Équipe 10 - LEGRAIN-HORAIRES

Visualisation et construction d’horaires d’infirmières
Plan de tests logiciels

Version 6.0

Historique des révisions

Date	Version	Description	Auteur
2023-01-30	1.0	Première version du plan de tests logiciel	Mark Bekhet
2023-01-12	2.0	Deuxième version du plan de tests logiciel -Mise à jour existence à tester (Section 2) -Mise à jour jalons du projet (Section 5)	Mark Bekhet Caroline Des Rochers
2023-02-16	3.0	Troisième version du plan de test logiciel - Révision final du plan de tests logiciel avant le remise de mi-session	Caroline Des Rochers Mark Bekhet Kyrollos Bekhet Zoé Paradis Malgorzata Niczyporuk
2023-04-12	4.0	Quatrième version du plan de test logiciel - Correction suite aux commentaires de la correction de mi-session	Caroline Des Rochers
2023-04-14	5.0	Quatrième version du plan de test logiciel - Mise à jour existence à tester (Section 2) - Mise à jour jalons du projet (Section 5)	Caroline Des Rochers
2023-04-18	6.0	Cinquième version du plan de tests logiciel -Révision final du plan de test logiciel	Caroline Des Rochers Mark Bekhet Kyrollos Bekhet Zoé Paradis Malgorzata Niczyporuk

Table des matières

Plan de tests logiciels

1. Introduction

Ce document décrit les types de tests que nous allons effectuer afin d'assurer la qualité du système logiciel. Dans la prochaine section nous allons identifier les exigences. Dans la troisième section, nous allons décrire les stratégies des tests. En premier, nous allons décrire les différents types de tests , ensuite nous allons mentionner les outils nécessaires afin de réaliser ces tests et finalement les ressources utilisées pour les tests, à la fois humaines et matérielles.

2. Exigences à tester

Dans le tableau qui suit, on retrouve une liste des éléments que nous allons tester, soit des exigences fonctionnelles et non fonctionnelles de notre SRS. L'objectif est d'arriver à ultimement établir la traçabilité entre chaque exigence de notre SRS et leurs cas de test respectifs.

Exigences	Types de tests associés
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.	Test de fonction Test de l'interface usager Test unitaire
3.1.2 Le système doit permettre à l'administrateur de créer un compte	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.2.1 Le système doit permettre à l'utilisateur de créer un contrat	Test de fonction Test de l'interface usager Test unitaire
3.2.3 Le système doit permettre à l'utilisateur d'ajouter une liste de contraintes dans le contrat	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.2.7 Le système doit permettre à l'utilisateur de modifier un contrat	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.2.8 Le système doit permettre à l'utilisateur de supprimer un contrat	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.2.9 Le système doit permettre à l'utilisateur d'enregistrer un contrat	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire

3.2.10 Le système doit permettre à l'utilisateur de créer un groupe de contrats	Test de fonction Test de l'interface usager Test unitaire
3.2.11 Le système doit permettre à l'utilisateur de modifier un groupe de contrats	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.2.12 Le système doit permettre à l'utilisateur de supprimer un groupe de contrats	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.2.13 Le système doit permettre à l'utilisateur d'enregistrer un groupe de contrats	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.3.1 Le système doit permettre à l'utilisateur de définir une infirmière.	Test de fonction Test de l'interface usager Test unitaire
3.3.2 Le système doit permettre à l'utilisateur de modifier une infirmière.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.3.3 Le système doit permettre à l'utilisateur de supprimer une infirmière.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.3.4 Le système doit permettre à l'utilisateur de définir un groupe d'infirmières.	Test de fonction Test de l'interface usager Test unitaire
3.3.5 Le système doit permettre à l'utilisateur de modifier un groupe d'infirmières.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.3.6 Le système doit permettre à l'utilisateur de supprimer un groupe d'infirmières.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.3.7 Le système doit permettre à l'utilisateur d'enregistrer une infirmière.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.3.8 Le système doit permettre à l'utilisateur d'enregistrer un groupe d' infirmières.	Test de fonction Test de l'interface usager

	Test d'intégrité de données Test unitaire
3.4.1 Le système doit permettre à l'utilisateur de définir un quart de travail.	Test de fonction Test de l'interface usager Test unitaire
3.4.2 Le système doit permettre à l'utilisateur de spécifier un type de quart de travail.	Test de fonction Test de l'interface usager Test unitaire
3.4.3 Le système doit permettre à l'utilisateur de créer un groupe de quart de travail.	Test de fonction Test de l'interface usager Test unitaire
3.4.4 Le système doit permettre à l'utilisateur de modifier un quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.4.5 Le système doit permettre à l'utilisateur de supprimer un quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.4.6 Le système doit permettre à l'utilisateur d'enregistrer un quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.4.7 Le système doit permettre à l'utilisateur de modifier un type de quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.4.8 Le système doit permettre à l'utilisateur de supprimer un type de quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.4.9 Le système doit permettre à l'utilisateur d'enregistrer un type de quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.4.10 Le système doit permettre à l'utilisateur de modifier un groupe de quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.4.11 Le système doit permettre à l'utilisateur de supprimer un groupe de quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire

3.4.12 Le système doit permettre à l'utilisateur d'enregistrer un groupe de quart de travail.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.5.1 Le système doit permettre à l'utilisateur de générer un horaire	Test de fonction Test de l'interface usager Test unitaire
3.5.2 Le système doit permettre à l'utilisateur de définir le problème d'un horaire à générer	Test de fonction Test de l'interface usager Test unitaire
3.5.3 Le système doit permettre à l'utilisateur d'annuler la définition du problème de l'horaire à générer	Test de fonction Test de l'interface usager Test unitaire
3.5.4 Le système doit permettre à l'utilisateur de modifier la définition du problème.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.5.5 Le système doit permettre à l'utilisateur d'enregistrer la définition du problème.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.5.6 Le système doit permettre à l'utilisateur de supprimer la définition d'un problème	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.6.1 Le système doit permettre à l'utilisateur de visualiser un horaire.	Test de fonction Test de l'interface usager Test unitaire
~~3.6.2 Le système doit permettre à l'utilisateur de filtrer l'affichage d'un horaire.~~	~~Test de fonction~~ ~~Test de l'interface usager~~ ~~Test unitaire~~
3.6.3 Le système doit permettre à l'utilisateur d'être notifié lorsque nouvel horaire associés à ces demandes de génération d'horaire vient d'être complété.	Test de fonction Test de l'interface usager Test unitaire
3.7.1 Le système doit permettre à l'utilisateur de consulter son historique de génération d'horaires.	Test de fonction Test de l'interface usager Test unitaire
3.7.2 Le système doit permettre à l'utilisateur de revenir à une version antérieure d'un horaire généré.	Test de fonction Test de l'interface usager Test unitaire

3.7.3 Le système doit permettre à l'utilisateur de consulter l'historique d'édition d'un horaire.	Test de fonction Test de l'interface usager Test unitaire
3.7.4 Le système doit permettre à l'utilisateur de consulter son historique de création de problèmes.	Test de fonction Test de l'interface usager Test unitaire
3.8.1 Le système doit permettre à l'utilisateur d'éditer la demande d'un horaire qu'il a déjà généré.	Test de fonction Test de l'interface usager Test unitaire
3.8.2 Le système doit permettre à l'utilisateur de demander une optimisation d'horaire après une édition.	Test de fonction Test de l'interface usager Test unitaire
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 ré-optimiser.	Test de fonction Test de l'interface usager Test unitaire
~~3.9.1 Le système doit permettre à l'utilisateur de partager un horaire avec un autre utilisateur du système.~~	~~Test de fonction~~ ~~Test de l'interface usager~~ ~~Test unitaire~~
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.	Test de fonction Test de l'interface usager Test de charge Test unitaire
3.11.1 Le système doit permettre à l'utilisateur d'enregistrer un horaire localement.	Test de fonction Test de l'interface usager Test unitaire
~~3.11.2 Le système doit permettre à l'utilisateur d'enregistrer un contrat localement.~~	~~Test de fonction~~ ~~Test de l'interface usager~~ ~~Test unitaire~~
3.11.3 Le système doit permettre à l'utilisateur d'enregistrer la définition d'un problème localement.	Test de fonction Test de l'interface usager Test unitaire
3.11.4 Le système doit permettre à l'utilisateur d'enregistrer un profil d'hôpital localement.	Test de fonction Test de l'interface usager Test unitaire
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.	Test de fonction Test de l'interface usager Test unitaire
3.11.6 Le système doit permettre à l'utilisateur d'enregistrer le gabarit d'un profil d'hôpital en format CSV localement.	Test de fonction Test de l'interface usager Test unitaire

~~3.12.1 Le système doit permettre à l'utilisateur de téléverser la définition d'un problème.~~	~~Test de fonction~~ ~~Test de l'interface usager~~ ~~Test unitaire~~
~~3.12.2 Le système doit permettre à l'utilisateur de téléverser un contrat.~~	~~Test de fonction~~ ~~Test de l'interface usager~~ ~~Test unitaire~~
~~3.12.3 Le système doit permettre à l'utilisateur de téléverser le dossier d'une infirmière.~~	~~Test de fonction~~ ~~Test de l'interface usager~~ ~~Test unitaire~~
~~3.12.4 Le système doit permettre à l'utilisateur de téléverser la définition d'un quart de travail.~~	~~Test de fonction~~ ~~Test de l'interface usager~~ ~~Test unitaire~~
3.12.5 Le système doit permettre à l'utilisateur de téléverser le profil d'un hôpital.	Test de fonction Test de l'interface usager Test unitaire
3.13.1 Le système doit permettre à l'utilisateur de créer un profil d'hôpital.	Test de fonction Test de l'interface usager Test unitaire
3.13.2 Le système doit permettre à l'utilisateur de dupliquer un profil d'hôpital.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.13.3 Le système doit permettre au propriétaire du profil de l'hôpital de le supprimer.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.13.4 Le système doit permettre à l'utilisateur de partager un profil d'hôpital avec une sélection d'utilisateurs.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.13.5 Le système doit permettre à l'utilisateur de consulter les profils d'hôpital partager avec lui.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.13.6 Le système doit permettre à l'utilisateur ayant accès à un profil de pouvoir le modifier.	Test de fonction Test de l'interface usager Test d'intégrité de données Test unitaire
3.14.1 Le système doit permettre à l'utilisateur de définir une compétence.	Test de fonction Test de l'interface usager Test unitaire
3.14.2 Le système doit permettre à l'utilisateur de supprimer une compétence.	Test de fonction Test de l'interface usager

	Test d’intégrité de données Test unitaire
4.1.1 Un utilisateur novice doit pouvoir utiliser l’application de manière productive après 10 minutes d’exploration.	Test d’interface usager
4.1.2 Un utilisateur spécialisé doit pouvoir utiliser l’application de manière productive immédiatement sans aucune formation.	Test d’interface usager
4.1.3 L’interface graphique de l’application doit suivre le critère d’homogénéité de Bastien et Scapin	Test d’interface usager
4.1.4 L’interface graphique de l’application doit suivre le critère de guidage de Bastien et Scapin.	Test d’interface usager
4.1.5 Un utilisateur doit pouvoir naviguer sur l’application lorsqu'une optimisation roule en arrière-plan.	Test de charge Test d’interface usager
4.2.1 En cas de panne, le système doit recommencer les opérations de génération d’horaire qui étaient en cours automatiquement.	Test d’échec/récupération
4.2.2 En cas de panne, le système doit redémarrer sans aucune perte d’information.	Test d’échec/récupération
4.3.1 Le système doit supporter au moins 3 demandes d’optimisation d’horaire simultanément.	Test de charge
4.3.3 Le système doit assurer le traitement asynchrone de l’opération de génération d’horaire.	Test de charge
4.5.1 L’application Web doit pouvoir tourner sur n’importe quel ordinateur.	Test d’installation
4.5.3 L’application doit pouvoir rouler entièrement de manière locale.	Test d’installation
4.5.6 Le serveur doit utiliser un volume persistant.	Test d’échec/récupération
4.6.1 Le système doit utiliser des “salted hashes” afin de protéger le mot de passe une fois le compte créé	Test de fonction Test unitaire

3. Stratégie de test

Dans la section qui suit, nous allons résumer la stratégie de test que nous allons utiliser afin de tester le logiciel. Ainsi, 6 types de tests sont prévus dans notre stratégie, soit les tests de fonction, les tests d'interface d'usager, les tests d'intégrités des données, les tests de charges, les tests d'échec/récupération et les tests d'installation.

3.1. Types de test

3.1.1. Tests de fonction

Objectif de test:	Ce test vise à vérifier que tous les cas d'utilisation s'effectuent correctement avec les différentes classes d'équivalences.
Technique:	Afin d'accomplir ce type de test, nous allons définir des classes d'équivalences pour les entrées du système afin de tester les limites des fonctionnalités.
Critère de complétion:	La sortie du programme correspond au résultat attendu et le programme ne lance pas des erreurs inattendues

3.1.2. Tests d'interface usager

Objectif de test:	Ce test vise à vérifier que l'interface graphique de l'application web est facile d'utilisation ainsi qu'elle est intuitive. Cela vise à s'assurer qu'un utilisateur quelconque puisse naviguer l'application sans avoir besoin de formation, c'est-à-dire que l'interaction d'un utilisateur avec l'application est adéquate.
Technique:	Test utilisateur. 3 utilisateurs de niveaux différents soit expert, notre client, moyen (l'étudiante au doctorat de notre client) et un utilisateur novice devront naviguer à travers l'application et réaliser des tâches en suivant les directives du testeur.
Critère de complétion:	L'utilisateur a-il atteint l'objectif de la tâche. Le temps de chacun des utilisateurs pour réaliser une tâche, afin de comparer entre les divers niveaux d'utilisateur. Nombre d'erreurs effectuées par l'utilisateur lors de l'exécution de la tâche.On peut également noter le niveau de satisfaction et de compréhension de l'utilisateur.
Considérations spéciales:	Si un utilisateur apporte une complainte par rapport à une composante. Un bogue doit être créé sur Jira. Avant le début de l'exercice, le développeur va expliquer le but du logiciel ainsi que les cas d'utilisation de l'application. Ensuite, il laissera l'utilisateur essayer d'accomplir les cas d'utilisation mentionnés. Le développeur va devoir noter les commentaires de l'utilisateur ainsi que le temps pour réaliser les tâches.

3.1.3. Tests d'intégrité des données

Objectif de test:	Assurer que les données sont sauvegardées correctement dans la base de données et si elles sont exactes en fonction des exigences. Les tests s'assurent aussi que les données ne soient pas modifiées ou bien corrompues lors de l'accès à la base de données.
Technique:	Après une opération d'insertion d'une entrée dans la base de données, nous utilisons un logiciel comme Studi3T pour lancer une requête vers la base de données afin de s'assurer que la nouvelle entrée existe comme prévue.
Critère de complétion:	Les opérations sur la base de données s'effectuent correctement et retournent les bonnes valeurs.

3.1.4. Tests de charge

Objectif de test:	Assurer que notre application assure la gestion asynchrone de la génération d'horaires.
Technique:	Demander la génération de plusieurs horaires en parallèle en effectuant un test manuel.
Critère de complétion:	Le programme est capable de générer un minimum de 5 horaires en parallèle sans tomber en panne.
Considérations spéciales:	En cas d'échec, la stratégie du déploiement doit être revisitée afin de créer plus de nœuds qui s'occupent de la génération d'horaires. Idéalement, l'ordinateur de l'utilisateur est de 8 GB de RAM afin de supporter tous les conteneurs Docker

3.1.5. Tests d'échec/récupération

Objectif de test:	S'assurer qu'un nœud peut reprendre la génération des horaires après avoir tombé en panne d'une manière soudaine.
Technique:	Arrêter un nœud qui était en train de générer un horaire de manière abrupte afin de vérifier qu'il reprenne la génération d'horaire interrompue.
Critère de complétion:	Le nœud redémarre automatiquement et reprend la génération de l'horaire qui était en cours.

3.1.6. Tests d'installation

Objectif de test:	S'assurer que les différents nœuds impliqués communiquent de manière cohésive.
Technique:	Manuellement lancer la commande docker-compose up et vérifier les logs des différents conteneurs afin de vérifier qu'ils puissent communiquer ensemble. Procéder à un test utilisateur si nécessaire.
Critère de complétion:	Les différents conteneurs communiquent de manière cohésive

3.1.7. Tests unitaires (backend)

Objectif de test:	S'assurer du fonctionnement du code et des types de retour et afin d'éviter les erreurs subtiles durant la programmation.
Technique:	À la suite de chaque changement, une pipeline est lancée par gitlab afin de rouler les tests unitaires et d'assurer qu'ils réussissent. Le cadriciel Pytest est utilisé afin de rouler les tests unitaires.
Critère de complétion:	La couverture des branches du code doit être à 80% et plus.

3.2. Outils

Les outils suivants seront utilisés au sein de la discipline de test:

Type de test	Outil
Test de fonction	pytest/python
Test d'interface utilisateur	Google chrome / firefox Docker / Docker-compose
Test d'intégrité de données	Robot 3t / DB shell
Test de charge Test de charge pour les containers Spécifié des spectres de machines	Docker/Docker-compose Postman
Test d'échec/ récupération	Docker/Docker-compose
Test d'installation	Docker/Docker-compose

4. Ressources

La section qui suit présente les ressources humaines et matérielles relatives à la discipline de test.

4.1. Équipe de test

Rôle	Membre de l'équipe	Responsabilités
Testeur du serveur	Mark Bekhet	Test d'intégration Test de charge Test de récupération/d'échec Test d'installation Tests d'intégrité des données Test unitaire
Designer de l'application web	Zoé Paradis	Tests d'interface usagée de l'application web
Testeur de fonctionnalité	Kyrollos Bekhet	Tests d'interface usager de l'application web
Designer de l'application web	Malgorzata Niczyporuk	Test d'interface usager de l'application web
Testeur de fonctionnalité	Caroline Des Rochers	Test d'interface usager de l'application web

4.2. Système

Comme l'application est désignée afin de rouler sur un ordinateur local, les tests auront lieu aussi sur un ordinateur local. Afin d'assurer le bon déroulement des tests, l'ordinateur sur lequel le serveur va rouler est d'au moins 8 GB de mémoire. Au cas où ceci n'est pas possible, il sera possible de louer une instance AWS ou de n'importe quel autre fournisseur de service nuage afin d'y rouler les différents serveurs. Ainsi l'ordinateur va servir à accéder à l'application web à partir du DNS ou le public IP donné par le fournisseur de service cloud. Afin de rouler les différents serveurs, il sera possible de les rouler en utilisant la commande docker-compose up.

5. Jalons du projet

Jalon	Effort	Date de début	Date de fin
Rédaction du plan de tests	25h	2023-01-27	2023-02-16
Planification et design des tests de prototypes (*Front-end* et *Back-end*)	6h	2023- 02-10	2023-02-12
Implémentation des tests du prototype	6h	2023- 02-13	2023-02-18
Exécution des tests du prototype	3h	2023-02-14	2023-02-19
Planification et designs des tests associés aux exigences fonctionnelles 3.1, 3.2.1 à 3.2.4, 3.3.1, 3.4.1, 3.4.2 et 3.4.4 à 3.4.9 du SRS.	8h	2023-02-21	2023-02-25
Implémentation des tests associés aux exigences fonctionnelles 3.1, 3.2.1 à 3.2.4, 3.3.1, 3.4.1, 3.4.2 et 3.4.4 à 3.4.9 du SRS.	8h	2023-02-26	2023-03-02
Exécution des tests associés aux exigences fonctionnelles 3.1, 3.2.1 à 3.2.4, 3.3.1, 3.4.1, 3.4.2 et 3.4.4 à 3.4.9 du SRS.	4h	2023-02-28	2023-03-06
Planification et designs des tests associés aux exigences fonctionnelles 3.2.2 à 3.2.10, 3.4.3, 3.4.10 à 3.4.12, 3.5.2, 3.11.3,3.11.4, 3.11.6, 3.123.13 et 3.14 du SRS.	8h	2023-03-07	2023-03-11
Implémentation des tests associés aux exigences fonctionnelles 3.2.2 à 3.2.10, 3.4.3, 3.4.10 à 3.4.12, 3.5.2, 3.11.3,3.11.4, 3.11.6, 3.123.13 et 3.14 du SRS.	8h	2023-03-12	2023-03-18
Exécution des tests associés aux exigences fonctionnelles 3.2.2 à 3.2.10, 3.4.3, 3.4.10 à 3.4.12, 3.5.2, 3.11.3, 3.11.4, 3.11.6, 3.123.13 et 3.14 du SRS.	4h	2023-03-14	2023-03-20
Planification et designs des tests associés aux exigences fonctionnelles 3.5.3 à 3.5.6, 3.10, 3.6, 3.7, 3,11.1, 3.2.10, 3.2.10, 3.8 du SRS.	8h	2023-03-21	2023-03-25

Implémentation des tests associés aux exigences fonctionnelles 3.5.3 à 3.5.6, 3.10, 3.6, 3.7, 3,11.1, 3.2.10, 3.2.10, 3.8 du SRS.	8h	2023-03-26	2023-04-02
Exécution des tests associés aux exigences fonctionnelles 3.5.3 à 3.5.6, 3.10, 3.6, 3.7, 3,11.1, 3.2.10, 3.2.10, 3.8 du SRS.	4h	2023-03-28	2023-04-03
Planification et designs des tests associés aux exigences non-fonctionnelles 4.1.5, 4.2.1, 4.2.2, 4.3.1, 4.3.3, 4.5.1, 4.5.3, 4.5.6 et 4.6.1 du SRS.	10h	2023-03-21	2023-03-25
Implémentation des tests associés aux exigences non-fonctionnelles 4.1.5, 4.2.1, 4.2.2, 4.3.1, 4.3.3, 4.5.1, 4.5.3, 4.5.6 et 4.6.1 du SRS.	10h	2023-03-26	2023-04-02
Exécution des tests associés aux exigences non-fonctionnelles 4.1.5, 4.2.1, 4.2.2, 4.3.1, 4.3.3, 4.5.1, 4.5.3, 4.5.6 et 4.6.1 du SRS.	5h	2023-03-28	2023-04-03
Planification et design des tests utilisateurs de la version bêta. Tests associés aux exigences non-fonctionnelles 4.1.1, 4.1.2, 4.1.3, 4.1.4, 4.1.5 du SRS.	15h	2023-03-21	2023-03-27
Planification des tests associés aux fonctionnalités ajouté ou bien modifié suite à la remise de la version bêta.	4h	2023-04-05	2023-04--08
Implémentation des tests associés aux fonctionnalités ajouté ou bien modifié suite à la remise de la version bêta.	10h	2023-04-07	2023-04-10
Exécution des tests associés aux fonctionnalités ajoutées ou bien modifié suite à la remise de la version bêta.	10h	2023-04-07	2023-04-12