Équipe 10 - LEGRAIN-HORAIRES

Visualisation et construction d'horaires d'infirmières
Plan de développement logiciel

Version 10.0

Historique des révisions

Date	Version	Description	Auteur
2023-01-26	1.0	Première version du plan de développement logiciel	Caroline Des Rochers Mark Bekhet
2023-02-02	2.0	Deuxième version du plan de développement logiciel -Mise à jour plan de projet (Section 4.1)	Caroline DesRochers Mark Bekhet
2023-02-12	3.0	Troisième version du plan de développement logiciel -Mise à jour suivi de projet et contrôle (Section 4.2) -Mise à jour de la traçabilité dans le calendrier du projet (Section 4.1.3)	Caroline Des Rochers Mark Bekhet
2023-02-16	4.0	Révision plan de développement logiciel	Caroline Des Rochers Mark Bekhet Kyrollos Bekhet Zoé Paradis Malgorzata Niczyporuk
2023-03-06	5.0	Cinquième version du plan de développement logiciel - Mise à jour plan de projet (Section 4.1)	Caroline Des Rochers
2023-03-07	6.0	Sixième version du plan de développement logiciel - Mise à jour plan de projet (Section 4.1)	Caroline Des Rochers
2023-03-20	7.0	Septième version du plan de développement logiciel - Mise à jour plan de projet (Section 4.1)	Caroline Des Rochers
2023-04-12	8.0	Huitième version du plan de développement logiciel - Mise à jour du plan de projet suite au corrections de mis session	Caroline Des Rochers
2023-04-14	9.0	Neuvième version du plan de développement logiciel - Mise à jour Mise à jour de la traçabilité du plan de projet (Section 4.1)	Caroline Des Rochers
2023-04-18	10.0	Dixième version du plan du plan de développement logiciel - Révision final du plan de développement logiciel	Caroline Des Rochers Mark Bekhet Kyrollos Bekhet Zoé Paradis Malgorzata Niczyporuk

Table des matières

Plan de développement logiciel

1. Introduction

Le document qui suit présente le plan de développement logiciel de l'application web de visualisation et construction d'horaires d'infirmières et est organisé de la façon suivante. La section 2 présente la vue d'ensemble du projet, soit le but, la portée et les objectifs du projet ainsi que les biens livrables attendus. L'organisation du projet sera décrite dans la section 3. Plus précisément, on y retrouve la structure d'organisation de l'équipe de projet, les interfaces externes et les responsabilités des membres de l'équipe de développement. Quant à elle, la section 4 décrit le processus de gestion du projet, soit le plan de projet et le suivi et contrôle de celui-ci. Le plan de projets fournit la planification des diverses phases, les objectifs à atteindre pour chaque itération ainsi qu'un calendrier du projet résument les phases et les itérations avec des dates de début/fin et des critères de réalisation pour chacune d'entre elles.

2. Vue d'ensemble du projet

2.1 But du projet, portée et objectifs

À travers plusieurs projets de recherche, un outil d'optimisation d'horaires d'infirmière a été implémenté par le groupe de recherche d'Antoine Legrain. Actuellement, le code C++ n'a aucune interface graphique. Par conséquent, l'objectif de ce projet est de créer une application web qui facilite l'utilisation du code C++ lors de démonstration dans les hôpitaux. Il s'agit d'une application web de type client-serveur qui permet à l'utilisateur d'entrer des contraintes liées au fonctionnement d'un hôpital ainsi que les préférences des infirmières afin de générer un horaire optimisé. En plus de pouvoir entrer les contraintes, les informations sur les infirmières et de visualiser les horaires, l'application doit permettre à l'utilisateur de demander une réoptimisation après avoir fixé certaines contraintes. Ainsi, l'ajout d'une interface graphique vise à faciliter l'interaction avec le code développé en C++ et du même coup être utilisé lors de démonstration.

2.2 Hypothèses et contraintes

2.2.2 Ressources humaines

La main-d'œuvre qui est disponible pour ce projet est limitée à 5 développeurs qui travaillent à temps partiel en moyenne 21 heures par semaine, ce qui revient 273 heures par personnes sur une période de 13 semaines. Ainsi 1365 heures seront allouées pour la réalisation du projet. L'équipe de développeurs est composée de 5 étudiants de quatrième année en génie logiciel. On émet l'hypothèse que ces 5 étudiants ont reçu la formation adéquate pour réaliser ce projet et occuper les diverses responsabilités nécessaires afin d'assurer le bon déroulement du projet. Ceci dit, les développeurs ont estimé qu'ils ont la capacité d'utiliser les connaissances et les principes appropriés pour identifier, formuler, analyser, résoudre et concevoir des solutions à des problèmes d'ingénierie. Bien qu'il soit déjà familier avec la majorité des technologies qui seront utilisées pour réaliser l'application web de visualisation et construction d'horaire d'infirmière, un certain temps d'adaptation sera nécessaire pour se familiariser avec le code C++ qui optimise les horaires d'infirmières qui a été fournies par le client. Ainsi, il est important de prendre en considération ce temps dans les heures allouées pour la réalisation du projet.

2.2.3 Équipement

L'équipement requis par chaque développeur pour le développement de l'application web de visualisation et construction d'horaire est un environnement de travail au choix qui inclut au minimum un ordinateur muni de l'application Docker et npm. De plus, les différents services utilisés pour la réalisation du projet doivent être gratuits et l'application doit être en mesure de rouler localement.

2.2.4 Échéancier

L'équipe de développeur a eu sa première rencontre avec son client dans la semaine du 9 janvier 2023 et doit lui présenter un prototype le 17 février, une version bêta de l'application le 4 avril 2023, puis le code source de l'application web de visualisation et de construction d'horaire d'infirmière le 18 avril 2023. Une rencontre hebdomadaire avec le client aura lieu tous les lundis afin de s'assurer de satisfaire les besoins du client et de le tenir au courant des avancements. On émet les hypothèses que les dates de début et de fin pour les différents lots de travail de l'échéancier sont respectées et que les changements d'exigences peuvent être ajoutés dans l'échéancier sans créer de conflits, afin de faire face aux ajustements continuels de la planification et des attentes. En plus de ces trois soumissions aux clients, l'équipe de développeurs aura à réaliser 5 livrables au cours des 13 semaines de développement. Ceux-ci seront soumis au superviseur de projet. Un suivi hebdomadaire avec le superviseur aura lieu tous les lundis afin de s'assurer de l'avancement du projet et d'aborder les obstacles rencontrés.

2.3 Biens livrables du projet

1er livrable : Remise initiale — 12 janvier 2023

Artefacts:
- Modélisation de notre processus logiciel
- Compte rendu de la première rencontre avec le client
- Description des responsabilités au sein de l'équipe
- Horaire hebdomadaire de travail collaboratif

2e livrable: SRS — 19 janvier 2023

Artefacts:
- Spécification des requis du système (SRS)

3e livrable: Plan de développement logiciel — 26 janvier 2023

Artefacts:
- Plan de développements logiciel

4e livrable: Mi-session — 17 Février 2023

Artefacts:
- Modélisation de notre processus logiciel (Mise à jour)
- Spécification des requis du système (SRS) (Mise à jour)
- Plan de développement logiciel (Mise à jour)
- Plan de tests

Code:
- Prototype de l'application web *Nurse Scheduler* — 20 février 2023

Présentation:
- Présentation orale devant client et superviseur — 20 février 2023

4e livrable: Bêta — 4 avril 2023

Code:
- Version bêta du produit final de l'application web de visualisation d'horaire d'infirmière

5e livrable: Final — 18 avril 2023

Artefacts:
- Modélisation de notre processus logiciel (Mise à jour)
- Spécification des requis du système (SRS) (Mise à jour)
- Plan de développement logiciel (Mise à jour)
- Plan de tests (Mise à jour)
- Résultats de tests
- Documentation usager

Code:
- Code source de l'application web de visualisation d'horaire d'infirmière

Présentation:
- Présentation orale devant client et superviseur — 19 avril 2023

3. Organisation du projet

3.1 Structure d'organisation

Nous sommes une équipe de 5 développeurs qui travaillent sous la supervision d'Olivier Gendreau. À titre de superviseur, Olivier est responsable du suivi du projet, de l'évaluation des artéfacts et du produit final. Le superviseur a accès à nos outils de gestion et une rencontre de suivi hebdomadaire avec lui a lieu tous les lundis.

Au niveau des ressources humaines, Dominique Arseneault s'assure du suivi et de l'évaluation de la dynamique et l'organisation du travail en équipe collaborative.

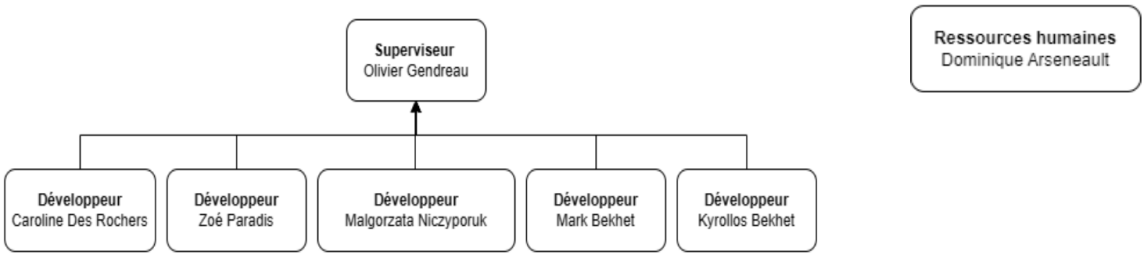

Figure 1: Structure organisationnelle

3.2 Interfaces externes

Dans le cadre de notre projet de visualisation et construction d'horaires externes, nous interagissons avec Antoine Legrain, professeur adjoint au département de mathématiques et de génie industriel et son étudiante au doctorat Flore Caye. Plus particulièrement, nous agissons à titre de consultants pour notre client (Antoine Legrain), afin de lui offrir une interface graphique qui satisfait ces attentes, sous forme d'application web pour son code actuel en C++ qui optimise des horaires. La communication avec le client et son étudiante se fait via courriel ainsi que la plateforme de communication collaborative Slack en plus d'une rencontre hebdomadaire en présentiel.

3.3 Responsabilités

En plus des divers rôles que nous allons occuper lors de la réalisation du projet, chacun des membres de l'équipe aura une responsabilité qui lui est propre. Ceci dit, nous n'aurons pas à faire l'entièreté des tâches reliées à cette responsabilité, mais plutôt veiller à ce que les tâches en lien avec cette responsabilité soient réalisées. Ainsi les 6 responsabilités au sein de l'équipe ont été réparties de la sorte:

Coordination/Planification

Zoé sera la personne-ressource au niveau de l'organisation de notre projet. Ceci dit, elle devra s'assurer que nous sommes organisés dans une optique de gestion de projet, soit en nous assurant qu'il y a une certaine coordination et planification qui est faite. À titre d'exemple, Zoé sera responsable à la fin des rencontres de s'assurer que nous avons effectué un plan des tâches que nous devons accomplir pour la prochaine rencontre et ainsi que la gestion du Jira.

Système (client)

Malgorzata sera responsable de la communication avec le client. Elle aura donc la responsabilité de s'assurer que le système soit bien défini et que celui-ci réponde aux attentes du client, afin de lui remettre le produit désiré. Tout au long de la session, le client va clarifier ces besoins et ces attentes, Malgorzata devra donc s'assurer que notre définition

du système et celle du client restent les mêmes. À titre d'exemple, elle devra s'assurer de faire des résumés des rencontres avec les membres de l'équipe qui n'ont pas assisté aux rencontres ainsi que de faciliter les communications entre l'équipe et le client.

Processus/Méthodologie
Caroline occupera la responsabilité Processus/Méthodologie. Plus précisément, elle sera responsable de s'assurer que le processus est adéquat à la réalité du projet, soit la visualisation et construction d'horaires d'infirmières. Ainsi, de s'assurer que la modélisation de notre processus logiciel avec *ProcessEdit* est suivie tout au long du projet.

Technique (outils)
Kyrollos occupera la responsabilité technique. Ainsi, il devra s'assurer qu'il y a eu une certaine réflexion derrière nos choix technologiques tout au long du projet. Il est important de mentionner que Kyrollos ne sera pas le seul à prendre les décisions techniques : choix des technologies / langages de programmation / API … mais bien celui qui s'assure qu'il y a eu un travail de recherche derrière les décisions techniques prises.

Produit (configuration)
Mark sera notre gestionnaire de configuration et ainsi s'assurer que le processus de configuration est adéquat et respecté par les autres membres de l'équipe. À titre d'exemple, il aura la tâche de s'assurer que des règles ont été établies concernant les "*Merge Request"* et la revue par les pairs en accord avec les autres membres de l'équipe ainsi que l'approche CI/CD.

Assurance qualité
Caroline aura la responsabilité de réfléchir à ce qu'est l'assurance qualité propre à notre projet et s'assurer d'identifier le type de test adéquat à réaliser. En d'autres mots, Caroline aura l'objectif de s'assurer que le produit final va au-delà des exigences fonctionnelles, en rappelant l'importance des requis non fonctionnels. Dans le but de s'assurer que notre produit ajoute de la valeur pour le client.

4. Processus de gestion

4.1 Plan de projet

4.1.1 Planification des phases

Le cycle de vie de notre produit comprend les quatre phases suivantes:

1. Identification des besoins / Requis
   → 9 janvier au 20 janvier 2023

   Au cours de la phase d'identification des besoins, l'identification et clarification des besoins de notre client, la définition des exigences du système, l'élaboration du processus logiciel et la définition des assignations des responsabilités de chaque membre de l'équipe doivent avoir lieu. Ainsi à la fin de cette phase, le SRS, modèle du processus logiciel et la description des responsabilités des 5 développeurs devront être livrée.

2. Élaboration de la solution
   → 21 janvier au 20 février 2023

   Au cours de la phase d'élaboration de la solution, la planification du projet, l'élaboration de l'architecture, la conception, l'élaboration de la stratégie de tests, le développement d'un prototype de solution et des suivis hebdomadaires avec le client et le superviseur doivent avoir lieu. Ainsi, à la fin de cette phase, la planification de projet, le document d'architecture logicielle, la planification de tests, le prototype de solution et la présentation orale devront être livrés.

3. Réalisation du produit
   → 20 février au 4 avril 2023

   Au cours de la phase de réalisation du produit, l'implémentation de la solution, l'exécution des tests logiciels, et des suivis hebdomadaires avec le client et le superviseur doivent avoir lieu. Ainsi, à la fin de cette phase un produit en version bêta de l'application devra être livré.

4. Terminaison du projet
   → 5 avril au 18 avril 2023

   Au cours de la phase de terminaison du projet, les tests d'acceptation, le transfert de l'application au client et l'évaluation du projet, et la clôture du projet doivent avoir lieu. Ainsi, à la fin de cette phase, le produit final, la mise à jour des livrables de la phase de définition du projet, des résultats de tests et la présentation orale finale devront être livrés.

4.1.2 Objectifs d'itération

Un projet est une séquence d'itération, le but ultime d'une itération est de gérer le risque. Ainsi, pour que nos itérations soient efficaces, il est important de bien définir les objectifs à atteindre pour chacune d'entre elles. Chacune des phases du cycle de vie de notre produit possède un nombre variable d'itérations ayant chacune leur propre objectif.

La phase d'identification des besoins possède 2 itérations:

Objectifs itération 1 (1 semaine)
→ 9 janvier au 12 janvier 2023

- Identification des besoins du client
- Établir les responsabilités au sein de l'équipe
- Élaboration du processus logiciel

Objectif itération 2 (1 semaine)
→ 13 janvier au 19 janvier 2023

- Clarification des besoins du client
- Premier jet du SRS

La phase d'élaboration de la solution possède 4 itérations:

Objectif itération 3 (1 semaine)
→ 20 janvier au 26 janvier 2023

- Plan de développement logiciel
- Premier jet de la maquette de l'interface utilisateur (Balsamiq)
- Fichier des données associé à une infirmière
- Liste des contraintes qu'on retrouve dans un contrat
- Mise en place du CI/CD sur GitLab

Objectif itération 4 (1 semaine)
→ 27 janvier au 2 février 2023

- Section 1 à 3 de la stratégie de tests (Planification des tests)
- Section 1 à 4 du document de l'architecture de la solution
- Schéma relationnel des contraintes d'un contrat
- Maquette de l'interface utilisateur (Balsamiq)
- Prototype :
    - Fichier .txt d'entré et de sortie
    - Back-end du prototype

Objectif itération 5 (1 semaine)
→ 3 février au 9 février 2023

- Section 4 et 5 de la stratégie de tests (Planification des tests)
- Section 5 à 7 du document d'architecture de la solution
- Prototype:
    - Création d'un formulaire basique de génération d'horaire
    - Visualisation d'un horaire simplifié
    - Communication entre le *front-end* et le *back-end*

Objectif itération 6 (1 semaine)
→ 10 février au 20 février 2023

- Mise à jour des artéfacts
- Maquette officielle (Figma)
- Prototype: Version simplifiée de la génération visualisation d'horaire
- Présentation orale

La phase de réalisation du produit possède 3 itérations:

Objectif itération 7 (2 semaines)
→ 21 février au 6 mars 2023

- Formulaires pour rentrer les contraintes
- Contrat (création du formulaire pour assembler les contraintes)
- Dossier infirmière
- Quart de travail
- Compétences
- Authentification
- Tests associés à ces fonctionnalités

Objectif itération 8 (2 semaines)
→ 7 mars au 20 mars 2023

- Groupe de contrat
- Formulaire de génération d'horaire
- Visualisation d'horaire
- Profil d'hôpital (Téléversement + téléchargement)
- Tests associés à ces fonctionnalités

Objectif itération 9 (2 semaines)
→ 21 mars au 4 avril 2023

- Reoptimisation d'horaire
- Historique
- Partage d'un horaire
- Tests associés à ces fonctionnalités
- Version Bêta de l'application web
- Tests utilisateur
- Validation avec le client

La phase de terminaison du projet possède 2 itérations:

Objectif itération 10 (1 semaine)
→ 5 avril au 11 avril 2023

- Rétroaction du client sur la version bêta
- Version finale de l'application
- Compléter les résultats des tests
- Compléter la documentation usager

Objectif itération 11 (1 semaines)
→ 12 avril au 18 avril 2023

- Présentation orale final

- Mise à jour des livrables de la phase de définition du projet
- Transfert de l'application final au client

4.1.3 Calendrier du projet

Phase 1: identification des besoins / Requis

It.	Date de début et de fin	Tâche	Effort estimé (h-p)	Traçabilité (ID SRS)
1 (40h)	9 janvier au 12 janvier 2023	Présentation du projet	5	n/a
		Première rencontre avec le client	5	
		Contre-rendu de la première rencontre avec le client	10	
		Élaboration du processus logiciel	10	
		Établir et définir les responsabilités au sein de l'équipe	10	
2 (70h)	13 janvier au 19 janvier 2023	Planification de l'itération	5	n/a
		Suivi avec le client	5	n/a
		Suivi avec le superviseur + débriefing	5	n/a
		SRS	50	n/a
		Retour sur l'itération	5	n/a

Phase 2 : Élaboration de la solution

It.	Date de début et de fin	Tâche	Effort estimé (h-p)	Traçabilité (ID SRS)
3 (100h)	20 janvier au 26 janvier 2023	Planification de l'itération	5	n/a
		Suivi avec le client	5	n/a

		Suivi avec le superviseur + débriefing	5	n/a
		Plan de développement logiciel	40	n/a
		Faire la liste des contraintes	15	n/a
		Mise en place du CI/CD	5	n/a
		Créer un gabarit du fichier des données qui seront envoyés de l'application web au code C++	10	n/a
		Premier jet de la maquette de l'interface utilisateur	10	n/a
		Retour sur l'itération	5	n/a
4 (100h)	27 janvier au 2 février 2023	Planification de l'itération	5	n/a
		Suivi avec le client	5	n/a
		Suivi avec le superviseur + débriefing	5	n/a
		Architecture logicielle: - Introduction - Objectifs et contraintes architecturaux - Vue des cas d'utilisation - Vue logique	20	n/a
		Élaboration du plan de test: - Introduction - Exigences à tester - Stratégie de test	15	n/a
		Avancement de la maquette de l'interface utilisateur	30	n/a
		Commencer le prototype: - Création du docker compose - Création des *endpoints* pour le back-end du prototype - Création du serveur - Fichier .txt d'entrée et de sortie	10	n/a
		Validation des contraintes et format de fichier	5	n/a
		Retour sur l'itération	5	n/a

5 (100h)	3 février au 9 février 2023	Planification de l'itération	5	n/a
		Suivi avec le client	5	n/a
		Suivi avec le superviseur + débriefing	5	n/a
		Architecture logicielle: - Vue des processus - Vue de déploiement - Taille et performance	15	n/a
		Élaboration du plan de test: - Ressources - Jalons du projet	10	n/a
		Réalisation de la maquette officielle avec *Figma*	15	n/a
		Avancer le prototype: - Mise à jour des fichiers .txt d'entrées et de sorties - Visualisation d'un horaire simplifié - Création d'un formulaire basique de génération d'horaire - Assurer la communication entre le *front-end* et le *back-end*	40	3.6.1 3.5.2.1 3.5.2.2
		Retour sur l'itération	5	n/a
6 (110h)	10 février au 20 février 2023	Planification de l'itération	5	n/a
		Suivi avec le client	5	n/a
		Suivi avec le superviseur + débriefing	5	n/a
		Dernier ajustement sur le prototype + tester le prototype	30	n/a
		Mise à jour des artéfacts: - SRS - Processus logiciel - Plan de développement logiciel - Stratégie de tests - Architecture de la solution	30	n/a
		Présentation orale - PowerPoint - Démo - Maquette officielle (*Figma*)	30	n/a

		Retour sur l'itération	5	n/a

Phase 3 : Réalisation du produit

It.	Date de début et de fin	Tâche	Effort estimé (h-p)	Traçabilité (ID SRS)
7 (180h)	21 février au 6 mars 2023	Planification de l'itération	5	n/a
		Suivi avec le client	10	n/a
		Suivi avec le superviseur + débriefing	10	n/a
		Créer les différents formulaires des contraintes: - *Complete Weekend* - *Identical Shift Types During Weekend* - *Minimum and Maximum Number of Consecutive Type* - *Minimum and Maximum Number of weekends in Four Weeks* - *Minimum and Maximum of Number of Assignments in Four Weeks* - *Minimum and Maximum of working hours in Four Weeks* - *Minimum and Maximum of Consecutive Working Weekends* - *Number Of Free Days After Shift* - *Unwanted Patterns* - *Unwanted Shift* - *Unwanted Skill*	50	3.2.3 3.2.4
		Contrat - Création du formulaire pour assembler les contraintes	20	3.2.1 à 3.2.4
		Gestion de l'authentification et gestion de comptes	20	3.1
		Créer le formulaire d'un dossier d'infirmière + logique de création	15	3.3.1
		Créer le formulaire de définition de quart de travail et type de quart de travail + logique	15	3.4.1 3.4.2 3.4.4 à 3.4.9
		Tests unitaires des fonctionnalités réalisés pendant l'itération (côté *back-end*) Tests associés aux exigences réalisés pendant l'itération, basée sur les tests établis dans le PTL	30	4.4

		Retour sur l'itération	5	n/a
8 (200h)	7 mars au 20 mars 2023	Planification de l'itération	5	n/a
		Suivi avec le client	10	n/a
		Suivi avec le superviseur + débriefing	10	n/a
		Intégration des commentaires du client - Groupe d'infirmière - Groupe de quart de travail - Définition de compétence - Revue du CSS - Revue de la logique des contraintes	20	3.3.4 3.4.3 3.4.10 à 3.4.12 3.14
		Infirmière - Annuler/Modifier/Enregistrer	10	3.3.2 3.3.3 3.3.7
		Groupe d'infirmière - Annuler/Modifier/Enregistrer	10	3.3.5 3.3.6 3.3.8
		Contrat - Logiques associés au contrat (Annuler/Modifier/Enregistrer) - Création de groupe de contrat	20	3.2.7 à 3.2.10
		Génération d'horaire - Préférences infirmières - Demande de l'hôpital	30	3.5.2
		Création d'un profil - Duplication profil - Partage profil - Suppression profil	30	3.13
		Téléchargement: - Téléchargement d'un profil - Problème	15	3.11.3 3.11.4 3.11.6
		Téléversement - Téléversement d'un profil	15	3.12
		Tests unitaires des fonctionnalités réalisés pendant l'itération (côté *back-end*) Tests associés aux exigences réalisés pendant l'itération, basée sur les tests établis dans le PTL	20	4.4
		Retour sur l'itération	5	n/a

9 (260h)	21 mars au 4 avril 2023	Planification de l'itération	5	n/a
		Suivi avec le client	10	n/a
		Suivi avec le superviseur + débriefing	10	n/a
		Logique de la génération d'horaire - Annuler/Modifier/Supprimer/Enregistrer	20	3.5.3 à 3.5.6
		Gestion asynchrone de l'optimisation d'horaire	10	3.10
		Visualisation d'horaire: - Consulter l'horaire en temps réel - Gestion de la visualisation dynamique des horaires - Notification	30	3.6
		Historique: - Génération d'horaires - Gestion des version d'horaires - Gestion demande de réoptimisation	30	3.7
		Téléchargement: - Exportation des horaires	5	3.11.1
		Intégration des commentaires du client - Création des groupes de contrat - Revue du CSS	20	3.2.10
		Réoptimisation d'horaire - Soumettre demande de réoptimisation d'horaire d'horaire - Préférence d'un horaire	20	3.8
		Tests unitaires des fonctionnalités réalisés pendant l'itération (côté *back-end*) Tests associés aux exigences réalisés pendant l'itération, basé sur les tests établis dans le PTL	20	4.4
		Tests des exigences non-fonctionnelles : sécurité, performance, fiabilité Tests utilisateurs de la version Bêta (utilisabilité)	35	n/a
		Tampon: Temps prévu pour faire face aux imprévue et s'ajuster	20	n/a
		Livraison de la version Bêta	20	n/a
		Retour sur l'itération	5	n/a

Phase 4 : Terminaison du projet

It.	Date de début et de fin	Tâche	Effort estimé (h-p)	Traçabilité (ID SRS)
11 (110h)	5 avril au 11 avril 2023	Planification de l'itération	5	n/a
		Suivi avec le client pour l'évaluation de la version bêta de l'application	5	n/a
		Correction de l'application suite aux rétroactions du client - Historique des infirmières - Statistiques - Rapport d'erreur de génération d'horaire - Revue du CSS	20	3.5.2.9 3.6.4 3.11.5
		Suivi avec le superviseur + débriefing	5	n/a
		Résultats des tests	30	n/a
		Création de la documentation usager	30	4.7
		Tests unitaires des fonctionnalités réalisés pendant l'itération (côté *back-end*) Tests associés aux exigences réalisés pendant l'itération, basé sur les tests établis dans le PTL	10	4.4
		Retour sur l'itération	5	n/a
12 (100h)	12 avril au 18 avril 2023	Planification de l'itération	5	n/a
		Suivi avec le client	5	n/a
		Suivi avec le superviseur + débriefing	5	n/a
		Mise à jour de la documentation usager	20	4.7
		Préparation de la présentation orale finale	20	n/a
		Rencontre pour transférer l'application finale au client et la documentation usager	10	n/a
		Tampon (Ajustement imprévu)	30	n/a

		Retour sur l'itération	5	n/a

4.2 Suivi de projet et contrôle

4.2.1 Gestion des exigences

Dans le cadre de notre projet, l'objectif ultime est de réaliser un produit qui répond parfaitement au besoin de notre client. Pour s'assurer de satisfaire les attentes de notre client, notre projet sera divisé en itération et à la fin de chacune d'entre elles une rencontre de validation avec le client aura lieu. Ces rencontres permettront à l'équipe de développeurs d'avoir un retour direct de la part du client et s'assurer que le travail effectué correspond bien à ses besoins qui risquent d'évoluer au fil du projet, tout en s'assurant de rester dans les périmètres du projet établis en début de projet. Ainsi, l'équipe de développement a mis en place des mécanismes pour faire face au changement des exigences. Tout d'abord, si un changement dans les exigences est nécessaire, une évaluation de la charge de travail supplémentaire engendrée par ce changement sera réalisée en équipe. Dans le cas où le temps alloué n'est pas suffisant, une rencontre avec le client aura lieu afin de l'aviser de la situation et du même coup gérer ces attentes. Dans le cas où le temps le permet, une rencontre d'équipe aura lieu afin de bien planifier et anticiper les impacts de ce changement. Ensuite, la liste des exigences et le SRS seront modifiés. Puis, le calendrier du projet ainsi que les différentes tâches sur Jira seront revus et adaptés en conséquence. Si nécessaire, le document d'architecture et la stratégie de test seront mis à jour.

4.2.2 Contrôle de la qualité

Afin de s'assurer de fournir un produit de qualité à notre client, des méthodes ont été mises en place par l'équipe de développeurs pour contrôler la qualité des biens livrables du projet ainsi qu'un processus pour entreprendre des actions correctives. Au niveau des artéfacts, avant la remise de chacun d'eux à notre superviseur ou bien à notre client, une revue en équipe a lieu afin de s'assurer de la qualité du contenu et le logiciel Antidote est utilisé pour s'assurer de la qualité de la langue. Au niveau du code écrit, une fois une fonctionnalité terminée, des tests doivent être réalisés, afin de s'assurer que celles-ci répondent aux exigences qu'on retrouve dans le SRS. De plus, une revue de code par les pairs doit être faite avant d'ajouter cette nouvelle fonctionnalité sur la branche "dev". Ainsi, toute modification ou intégration de code doit passer par un *merge request*. Au moins deux autres membres de l'équipe de développeurs autre que le propriétaire du ''merge'' doivent l'accepter. Si jamais, le développeur qui effectue la revue de code identifie des erreurs et juge que des corrections doivent être effectuées, il doit laisser un commentaire clair et précis de la correction qu'il veut adopter et créer un ticket sur Jira. Ce ticket doit être assigné au développeur qui à créer le code et avoir une description claire et pertinente.

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.

1 - Mauvaise de la vision du client				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
3	Mauvaise compréhension de la vision du client et identification des besoins. Ainsi le produit livré ne répond pas aux attentes du client.	C	Complétude Exactitude	Faire des rencontres de suivi hebdomadaire avec le client afin de gérer les attentes du client et de recevoir de la rétroaction de manière continue. Création d'une maquette afin d'établir le flux de travail de l'application. Adopter la méthode échoue tôt!

2 - Risque de débordement du nombre de fonctionnalités				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
4	Difficultés à définir le périmètre (scope) du projet qui peut être lié à une mauvaise vision du client. Mauvaise ressource humaine et de temps.	C	Complétude	Analyser en amont l'ensemble des besoins du client et établir à l'avance les fonctionnalités de base qui devront être prises en compte dans le développement afin de limiter l'ajout de nouvelles fonctionnalités en cours de projet et du même coup limiter les délais incontrôlés. Développer l'application de manière modulaire.

3 -Mauvaise Intégration avec le code C++ du client				
Ampleur	**Description**	**Impact**	**Facteurs**	**Stratégie de gestion**
5	Mauvais transfert des données entre l'application web et le code c++ d'optimisation d'horaire d'infirmière qui nous a été fourni par le client.	M	Fiabilité	Tester la communication entre le code C++ et l'application dans les premières itérations la phase de réalisation du produit. Valider les fichiers d'entrées et de sorties avec le client dès le prototype.

4.2.4 Gestion de configuration

Dès qu'un changement doit avoir lieu dans le développement du produit ou bien qu'un problème a été identifié, un ticket doit être créé sur Jira. Celui-ci doit contenir toutes les informations pertinentes : description, niveau de gravité,

capture d'écran si besoin, la version, etc. et être assigné à un développeur de l'équipe. Le développeur qui a été assigné à la tâche doit créer une nouvelle branche sur GitLab pour travailler. En effet, GitLab est l'interface qui offre le système de gestion de version que l'équipe de développeurs a décidé d'utiliser. Une branche de fonctionnalité doit se nommer feature/nom-du-feature et une branche de correction de bogue doit se nommer hotfix/nom-du-bug. Une fois la correction effectuée, une demande de merge request est faite et une tâche de revue de code est créée et assignée à un autre développeur. Les différentes versions du produit seront nommées de la sorte : nom du projet suivi du numéro de version. Par exemple, "Nurse_Scheduling_V1", qui signe qu'on est à la version 1 de l'application Nurse Scheduling. Le numéro de version est incrémenté entre chaque itération.

La gestion des artéfacts a lieu à partir d'un dossier partagé sur Google Drive auquel tous les membres de l'équipe ont accès. Les artéfacts du projet seront nommés, de la manière suivante: numéro d'équipe de projet soit 10 suivis du nom du document et de la version. Par exemple, Eq10_PDL_V1, qui signe qu'on est à la version 1 du plan de développement logiciel. La version est mise à jour entre chaque soumission au superviseur.