

UNIVERSITÉ DU QUÉBEC À MONTRÉAL

MGL7460-90 HIVER 2020

REALISATION ET MAINTENANCE DE LOGICIELS

PROJET TECHNIQUE / RAPPORT LÉGER

PRÉSENTÉ PAR

MAJED ABIDI

JEAN-PIERRE PASCAL SUDRE

27 MARS 2020

Équipe Enseignante :

Professeur : Sébastien Mosser (UQAM)

Laboratoires : Jean-Philippe Gélinas (SFL, Lévis), Jonatan Cloutier (SFL, Montréal)

Table des matières

1. Introduction.....	3
2. Objectifs.....	3
3. Réalisation de l'application et mise en place du pipeline continu	3
4. Les forces et faiblesses de l'application développée et du pipeline.	5

1. Introduction

Dans ce projet technique l'hypothèse de travail repose sur le fait que nous nous intéressons à la définition d'un logiciel de gestion de produits bancaires.

Un premier module est à destination des employés de la banque, pour créer des comptes clients et gérer ceux-ci. Un second module est destiné aux clients, qui peuvent consulter les produits qu'ils ont dans la banque, et y apporter des modifications. Ces deux modules partagent une dépendance vers le système de la banque.

Les modules des employés et des clients fonctionnent sous la forme d'application en ligne de commande, non interactive. La communication entre les modules est basée sur des communications directes entre les classes. Nous ne prenons pas en considération la sécurité, ni la persistance.

Nous avons choisi de développer notre application avec Eclipse IDE car **n'ayant pas une expertise avancée en développement ni des rôles de développeurs dans notre compagnie**, nos connaissances de Maven et/ou Gradle étaient extrêmement limitées et nous avons dû pallier cet handicap par plus de recherches personnelles accompagnées de compromis dans le développement de notre application.

2. Objectifs

L'objectif est la réalisation de l'application et la mise en place du pipeline de déploiement continu. L'application reste volontairement simpliste pour les besoins du projet technique. Les fonctionnalités sont données pour les cas nominaux. Nous avons mis en place la gestion d'erreur ainsi que la gestion des tests.

3. Réalisation de l'application et mise en place du pipeline continu

Dans le cadre de cette analyse, nous avons commencé par effectuer les opérations suivantes :

3.1. Dépôt Git public créé pour le projet

<https://github.com/Majeddesj/ProjetTechnique.git>

3.2. Mise en place d'un système de suivi d'exigences

3.2.1. Définition des user stories associée au projet



<https://github.com/Majeddesj/ProjetTechnique/issues>

3.2.2. Estimation des stories en termes de risque technique et de valeur métier



3.2.2.1. Parties prenantes

- Employé
- Client

3.2.2.2. Liste des exigences

Priorités requises / Parties prenantes	Employé 	Client 
Doit	Créer nom client	Lister tous les produits du client
Devrait	Lister les produits du compte client	Lister tous les produits auquel le client a accès
Devrait	Élever le statut du client qui accède à de nouveaux produits	Se dissocier d'un produit avec approbation éventuelle employé
Devrait	Diminuer le statut du client et restreindre les produits auxquels il accède	
Pourrait	Accepter l'ajout du produit pour le client	S'associer à un produit avec approbation éventuelle employé
Voudrait	Rejeter le produit demandé par le client	
Voudrait	Lister les clients avec des produits en attente de validation	

3.2.2.3. Évaluation du risque

Priorités requises / Parties prenantes	Employé 	Client 
Doit	Créer nom client	Lister tous les produits du client
Devrait	Lister les produits du compte client	Lister tous les produits auquel le client a accès
Devrait	Élever le statut du client qui accède à de nouveaux produits	Se dissocier d'un produit avec approbation éventuelle employé
Devrait	Diminuer le statut du client et restreindre les produits auxquels il accède	
Pourrait	Accepter l'ajout du produit pour le client	S'associer à un produit avec approbation éventuelle employé
Voudrait	Rejeter le produit demandé par le client	
Voudrait	Lister les clients avec des produits en attente de validation	

3.2.3. Stories tracée au code via les commits

Les user stories ont été tracées au code comme suit :

branche Feature-add-commands-for-employees: commit : Ajouter commande liste produit par client EMP [1#card-34319974, 1#card-34321111]

3.3. Développement des 3 modules

3.3.1. Mise en place d'un système de build tenant compte des dépendances

Les dépendances ont été créées avec Maven qui aide à définir, créer et maintenir des builds reproductibles avec des chemins de classe et des versions de bibliothèque bien définis.

3.4. Tests du système

3.4.1. Tests unitaires pour chaque module

Les tests unitaires nous offrent un certain nombre d'avantages :

- Augmentent notre confiance lorsque nous changeons de code
- Servent de documentation
- Rendent notre code plus réutilisable

Au vu de cette définition de nos tests unitaires et des raisons d'utilisation, nous avons choisi d'utiliser JUnit5 pour effectuer nos tests unitaires.

3.4.2. Tests d'intégration entre les modules "interface" (employé et client) et le module de la banque

[Pas encore adressé, seront livrés et documentés dans livraison finale du Projet Technique – Final 20-04-2020.](#)

3.4.3. Scénario d'acceptations pour valider automatiquement les stories

[Pas encore adressé, seront livrés et documentés dans livraison finale du Projet Technique – Final 20-04-2020.](#)

3.5. Pipeline de déploiement continu

3.5.1. Au push sur le dépôt, lancement des tests, fabrication des images docker, envoi dans le registre.

Pipeline avec Gitbash : commit / push / tests.

3.6. Mise en place de scénarios de démonstration

3.6.1. Démonstration de la valeur ajoutée du pipeline de déploiement continu pour le développement du produit.

[Pas encore adressé, mais le pipeline va être basé sur Azure sera livré et documenté dans livraison finale du Projet Technique – Final 20-04-2020.](#)

4. Les forces et faiblesses de l'application développée et du pipeline.

4.1. Les forces de l'application développée

L'application dans son état actuel est fonctionnelle, les dépendances sont gérées efficacement par Maven, ce qui nous permet d'ajouter facilement des bibliothèques tierces à notre application. Les tests unitaires ont été effectués avec succès. Le code développé est de bonne qualité.

4.2. Les faiblesses de l'application développée

Les user stories n'ont pas été tracés au code dans les commits en raison d'une contrainte, le head a été détaché de notre github et nous avons eu des difficultés à le réattacher. Pour cette raison nous avons décidé de ne pas forcer rebase de nos commits pour ajouter le lien vers nos user stories.

Nos commits sont de cet ordre :

Ajouter commande liste produit par client EMP [1#card-34319974, 1#card-34321111]

4.3. Les forces du pipeline

Va être adressé avec Azure pipeline, afin d'automatiser et simplifier le déploiement de l'application. Avec cette approche d'intégration continue et de déploiement continu généré automatiquement et entièrement intégré, l'application peut être mise à jour chaque fois que le code source est modifié. Nous pourrions créer et suivre les backlogs, gérer des dépôts de code et améliorer la collaboration entre les membres de l'équipe.

4.4. Les faiblesses du pipeline

Actuellement il s'agit d'un Pipeline avec Gitbash : commit / push / tests.