Poly Paint Pro

Plan de tests logiciels

Version 1.0

Historique des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Auteur** |
| 2018-11-28 | 1.0 | Rédaction des exigences à tester, des stratégies de test et des jalons | Ayman |
| 2018-11-28 | 1.1 | Rédaction de l’introduction, des stratégies de test restantes, des outils et des ressources | Pascal |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table des matières

[**1. Introduction**](#_1fob9te) **4**

[**2. Exigences à tester**](#_3znysh7) **4**

[**3. Stratégie de test**](#_2et92p0) **4**

[3.1. Types de test](#_tyjcwt) 4

[3.1.1. Tests de fonction](#_3dy6vkm) 4

[3.1.2. Tests d’interface usager](#_4d34og8) 5

[3.1.3. Tests d’intégrité des données](#_3rdcrjn) 5

[3.1.4. Tests de performance](#_26in1rg) 5

[3.1.5. Tests de stress](#_35nkun2) 5

[3.1.6. Tests de sécurité et de contrôle d’accès](#_3j2qqm3) 6

[3.2. Outils](#_l3wadtgy3rvc) 6

[**4. Ressources**](#_qsh70q) **6**

[4.1. Équipe de test](#_1pxezwc) 6

[4.2. Système](#_49x2ik5) 7

[**5. Jalons du projet**](#_2p2csry) **7**

**Plan de tests logiciels**

# 

# 1. Introduction

Le contenu du présent document sert à expliquer quelles mesures ont été prises afin d’assurer la qualité de notre logiciel. Pour ce faire nous allons présenter ce que nous nous attendons par rapport à la qualité de chaque fonctionnalité que nous avons développé afin de nous assurer de livrer un produit de haute qualité, c’est-à-dire un logiciel absent de bogues qui respecte la liste des exigences de notre SRS et du document pédagogique.

Pour ce faire, nous allons dans un premier temps énumérer les différentes exigences à tester, puis nous allons décrire plus en détail nos stratégies de tests ainsi que les outils utilisés pour les effectuer. Par la suite, nous allons énumérer les différentes ressources utilisées pour effectuer ces tests.

# 2. Exigences à tester

* Clavardage
* Édition des formes
* Édition collaborative
* Sauvegarde d’image et chargement
* Accessibilité des images
* Profil utilisateur et galerie
* Administration (site web)
* Tutoriel
* J’aime et commentaires
* Photo de profil
* Gestion des amis
* Partage d’URL
* Performance
* Sécurité
* Utilisabilité

# 3. Stratégie de test

## 3.1. Types de test

### 3.1.1. Tests de fonction

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que la fonctionnalité fonctionne correctement, qu’elle est cohérente avec son cas d’utilisation et qu’elle respecte le SRS. |
| Technique: | Le testeur doit exécuter la fonctionnalité et comparer les résultats obtenus avec le SRS et le cas d’utilisation |
| Critère de complétion: | Les résultats de la fonctionnalité sont conformes au SRS et au cas d’utilisation |
| Considérations spéciales: | Le test est réalisé en boîte noire. De ce fait, il ne prend pas en considération le code source de la fonctionnalité. |

### 3.1.2. Tests d’interface usager

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que l’utilisateur peut accéder à toutes les interfaces qui lui sont permises et que la navigation se fait correctement.  Vérifier le bon fonctionnement des boutons et les champs de texte. |
| Technique: | Le testeur doit tester l’application avec les différents types d’utilisateurs et naviguer entre les différentes interfaces disponibles. Il doit également tester les boutons et saisir des données valides et invalides dans les champs de texte. |
| Critère de complétion: | La navigation se fait correctement et les permissions des différents utilisateurs sont respectées.  Les champs de textes n’acceptent pas les données invalides. |
| Considérations spéciales: |  |

### 3.1.3. Tests d’intégrité des données

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que la base de données ne peut contenir des données invalides. |
| Technique: | Envoyer des requêtes HTTP contenant des données valides et invalides. |
| Critère de complétion: | La base de données contient seulement les données valides envoyées. |
| Considérations spéciales: |  |

### 3.1.4. Tests de performance

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que les clients peuvent charger les images dans un délai raisonnable. |
| Technique: | Si le chargement d’une page d’un de nos clients prend un temps déraisonnable, le test échoue. |
| Critère de complétion: | Toutes les pages des applications des clients doivent être chargées dans un délai raisonnable. |
| Considérations spéciales: |  |

### 3.1.5. Tests de stress

|  |  |
| --- | --- |
| Objectif de test: | Plusieurs utilisateurs peuvent accéder concurrentiellement aux images et aux clients. |
| Technique: | Plusieurs personnes tentent d’accéder en même temps à une image et y faire des modifications simultanément, sans avoir une perte de performance. |
| Critère de complétion: | Aucun parti ne doit voir une baisse de performance ou avoir une défaillance. |
| Considérations spéciales: |  |

### 3.1.6. Tests de sécurité et de contrôle d’accès

|  |  |
| --- | --- |
| Objectif de test: | Vérifier que l’accès des données est respecté et que les différents types d’utilisateurs respectent les permissions qui leur sont attribuées |
| Technique: | Le testeur doit accéder à l’application en utilisant les différents types utilisateurs et en essayant d’accéder aux différentes données protégées. |
| Critère de complétion: | Chaque type d’utilisateur respecte les permissions qui leur ont été attribuées.  Toutes les données protégées peuvent être accédé si leurs conditions d’accès sont respectées |
| Considérations spéciales: |  |

### 

## 3.2. Outils

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

|  |  |
| --- | --- |
| **Type de test** | **Outil** |
| Tests de fonction | iPad, PC Windows 10 |
| Tests d’interface usager | iPad, PC Windows 10, Utilisateur |
| Tests d’intégrité des données | Postman, DataGrip |
| Tests de performance | Serveur Node |
| Tests de stress | iPad, PC Windows 10 |
| Tests de sécurité et contrôle d’accès | iPad, PC Windows 10 |

# 4. Ressources

## 4.1. Équipe de test

|  |  |  |
| --- | --- | --- |
| **Rôle** | **Membre de l’équipe** | **Responsabilités** |
| Testeur des exigences non fonctionnelles | Pascal | Écriture et exécution des tests non fonctionnels |
| Concepteur de tests client lourd | Ayman | Écriture des tests fonctionnels du client lourd |
| Testeur du client lourd | Sébastien | Exécution des tests fonctionnels du client lourd |
| Concepteur de tests client léger | Audrey | Écriture des tests fonctionnels du client léger |
| Testeur du client léger | Éric | Exécution des tests fonctionnels du client léger |
| Testeur externe | William Li (Frère de Eric) | Des exigences non fonctionnelles et quelques exigences fonctionnelles |

## 4.2. Système

PolyPaintPro est une application écrite en Swift pour son client léger déployée sur iOS et en C# pour son client lourd. Le client léger utilise Xcode, le client lourd, Visual Studio. Afin de permettre la communication entre ces clients, l’application a aussi une base de données. Sa base de données est sur RDS d’AWS et son serveur est déployé sur EC2 à l’aide de docker. Le serveur est développé sur Visual Studio Code. Finalement, son site web est fait en Angular6 et est aussi développé sur Visual Studio Code.

# 5. Jalons du projet

|  |  |  |  |
| --- | --- | --- | --- |
| **Jalon** | **Effort** | **Date de début** | **Date de fin** |
| Définir les cas de test | 3 | 2018-11-22 | 2018-11-23 |
| Implémenter les tests de fonction | 2 | 2018-11-27 | 2018-11-28 |
| Implémenter les tests d’interface utilisateur | 1 | 2018-11-27 | 2018-11-27 |
| Implémenter les tests de performance | 1 | 2018-11-27 | 2018-11-27 |
| Implémenter les tests de stress | 1 | 2018-11-27 | 2018-11-27 |
| Implémenter les tests sécurité et de contrôle d’accès | 1 | 2018-11-27 | 2018-11-27 |
| Exécution des tests | 4 | 2018-11-28 | 2018-11-28 |