Équipe 1

# Reconnaissance automatique des matricules et compilation des notes pour optimiser le travail des enseignants

## Plan de développement logiciel

**Version 3.0**

# Historique des révisions

| Date | Version | Description | Auteur |
|---|---|---|---|
| 2022-09-16 | 1.0 | Rédaction initiale du plan de développement logiciel | Équipe 1 |
| 2022-09-27 | 2.0 | Modification du document suite à des commentaires de Olivier | Équipe 1 |
| 2022-10-06 | 2.1 | Repriorisation des tâches | Équipe 1 |
| 2022-10-19 | 2.2 | Mise à jour des tâches suite à la semaine de relâche | Équipe 1 |
| 2022-11-01 | 2.3 | Mise à jour des tâches de l'itération 6 | Équipe 1 |
| 2022-11-16 | 2.4 | Mise à jour des tâches de l'itération 7 | Équipe 1 |
| 2022-12-07 | 3.0 | Recorrection complète du plan suite à la correction de la remise de mi-session | Équipe 1 |

# Table des matières

# Plan de développement logiciel

## 1. Introduction

Ce document représente la planification que notre équipe envisage quant au développement du logiciel. Par conséquent, plusieurs aspects du développement seront décrits dans différentes sections distinctes. Tout d'abord, dans la partie 2, nous présenterons un énoncé de travail dans lequel nous expliquerons la solution proposée, les hypothèses et les contraintes que notre projet doit respecter, ainsi que les livrables du projet plus tard. Par la suite, dans la section 3, nous allons décrire la structure organisationnelle de l'équipe de projet, y compris la direction et les autres autorités responsables de la supervision. Nous allons aussi décrire comment le projet s'interface avec des groupes externes ainsi qu'identifier les responsabilités des membres de l'équipe de développement. Pour ce qui est de la section 4, elle se décompose en deux parties. En premier temps, on présentera le processus de projet en précisant le lot de travail, l'effort estimé, les dates de début et de fin de chaque sprint. Aussi, on commentera l'effort total estimé, en nombre d'heures, de chacun de ces sprints. Par la suite, la deuxième partie de la section 4 va entamer la gestion et suivi de l'avancement qui portera plus précisément sur la gestion des exigences, le contrôle de la qualité, la gestion des risques et la gestion de configuration.

## 2. Vue d'ensemble du projet

### 2.1 But du projet, portée et objectifs

Nous proposons comme solution une application Web supportée par tout navigateur moderne sur un ordinateur. L'application se nommant RMN sera une plateforme qui importe des copies d'examens pour accomplir la reconnaissance automatique des matricules et la compilation des notes. Son but est de sauver du temps précieux et d'optimiser le travail des enseignants. L'application Web sera connectée à un serveur, ce qui va permettre à plusieurs utilisateurs d'interagir avec en même temps. Pour des raisons de sécurité, l'application contiendra deux types d'utilisateurs: enseignants et administrateurs. Les administrateurs ont le pouvoir de créer un nouveau compte pour un enseignant. Le système va imposer à un enseignant de se connecter avec un pseudonyme et mot de passe unique pour accéder à l'application Web.

À part la fonctionnalité principale décrite plus haut, le client Web sera en mesure d'accomplir plusieurs autres fonctionnalités, en se basant sur des requêtes de nos clients. Les spécifications détaillées sont écrites dans le document de spécification des requis (SRS). Veuillez vous y référer pour plus de détails.

### 2.2 Hypothèses et contraintes

L'équipe de développement doit être composée de 5 membres et dont tous sont des étudiants en génie logiciel à l'école Polytechnique. De plus, on suppose qu'on aura du soutien technique de la part du professeur Antoine Legrain sur la documentation et fonctionnement de son programme fourni. Chaque membre de l'équipe de développement devra fournir un minimum de 18 heures par semaine, sur une session de 15 semaines, pour un total de développement de 1350 heures-personnes. Nous supposons qu'aucun membre de l'équipe n'abandonnera le projet en cours de développement et qu'aucun conflit entre les membres ne causera de retards ou de problèmes de développement.

Du côté matériel, le groupe aura accès à leurs ordinateurs personnels pour développer le logiciel. Ils auront également accès aux installations universitaires pour des réunions et des suivis hebdomadaires, ainsi qu'à une connexion Internet fiable. On suppose que des modèles d'examens seront fournis pour pouvoir tester l'efficacité de l'application. Le serveur sera hébergé à l'intérieur des serveurs de Polytechnique Montréal à travers une machine virtuelle, alors que la base de données serait hébergée sur MongoDB.

Un livrable mi-session sera effectué le 7 octobre, qui contiendra tous les documents relatifs au développement, ainsi qu'un prototype initial fonctionnel. Ensuite, une version Beta de notre logiciel est attendue le 23 novembre. Le produit final sera livré le 7 décembre 2022. On suppose que l'échéancier est conforme et sera respecté par tous les membres de l'équipe.

### 2.3 Biens livrables du projet

**Livrable de mi-session** (7 octobre 2022) :
- Processus Logiciel
- SRS
- Document d'architecture logicielle
- Plan de développement logiciel
- Plan de tests

**Livrable Beta** (23 novembre 2022)
- Déploiement de la version Beta

**Livrable final** (7 décembre 2022)
- Processus Logiciel
- SRS
- Document d'architecture logicielle
- Plan de développement logiciel
- Plan de tests
- Résultats de tests
- Code Source
- Documentation usager

## 3. Organisation du projet

### 3.1 Structure d'organisation

L'équipe de développement est composée de cinq étudiants en 4$^{e}$ de génie logiciel. Le responsable académique du projet et un de nos superviseurs est le professeur Olivier Gendreau.

Le professeur Jean-Michel Cardin, professeur au Cégep de Lanaudière à L'Assomption, ainsi que Antoine Legrain, professeur à Polytechnique de Montréal, seront aussi des superviseurs du projet ayant le rôle de clients.

### 3.2 Interfaces externes

Les groupes externes en lien avec le projet sont les deux clients avec qui on fait affaire, comme décrits plus haut. La communication avec les deux clients se fait par courriel pour la communication écrite, ainsi que *Teams* pour des visioconférences. À l'interne, c'est Mohammed Ariful Islam qui se charge d'entamer un contact avec les clients (voir 3.3 Responsabilités). Pour les téléconférences, l'équipe de développement au complet est généralement présente avec le ou les clients.

### 3.3 Responsabilités

Mohammed Ariful Islam

Responsabilité: Système (client): Personne qui va s'assurer que le système répond au besoin du client. Donc, il s'occupera de communiquer et valider avec le client que les fonctionnalités qui seront implémentées sont bien comprises par l'équipe.

Andy Lam

Responsabilité: Processus/Méthodologie: Personne qui va définir un processus de développement logiciel adéquat pour la session. De plus, il va se charger de la surveillance pour s'assurer que ce processus est bien respecté. Cette personne s'occupe largement de conceptualiser ce processus sur *processEDIT*.

Aleksandar Stijelja

Responsabilité: Assurance qualité: Personne qui s'occupe de mettre en place un système d'évaluation de la qualité du logiciel tout au long du développement, dans le but de retenir cette qualité et minimiser les erreurs. C'est la personne qui s'occupe largement de définir un plan de tests et d'appliquer ces tests.

Jason Thai

Responsabilité:

- Coordination/planification: Personne qui s'occupera de la gestion du projet. Il s'occupera donc de la planification du projet et s'assurera que le plan est bien suivi.
- Technique (outils): Personne qui s'occupera de déterminer les outils de développement qui seront potentiellement utilisés pour le développement du système.

Anis Zouatene

Responsabilité: Produit (configuration): Personne qui sera responsable de la configuration des services qui assureront la maintenance du système. Autrement dit, il assurera que les pipelines sont bien configurés par exemple.

## 4. Processus de gestion

### 4.1 Plan de projet

#### *4.1.1 Planification des phases*

| Phases | Jalons | Critères de réalisation |
|---|---|---|
| Identification des besoins | SRS | Les besoins des clients ont été identifiés. |
| | | Les besoins ont été clarifiés avec les clients. |

| | | |
|---|---|---|
| | | Les outils de développements ont été identifiés. |
| | | L'environnement de développement a été configuré pour rouler le module fourni. |
| | | Le module fourni a été clarifié avec le client. |
| | | Le document de requis (SRS) a été rédigé. |
| | | Le document de requis a été révisé avec Olivier. |
| | Modèle du processus logiciel | Une analyse des étapes du processus logiciel a été faite. |
| | | Les phases du processus logiciel ont été identifiées. |
| | | Les jalons du processus logiciel ont été identifiés. |
| | | Le modèle du processus logiciel a été créé. |
| | | Le modèle du processus logiciel a été révisé avec Olivier. |
| | Description des responsabilités | Les responsabilités ont été identifiées. |
| | | Les responsabilités ont été clarifiées. |
| | | Une distribution des responsabilités a été faite. |
| | | La description des responsabilités a été rédigée. |
| | | La description des responsabilités a été révisée avec Olivier. |
| Définition du projet | Planification du projet | Les phases du produit ont été identifiées. |
| | | La planification des phases a été faite. |
| | | Les objectifs des itérations ont été identifiés. |
| | | Les tâches ont été identifiées. |
| | | La planification des calendriers a été faite. |
| | | La planification du projet a été rédigée. |
| | | La planification du projet a été révisée avec Olivier. |
| | Document d'architecture logicielle | Les objectifs et contraintes architecturaux ont été identifiés. |
| | | Une description des objectifs et contraintes architecturaux a été rédigée. |

| | | |
|---|---|---|
| | | Les cas d'utilisation du logiciel ont été identifiés. |
| | | La vue des cas d'utilisation a été modélisée.. |
| | | Une analyse sur les modules significatifs du modèle de design a été faite. |
| | | Un diagramme de paquetage a été modélisé. |
| | | Les diagrammes de classes ont été modélisés. |
| | | Une description de la vue logique a été rédigée. |
| | | Une analyse sur les différentes interactions du processus du système a été faite. |
| | | Un diagramme de séquence a été modélisé. |
| | | Une description de la vue des processus a été faite. |
| | | Une analyse sur les requis du système dans lequel le logiciel sera déployé a été faite. |
| | | La vue de déploiement a été modélisée. |
| | | Une description des caractéristiques de taille et performance a été rédigée. |
| | | Le document d'architecture logiciel a été révisé avec Olivier. |
| | Planification de tests | Les exigences à tester ont été identifiées. |
| | | Les différents types de tests ont été identifiés. |
| | | Une description des différents types de tests a été rédigée. |
| | | Les outils requis utilisés au sein de la discipline de test ont été identifiés. |
| | | Une description des outils identifiés a été rédigée. |
| | | L'équipe de test a été identifiée. |
| | | Les ressources système utilisées au sein de la discipline de test ont été identifiées. |
| | | Les jalons utilisés au sein de la discipline de test ont été identifiés. |
| | | Le document plan de test a été révisé avec Olivier. |
| | Prototype de solution | Les tickets des fonctionnalités pertinentes au prototype ont été créés sur Jira. |

| | | |
|---|---|---|
| | | Les tâches sont distribuées entre les membres de l'équipe pour assurer l'efficacité du développement du prototype. |
| | | Le prototype de solution est développé. |
| | | Les fonctionnalités du prototype sont testées à l'aide du plan de tests. |
| | Présentation orale de mi-session | Détermination des points à divulguer dans la présentation orale. |
| | | Réalisation de la présentation orale. |
| | | La présentation orale est corrigée par Olivier. |
| Réalisation du produit | Produit en version bêta | Les tickets des fonctionnalités pertinentes au produit bêta ont été créés sur Jira. |
| | | Les tâches sont distribuées entre les membres de l'équipe pour assurer l'efficacité du développement du produit. |
| | | Le produit bêta est développé. |
| | | Les fonctionnalités du produit bêta sont testées à l'aide du plan de tests. |
| | | Le produit bêta a été vérifié par les clients pour s'assurer que les exigences des clients sont suivies. |
| Terminaison du projet | Produit final | Les corrections nécessaires soulevées par les clients, s'il y en a, sont entamées avant de compléter le produit final. |
| | | Les tickets des fonctionnalités restantes à développer sont créés sur Jira. |
| | | Les tâches sont distribuées entre les membres de l'équipe pour assurer l'efficacité de la finalisation du produit. |
| | | Le produit final est développé. |
| | | Le produit final est testé à l'aide du plan de tests. |
| | Mise à jour des livrables de la phase de définition du projet | Corrections des livrables à partir des points soulevés dans les remises antérieures. |
| | | Remise finale des livrables à Olivier. |
| | Résultats de tests | Mise à jour du document de résultats de tests tout au long du développement jusqu'au produit final. |
| | | Remise du document de résultats de tests à Olivier. |

| | | |
|---|---|---|
| | Présentation orale finale | Détermination des points à divulguer dans la présentation orale. |
| | | Réalisation de la présentation orale. |
| | | La présentation orale est corrigée par Olivier. |

### *4.1.2 Objectifs d'itération*

Itération 1:

- Création du processus logiciel

Itération 2:

- Rédaction du SRS

Itération 3:

- Rédaction du plan de développement logiciel

Itération 4:

- Conteneuriser le serveur avec Docker
- Créer la page d'accueil de l'application web
- Implémenter l'architecture Kubernetes

Itération 5:

- Adapter le module de reconnaissance pour répondre aux besoins des clients
- Permettre la création de comptes dans l'application web

Itération 6:

- Implémentation la fonctionnalité permettant la vérification manuelle des notes et matricules détectés

Itération 7:

- Implémentation de la fonctionnalité de définition de gabarit
- Permettre l'importation des fichiers à partir des services d'entreposage de fichiers

Itération 8:

- Rédaction du document de résultat de tests
- Correction des artéfacts
- Implémentation de la gestion de compte

### *4.1.3 Calendrier du projet*

| It. | Date de début et de fin | Tâche | Effort estimé (h-p) | Traçabilité (ID SRS) |
|---|---|---|---|---|
| **Phase d'identification des besoins** | | | | |
| 1 (90h) | 29 août au 2 septembre | Première rencontre avec les clients | 20 | s/o |

| | | | | |
|---|---|---|---|---|
| | (<1 semaine) | Conception initiale du processus logiciel | 40 | s/o |
| | | Rédaction du compte-rendu de la rencontre | 20 | s/o |
| | | Recherche sur les technologies à utiliser | 10 | s/o |
| | | | **90** | |
| 2<br>(90h) | 3 septembre<br>au<br>9 septembre<br><br>(1 semaine) | Révision des requis avec les clients | 10 | s/o |
| | | Première rédaction du SRS | 50 | s/o |
| | | Exploration du module de reconnaissance automatique | 30 | s/o |
| | | | **90** | |
| **Phase de la définition du projet** | | | | |
| 3<br>(75h) | 10 septembre<br>au<br>23 septembre<br><br>(2 semaine) | Première rédaction du plan de développement logiciel | 25 | s/o |
| | | Recherche sur le déploiement du logiciel | 30 | s/o |
| | | Correction du SRS | 5 | s/o |
| | | Correction du processus logiciel | 5 | s/o |
| | | Correction du plan de développement logiciel | 5 | s/o |
| | | Correction du SRS | 5 | s/o |
| | | | **75** | |
| 4<br>(176h) | 24 septembre<br>au<br>7 octobre<br><br>(2 semaines) | Création de la page d'accueil dynamique | 10 | s/o |
| | | Implémentation du serveur avec l'API de base | 30 | s/o |
| | | Implémentation de l'importation de copies par téléversement local | 10 | 3.2.1.3 |
| | | Implémentation de l'importation de fichiers de notes par téléversement local | 10 | 3.2.2.3 |
| | | Implémentation de la création d'une tâche | 15 | 3.2.3<br>3.2.4<br>3.2.5 |
| | | Intégration du module de reconnaissance automatique | 10 | 3.4 |
| | | Exportation des copies | 15 | 3.7 |
| | | Première rédaction du document d'architecture | 40 | s/o |
| | | Première rédaction du plan de tests | 15 | s/o |
| | | Encapsuler le serveur dans un Docker Image | 5 | s/o |

| | | | | |
|---|---|---|---|---|
| | | Création de l'interface permettant de visualiser la liste de tâches | 8 | 3.1.3 |
| | | Implémentation des requêtes pour recueillir les informations sur la liste de tâches d'un utilisateur | 8 | 3.1.3 |
| | | | **176** | |
| **Phase de la réalisation du produit** | | | | |
| 5<br>(185h) | 8 octobre au<br>25 octobre<br><br>(~2.5<br>semaines) | Implémenter l'architecture Kubernetes avec l'image Docker | 50 | s/o |
| | | Création de l'interface de la page de connexion | 15 | 3.1.1 |
| | | Création de l'interface de la page de création de compte pour les administrateurs | 15 | 3.1.2 |
| | | Adapter(restructurer) le module de reconnaissance automatique pour la vérification manuelle | 25 | s/o |
| | | Création de la page de vérification manuelle (visualisation des résultats du traitement) | 30 | 3.5.1<br>3.5.2<br>3.5.3<br>3.5.4<br><br>3.5.6 |
| | | Implémentation de la modification de résultats lors de la vérification manuelle | 40 | 3.1.5.6 |
| | | | **175** | |
| 6<br>(185h) | 25 octobre<br>au<br>8 novembre<br><br>(2 semaines) | Implémenter la validation par copie | 30 | 3.5.1 |
| | | Implémenter l'estimation du temps d'exécution | 20 | 3.1.3.3 |
| | | Implémenter la mise à jour automatique des statuts des tâches en cours | 5 | 3.1.3 |
| | | Implémenter la sélection des matricules à partir d'une liste prédéfinie | 10 | 3.5.6.1 |
| | | Implémenter le filtrage des tâches en cours | 5 | 3.1.3.7 |
| | | Implémenter la suppression des tâches en cours | 5 | 3.1.3.9 |
| | | Implémentation de la création de comptes au niveau du serveur | 10 | 3.1.2 |
| | | Implémentation de la connexion au niveau du serveur | 20 | 3.1.1 |
| | | Implémentation de la création de rectangles de détection lors de la définition de gabarit | 40 | 3.3.1.3 |

| | | | | |
|---|---|---|---|---|
| | | Implémentation de l'importation d'un fichier lors de la création d'un gabarit | 40 | 3.1.3.1.1<br>3.1.3.1.2 |
| | | | **185** | |
| 7<br>(270h) | 8 novembre<br>au<br>23 novembre<br><br>(~2.5 semaines) | Adapter(restructurer) le module de reconnaissance automatique pour la partie du gabarit | 15 | 3.3.1.3<br>3.3.3.2 |
| | | Implémentation du déplacement de rectangles présents dans la définition du gabarit | 15 | 3.3.1.3.3 |
| | | Implémentation de la suppression de rectangles présents dans la définition du gabarit | 20 | 3.3.1.4<br>3.3.3.3 |
| | | Implémentation de la modification du type de rectangle de détection | 10 | 3.3.1.5<br>3.3.3.4 |
| | | Implémentation de la suppression d'un gabarit existant | 15 | 3.3.2 |
| | | Implémentation du redimensionnement de rectangles présents dans la définition du gabarit | 25 | 3.3.1.3.4<br>3.3.1.5.2 |
| | | Implémentation du choix du format des copies (PDF ou ZIP) | 15 | 3.2.1.4 |
| | | Implémentation de l'importation de fichiers à partir de DropBox | 5 | 3.2.1.1<br>3.2.2.1 |
| | | Implémentation de l'importation de fichiers à partir de OneDrive | 5 | 3.2.1.2<br>3.2.2.2 |
| | | Implémenter la mise à jour automatique de la somme des notes lors d'un changement | 5 | 3.5.5.2.1 |
| | | Implémentation de la page de profil utilisateur | 15 | 3.1.4 |
| | | Implémentation de la sauvegarde automatique des chiffres identifiés | 20 | 3.6 |
| | | Implémentation de la segmentation des copies de sorties en groupe de fichiers compressés | 10 | 3.7.1.2 |
| | | Implémentation de l'arborescence Moodle des fichiers de sorties | 10 | 3.1.4.4.1 |
| | | Configuration de l'environnement de production (Kubernetes, VM) | 50 | s/o |
| | | Implémenter l'ajout d'une page de présentation pour tous les fichiers donnés en entrée | 30 | 3.8 |

| | | | | |
|---|---|---|---|---|
| | | | **265** | |
| **Phase de la Terminaison du produit** | | | | |
| 8<br>(220h) | 24 novembre<br>au<br>7 décembre<br><br>(2 semaines) | Créer un gabarit valide pour le client | 10 | s/o |
| | | Débogages et ajustements suite aux commentaires du client | 60 | s/o |
| | | Rédaction du guide d'utilisation du logiciel | 10 | s/o |
| | | Rédaction du guide d'installation du logiciel | 10 | s/o |
| | | Corriger le SRS | 10 | s/o |
| | | Corriger le plan de développement | 10 | s/o |
| | | Corriger le plan de tests | 10 | s/o |
| | | Corriger le schéma du processus logiciel | 10 | s/o |
| | | Corriger le document d'architecture logicielle | 15 | s/o |
| | | Rédaction du document de résultat de tests | 30 | s/o |
| | | | **220** | |
| Total | | | **1276** | |

### 4.2 Suivi de projet et contrôle

#### *4.2.1 Gestion des exigences*

L'équipe a rédigé une liste d'exigences qui a été validée par les clients, puis a compilé un document de spécification des exigences (SRS) basé sur la liste des exigences. Les tâches de chaque membre seront créées textuellement à partir du document de spécification des exigences sur la plateforme JIRA.

Si une modification aux exigences a lieu en cours de développement, nous ferons une rencontre d'équipe pour vérifier qu'elle répond encore aux exigences des clients. Si c'est le cas, une mise à jour du document de spécification des requis sera effectuée. Par contre, si nous estimons que cette dernière aura un gros impact sur la planification, nous allons devoir par conséquent modifier les échéances. Si un changement d'exigence affecte des fonctionnalités qui ont déjà été implémentées, nous créerons de nouvelles tâches sur le projet du Jira. Elles seront planifiées pour un sprint futur. Si le changement affecte des fonctionnalités qui n'ont pas été implémentées, nous modifierons seulement les tâches affectées dans le Jira.

#### *4.2.2 Contrôle de la qualité*

Premièrement, pour ce qui des artefacts, ils vont être relus et vérifiés manuellement par tous les membres

de l'équipe. Par la suite, le logiciel Antidote sera utilisé pour corriger toute faute de syntaxe ou de grammaire qui reste.
Ensuite, pour assurer un bon contrôle de la qualité du code, nous utiliserons plusieurs programmes conçus pour optimiser les revues d'ajout de fonctionnalités. Tout d'abord, nous utiliserons la plate-forme Gitlab, et nous utiliserons la fonctionnalité *merge request* pour nous assurer que les intégrations ne sont ajoutées qu'après avoir été examinées et validées par d'autres membres de l'équipe. Par conséquent, si l'auteur de la *merge request* ne remarque pas un problème ou un défaut, il y a plus de chances d'être remarqué par d'autres membres de l'équipe. Deuxièmement, le développement se fera en parallèle à l'aide d'une fonctionnalité de branches, de sorte que les fonctionnalités ne seront ajoutées que lorsqu'elles seront entièrement prêtes et testées. Troisièmement, si un membre de l'équipe trouve un bogue ou un problème, nous créerons une tâche Jira en conséquence et planifions la chronologie s'il s'agit d'un bogue majeur affectant nos ressources trop longtemps. Finalement, nous allons créer un plan de tests qui va couvrir tous les cas d'utilisations. Le plan de test va s'assurer que tout s'exécute normalement et qu'il y a le moins de bogues possible.

### *4.2.3 Gestion de risque*

La description des risques suit la convention suivante :

- Ampleur : sur une échelle de 1 à 10, 10 étant le risque le plus élevé. Cette analyse est basée sur la probabilité d'occurrence du risque, ainsi que ses impacts.
- Description : description textuelle du risque ainsi que les problèmes attendus.
- Impact : échelle définissant la portée du risque
    - C – critique (affecte le projet en entier)
    - E – élevé (affecte les fonctionnalités principales du système)
    - M – moyen (devrait être maîtrisable en appliquant une stratégie d'atténuation adéquate)
    - F – faible (l'acceptation du risque est une stratégie envisageable)
- Facteurs : aspects (**métriques**) du système pouvant être compromis.
- Stratégie de gestion : mesures à prendre afin de gérer le risque.

| **01 - Gestion de requêtes simultanées** | | | | |
|---|---|---|---|---|
| **Ampleur** | **Description** | **Impact** | **Facteurs** | **Stratégie de gestion** |
| **6** | Étant donné que les documents à traiter comportent plusieurs pages et plusieurs documents devront être à la fois, il y a un risque que le logiciel prenne un temps considérable pour identifier les notes et les matricules. Ainsi, à mesure que d'autres tâches s'ajoutent au système, celui-ci deviendra indisponible. | E | Temps d'indisponibilité du système | Lors de la planification de l'architecture du logiciel, il faudra séparer le module de reconnaissance de chiffre pour le garder modulaire. Une architecture `Scalable` devra être mise de l'avant pour s'assurer qu'une tâche soit faite dans des délais raisonnables. |

| **02 - Difficulté de l'intégration du module de reconnaissance** |
|---|

| Ampleur | Description | Impact | Facteurs | Stratégie de gestion |
|---|---|---|---|---|
| **5** | Dans le cadre de ce projet, un module de reconnaissance de notes et de matricules sera intégré dans l'application. Ce module a été conçu par Antoine Legrain pour ses propres besoins. Il sera ainsi nécessaire d'adapter le module afin de pouvoir respecter les exigences de notre application. | M | Temps de développement de l'application | Une analyse complète du module de reconnaissance de chiffre doit être complétée tôt dans le processus de développement de l'application. Une séance de clarification du module doit être faite avec le professeur Antoine. |

| 03 - Intégration du système de définition d'un gabarit | | | | |
|---|---|---|---|---|
| **Ampleur** | **Description** | **Impact** | **Facteurs** | **Stratégie de gestion** |
| **2** | Dans ce projet, on donne la possibilité à l'utilisateur de créer/supprimer un gabarit, contenant des rectangles de détection et de transcription. Avec ce gabarit, le logiciel pourra intégrer le module de reconnaissance des notes pour les retranscrire dans les rectangles de transcription. Pour ce faire, on aura besoin d'importer le fichier PDF d'une copie et la transformer en type canvas pour pouvoir établir les rectangles de détection. Donc, il y a un risque concernant la conversion pour le type canvas. | M | Temps de développement de l'application | Avant de commencer l'implémentation du système de gabarit, il faudra faire une recherche pour s'assurer qu'il existe une librairie qui sera compatible pour pouvoir faire la conversion d'un fichier PDF à un canvas. |

| 04 - Répondre aux besoins différents des clients | | | | |
|---|---|---|---|---|
| **Ampleur** | **Description** | **Impact** | **Facteurs** | **Stratégie de gestion** |
| **4** | Pour ce projet, des besoins différents ont été identifiés entre les clients. Effectivement, selon le contexte d'utilisation du logiciel par les professeurs, le module de reconnaissance aura besoin d'effectuer des tâches sur différents éléments.. Donc, il y a un risque que les besoins des clients ne soient pas tous répondus. | E | Niveau de satisfaction de l'utilisation du logiciel | Une clarification des besoins doit être faite auprès des deux clients. Une solution généraliste qui couvre les besoins des deux clients doit être analysée avant de commencer l'implémentation. Un suivi régulier sera aussi fait afin de s'assurer qu'on réponde toujours aux besoins des clients. |

### *4.2.4 Gestion de configuration*

Lorsqu'un changement est demandé ou qu'un problème est découvert, nous allons créer une tâche Jira qui sera assignée à un membre de l'équipe. Une description et une priorité seront ajoutées dans cette tâche. Si c'est un changement qu'on doit faire, le membre responsable de cette tâche crée une branche avec un nom débutant par « edited » et terminant par un titre descriptif du changement. Quant à un bogue, le nom de la branche débutera par « bug-fix » et elle se terminera par un titre descriptif du bogue.

Pour les artéfacts, nous allons nommer chacun des documents selon le nom de l'artéfact. Par exemple, le document du plan de développement logiciel sera nommé « Plan_de_développement_logiciel_Eq1». Suite à la rédaction initiale d'un artéfact, nous commençons à la version 1.0. Par la suite, à chaque modification majeure de l'artéfact, le premier chiffre de version sera incrémenté de 1 et le second chiffre reviendra à 0. Si on ne fait qu'une modification qui n'a pas un grand impact sur l'artéfact, on incrémente le second chiffre de 1. L'historique des versions de chaque artéfact se trouve sur la page suivante la page couverture et contiendra les descriptions associées à chaque version ainsi que les détails des ajouts et des modifications, et les auteurs de chaque révision.