|  |
| --- |
| HE-ARC |
| BorrowThings |
| Projet qualité de logiciel |

|  |
| --- |
| Kiliç Ergün, Claude Edwin  24/04/2020 |

Table des matières

[1. Introduction 2](#_Toc38555623)

[2. Concept de test 2](#_Toc38555624)

[2.1 Attentes du projet en termes de qualité de produit 2](#_Toc38555625)

[2.2 Objectifs de test 2](#_Toc38555626)

[2.3 Périmètre de test 2](#_Toc38555627)

[3. Plan de Test 2](#_Toc38555628)

[3.1 Périmètre de test 2](#_Toc38555629)

[3.1.1 Périmètre fonctionnel 2](#_Toc38555630)

[3.1.2 Type de test 2](#_Toc38555631)

[3.1.3 Niveau de test 2](#_Toc38555632)

[3.2 Stratégie détaillée de test 2](#_Toc38555633)

[3.3 Organisation 2](#_Toc38555634)

[3.4 Infrastructure et logistique de test 2](#_Toc38555635)

[3.5 Gestion des risques 2](#_Toc38555636)

[4. Rapport de test 2](#_Toc38555637)

[4.1 Test unitaire 2](#_Toc38555638)

[4.2 Test d’intégration 2](#_Toc38555639)

[4.3 Test de charge 2](#_Toc38555640)

[4.4 Test fonctionnels 2](#_Toc38555641)

[4.5 Analyse SonarCloud 2](#_Toc38555642)

[5. Conclusion 2](#_Toc38555643)

# Introduction

BorrowThings est un site web permettant l’organisation d’emprunt d’objet au sein d’une école. Les utilisateurs (enseignants) peuvent enregistrer les différents emprunts de matériel effectué dans une classe. Un emprunt est effectué par un utilisateur à une date précise et celui-ci devra le restituer avant une date limite sous peine de recevoir une notification de retards.

Ce projet a été effectué durant les cours de Java Entreprise Edition et Qualité du Logiciel. Ce rapport présente la partie réalisée pour le cours de qualité du logiciel. Les chapitres qui suivent vont définir les points qui ont été réaliser afin d’avoir une application fiable et fonctionnelle.

# Concept de test

## Attentes du projet en termes de qualité de produit

* Site pas trop lourd à charger.
* Site conformes aux normes standard du Web
* Les administrateurs sont les seuls à pouvoir modifier un utilisateur
* Les administrateurs sont les seuls à pouvoir accéder à la page d’admin et d’utilisateurs
* Les enseignants et administrateurs sont les seul à pouvoir enregistrer des emprunts
* Interface du site ergonomique
* Interface fonctionnelle sur mobile
* Bonne structure de projet afin de faciliter la maintenance et évolution futur Objectifs de test

## Périmètre de test

Nous avons utilisé trois méthodes de tests et une méthode d’analyse de qualité du code

* Tests de performance avec OctoPerf
* Tests d’intégration avec Katalon
* Tests d’acceptances avec des utilisateurs
* Contrôle qualité de code avec SonarCloud

# Plan de Test

## Périmètre de test

### Périmètre fonctionnel

* Connexion au site
* Inscription au site
* Ajouter et voir un utilisateur (seulement administrateur)
* Ajouter un rôle
* Ajouter, modifier et supprimer une salle
* Ajouter, modifier et supprimer un objet
* Emprunter un objet
* Temps de réponse acceptable sur le site
* Ergonomie du site

### Type de test

* Fonctionnalité
  + Les différents CRUD sont fonctionnelles sauf avec les items où le champs description (textarea) n’est pas pris en compte
  + La gestion des rôles est fonctionnelle
* Fiabilité
  + Resistance à la charge d’utilisateur (10 utilisateurs avec la version gratuite d’Octoperf)
* Ergonomie
  + Site suivant une logique pouvant être compris par n’importe qui
* Efficience
  + Aucune latence sur les accès des pages
* Maintenabilité
  + Le code suit une bonne structure (séparation du code) ce qui permet de facilement maintenir le site
  + Aucun problème détecter dans le code avec SonarCloud
* Portabilité
  + Affiche presque 100% fonctionnelle sur toutes les plateformes cependant gros problème avec l’affichage du menu. Ceci aurait pu être corriger avec plus de temps

### Niveau de test

* Test de performance avec OctoPerf
* Test de validation avec des personnes ne connaissant pas le développement du site
* Test d’intégration avec Katalon

## Stratégie détaillée de test

* Inscription au site
  + Vérifier que le compte qui s’inscrit au site possède bien le rôle étudiant par défaut
* Connexion au site
  + Vérifier qu’un compte peut se connecter si il est actif
  + Message d’erreur si mauvais identifiants
* Pour chaque fonctionnalité CRUD
  + Tester la création
  + Tester la modification
  + Tester la Suppression
* Contrôle des accès aux pages d’utilisateur
  + Seulement accessible par l’administrateur
* Contrôle des accès aux pages d’emprunts
  + Seulement accessible par un enseignant
* Performance du site
  + Lancement d’un test grâces à OctoPerf permettant de simuler une charge de 10 personnes sur le site.
* Ergonomie
  + Demande à des personnes externes d’utilisateur le site
  + Tests d’affichage sur différentes plateforme

## Gestion des risques

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Description | Source | Catégorie | Probabilité | Impact | Gravité | Remarque |
| Problème avec les environnements fournis | Externe | Timing | 5 | 10 | 0.5 | Quelques problèmes de Crash sur l’outil Jenkins + Problème de lenteur du au travail sur une machine virtuelle |
| Développement en retard | Interne | Timing | 4 | 15 | 0.6 | Changement d’environnement de travail + Mobilisation à l’armée qui n’ont pas pu être prévu |
| Manque d’expérience sur les tests techniques | Interne | Compétence | 8 | 10 | 0.8 | Premier projet avec autant d’implication dans l’intégration continu + Tests |
| Connaissances des outils | Interne | Compétence | 6 | 15 | 0.9 | Outils nouveaux + Licence gratuite à faire cohabiter avec un environnement de développement local |
|  |  |  |  |  |  |  |

# Rapport de test

## Test d’intégration

Pour les tests d’intégration, Nous avons tester que le login et la consultation du menu était fonctionnelle. Plusieurs autres tests d’intégration été prévu afin de tester toutes les fonctionnalités CRUD dans notre application malheureusement le manque de temps nous à rattraper

## Test de charge

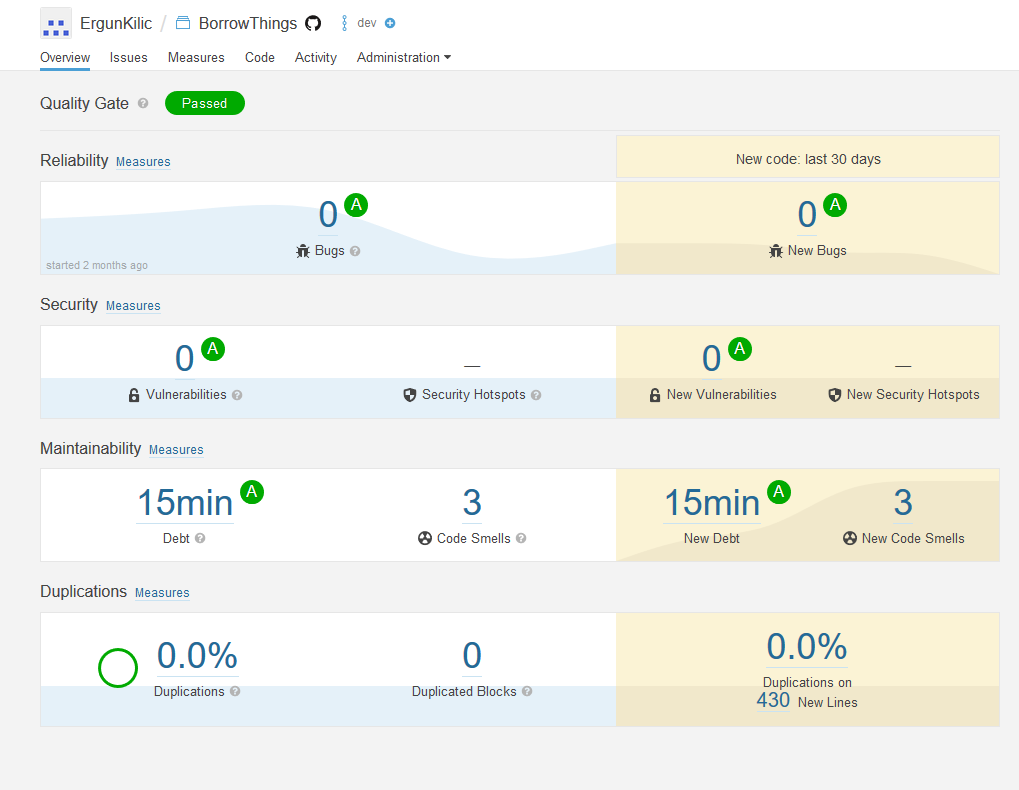
Pour les tests de charge, nous avons utiliser OctoPerf à qui nous avons fournis des fichier « .har » que nous avons enregistrer avec notre navigateur. C’est fichier contienne des simulations d’utilisateur de note site web. L’objectif était d’avoir un temps de réponse moyen très bas avec 10 utilisateurs simultané.

## Test fonctionnels

L’objectif était de tester les différentes fonctionnalités offertes par le site.

|  |  |
| --- | --- |
| **Fonctionnalité** | **Remarque** |
| Connexion | Fonctionnelle |
| Register | Fonctionnelle |
| Accès à la page user seulement par les utilisateurs possédant le rôle « Admin » | Fonctionnelle |
| Ajouter un rôle | Fonctionnelle |
| CRUD sur les salles de classes | Fonctionelle |
| CRUD sur les objets | 90% fonctionnelle petit problème avec la description qui n’est pas pris en compte |
| CRUD Emprunt | Non fonctionnelle suite aux derniers merges la fonctionnalité n’est plus fonctionnelle est n’as pas pu être corriger à temps |
| Pagination | Manquante |
| Notification | Manquante |
| Barre de recherche | Manquante |

## Analyse SonarCloud

L’objectif était d’avoir aucun bug et vulnérabilité sur notre sites

Nous pouvons voir qu’il reste quelques « Code smells » qui correspond à des bouts de code en commentaire.

# Conclusion

Finalement, la qualité de logiciel nous aura permis de voir les différents outils pouvant nous permettre un rendu de qualité. Elle nous permet de faire le point sur les manques et faiblesse de notre application mais aussi de valider certaines fonctionnelles et leur bon fonctionnement. Ceci nous aura été bénéfique d’un point de vue développeur car malheureusement durant les projets d’école cette partie est souvent omis par les étudiants alors qu’elle est primordiale en entreprise