CODAZZI Rama
DAMOTTE Alan
GUI Kai
HAMMERER Jérémy
TORCK Quentin

RICM5

Document d'évaluation système

Groupe 2 – Site de vente de stickers personnalisés

My Stick It

Etat de chaque composant : non réalisé, réalisé, réalisé et validé

Nous avons implémentés plusieurs composants. L'application permet de gérer des clients, des produits, des paniers, des commandes, et des historiques de commandes. Ces composants ont été testés mais non validés par des tests unitaires formels JUnit.

La gestion des catégories de produit a été implémentée mais non testé et est non fonctionnelle actuellement. En effet, nous avons eu quelques difficultés à implémenter sur les pages jsp les listes déroulantes dynamiques permettant l'association d'une catégorie à un produit.

Les fonctionnalités principales ont donc été implémentées. Toutefois, des évolutions et améliorations sont possibles. En effet, concernant la création de nouveaux stickers via l'interface administrateur, les images sont enregistrées sur le serveur dans un dossier dédié. Une évolution possible serait d'utiliser des blobs stockés dans la base de données. Il faudrait néanmoins vérifier l'efficacité de ce genre de solution moins naturelle et plus coûteuse lors de requêtage, mais permettant des back-up plus simples et efficaces.

Une autre amélioration envisagée est la gestion des formulaires. Actuellement, un objet POJO permet de gérer les formulaires et vérifier l'intégrité des données. Une autre manière permettant de simplifier le fonctionnement des formulaires est l'utilisation de regexp et annotations directement dans la déclaration des attributs des entités.

Pour la partie IHM, nous avons utilisé Bootstrap afin d'obtenir un site web au design responsive. Les principales fonctionnalités ont été implémentées en essayant de respecter au mieux le cahier des charges et ce que nous avions défini lors de la phase de conception. Il reste toutefois certains problèmes à corriger et quelques points de l'interface à améliorer.

Concernant les fonctionnalités supplémentaires, voici la liste des fonctionnalités avancées qui ont été implémentées :

- Gestion de la confidentialité avec SSL/TLS lors des phases de login, signin, et de paiement actuellement nous avons configuré Glassfish de façon à pouvoir se connecter au site via https. Nous n'avons cependant pas eu le temps de mettre en place la confidentialité uniquement lors des phases critiques.
- Gestion de la concurrence et de la reprise sur panne avec des transactions ACID (utilisation de pull de connexion bonecp)
- Gestion asynchrone et transactionnelle de l'envoi des courriels via JMS et des EJBTimer : envoi de courriels lors de la phase de paiement (lorsqu'il est validé) à l'adresse du client et implémentation d'une newsletter grâce au EJBTimer.
- Performances (résultat de l'injection de charge avec Apache JMeter)
- Déploiement de l'application eCOM distribuée sur Microsoft Azure
- Intégration en continue avec Jenkins : build et déploiement automatisé

Fonctions réalisées, réalisées et testées, et celles restant à réaliser

Pour rappel voici la liste des tâches que nous devions réaliser et leur priorité :
Commander des stickers : (1)

- Stickers prédéfinis pour la V0 du site (1)
- Stickers personnalisés pour la V1 du site (4)
- Gérer le panier utilisateur et payer (2)
- Créer un compte utilisateur (3)
- Se connecter/Se déconnecter (3)
- Gérer son compte (5)
- Supprimer son compte (6)

Au terme du projet il est possible pour un utilisateur de s'insrire, de se connecter, se déconnecter, de gérer son panier et de commander des stickers prédéfinis. Toutes ces fonctionnalités ont été testées au cours du développement et des tests de charges ont été effectués. Il faudrait cependant mettre en place des tests unitaires pour s'assurer du bon fonctionnement en toutes circonstances. Pour ce faire, nous aurions très certainement utilisé JUnit mais par manque de temps et de connaissance de cet outil, nous n'avons pas mis en place de tests unitaires. Toutefois, la création de scénario de test avec Apache JMeter pemet tout de même d'obtenir à la fois des tests de charges et des tests unitaires.

Les fonctionnalités qui ne sont pas encore implémentées sont la création de stickers personnalisés via un outil de dessin, la gestion et la suppression d'un compte utilisateur. Une évolution également possible serait l'envoi d'email de confirmation (avec lien d'activation) lors de l'inscription.

Propriétés non fonctionnelles garanties par votre réalisation, et celles restant à considérer pour finaliser votre application

Voici la liste des exigences non fonctionnelles que nous avions définies en début de projet au sein de notre SRS :

- Portabilité : doit fonctionner sur ordinateur
- Compatibilité : doit fonctionner sur Chrome et Mozilla
- Utilisabilité : pas d'expérience requise. Interface clair et facile d'utilisation
- Robustesse : Le système doit pouvoir fonctionner malgré un grand nombre de connexions simultanées. De plus, en cas de panne du système, celui-ci doit se reconstruire automatiquement
- Sécurité : La fonctionnalité de paiement doit être sécurisée + ACID (tout ou rien)
- Atomicité : L'utilisation de transactions pour le paiement. La transaction doit être terminée, sinon, rien n'est fait.

Au terme du projet, nous pouvons dire que notre application garantit la plupart de ces propriétés.

Toutefois, concernant la portabilité, il faudrait vérifier le bon fonctionnement également sur environnement mobile (smartphone, tablette).

Concernant l'utilisabilité, certaines améliorations doivent être encore apportées à l'IHM de notre application afin que celle-ci respecte vraiment les attentes des clients potentiels (interface claire et précise). En effet, l'ajout de certains aspects visuels pourrait aider à la visualisation de l'utilisateur (icône plutôt que du texte).

Concernant la robustesse, les tests de charges ont montré des faiblesses dans l'implémentation du pool de connexion BoneCP. La limite de connexion parallèle à la base de données a été fixée à 20. Il faudrait dans un cas réel augmenter cette limite. Ceci se fait très rapidement, et pour notre utilisation de l'application (environnement de test sur des machines peu puissantes) nous n'avons pas jugé nécessaire de modifier cette limite. De plus, le nombre de connexion simultanée à la base de données SQL sur le Cloud Azure est encore plus limité (inférieur à notre configuration de BoneCP, seulement 4 connexions simultanées).

Temps consacré à la conception système

Le temps consacré à la conception (IHM et système) a été d'environ 1 mois et demi. N'ayant pas de réelles connaissances en Java EE et les technologies associées, il nous a dans un premier temps été difficile d'imaginer l'architecture finale de notre système. De plus cette étape était étroitement liée à la phase de conception IHM qui elle aussi a pris du temps.

Temps consacré au développement système

Le développement système des principales fonctionnalités a duré approximativement 6 semaines (du 21 Octobre au 27 Novembre). Au-delà de cette date, le travail sur la partie système (cœur de l'application) consistait essentiellement en l'amélioration de certaines fonctionnalités et la correction de certains bogues.

Les 2 dernières semaines de développement ont permis de travailler sur les fonctionnalités avancées : gestion de l'envoi de mail, ajout d'une newsletter, gestion de la confidentialité, intégration sur le Cloud Azure avec Jenkins, tests de charge et de performance.

Principales difficultés rencontrées

La première difficulté majeure du projet a été la compréhension des technologies utilisées et notamment le fonctionnement de Java EE. Aucun des membres du groupe n'ayant d'expérience dans le domaine, il a fallu passer plusieurs journées entières à la compréhension et au déploiement des premières beans et fonctionnalités du projet. Puis au fur et à mesure du développement, nous avons dû faire face à d'autres difficultés sur lesquelles il a fallu travailler plusieurs heures: gestion des beans distantes, gestion des transactions ACID.

La seconde difficulté a été l'intégration du projet à Maven. Encore une fois, nos connaissances étant limité sur ce sujet, nous avons rencontrés certains problèmes de dépendances lors de la compilation que nous devions régler petit à petit jusqu'à obtention de l'ear.

Une autre difficulté a été la prise en main du Cloud Azure. Nous avons décidé, d'utiliser le Cloud proposé par Microsoft car une offre étudiante nous a été fournie. Cependant après quelques essais, nous nous sommes vite aperçus que la plupart des services dont nous aurions besoin pour mettre en ligne notre site web n'étaient pas disponibles dans cette offre. Nous avons dû donc passer à la version d'essai d'une durée de 1 mois. Un autre problème que nous avons rencontré est que nous avons constaté qu'en utilisant une base de données MySQL proposée par azure, notre site avait par moment une grande perte de performance. L'origine de ce problème nous est encore inconnue mais ne semble pas être lié à l'application car après déploiement de la base de données sur la même machine virtuelle que celle contenant notre serveur Glassfish.

Pour terminer, la difficulté principale de ce projet est sans doute le temps qui nous est accordé à sa réalisation. Nous n'avons pas beaucoup d'occasion de travailler ensemble pour avancer plus rapidement et les projets annexes peuvent ralentir le développement. Il aurait peut-être été préférable d'allouer (comme les projets de fin de 3ème année), plusieurs semaines complètes durant lesquelles nous aurions travaillé uniquement sur le projet ECOM.