Rapport de Tests du projet IDDY

Introduction

Ce rapport présente la façon dont ont été effectués les tests pour l'application IDDY développée en JEE. S'y trouvent également es résultats de ces tests ainsi qu'une brève analyse et interprétation pour chacun d'entre d'eux.

Objectifs et portée des tests

L'application a été testée selon deux méthodes : les tests fonctionnels avec Selenium et les tests de charge avec JMeter selon les différents scénarios de test. L'objectif étant avant tout de documenter la méthode plutôt que les tests eux-mêmes, étant donné que l'application possède un nombre limité de fonctionnalités.

Cas de tests

Les différents cas de test mis en pratique pour tester l'application sont décrits dans le document "spécification de tests". Le but de ce document étant avant tout d'interpréter les résultats, les cas de tests ne sont pas rappelés en détail ici.

Méthodologie pour les tests fonctionnels

Les tests fonctionnels ont été effectués avec Selenium sur une exécution locale de l'application. Les fonctionnalités principales ont été testées pour garantir une utilisation correcte de la plateforme web développée dans le cadre du projet. Les tests sont réalisés en enregistrant des séquences de cliques et d'entrées utilisateur dans un contexte précis. Elles peuvent ensuite être relancées avec Selenium de façon automatisée. Les tests sont ensuite exportés au format HTML compatible Selenium et sous forme de code Java Junit 4.0 avec Webdriver.

Méthodologie pour les tests de charge

Les tests de performance sont réalisés avec le programme JMeter. Le programme installé sur Windows dans notre cas précis permet d'automatiser l'exécution et l'envoi des requêtes HTTP sur notre application locale. Le but étant d'effectuer un grand nombre de requêtes pour déterminer le comportement de la plateforme face à une charge élevée. Cette suite de test est exportée sous la forme d'un fichier compatible avec JMeter d'extension .jmx

Résultats des tests

Ce chapitre présente les résultats des différents tests effectués et leur interprétation.

Tests fonctionnels

La suite de tests effectué avec le plugin Selenium du navigateur Firefox présente des résultats parfaitement satisfaisants :

Figure 1 : Sortie de tests Selenium

L'ensemble de la suite de test étant en vert après l'exécution, ceci indique que tout s'est déroulé selon le comportement souhaité et valide donc le bon fonctionnement de chacune de *features* testées.

Les tests sont effectués dans un ordre logique à respecter. En effet, les derniers tests dépendent de la bonne réussite des premiers. Cela est principalement dû au fait qu'il faut avoir créé un compte sur la plateforme avant de pouvoir accéder à certaines des fonctionnalités testées (comme l'action de s'abonner : Follow / Unfollow). Il est donc important de respecter les deux règles initiales ci-dessous pour garantir un contexte de test valide :

- Pas d'utilisateur nommé *testCaseUser*
- L'utilisateur user1 doit exister

Les fonctionnalités d'inscription, de connexion, de recherche, d'abonnement et de désabonnement ainsi que de création d'une entité sont validées.

Tests de charge

Les tests de charges proposent trois scénarios par fonctionnalité testée. Ceux-ci sont décrits précisément dans le document “spécification de tests”. Pour rappel, les scénarios sont les suivants :

- Charge normale (*soft load*)
- Charge moyenne (*medium load*)
- Charge élevée (*high load*)

Et les fonctionnalités testées sont :

- Simple affichage de la page d’accueil
- La connexion à la plateforme
- La recherche

Figure 2 - Les scénarios JMeter

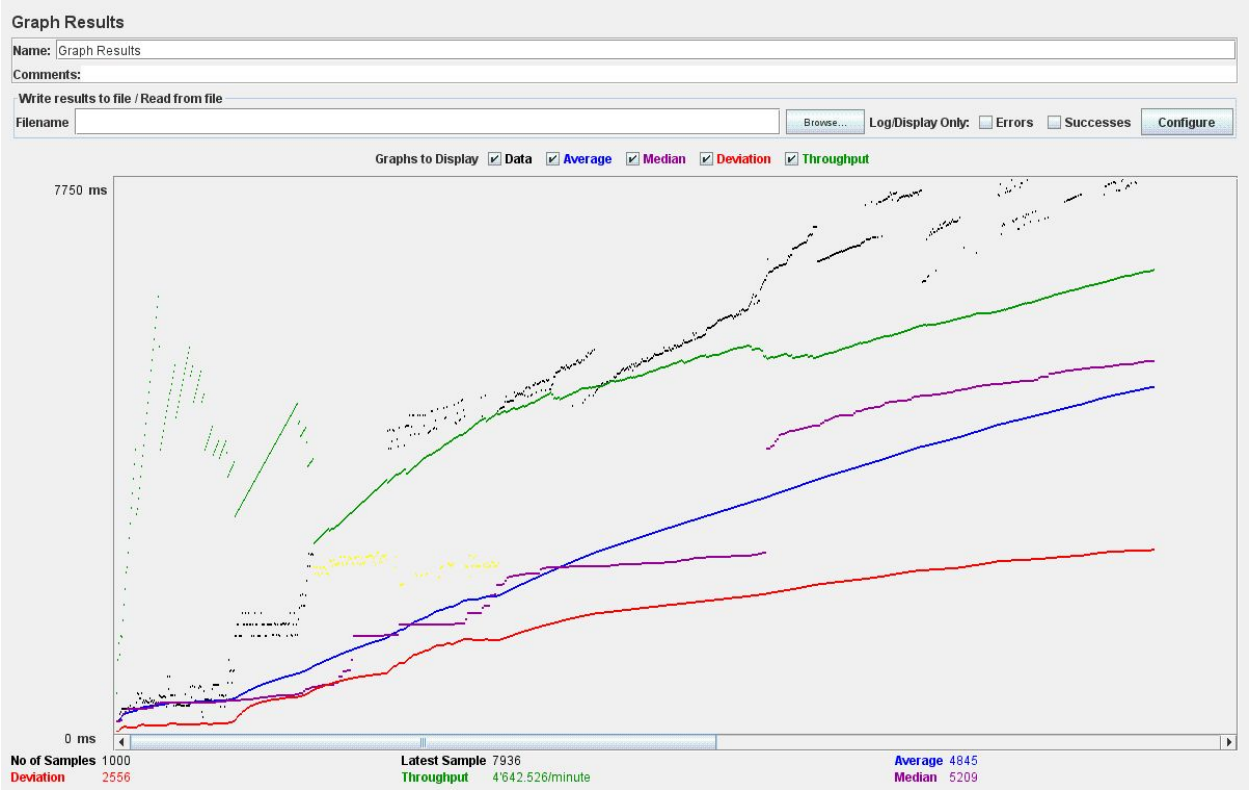
Pour des questions de visibilité et d’intérêt des tests, seuls les graphes des tests à charge élevée seront présentés dans ce document. En effet, en *soft load* et en *medium load*, l’application a un comportement tout à fait normal et rien ne semble montrer que le fonctionnement est sous stress.

Interprétation des graphiques à la page suivante.

Graphique du test de page d'accueil High Load

Le graphique ci-dessous présente le résultat du test de charge en *high load* sur la page d'accueil.

L'axe X représente le nombre d'utilisateurs simultanés et l'axe Y le temps de réponse de l'application.



On peut constater qu'une adaptation des ressources attribuées à l'application à mesure que le nombre d'utilisateurs augmente (probablement un automatisme de la JVM).

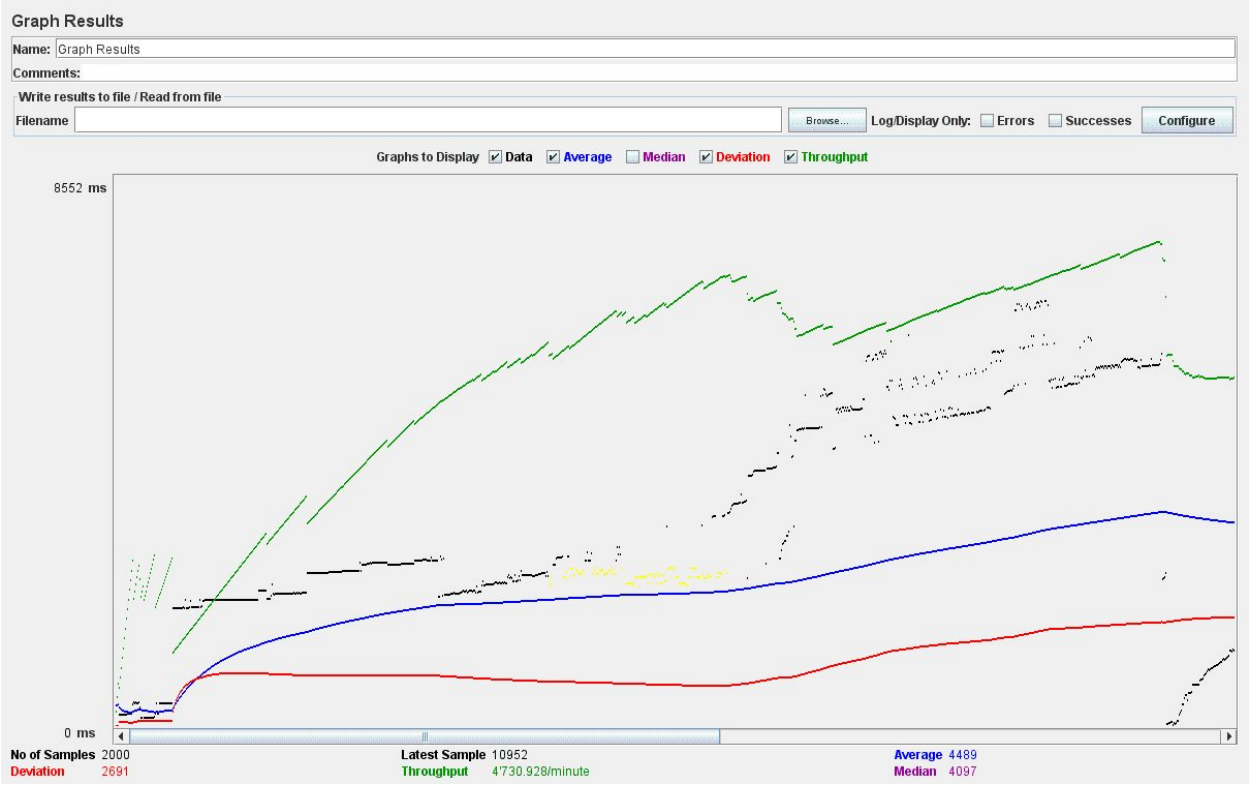
Les séries intéressantes sont :

- La série noire qui représente, pour le thread x, le temps de réponse en ms.
- La série bleue qui représente la moyenne du temps de réponse, combiné avec les échantillons précédents.

Le temps de réponse augmente petit à petit et pourrait devenir problématique avec un nombre beaucoup plus élevé de connexions.

Graphique du test Login + Search en High Load

Les codes couleurs correspondent à ceux du graphique précédent.



On remarque ici que le temps de réponse reste stable jusqu'à un certain nombre de requêtes, puis commence à augmenter avant de se stabiliser à nouveau. Sur la droite du graphe, le JVM ajuste les ressources attribuées à 'application et le temps de réponse commence donc à redescendre légèrement.

Discussion sur les résultats

Les tests fonctionnels passent sans problème. La plateforme développée contenant principalement des fonctionnalités basiques, il aurait été moins évident de tester des *features* très complexes telles que celles que l'on peut trouver dans une application normale aujourd'hui.

Pour les tests de charge, il aurait été intéressant d'ajouter un scénario de test de charge *extrème* afin de déterminer si oui ou non l'application peut montrer des signes de faiblesse voire même crasher complètement. Et ceci sur des durées plus conséquentes, une dizaine de minutes par exemple.

En ce qui concerne les tests effectués, ils sont entièrement satisfaisants. De plus, il serait intéressant également d'effectuer des tests directement sur la base de données pour contrôler le bon fonctionnement et l'intégrité des données sauvegardées.