourd  Noter le résultat des tests  Noter la cohérence des clients |
| Rédiger les cas de tests client lourd | **Yuhan Li** | Rédaction des cas de tests pour le client lourd |
| Testeur client lourd et serveur | **Nu Chan Nhien Ton** | Réalisation des cas de tests pour le client lourd  Réalisation des cas tests pour le server/base de données |
| Rédiger les cas de tests client léger | **Yanis Toubal** | Rédaction des cas de tests pour le client léger |

## 4.2. Système

L’application PAINseau est construite sur Electron du côté du client lourd et il faut donc rouler l’exécutable sur un ordinateur avec Windows 10. Du côté du client léger, l’application roule sur Android Studio et il faut donc une tablette Android pour l’utiliser. Le serveur a été déployé sur Heroku et ne nécessite donc pas d’exécution au préalable. Il faudra s’assurer de réveiller le serveur avant puisqu’il rentre en mode sommeil après un certain temps d’inactivité.

# 5. Jalons du projet

|  |  |  |  |
| --- | --- | --- | --- |
| **Jalon** | **Effort** | **Date de début** | **Date de fin** |
| Conception des cas de test client léger | 3 | 08-14-2021 | 08-16-2021 |
| Conception des cas de test client lourd | 3 | 08-14-2021 | 08-16-2021 |
| Exécuter la première version des cas de test client léger | 2 | 08-15-2021 | 08-16-2021 |
| Exécuter la première version des cas de test client lourd | 2 | 08-15-2021 | 08-16-2021 |
| Réviser les tests de client léger | 1 | 08-17-2021 | 08-17-2021 |
| Réviser les tests de client lourd | 1 | 08-17-2021 | 08-17-2021 |
| Re-exécuter les tests de client léger | 2 | 08-18-2021 | 08-18-2021 |
| Re-exécuter les tests de client lourd | 2 | 08-18-2021 | 08-18-2021 |