Équipe 10 - LEGRAIN-HORAIRES

Visualisation et construction d’horaires d’infirmières

Spécifications des requis du système (SRS)

Version 7.0

Historique des révisions

Date	Version	Description	Auteur
2023-01-19	1.0	Première version du SRS	Caroline Des Rochers Mark Bekhet
2023-01-26	2.0	Deuxième version du SRS - Mise à jours des exigences fonctionnelles (Section 3)	Caroline Des Rochers
2023-02-12	3.0	Troisième version du SRS - Mise à jour des exigences fonctionnelles (Section 3) - Ajout annexe B (Contraintes)	Caroline Des Rochers Mark Bekhet Kirollos Bekhet
2023-02-16	4.0	Quatrième version du SRS -Révision finale du SRS avant la remise de mi-session	Caroline Des Rochers Mark Bekhet Kyrollos Bekhet Zoé Paradis Malgorzata Niczyporuk
2023-04-10	5.0	Cinquième version du SRS - Correction suite aux commentaires de la correction de mi-session	Caroline Des Rochers
2023-04-12	6.0	Sixième version du SRS - Mise à jour des exigences fonctionnelles suite aux ajouts de nouvelles fonctionnalités (Section 3) - Mise à jour de l'annexe A et B	Caroline Des Rochers
2023-04-18	7.0	Septième version du SRS - Révision final du SRS avant la remise de mi-session	Caroline Des Rochers Mark Bekhet Kyrollos Bekhet Zoé Paradis Malgorzata Niczyporuk

Table des matières

Spécifications des requis du système (SRS)

1. Introduction

1.1. But

Le SRS décrit le comportement externe d'une application. Il décrit aussi les exigences non fonctionnelles, les contraintes de conception, ainsi que les autres facteurs nécessaires à la description complète des exigences du logiciel à développer.

1.2. Définitions, acronymes et abréviations

La définition de tous les mots, acronymes et abréviations nécessaires à l'interprétation adéquate de ce SRS est fournie par référence à un glossaire dans l'annexe A et dans l'annexe B.

1.3. Vue d'ensemble du document

Dans les sections qui suivent, on retrouvera tout d'abord une description globale du produit, soit une application web permettant la visualisation et la construction d'horaires d'infirmières. Dans la section 2, nous définissons les caractéristiques des usagers ainsi que les différentes interfaces. Puis, les sections 3 et 4 sont consacrées à la description des exigences fonctionnelles et non fonctionnelles du logiciel.

1.4. Références

Ce document fait référence au document suivant:

- Projet 10- Polytechnique-Legrain-Horaires-Document de vision

2. Description globale

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.

2.1. Caractéristiques des usagers

Le logiciel développé dans le cadre de ce projet sera utilisé lors de démonstration dans les hôpitaux. Ainsi, nos usagers seront autant les fonctionnaires présents lors des démonstrations que les agents administratifs qui testeront le logiciel. Il s'agit d'usagers âgés de 20 à 60 ans n'ayant pas nécessairement de connaissance poussée en informatique. Le milieu hospitalier est bilingue avec une majorité anglophone, l'application est développée en anglais. Au niveau matériel, les utilisateurs devront posséder un ordinateur.

2.2. Interfaces

2.2.1. Interfaces usagers

L'interface graphique web qui sera développée devra être simple et facile d'utilisation. Elle devra accepter des entrées génériques pour définir les problèmes. Le langage visuel de l'application devra respecter les critères ergonomiques de Bastien et Scapin soit le guidage, la charge de travail, le contrôle explicite, l'adaptabilité, la gestion des erreurs, l'homogénéité cohérence, la signifiance des codes et dénominations et la comptabilité. De plus, l'ensemble des thèmes de couleur devront faire ressortit les fonctionnalités importantes et guider l'utilisation du logiciel . Finalement, les utilisateurs ne nécessitent aucune connaissance avancée d'informatique afin d'utiliser le logiciel.

2.2.2. Interfaces matérielles

Pour utiliser l'application web, les interfaces matérielles seront les périphériques standards d'un ordinateur (souris, clavier, etc.). L'application doit rouler localement sur n'importe quels ordinateurs et serveurs possédant au moins 8 Go de mémoire vive. Pour démarrer le serveur, ce dernier devra être accessible au minimum par une ligne de commande.

2.2.3. Interfaces logicielles

Au cours de ce projet, une interface graphique sera développée. Il s'agit d'une application web qui doit être supportée par les navigateurs modernes comme Chrome, Microsoft Edge ou Mozilla Firefox. L'application web sera codée en Typescript avec le cadriciel Angular. En plus de l'interface web, un serveur va être développé pour faire le traitement et la génération des horaires des infirmières. Le serveur va utiliser le code C++ fourni par le client pour générer les horaires des infirmières. Il est important de noter qu'une des exigences du client est que l'application, interface graphique et serveur, doit rouler en local en utilisant n'importe quel ordinateur/serveur. Par conséquent, le serveur va être un serveur Flask développé en Python en raison de la simplicité de son utilisation et sa compatibilité avec tous les systèmes d'exploitation comme Windows, Linux et Mac. Des conteneurs Docker vont être utilisés pour faire le déploiement de l'application web et du serveur en local afin de pouvoir automatiquement recommencer l'application en cas d'erreur fatale ainsi que de bien isoler les applications du reste du système.

2.2.4. Interfaces de communication

Le serveur va être développé en Python. Ce choix technologique est basé sur la simplicité du langage à communiquer avec le code C++. En effet, il existe plusieurs librairies en Python qui permettent de faire le lien entre du code Python et du code C++ comme boost, pyBind11 et autres librairies. En plus, l'utilisation de Python comme langage de programmation va nous permettre de partir un serveur rapidement puisque notre serveur sera un serveur Flask. Les dépendances qui vont être utilisées par le serveur sont incluses dans le gestionnaire de paquets Pip. Le serveur va être déployé localement à l'intérieur d'un conteneur Docker afin d'offrir plus de sécurité en isolant le serveur du reste des applications de l'ordinateur. Aussi, Docker va nous allouer une plus grande simplicité concernant l'exposition des ports. Enfin, le conteneur Docker va nous permettre de redémarrer automatiquement le serveur en cas d'une erreur fatale.

2.3. Contraintes générales

Plusieurs contraintes sont à prendre en compte dans le développement de l'application web de visualisation et construction d'horaires d'infirmières, afin de s'assurer de son bon fonctionnement et de la satisfaction de notre client. Tout d'abord, un code C++ qui optimise les horaires d'infirmières, mais qui n'a aucune interface graphique nous a été fourni par le client. Nous devons donc le prendre en compte lors de nos décisions d'architecture et technologiques. De plus, l'application doit fonctionner localement, être en mesure de lancer plusieurs optimisations simultanément en arrière-plan. Le serveur doit donc assurer le support de plusieurs requêtes d'optimisation de façon simultanée. Il doit également assurer la sauvegarde des différents profils d'hôpital, ainsi que les différents horaires générés auparavant. La persistance des données est également importante et celle-ci sera assurée par des volumes qui vont être sauvegardés localement et qui vont être accessibles par les différents conteneurs Docker. L'application sera utilisée comme prototype pour les différents hôpitaux, la qualité visuelle de l'interface web est donc importante pour assurer une bonne expérience utilisateur. Celle-ci doit également être facile d'utilisation, puisqu'elle vise à simplifier la tâche de ces utilisateurs.

2.4. Hypothèses et dépendances

Nous supposons que l'utilisateur a accès à un ordinateur avec un navigateur moderne. Nous supposons également que l'utilisateur a l'engin Docker déjà installé sur son ordinateur pour rouler l'application durant la démonstration.

3. Exigences fonctionnelles

La section suivante présente l'ensemble des exigences fonctionnelles supportées sur l'application client et serveur. Celles-ci sont séparées en groupe de fonctionnalités avec leurs types. De plus, il est à noter que toutes les exigences qui suivent sont essentielles à moins d'indication contrainte.

3.1 Authentification [Essentiel]

3.1.1 Le système doit permettre à l'utilisateur de s'authentifier à l'application en entrant son nom d'utilisateur et son mot de passe.

3.1.2 Le système doit permettre à l'administrateur de créer un compte.

3.1.2.1 Le système doit permettre à l'administrateur de fournir un nom d'utilisateur lors de la création d'un compte.

3.1.2.2 Le système doit permettre à l'administrateur de fournir un mot de passe lors de la création d'un compte.

3.2 Contrat [Essentiel]

3.2.1 Le système doit permettre à l'utilisateur de créer un contrat.

3.2.2 Le système doit permettre à l'utilisateur de donner un nom à un contrat.

3.2.3[1] Le système doit permettre à l'utilisateur d'ajouter une liste de contraintes dans le contrat.

3.2.3.1 Le système doit permettre à l'utilisateur de spécifier la contrainte *Identical Shift Types During Weekend.*

3.2.3.2 Le système doit permettre à l'utilisateur de spécifier la contrainte *Complete Weekends.*

~~3.2.3.3 Le système doit permettre à l'utilisateur de spécifier la contrainte *MaxNumAssignmentsInFourWeeks.*~~

~~3.2.3.4 Le système doit permettre à l'utilisateur de spécifier la contrainte *MinNumAssignmentsInFourWeeks.*~~

~~3.2.3.5 Le système doit permettre à l'utilisateur de spécifier la contrainte *AlternativeShift.*~~

~~3.2.3.6 Le système doit permettre à l'utilisateur de spécifier la contrainte *MaxConsecutiveWorkingWeekends.*~~

~~3.2.3.7 Le système doit permettre à l'utilisateur de spécifier la contrainte *MinConsecutiveWorkingWeekends.*~~

3.2.3.8 Le système doit permettre à l'utilisateur de spécifier la contrainte *Unwanted Patterns.*

3.2.3.8.1 Le système doit permettre à l'utilisateur de spécifier un *Pattern.*.

~~3.2.3.9 Le système doit permettre à l'utilisateur de spécifier la contrainte *TotalWeekendsInFourWeeks.*~~

[1] Pour la définition des différentes contraintes, consultez Annexe B: Définition des contraintes

3.2.3.10 Le système doit permettre à l'utilisateur de spécifier la contrainte *Number Of Free Days After Shift.*

~~3.2.3.11 Le système doit permettre à l'utilisateur de spécifier la contrainte *MaxConsecutiveShiftType.*~~

~~3.2.3.12 Le système doit permettre à l'utilisateur de spécifier la contrainte *MinConsecutiveShiftType.*~~

3.2.3.13 Le système doit permettre à l'utilisateur de spécifier la contrainte *Unwanted Skills.*

3.2.3.14 Le système doit permettre à l'utilisateur de spécifier la contrainte *Unwanted Shift.*

3.2.3.15 Le système doit permettre à l'utilisateur de spécifier la contrainte *Minimum and Maximum Number of Weekends in Four Weeks.*

3.2.3.16 Le système doit permettre à l'utilisateur de spécifier la contrainte *Minimum and Maximum Number of Consecutive Type.*

3.2.3.17 Le système doit permettre à l'utilisateur de spécifier la contrainte *Minimum and Maximum of Consecutive Working Weekends.*

3.2.3.18 Le système doit permettre à l'utilisateur de spécifier la contrainte *Minimum and Maximum of Number of Assignments in Four Weeks.*

3.2.3.19 Le système doit permettre à l'utilisateur de spécifier la contrainte *Minimum and Maximum of working hours in Four Weeks.*

3.2.4 Le système doit permettre à l'utilisateur de spécifier le poids d'une contrainte.

~~3.2.5. Le système doit permettre à l'utilisateur d'ajouter une compétence au contrat.~~

~~3.2.6 Le système doit permettre à l'utilisateur de téléverser une liste de contraintes.~~

3.2.7 Le système doit permettre à l'utilisateur de modifier un contrat.

3.2.8 Le système doit permettre à l'utilisateur de supprimer un contrat.

3.2.9 Le système doit permettre à l'utilisateur d'enregistrer un contrat.

3.2.10 Le système doit permettre à l'utilisateur de créer un groupe de contrat.

3.2.10.1 Le système doit permettre à l'utilisateur de donner un nom au groupe de contrat.

3.2.10.2 Le système doit permettre à l'utilisateur d'ajouter un ensemble de contrat dans le groupe.

3.2.11 Le système doit permettre à l'utilisateur de modifier un groupe de contrat.

3.2.12 Le système doit permettre à l'utilisateur de supprimer un groupe de contrat.

3.2.13 Le système doit permettre à l'utilisateur d'enregistrer un groupe de contrat.

3.3 Infirmière [Essentiel]

3.3.1 Le système doit permettre à l'utilisateur de définir une infirmière.

3.3.1.1 Le système doit permettre à l'utilisateur d'inscrire l'identifiant unique de l'infirmière.

3.3.1.2 Le système doit permettre à l'utilisateur d'inscrire le nom de l'infirmière.

3.3.1.3 Le système doit permettre à l'utilisateur d'inscrire le prénom de l'infirmière.

~~3.3.1.4 Le système doit permettre à l'utilisateur d'ajouter l'ensemble des compétences de l'infirmière.~~

3.3.1.5 Le système doit permettre à l'utilisateur de sélectionner l'ensemble des contrats de l'infirmière.

3.3.1.6 Le système doit permettre à l'utilisateur de sélectionner l'ensemble des groupes de contrat de l'infirmière.

3.3.2 Le système doit permettre à l'utilisateur de modifier une infirmière.

3.3.3 Le système doit permettre à l'utilisateur de supprimer une infirmière.

3.3.4 Le système doit permettre à l'utilisateur de créer un groupe d'infirmières.

3.3.4.1 Le système doit permettre à l'utilisateur de donner un nom au groupe d'infirmières.

3.3.4.2 Le système doit permettre à l'utilisateur d'ajouter un ensemble d'infirmières dans le groupe.

3.3.4.3 Le système doit permettre à l'utilisateur de sélectionner l'ensemble des contrats du groupe d'infirmières.

3.3.4.4 Le système doit permettre à l'utilisateur de sélectionner l'ensemble des groupes de contrat du groupe d'infirmières.

3.3.5 Le système doit permettre à l'utilisateur de modifier un groupe d'infirmières.

3.3.6 Le système doit permettre à l'utilisateur de supprimer un groupe d'infirmières.

3.3.7 Le système doit permettre à l'utilisateur d'enregistrer une infirmière.

3.3.8 Le système doit permettre à l'utilisateur d'enregistrer un groupe d' infirmières.

3.4 Quart de travail [Essentiel]

3.4.1 Le système doit permettre à l'utilisateur de définir un quart de travail.

3.4.1.1 Le système doit permettre à l'utilisateur de donner un nom au quart de travail.

3.4.1.2 Le système doit permettre à l'utilisateur d'inscrire l'heure de début du quart de travail.

3.4.1.3 Le système doit permettre à l’utilisateur d’inscrire l’heure de fin du quart de travail.

3.4.2 Le système doit permettre à l’utilisateur de spécifier un type de quart de travail.

3.4.2.1 Le système doit permettre à l’utilisateur de donner un nom au type de quart de travail.

3.4.2.2 Le système doit permettre à l’utilisateur d’ajouter un semble de quart de travail dans le type de quart de travail.

3.4.3 Le système doit permettre à l’utilisateur de créer un groupe de quart de travail.

3.4.3.1 Le système doit permettre à l’utilisateur de donner un nom au groupe de quart de travail.

3.4.3.2 Le système doit permettre à l’utilisateur d’ajouter un semble de quart de travail dans le groupe de quart de travail.

3.4.3.3 Le système doit permettre à l’utilisateur d’ajouter un semble de type de quart de travail dans le groupe de quart de travail.

3.4.4 Le système doit permettre à l'utilisateur de modifier un quart de travail.

3.4.5 Le système doit permettre à l’utilisateur de supprimer un quart de travail.

3.4.6 Le système doit permettre à l’utilisateur d’enregistrer un quart de travail.

3.4.7 Le système doit permettre à l’utilisateur de modifier un type de quart de travail.

3.4.8 Le système doit permettre à l’utilisateur de supprimer un type de quart de travail.

3.4.9 Le système doit permettre à l’utilisateur d’enregistrer un type de quart de travail.

3.4.10 Le système doit permettre à l’utilisateur de modifier un groupe de quart de travail.

3.4.11 Le système doit permettre à l’utilisateur de supprimer un groupe de quart de travail.

3.4.12 Le système doit permettre à l’utilisateur d’enregistrer un groupe de quart de travail.

3.5 Génération d’horaire [Essentiel]

3.5.1 Le système doit permettre à l’utilisateur de générer un horaire.

3.5.2 Le système doit permettre à l’utilisateur de définir le problème d’un horaire à générer.

3.5.2.1 Le système doit permettre à l’utilisateur de sélectionner l’horizon de planification de l’horaire à générer.

3.5.2.2 Le système doit permettre à l’utilisateur d’ajouter un ensemble d'infirmières à inclure dans l’horizon de planification de l’horaire à générer.

~~3.5.2.3 Le système doit permettre à l'utilisateur de téléverser une liste d'infirmières à inclure dans l'horizon de planification de l'horaire à générer. [Souhaitable]~~

3.5.2.4 Le système doit permettre à l'utilisateur de spécifier l'ensemble des préférences d'une infirmière.

3.5.2.4.1 Le système doit permettre à l'utilisateur de spécifier le poids d'une préférence.

3.5.2.5 Le système doit permettre à l'utilisateur de spécifier la demande de l'hôpital.

3.5.2.5.1 Le système doit permettre à l'utilisateur de spécifier le poids d'une demande de l'hôpital.

~~3.5.2.6 Le système doit permettre à l'utilisateur de définir une fin de semaine.~~

3.5.2.7 Le système doit permettre à l'utilisateur de sélectionner la liste des quarts de travail à inclure dans l'horizon de planification de l'horaire à générer.

3.5.2.8 Le système doit permettre à l'utilisateur de sélectionner la liste des compétences à inclure dans l'horizon de planification de l'horaire à générer.

3.5.2.9 Le système doit permettre à l'utilisateur de définir l'historique de travail d'une infirmière.

3.5.3 Le système doit permettre à l'utilisateur d'annuler la définition du problème de l'horaire à générer.

3.5.4 Le système doit permettre à l'utilisateur de modifier la définition d'un problème.

3.5.5 Le système doit permettre à l'utilisateur d'enregistrer la définition d'un problème.

3.5.6 Le système doit permettre à l'utilisateur de supprimer la définition d'un problème

3.6 Visualisation d'horaire [Essentiel]

3.6.1 Le système doit permettre à l'utilisateur de visualiser un horaire.

~~3.6.2 Le système doit permettre à l'utilisateur de filtrer l'affichage d'un horaire.~~

~~3.6.2.1 Le système doit permettre à l'utilisateur de filtrer l'affichage par infirmière.~~

~~3.6.2.2 Le système doit permettre à l'utilisateur de filtrer l'affichage par quart de travail [souhaitable]~~

~~3.6.2.3 Le système doit permettre à l'utilisateur de filtrer l'affichage par compétence. [souhaitable]~~

3.6.3 Le système doit permettre à l'utilisateur d'être notifié lorsque nouvel horaire associé à ces demandes de génération d'horaire vient d'être complété.

3.6.4 Le système doit permettre à l'utilisateur de consulter l'ensemble des statistiques associé à l'horaire généré. [souhaitable]

3.7 Historique [Essentiel]

3.7.1 Le système doit permettre à l'utilisateur de consulter son historique de génération d'horaires.

3.7.2 Le système doit permettre à l'utilisateur de revenir à une version antérieure d'un horaire généré.

3.7.3 Le système doit permettre à l'utilisateur de consulter l'historique d'édition d'un horaire. [Souhaitable]

3.7.4 Le système doit permettre à l'utilisateur de consulter son historique de création de problèmes. [Souhaitable]

~~3.7.5 Le système doit permettre à l'utilisateur de consulter son historique de création de contrats. [Souhaitable]~~

3.8 Réoptimisation d'horaire [Essentiel]

3.8.1 Le système doit permettre à l'utilisateur d'éditer la demande d'un horaire qu'il a déjà généré.

3.8.2 Le système doit permettre à l'utilisateur de demander une optimisation d'horaire après une édition.

~~3.8.2.1 Le système doit permettre à l'utilisateur de fixer une entrée dans l'horaire afin de faire une optimisation.~~

3.8.3 Le système doit permettre à l'utilisateur de fixer des préférences sur un horaire qu'il a déjà généré avant de le réoptimiser.

~~3.9 Partage d'horaire [Essentiel]~~

~~3.9.1 Le système doit permettre à l'utilisateur de partager un horaire avec une sélection d'utilisateurs du système.~~

~~3.9.2 Le système doit permettre à l'utilisateur de consulter un horaire qui lui a été partagé par un autre utilisateur du système.~~

3.10 Gestion asynchrone [Essentiel]

3.10.1 Le système doit permettre à l'utilisateur d'utiliser l'ensemble des fonctionnalités lorsqu'un horaire est en train d'être généré en arrière-plan.

3.11 Téléchargement [Essentiel]

3.11.1 Le système doit permettre à l'utilisateur d'enregistrer un horaire localement.

~~3.11.2 Le système doit permettre à l'utilisateur d'enregistrer un contrat localement.~~

3.11.3 Le système doit permettre à l'utilisateur d'enregistrer la définition d'un problème localement.

3.11.4 Le système doit permettre à l'utilisateur d'enregistrer un profil d'hôpital localement.

3.11.5 Le système doit permettre à l'utilisateur d'enregistrer le rapport d'erreur d'un horaire généré qui a un statut d'échec localement.

3.11.6 Le système doit permettre à l'utilisateur d'enregistrer le gabarit d'un profil d'hôpital en format CSV localement.

3.12 Téléversement [Essentiel]

~~3.12.1 Le système doit permettre à l'utilisateur de téléverser la définition d'un problème.~~

~~3.12.2 Le système doit permettre à l'utilisateur de téléverser un contrat.~~

~~3.12.3 Le système doit permettre à l'utilisateur de téléverser le dossier d'une infirmière.~~

~~3.12.4 Le système doit permettre à l'utilisateur de téléverser la définition d'un quart de travail.~~

3.12.5 Le système doit permettre à l'utilisateur de téléverser le profil d'un hôpital.

3.13 Profil d'hôpital [Essentiel]

3.13.1 Le système doit permettre à l'utilisateur de créer un profil d'hôpital.

3.13.1.1 Le système doit permettre à l'utilisateur de donner un nom au profil.

3.13.2 Le système doit permettre à l'utilisateur de dupliquer un profil d'hôpital.

3.13.3 Le système doit permettre au propriétaire du profil de l'hôpital de le supprimer.

3.13.4 Le système doit permettre à l'utilisateur de partager un profil d'hôpital avec une sélection d'utilisateurs.

3.13.4.1 Le système doit permettre à l'utilisateur de partager un quart de travail.

3.13.4.2 Le système doit permettre à l'utilisateur de partager un groupe de quart de travail.

3.13.4.3 Le système doit permettre à l'utilisateur de partager un type de quart de travail.

3.13.4.4 Le système doit permettre à l'utilisateur de partager un dossier d'infirmière.

3.13.4.5 Le système doit permettre à l'utilisateur de partager un groupe d'infirmières.

3.13.4.6 Le système doit permettre à l'utilisateur de partager un contrat.

3.13.4.7 Le système doit permettre à l'utilisateur de partager un groupe contrat.

3.13.4.8 Le système doit permettre à l'utilisateur de partager un horaire.

3.13.4.9 Le système doit permettre à un utilisateur de partager un problème de génération d'horaire.

3.13.5 Le système doit permettre à l’utilisateur de consulter les profils d’hôpital partager avec lui.

3.13.6 Le système doit permettre à l’utilisateur ayant accès à un profil de pouvoir le modifier.

3.14 Compétences [Essentiel]

3.14.1 Le système doit permettre à l'utilisateur de définir une compétence.

3.14.2 Le système doit permettre à l'utilisateur de supprimer une compétence.

4. Exigences non-fonctionnelles

Cette section présente les exigences non-fonctionnelles du système liées à l'utilisabilité, la fiabilité, la performance, la sécurité et les exigences de la documentation usager en ligne et du système d'assistance. On retrouve également les contraintes de conception ainsi que les normes applicables au sein de l'équipe de développement.

4.1. Utilisabilité

4.1.1 Un utilisateur novice doit pouvoir utiliser l'application de manière productive après 10 minutes d'exploration.

4.1.2 Un utilisateur spécialisé doit pouvoir utiliser l'application de manière productive après 2 minutes d'exploration.

4.1.3 L'interface graphique de l'application doit suivre le critère d'homogénéité de Bastien et Scapin.

4.1.3.1 L'ensemble des fenêtres de l'application doivent suivre le même schéma d'agencement.

4.1.3.2 La sémantique des boutons de la souris doit être constante.

4.1.3.3 La position des informations doit être cohérente d'une fenêtre à l'autre.

4.1.3.4 Le vocabulaire doit être cohérent d'une fenêtre à l'autre.

4.1.3.5 Le format des données doit suivre le même format pour l'ensemble de l'application.

4.1.4 L'interface graphique de l'application doit suivre le critère de guidage de Bastien et Scapin.

4.1.4.1 Le titre d'une section doit apparaître en gras lorsque le curseur de l'utilisateur est sur celui-ci.

4.1.4.2 L'image d'une icône représentant une action opposée d'une autre icône doit être d'une couleur différente.

4.1.4.3 Un champ actif doit être rendu visible au moment de la saisie.

4.1.4.4 Une distinction visuelle doit être faite entre une section de fonction différente sur la page Web.

4.1.5 Un utilisateur doit pouvoir naviguer sur l'application lorsqu'une optimisation roule en arrière-plan.

4.2. Fiabilité

4.2.1 En cas de panne, le système doit recommencer les opérations de génération d'horaire qui étaient en cours automatiquement.

4.2.2 En cas de panne, le système doit redémarrer sans aucune perte d'information.

4.2.3 En cas de panne, le système doit se rétablir en moins de 5 secondes.

4.3. Performance

4.3.1. Le système doit supporter au moins 3 demandes d'optimisation d'horaire simultanément.

4.3.2. Le système doit permettre à l'utilisateur de se connecter à l'application en moins d'une seconde.

4.3.3 Le système doit assurer le traitement asynchrone de l'opération de génération d'horaire.

4.3.4 Le temps maximal de réponse à une requête doit être de 2 secondes.

4.3.5 Le système doit charger l'historique des actions d'édition d'un horaire en moins de 10 secondes.

4.3.6 Le système doit charger un horaire lors de son ouverture en moins de 5 secondes.

4.3.7 Le système doit assurer le partage d'un horaire aux autres utilisateurs en moins de 1 minute.

4.4. Maintenabilité

4.4.1. Le code doit être conforme avec les standards de qualité de code établie dans l'équipe.

4.4.1.1 Le code doit être testé à la fin de chaque sprint afin d'identifier rapidement les défauts du système.

4.4.1.2 Le code et les procédures doivent être documentés afin de faciliter la maintenance du système.

4.4.1.3 Le code doit être révisé avant d'être fusionné.

4.4.1.4 Un fichier de code doit contenir une entête de fichier.

4.4.1.5 Une fonction doit n'avoir qu'une seule responsabilité.

4.4.1.6 Une classe doit n'avoir qu'une seule responsabilité.

4.4.2 Les commentaires dans le code doivent être rédigés en anglais.

4.4.3 L'application doit avoir au minimum une couverture de code de 80%.

4.4.4 Le code du serveur doit respecter les normes du langage Python.

4.4.5 Le code de l'application Web doit respecter les normes du langage Typescript.

4.5. Contraintes de conception

4.5.1 L'application Web doit fonctionner dans tous les navigateurs modernes.

4.5.2 Le système, application web et le serveur, doit pouvoir rouler entièrement de manière locale.

4.5.3 Le serveur doit être développé en Python.

4.5.4 Le serveur doit utiliser un volume persistant.

4.5.5 Le serveur doit utiliser une base de données locale.

4.5.6 Le coût de développement de notre application doit être nul.

4.5.7 L'application doit être développée en mode itératif de 2 semaines.

4.6. Sécurité

4.6.1. Le système doit utiliser des "salted hashes" afin de protéger le mot de passe une fois le compte créé.

4.6.2. Le système doit utiliser des salted hashes afin de protéger le mot de passe à chaque fois qu'il est saisi.

4.6.3. Le système doit empêcher l'accès au mot de passe.

4.7. Exigences de la documentation usager en ligne et du système d'assistance

4.7.1 L'utilisateur doit avoir accès à de la documentation sur l'ensemble des fonctionnalités de l'application.

4.7.2 Le système doit présenter des messages d'erreur sans terme technique.

4.8. Normes applicables

4.8.1 Au moins deux personnes doivent participer à l'activité de revue de code.

4.8.2 Tous les membres de l'équipe doivent être impliqués dans la conception du projet.

4.8.3 L'équipe de développement doit citer les sources trouvées sur Internet.

4.8.4 L'équipe de développement doit respectecter les conditions des librairies utilisées.

4.8.5 Le logiciel de gestion de version qui doit être utilisé par l'équipe est GitLab

4.8.5.2 Une branche de fonctionnalité doit se nommer feature/nom-du-feature.

4.8.5.3 Une branche de correction de bogue doit se nommer hotfix/nom-du-bug.

4.8.5.4 La branche principale de développement doit être "dev".

4.8.5.5 Tous les tests doivent passer avant de faire un Merge Request d'une branche dans "dev".

4.8.5.6 Les messages de commits doivent énoncer les changements apportés.

ANNEXE A: Glossaire

Terme	Description
Administrateur	Un utilisateur ayant des permissions de création de comptes. Qui à un compte par défaut.
Angular	Angular est un framework pour clients, open source, basé sur TypeScript
Asynchrone	Qui s'effectue en arrière-plan sans impacter les fonctionnalités de l'application.
boost	Bibliothèque logicielle utilisée en C++.
C++	Langage de programmation
Compétence	Les qualifications d'une infirmière (ex.infirmière en chef, infirmière régulière, infirmière clinicienne, stagiaire, etc.)
Contrainte	L'ensemble des conditions qui doivent être respectées par l'horaire.
Contrat	Ensemble de contrainte
Convention collective	Les règles de l'hôpital doivent êtres respectées par toutes les infirmières.
Docker	Plateforme permettant de lancer une application dans des conteneurs logiciels.
Flask	Micro-framework basé sur Python pour créer des pages Web.
Historique de travail	Les quarts de travail réalisé par une infirmière dans la semaine qui précède l'horizon de planification de l'horaire à générer.
Horaire	Le résultat donné par le code C++
Horizon de planification	Période de temps pour laquelle l'horaire est établi.

Instance	Ensemble de données concrètes nécessaire pour générer un horaire par exemple la préférence du quart de travail par une infirmière.
Pip	Gestionnaire de paquets utilisé pour installer et gérer des paquets écrits en Python.
Poids	Niveau d'importance accordé à une contrainte.
Préférences	Volonté d'une infirmière de travailler ou de ne pas travailler pour une certaine plage horaire.
Problème	Un ensemble de contraintes tenu en compte lors de la génération de l'horaire.
Profil d'hôpital	Fait référence au problème associé à un hôpital, soit l'ensemble des infirmières, groupe d'infirmières, contrat, groupe de contrat, quart de travail, groupe de quart de travail, type de quart de travail, compétences et les problèmes de génération d'horaires et les horaires générés.
Propriétaires	Utilisateur qui a créé le profil de l'hôpital.
PyBind11	Bibliothèque qui expose les types C++ en python et vice-versa.
Python	Langage de programmation interprété, multiparadigme et multiplateformes.
Quart de travail	Division de la journée dans l'hôpital (jour/soir/nuit).
Salted hashes	Méthode visant à renforcer la sécurité des données.
shift	Un quart de travail: période pendant laquelle une infirmière travaille
shift type	Un ensemble de shift
Statistiques	L'ensemble des données qui indiquent à quel degré l'algorithme respecte les contraintes de génération d'horaire.
Typescript	Langage de programmation libre et open Source.

Utilisateur novice	Un utilisateur n'ayant jamais utilisé l'application.
Utilisateur spécialisé	Un utilisateur ayant déjà utilisé l'application pour plus de 30 minutes.

ANNEXE B: Glossaire des contraintes

Contrainte	Description
Complete Weekend	L'utilisateur doit travailler durant toutes les jours de la fin de semaine
Identical Shift Types During Weekend	L'infirmière doit avoir le même type de quart durant la fin de semaine.
Minimum and Maximum Number of Consecutive Type	Nombre minimum et maximum de fois qu'une infirmière peut travailler consécutivement le même type de quart dans l'horizon de planification.
Minimum and Maximum Number of weekends in Four Weeks	Nombre minimum et maximum de fin de semaine assignée dans une période de quatre semaines.
Minimum and Maximum of Number of Assignments in Four Weeks	Nombre minimum et maximum d'assignations d'un quart de travail dans une période de quatre semaines
Minimum and Maximum of working hours in Four Weeks	Nombre minimum et maximum d'heures consécutives que l'infirmière peut avoir dans l'horizon de planification pour un quart de travail
Minimum and Maximum of Consecutive Working Weekends	Nombre minimum et maximum de fins de semaine consécutives que l'infirmière peut avoir dans l'horizon de planification

Number Of Free Days After Shift	Nombre minimum de jours de repos qu'une infirmière doit avoir dans l'horaire après un certain type de quart de travail.
Pattern	Succession de type de quart de travail.
Unwanted Patterns	Matrice de succession de type de quart de travail interdit.
Unwanted Shift	Quart de travail qu'une infirmière ne peut pas avoir.
Unwanted Skill	Compétence qu'une infirmière ne peut pas avoir.