Université Cadi AYYAD
Ecole Nationale des Sciences Appliquées de Safi

Rapport d'avancement de stage PFE

Conception et réalisation d'une application d'automatisation de tests

Toumani Sidibe

2018/2019

Encadrant académique : **M. Harzalla**
Encadrant professionnel : **M. Fouad Faris**

Table des matières

Chapitre 1

Présentation du contexte du stage

1.1 Présentation de l'entreprise d'accueil

Adria Business & Technology est une entreprise spécialisée dans la transformation digitale des banques et dans leur accompagnement au cours de celle-ci.

Avec une connaissance profonde de l'industrie bancaire, Adria crée des produits bien conçus pour aider les banques à renforcer leur stratégie digitale en respectant leurs spécificités, contraintes, et exigences. Leur ambition est de se positionner comme un partenaire de référence auprès des établissements financiers dans leur transformation digitale.

FIGURE 1.1 – Adria business & technology

1.2 Présentation du périmètre du projet

Le périmètre dans lequel s'inscrit notre projet est celui d'une application devant être testée efficacement contre toutes les régressions éventuelles. Les tests fonctionnelles étant des tâches longues et fastidieuses, donc coûteuses, la nécessité de l'automatisation de ceux-ci devient une évidence.

C'est ainsi que notre projet intitulé *Réalisation d'une application de test de non régression* prend forme, le but étant de mettre en place une série de tests souples pouvant s'exécuter à la demande, notamment après chaque opération susceptible de générer des régressions.

Chapitre 2

Présentation de la méthodologie de travail

2.1 Présentation des outils de travail

Comme tout projet informatique, le projet repose sur une pile d'outils dont voici une brève description de chacun.

Selenium Outil d'automatisation de tests par excellence et de loin le plus utilisé sur le marché, Selenium a fait ses preuves au cours de ses années d'existence. En effet, il permet de donner un contrôle quasi exclusif au programmeur qui l'utilise, lui donnant ainsi des possibilités quasi illimitées.

Spring Boot Le célèbre *framework Spring Boot* fait partie de notre outillage et ce choix se justifie par, d'une part, la notoriété inconstestée de ce dernier et d'autre part, l'atomicité du livrable obtenu suite à l'utilisation de Spring Boot, ce qui est un requis pour le projet.

ReactJS L'un des *framework* les plus populaires en termes de couche de présentation pour les applications *web* modernes. Sa légèreté et sa simplicité d'utilisation en ont fait un outil de choix pour les IHM de notre application.

Git L'incontournable outil de contrôle de source le plus populaire nous a permi de garder un rythme d'avancement cohérent et a grandement facilité l'intégration, la revue ainsi que la qualité du code produit.

Les outils présentés ci-dessus ne représent qu'une infime partie de la grande quantité d'autres outils ayant été utilisés dans la réalisation du projet. Nous nous limitions cependant à ceux-ci pour ne pas détourner ce document de son objectif premier.

2.2 Présentation des techniques et paradigme de développement

Behavior Driven Development (BDD) Le *Behavior Driven Development* est une technique de développement qui consiste à développer une application de façon modulaire de sorte que chaque module réponde à une exigeance fonctionnelle exprimée par le bénéficiaire. Cette technique diffère du TDD (Test Driven Development) qui lui consiste à créer un ensemble de

tests unitaires, de le voir échouer avant d'implémenter concrètement les portions de code en vue de faire passer les tests.

Le BDD repose sur le concept de *scénarios* eux même constitués de *gherkins*. Un scénario est un cas d'utilisation constitué le plus souvent de la manière suivante :

— Une mise en contexte
— Une suite d'action
— Un résultat attendu

Un *gherkin* n'est quant à lui qu'un simple phrase explicite pouvant constituer la mise en contexte, un action ou l'expression d'un résultat attendu.

Exploitation du BDD Le BDD est à la base une technique de développement logicielle, mais il n'est pas rare que cette technique soit détournée de sa fonction primaire (à savoir le développement de nouveaux logiciels) au profit du test d'applications déjà existantes. Et c'est dans ce sens que nous avons exploité toute la puissance du concept et l'avons mis au service de notre application. Ainsi, les scénarios seraient les cas de tests pouvant être renseignés par un testeurs et notre application sera en charge d'exécuter le scénario, de constater les résultats obtenus et d'en générer un rapport d'exécution.

Chapitre 3

État d'avancement du projet

3.1 Fonctionnalités existantes

Les principales fonctionnalités existantes à ce jour sont :

CRUD et organisation des exécutions par le testeur Le testeur est en mesure de créer autant d'exécutions qu'il le souhaite, une exécution n'étant autre qu'une suite de *gherkins*. Ensuite, il a la possibilité d'organiser ses exécutions dans des *testcases* puis finalement les *testcases* dans des scénarios. Ce regroupement est purement logique et dépend intégralement du choix du testeur, le but étant de se répérer lors de futures investigations. Le regroupement permet également le lacement séquentiel d'exécutions appartenant au même scénario.

Lancement d'une exécution La fonctionnalité de lancement d'une exécution est au coeur de la logique métier de l'application. Elle passe par la compréhension puis l'interprétation des *gherkins* en actions à réaliser dans un navigateur. Un *gherkin* pouvant être sous la forme de «*Quand je clique sur le bouton valider*» ou encore «*Étant donné que je suis authentifié*», il est indispensable de morceller la phrase pour la convertir adéquatement.

Génération d'un rapport d'exécution Au terme d'une exécution, l'application persiste les données journalisées de celle-ci, pouvant ensuite être converties en un document consultable régroupant toutes les informations liées à·ux l'exécution·s. Le rapport peut être visionné directement depuis une interface *web* ou alors téléchargé au format PDF sur le terminal du testeur.

3.2 Fonctionnalités à implémenter

Il n'est pas difficile d'imaginer toute une batterie de fonctionnalités à ajouter pour porter l'ergonomie de l'application à son appogée mais le temps étant une contrainte, nous nous limiterons à celles que nous envisageons de réaliser dans la durée impartie du stage.

Lancement simultané de plusieurs exécutions Une limite notable de l'application est le lancement séquentiel des exécutions. Ainsi, il est nécessaire d'attendre la fin d'une exécution avant de pouvant lancer la suivante. Bien qu'il soit possible de mettre plusieurs exécutions à la suite, et ne demander aucune intervention humaine pour l'agencement des celles-ci, cette méthode reste lente. La fonctionnalité d'exécution simultanée permettrait de lancer plusieurs instances d'un navigateur, autant d'instances que d'exécutions, tous encapsulées et indépendantes les unes des

autres. Cela ramènerait le temps d'exécution d'une suite d'exécutions non plus à la somme de tous les temps individuels mais au maximum des temps. Le gain est considérable.

Réponse en temps réel En l'état actuel de l'application, il est nécessaire d'attendre la fin d'une exécution pour voir ce qui s'y est produit. Une exécution pouvant être longue, le testeur n'a ni indicateur de progression, ni possibilité d'annuler une exécution déjà lancée. Le phénomène se voit exacerbé lorsque c'est une suite d'exécution qui est lancée. Il est donc prévu de mettre en place un système permettant au testeur d'observer chacune des exécutions cours et d'interrompre éventuellement celles qu'il souhaite.

Gestion des utilisateur Idéalement, l'application serait munie d'un système de gestion d'utilisateurs et de droits d'accès imposant aux testeurs de s'authentifier avant d'accéder à l'application. D'autre part ce permettrait d'associer un auteur à chaque exécution créer ainsi qu'à chaque rapport d'exécution, le but étant de pouvoir fournir les justes autorisations aux testeurs.

Conclusion

Ce sont sur ces constats que nous ponctuons ce document. S'il y a une chose à ajouter à ce résumé, ce serait de noter que le démarrage du projet a été fastidieux car les idées n'étaient pas claires et bien établies. Par conséquent, il est actuellement facile de rajouter de nouvelles fonctionnalités à l'existant. C'est pourquoi nous sommes optimistes dans la réalisation fructueuse du projet.