Équipe #15

Engin de recherche et plateforme de maillage dans le domaine de l'IA en santé

Plan de développement logiciel

Version 3.0

Historique des révisions

Date	Version	Description	Auteur
2023-09-12	1.0	Rédaction de l'introduction et de la section 2	Antoine Déry
2023-09-12	1.1	Rédaction des objectifs d'itération et du calendrier du projet	Adam Halim Antoine Déry
2023-09-13	1.2	Rédaction de gestion de risque	Ahmed Sabsabi
2023-09-14	1.3	Rédaction de l'organisation du projet	Augustin Lompo
2023-09-14	1.4	Rédaction des sections 4.2.1, 4.2.2 et 4.2.4	Hichem Lamraoui
2023-09-14	1.5	Correction et relecture	Équipe #15
2023-09-30	2.0	Mise à jour et correction du calendrier de projet	Adam Halim Antoine Déry
2023-10-06	2.1	Révision pour prototype	Équipe #15
2023-11-24	3.0	Correction selon les rétroactions de la remise de mi-session	Équipe #15

Table des matières

Plan de développement logiciel

1. Introduction

Le plan de développement logiciel (PDL) décrit l'ensemble des travaux à réaliser pour livrer le projet. Plus précisément, le document fait état des objectifs du projet, des contraintes ainsi que des livrables. De plus, la structure organisationnelle de l'équipe de projet ainsi que les responsabilités de chaque membre sont décrites. Finalement, nous présentons les détails de notre processus de gestion ainsi qu'un échéancier détaillé et un plan de gestion des risques.

2. Vue d'ensemble du projet

2.1 But du projet, portée et objectifs

Le projet vise à développer une plateforme de répertoire des expertises de la Communauté de Pratique IA en Santé (CPIAS). Cette application web a comme objectif de faciliter l'intégration de l'IA dans le domaine de la santé en favorisant la collaboration entre les membres de la communauté. La plateforme sera conçue pour accueillir les membres de la CPIAS tels que des chercheurs, professionnels de la santé et étudiants dans le domaine de l'IA en santé.

Les objectifs se résument à développer et livrer un moteur de recherche permettant à un utilisateur d'énoncer une problématique et d'obtenir une liste d'experts pouvant y répondre. Ces suggestions seront classées selon différents critères pour recommander l'expert le plus adapté à la problématique.

Les livrables attendus seront un prototype de l'application (6 octobre 2023), une version bêta (20 novembre 2023) et la version finale de l'application (4 décembre 2023). Ces remises seront accompagnées d'artéfacts, soit les spécifications des requis du système (SRS), le document d'architecture logicielle, le plan de tests et les résultats des tests ainsi que le présent document.

2.2 Hypothèses et contraintes

Nous posons l'hypothèse qu'en cas de besoins, le client soit en mesure de répondre à nos questions et à nous fournir les informations dont nous avons besoin. Il est également attendu que le client fournisse les ressources nécessaires pour déployer l'application (abonnement à un service infonuagique, par exemple). Nous posons l'hypothèse qu'avec une contribution moyenne de 18 heures par semaine par personne, avec une équipe de 5 personnes et une contrainte de 15 semaines pour réaliser le projet, celui-ci prendra environ 1350 heures à réaliser.

2.3 Biens livrables du projet

La livraison des biens s'effectue en trois temps. D'abord, le 6 octobre 2023, l'équipe de projet devra livrer un prototype de l'application. L'objectif est d'établir les bases et de s'assurer que les exigences sont réalistes pour la portée du projet. Par la suite, la version bêta de l'application sera livrée le 20 novembre 2023. Celle-ci devra inclure l'ensemble des exigences jugées essentielles. Finalement, après les rétroactions du client sur la version bêta, des corrections seront apportées et la version finale sera livrée le 4 décembre 2023. Le contenu de chaque remise est présenté dans le Tableau 1.

Tableau 1 : Biens livrables du projet

Livrable	Artéfacts	Logiciel
Remise de mi-session (6 oct. 2023)	- SRS - PDL - Document d'architecture logicielle - Plan de tests logiciels - Processus	Prototype
Remise bêta (20 nov. 2023)	N/A	Version bêta
Remise finale (4 déc. 2023)	- Artéfacts de la remise de mi-session mis à jour - Résultats de tests	- Version finale - Code source - Documentation usager

3. Organisation du projet

3.1 Structure d'organisation

3.1.1 Membres de l'équipe de développement

Dans le cadre de réalisation du projet, notre équipe de développement est composée de 5 étudiants en génie logiciel : Ahmed Sabsabi, Adam Halim, Antoine Déry, Hichem Lamraoui et Augustin Lompo.

Chaque membre de l'équipe joue un rôle actif dans la réalisation du projet et est attribué une responsabilité qui sera identifiée et décrite dans la section 3.3

Afin d'assurer la bonne réalisation du projet, notre équipe est principalement supervisée par deux groupes:

3.1.2 Superviseurs académiques

D'une part, nous sommes supervisés par l'équipe académique constituée des professeurs Olivier Gendreau et Lévis Thériault avec qui nous effectuons un suivi hebdomadaire quant à l'avancement du projet. Ces suivis nous permettent entre autres de discuter de l'avancement du projet, mais aussi des difficultés que nous rencontrons et à cet effet recevoir des conseils académiques.

3.1.3 Équipe du CHUM

D'autre part, nous sommes aussi sous la supervision de l'équipe du CHUM constituée principalement de Pascale Béliveau et de ses collaborateurs. De façon similaire aux suivis hebdomadaires académiques, nous avons des rencontres hebdomadaires avec l'équipe CHUM dont le but est de faire état de l'avancement du projet.

3.2 Interfaces externes

3.2.1 Contacts internes

Les contacts internes au projet sont constitués de tous les membres de l'équipe de développement à savoir Ahmed Sabsabi, Adam Halim, Antoine Déry, Hichem Lamraoui et Augustin Lompo. À cet effet, tous les membres participent de façon active à la collaboration et à l'échange avec le client tant lors des réunions que dans les discussions dans les canaux de communications établies.

3.2.2 Contacts externes

Les contacts externes au projet sont constitués des membres de l'équipe CHUM à savoir Pascale Béliveau, Abid Mariem, Amal Khabou, Kahina Bensaadi, Lilia Brahimi, Yassine Benhajali et Vincent Beaulac.

Tel que mentionné dans la section *3.1.3* des réunions hebdomadaires sont organisées avec l'équipe CHUM. Lors de ces réunions, nous discutons principalement des besoins et des attentes du client, mais aussi de nos questionnements quant à certains points relatifs aux technologies, aux données et à la sécurité lorsque nécessaire. C'est également une occasion pour nous de faire part de notre état d'avancement et de recevoir la rétroaction de l'équipe CHUM.

Afin de faciliter davantage la communication, un canal Slack a également été mis en place par l'équipe CHUM.

3.3 Responsabilités

La répartition des responsabilités aux membres de l'équipe s'est faite en tenant compte des intérêts, mais aussi des compétences de chacun des membres; c'est ainsi que chaque membre s'est vu attribuer les responsabilités suivantes:

Antoine Déry travaille principalement sur le développement de la plateforme web, tout en apportant son soutien au développement du serveur lorsque nécessaire.

Adam Halim travaille à la fois sur le développement du serveur, mais aussi de l'intelligence artificielle responsable du système de recommandation de la plateforme.

Ahmed Sabsabi travaille à la fois sur le développement du serveur, surtout la portion qui se rattache aux données.

Augustin Lompo travaille à la fois sur le développement du serveur, mais aussi de l'intelligence artificielle responsable du système de recommandation de la plateforme.

Hichem Lamraoui travaille à la fois sur le développement du serveur et sur l'intelligence artificielle responsable du système de recommandation de la plateforme.

4. Processus de gestion

4.1 Plan de projet

4.1.1 Planification des phases

Le projet se divise en trois jalons : la livraison du prototype, la livraison de la version bêta et la remise de la version finale de l'application.

La livraison du prototype s'effectuera le 6 octobre 2023 et s'accompagne de la remise des principaux artéfacts, tel que montré au Tableau 1. Ce jalon sera complété lorsque ces artéfacts seront complétés. Pour ce qui est de l'application, celle-ci devra comporter une interface utilisateur avec, au minimum, une barre de recherche permettant d'envoyer des requêtes au serveur. De plus, le serveur sera en mesure de retourner les résultats pour des requêtes simples. Par exemple, le prototype sera en mesure de trouver les correspondances entre les mots-clés et les expertises des membres.

Par la suite, la livraison de la version bêta s'effectuera le 20 novembre 2023. Ce jalon sera complété lorsque l'application sera déployée avec toutes les exigences jugées essentielles. Ce livrable ne contient aucun artéfact.

Finalement, la version finale sera livrée le 4 décembre 2023. Ce jalon sera complété lorsque tous les artéfacts du prototype auront été révisés, en fonction des rétroactions reçues. De plus, le document des résultats de tests devra être rédigé. Pour ce qui est de l'application, celle-ci devra inclure les modifications demandées par le client suite à la livraison de la version bêta.

4.1.2 Objectifs d'itération

Dans le cadre de ce projet, l'équipe a décidé de réaliser des itérations de trois semaines puisque cela concorde bien avec les dates des différents livrables. Cette section montre les objectifs des différentes itérations jusqu'à la livraison finale.

Itération #1 (28 août 2023 au 17 septembre 2023)

- Élaborer l'horaire des périodes de travail et des rencontres.
- Créer le processus logiciel.
- Définir les requis de l'application (SRS).
- Choisir les technologies à utiliser.
- Rédiger le PDL.
- Débuter le développement du prototype.

Itération #2 (18 septembre 2023 au 8 octobre 2023)

- Mettre à jour les artéfacts précédents.
- Rédiger le plan d'architecture logicielle.
- Rédiger le plan de tests logiciels.
- Choisir et configurer le modèle d'intelligence artificielle.
- Créer et configurer le serveur.
- Développer l'interface de la page d'accueil.
- Créer la page des résultats de recherche.
- Déployer le prototype.

Itération #3 (9 octobre 2023 au 29 octobre 2023)

- Développer le système de recommandation d'expertise.
- Création du formulaire de correction d'information
- Créer la page de membres.
- Créer un formulaire d'inscription.

Itération #4 (30 octobre 2023 au 20 novembre 2023)

- Affiner le système de recommandation d'expertise.
- Implémenter la cartographie sur l'interface utilisateur
- Intégrer le système de recommandation d'expertise à l'interface utilisateur.
- Implémenter le mode « administrateur »
- Déployer la version *bêta.*
- Établir les points à aborder et fonctionnalités à présenter pour la présentation orale.

Itération #5 (21 novembre 2023 au 4 décembre 2023)

- Créer un support visuel pour la présentation orale.
- Exécuter les tests logiciels.
- Collecter les résultats de tests.
- Mettre à jour les artéfacts précédents
- Déployer la version finale.

4.1.3 Calendrier du projet

It.	Date de début et de fin	Tâche	Effort estimé (h-p)	Traçabilité (ID SRS)
1	28 août au 17 sept. (3 sem.)	Rencontre avec le client (1 h/semaine)	15	s/o
		Rencontres SCRUM (1 h/semaine)	15	s/o
		Rencontre avec le superviseur (1 h/semaine)	15	s/o
		Création du processus logiciel	7.5	s/o
		Rédaction du document de la remise initiale	7.5	s/o
		Rédaction du SRS	35	s/o
		Rédaction du PDL	30	s/o
		Recherche sur les technologies	30	s/o
		Essai de modèles & outils d'IA	25	s/o
		Mise en place et configuration d'un environnement	20	s/o
		Création de tâches & itérations sur Gitlab	20	s/o

		Création du serveur	15	s/o
		Création du projet React	15	s/o
		Création de la page d'accueil de la plateforme	30	3.1.2
		TOTAL	**280**	**s/o**
2	18 sept au 8 oct. (3 sem.)	Rencontre avec le client (1 h/semaine)	15	s/o
		Rencontres SCRUM (1 h/semaine)	15	s/o
		Planification de sprint (1 h/itération)	5	s/o
		Rencontre avec le superviseur (1 h/semaine)	15	s/o
		Rédaction du plan de tests logiciels	10	s/o
		Rédaction du document d'architecture logicielle	25	s/o
		Mise à jour des artéfacts	10	s/o
		Création de la page des résultats d'une recherche	40	3.2.3
		Extraction des données du fichier d'inscription	20	3.4.3 3.6.1 3.6.2
		Initialisation du modèle d'IA pour retourner des profils selon des mots simples	60	3.2.1 3.4.1 3.4.2
		Mise en place de la communication entre le client et le serveur (points d'entrées)	30	s/o
		Ajout d'un *linter* pour Typescript	5	s/o
		Ajout d'un *linter* pour Python	5	s/o
		Mise en place des pipelines sur GitLab	10	s/o
		Correction de bogues et réalisation de tests	15	s/o
		TOTAL	**280**	**s/o**
Remise du prototype : 6 octobre 2023				
		Rencontre avec le client (1 h/semaine)	15	s/o
		Rencontres SCRUM (1 h/semaine)	15	s/o
		Planification de sprint (1 h/itération)	5	s/o

3	9 oct. au 29 oct. (3 sem.)	Rencontre avec le superviseur (1 h/semaine)	15	s/o
		Création de la barre de navigation	20	3.1.3
		Création de la page des membres	40	3.1.4
		Création du formulaire de correction d'information	20	3.1.5
		Étendre les fonctionnalités de recherche pour rechercher des phrases	60	3.2.2 3.2.3
		Initialisation du système de recommandation d'expertise	60	3.3.1 3.3.2
		Correction de bogues et réalisation de tests	40	s/o
		TOTAL	**290**	**s/o**
4	30 oct. au 20 nov. (3 sem.)	Rencontre avec le client (1 h/semaine)	15	s/o
		Rencontres SCRUM (1 h/semaine)	15	s/o
		Planification de sprint (1 h/itération)	5	s/o
		Rencontre avec le superviseur (1 h/semaine)	15	s/o
		Création de la page « À propos »	10	3.1.6
		Création de la page « administrateur »	20	3.1.7
		Étendre les fonctionnalités de recherche pour rechercher des phrases (suite)	40	3.2.2 3.5.1 3.5.2 3.5.3
		Extraction et traitement des données de LinkedIn	40	3.5.4 3.6.3
		Implémentation de la cartographie	60	3.2.3
		Amélioration du système de recommandation d'expertise	60	3.3.2 3.3.3 3.3.4 3.3.5
		Correction de bogues et réalisation de tests	20	s/o
		TOTAL	**300**	**s/o**
Remise de la version bêta : 20 novembre 2023				
5	21 nov.	Rencontre avec le client (1 h/semaine)	10	s/o

	au 4 déc. (2 sem.)	Rencontres SCRUM (1 h/semaine)	10	s/o
		Planification de sprint (1 h/itération)	5	s/o
		Rencontre avec le superviseur (1 h/semaine)	10	s/o
		Mise à jour des artéfacts	30	s/o
		Rédaction du document des résultats de tests	20	s/o
		Peaufinage de l'interface	15	s/o
		Implémentation des améliorations souhaitées par le client	50	s/o
		Préparation de présentation finale	20	s/o
		Correction de bogues et réalisation de tests	30	s/o
		Ajout du lien de l'outil sur le site du CPIAS	10	3.1.1
		TOTAL	**210**	**s/o**
Remise de la version finale : 4 décembre 2023				

4.2 Suivi de projet et contrôle

4.2.1 Gestion des exigences

La gestion des exigences est un aspect essentiel du projet pour garantir que le produit s'aligne sur les fonctionnalités et les objectifs souhaités tout au long de son cycle de développement. Elle implique la collecte, le suivi et le contrôle des exigences du produit afin de faciliter la communication, de surveiller les progrès et de gérer les changements de manière efficace.

4.2.1.1 Mécanismes d'information et de contrôle

A. **Documentation des exigences:** Toutes les exigences du projet, y compris les exigences fonctionnelles et non fonctionnelles, seront répertoriées de manière exhaustive dans un document dédié aux spécifications des requis du système (SRS). Ce document servira de référence principale pour les exigences du projet.

B. **Collaboration via Google Drive:** Nous utiliserons Google Drive comme plateforme de collaboration pour partager les artéfacts du projet, y compris les spécifications des requis du système (SRS), le plan de développement du logiciel, et d'autres documents pertinents. Les parties prenantes pourront faire part de leurs commentaires directement dans ces documents.

C. **Réunions hebdomadaires SCRUM :** L'équipe de projet tiendra des réunions SCRUM hebdomadaires au cours desquelles les demandes de modification des exigences seront discutées et validées. Ces réunions serviront de forum pour discuter des tâches liées aux exigences et les classer par ordre de priorité.

D. **Réunions avec le client :** L'équipe de projet tiendra également des réunions hebdomadaires avec le client pour s'assurer que le produit est conforme à ses besoins. Ces réunions seront l'occasion pour le client d'examiner les progrès accomplis, de donner son avis et de valider les exigences.

E. **Communication sur Slack:** Un canal Slack a été mis en place comme moyen de communication en temps réel avec les parties prenantes. Ce canal sera utilisé pour les mises à jour rapides, les clarifications et les discussions informelles liées aux exigences.

4.2.1.2 Mesure et rapports

A. **Mesure :** Toutes les exigences du projet seront traduites et organisées en *« Epics » et « Issues »* sur GitLab. En fait, nous utiliserons l'outil *« Issues Analytics »* de GitLab pour suivre et analyser les métriques liées à la gestion des exigences. Cet outil fournira des informations sur l'avancement des problèmes, les délais d'exécution et d'autres paramètres pertinents.

B. **Traçabilité :** Chaque artéfact dans Google Drive sera muni d'un tableau de l'historique des révisions qui nous permettra de suivre toutes les modifications apportées à ces documents. L'historique des révisions comprendra la date de la modification, la version du document, la description de la modification et l'auteur de la modification.

4.2.2 Contrôle de la qualité

Le contrôle de la qualité permet de garantir que tous les produits livrables répondent aux normes de qualité définies. Il implique une vérification systématique de la qualité des artéfacts, y compris les documents du projet et le code source de l'application. En outre, il décrit le processus permettant de prendre des mesures correctives lorsque cela est nécessaire pour maintenir le niveau de qualité souhaité.

4.2.2.1 Contrôle de la qualité des artéfacts

A. Calendrier : Le contrôle de la qualité des artéfacts du projet est principalement effectué avant le début du développement de l'application. Des modifications peuvent être apportées au cours du développement en réponse aux commentaires des parties prenantes.

B. Méthodes :

- **Revues par les pairs :** Tous les artéfacts du projet, y compris les spécifications des requis du système (SRS), le plan de développement du logiciel, le document d'architecture du logiciel, le plan de test du logiciel et le processus de développement du logiciel, feront l'objet d'une révision par les pairs. Les membres de l'équipe et les parties prenantes participeront à ces revues afin d'identifier les incohérences, les erreurs et les points à améliorer.
- **Normes de documentation :** Nous respecterons les normes de documentation, en veillant à ce que chaque artéfact soit structuré, bien organisé et contienne des informations claires et concises. Les tableaux d'historique des révisions, tels que mentionnés précédemment, seront utilisés pour suivre les modifications et maintenir le contrôle des versions.
- **Rétroaction (*feedback*) :** Une boucle de rétroaction sera établie avec les parties prenantes du projet, ce qui leur permettra de donner leur avis et de suggérer des améliorations pour les artéfacts du projet. Leurs commentaires seront pris en compte au cours du processus de contrôle de la qualité.

C. Action corrective

- En cas de problèmes ou de divergences identifiés au cours du processus de contrôle de la qualité, des mesures correctives seront prises rapidement. Cela peut impliquer la révision et le réexamen de l'artéfact concerné afin de garantir l'alignement sur les normes de qualité.

4.2.2.2 Contrôle de la qualité du code source de l'application

A. Calendrier : Le contrôle de la qualité du code source de l'application sera un processus continu tout au long du cycle de développement.

B. Méthodes :

- **Développement incrémental de fonctionnalités :** Les différentes fonctionnalités de l'application seront développées dans des branches dédiées avant d'être fusionnées avec la branche “main”. Nous utiliserons le mécanisme de “Merge Request” offert par GitLab pour revoir le code et apporter les modifications nécessaires avant d'accepter sa fusion avec la branche “main”.
- **Révisions du code :** Toutes les modifications et tous les ajouts de code seront soumis à des revues de code par les membres de l'équipe. Les réviseurs de code évalueront le code pour s'assurer qu'il respecte les normes de codage, les meilleures pratiques et les problèmes potentiels. Les révisions de code seront effectuées avant de fusionner les modifications dans la base de code principale.
- **Outils de qualité du code :** Des outils de qualité du code et des “*linters*” (ESLint pour TypeScript et Pylint pour Python) seront utilisés pour faire respecter les normes de codage et identifier les mauvaises odeurs de code ou les violations de style.
- **Tests automatisés :** Ceci inclut des ***tests unitaires*** qui seront mis en œuvre pour s'assurer que toutes les fonctionnalités fonctionnent correctement au niveau du code. Nous utiliserons également l'outil “CI/CD pipelines” de GitLab pour les ***tests d'intégration***, afin de s'assurer que les changements de code s'intègrent parfaitement à la base de code principale et passent les tests automatisés.
- **Tests manuels :** Il s'agit des ***tests d'acceptation par l'utilisateur***, qui seront menés pour évaluer l'application dans des conditions réelles. Cela permettra de s'assurer que l'application répond aux attentes des utilisateurs et fonctionne comme prévu. En outre, le ***test du système de recommandation aux utilisateurs utilisant l'IA*** impliquera l'évaluation de ses performances et la nécessité de s'assurer qu'il fournit des recommandations significatives et précises aux utilisateurs. Cette évaluation comprendra des mesures telles que la précision, le rappel, le score F1 et d'autres mesures pertinentes pour évaluer l'efficacité du système.

C. Action corrective :

- L'outil “Issues” de GitLab sera utilisé comme principal système de suivi des problèmes pour saisir et gérer les problèmes liés aux exigences, les demandes de changement et les demandes d'amélioration. Cet outil facilitera le suivi des problèmes et des actions correctives liés au code source.

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.

R1 - Retard dans la fourniture des informations par le client				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
8	Si le client ne fournit pas en temps opportun les informations nécessaires, telles que les exigences spécifiques ou l'accès aux ressources infonuagiques, cela pourrait retarder le développement du projet.	E	Complétude, Adéquation, Performance	Établir une communication claire avec le client pour s'assurer de la disponibilité des informations requises. Définir des jalons pour les livrables du client.

R2 - Complexité technique inattendue				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
7	Des défis techniques non prévus peuvent surgir lors du développement de l'application, ce qui pourrait entraîner des retards ou des problèmes de qualité.	E	Réutilisabilité, Analysabilité, Exactitude	Effectuer une analyse technique approfondie avant le début du projet pour anticiper les défis possibles. Former l'équipe sur les technologies nécessaires.

R3 - Changement des exigences du client				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
9	Si le client modifie fréquemment les exigences du projet, cela peut entraîner des retards, des coûts supplémentaires et des conflits avec la portée initiale du projet.	E	Complétude, Adéquation	Établir un processus formel de gestion des modifications de portée. Documenter toutes les demandes de modification et évaluer leur impact sur le projet avant de les accepter.

R4 - Problèmes de sécurité des données				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Si des vulnérabilités de sécurité ne sont pas correctement gérées, les données sensibles des utilisateurs pourraient être compromises, ce qui aurait un impact négatif sur la réputation du projet.	C	Confidentialité, Intégrité	Mettre en place des protocoles de sécurité robustes, effectuer des audits de sécurité réguliers et se conformer aux réglementations en matière de protection des données.

R5 - Problèmes de compatibilité des navigateurs				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Les problèmes de compatibilité des navigateurs peuvent survenir si l'application web ne fonctionne pas correctement sur certains navigateurs, ce qui pourrait entraîner des inconvénients pour les utilisateurs	M	Adaptabilité	Effectuer des tests de compatibilité sur une gamme de navigateurs couramment utilisés et corriger les problèmes identifiés.

R6 - Perte de données accidentelle				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
4	Une perte de données accidentelle, telle qu'un fichier important supprimé par erreur, peut survenir et entraîner des retards mineurs dans le projet.	F	Résilience	Mettre en place des procédures de sauvegarde régulières et sensibiliser l'équipe à l'importance de la gestion des données.

R7 - Problèmes de communication interne				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
3	Des problèmes de communication interne au sein de l'équipe peuvent entraîner des malentendus, des retards mineurs et une coordination inefficace.	F	Adéquation, Exactitude	Établir des canaux de communication clairs au sein de l'équipe, organiser des réunions régulières de suivi et encourager une communication ouverte et transparente.

4.2.4 Gestion de configuration

Dans cette section, nous décrivons le processus de soumission, de révision et d'élimination des problèmes et des changements. Nous verrons également comment les artéfacts du projet ou du produit seront nommés, marqués et numérotés.

4.2.4.1 Soumission des problèmes et des changements

- **GitLab Issues :** Les problèmes, y compris les rapports de bogues, les demandes de fonctionnalités, les demandes de changement et les demandes d'amélioration, seront soumis à l'aide de la fonction "Issues" de GitLab. Les membres de l'équipe auront accès à la création et au suivi des problèmes dans un dépôt centralisé.
- **Google Drive :** Les artéfacts du projet, tels que les spécifications des requis du système (SRS), le

plan de développement du logiciel et d'autres documents, seront stockés et gérés dans Google Drive. Les parties prenantes et les membres de l'équipe peuvent utiliser cet outil pour soumettre et collaborer sur des problèmes non liés au code.

4.2.4.2 Processus de revue

- **Attribution des problèmes ("Issues"):** Les problèmes seront attribués aux membres de l'équipe concernés en fonction de leur expertise et de leurs responsabilités. Les personnes assignées seront responsables de la revue et de la résolution des problèmes.
- **Demandes de fusion ("Merge Request"):** Pour les changements liés au code, le mécanisme de "Merge Request" fourni par GitLab sera utilisé. Les modifications du code seront examinées par des pairs à l'aide de demandes de fusion avant d'être intégrées dans la base de code principale.
- **Revues par les pairs :** Des revues par les pairs seront effectuées pour les modifications de code ou de documents. Les réviseurs évalueront les changements proposés, fourniront une rétroaction et s'assureront qu'ils sont conformes aux objectifs du projet et aux normes de qualité.

4.2.4.3 Disposition des problèmes et des changements

- **Statut du problème :** Les problèmes ont trois statuts possibles : "Ouvert", "En cours" et "Fermé". L'état "Ouvert" signifie que le problème est actif et qu'il nécessite une attention particulière. L'état "En cours" signifie que le problème est en cours de résolution tandis que l'état "Fermé" indique que le problème a été résolu ou traité avec succès.
- **Approbation des modifications :** Les changements, y compris les changements de code et les modifications des artéfacts du projet, seront approuvés par les parties prenantes ou les membres de l'équipe concernés avant d'être implémentés. Le processus d'approbation garantit que les changements sont conformes aux objectifs et aux exigences du projet.

4.2.4.4 Nommage, marquage et numérotation des artéfacts

- **Nommage des artéfacts :** Les artéfacts du projet ou du produit seront nommés à l'aide de noms clairs et descriptifs qui reflètent leur contenu et leur objectif. Des conventions de dénomination seront établies pour assurer la cohérence des noms des artéfacts.
- **Marquage des artéfacts :** Des étiquettes seront appliquées aux artéfacts afin de les classer en fonction de leur cycle de vie. À la fin d'un cycle de vie, une étiquette portant le nom de ce cycle sera créée. Par exemple, "Prototype", "bêta" et "Version finale" seront utilisés comme étiquettes pour indiquer le stade d'un artéfact.
- **Numérotation des artéfacts :** La numérotation des artéfacts suivra le format ***X.Y***, où ***X*** représente la version majeure correspondant au cycle de vie de l'application (par exemple, 1 pour Prototype, 2 pour bêta, 3 pour Version finale), et ***Y*** représente la version mineure au sein de ce cycle de vie. Par exemple, un artéfact au stade bêta peut avoir un numéro de version comme "2.1" pour indiquer la première version mineure dans le cycle de vie bêta.

4.2.4.5 Contrôle de version

GitLab servira de système central de contrôle des versions pour gérer les modifications apportées au code source. Chaque version de changement de code sera suivie, étiquetée et documentée dans GitLab.