

ECOLE POLYTECHNIQUE DE L'UNIVERSITÉ FRANÇOIS RABELAIS DE TOURS

Département Informatique

64 avenue Jean Portalis

37200 Tours, France

Tél. +33 (0)2 47 36 14 14

[polytech.univ-tours.fr](http://polytech.univ-tours.fr)

## Projet Recherche & Développement 2018-2019

# Outil de gestion de parcours patient dans un hôpital de jour

**Tuteur académique**  
Yannick KERGOSIEN

**Étudiant**  
Romain ROUSSEAU (DI5)

11 février 2019



# Liste des intervenants

Nom	Email	Qualité
Romain ROUSSEAU	<a href="mailto:romain.rousseau@etu.univ-tours.fr">romain.rousseau@etu.univ-tours.fr</a>	Étudiant DI5
Yannick KERGOSIEN	<a href="mailto:yannick.kergosien@univ-tours.fr">yannick.kergosien@univ-tours.fr</a>	Tuteur académique, Département Informatique



# Avertissement

Ce document a été rédigé par Romain ROUSSEAU susnommé l'auteur.

L'Ecole Polytechnique de l'Université François Rabelais de Tours est représentée par Yannick KERGOSIEN susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnaît assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assument l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnaît qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.



## Pour citer ce document

Romain ROUSSEAU, *Outil de gestion de parcours patient dans un hôpital de jour*, Projet Recherche & Développement, Ecole Polytechnique de l'Université François Rabelais de Tours, Tours, France, 2018-2019.

```
@mastersthesis{
  author={ROUSSEAU, Romain},
  title={Outil de gestion de parcours patient dans un hôpital de jour},
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université François Rabelais de Tours},
  address={Tours, France},
  year={2018-2019}
}
```



# Table des matières

Liste des intervenants	a
Avertissement	b
Pour citer ce document	c
Table des matières	i
Table des figures	v
Liste des tableaux	vii
Introduction	1
<b>I Analyse préliminaire et détermination des objectifs</b>	<b>3</b>
<b>1 Contexte de la réalisation et objectifs</b>	<b>4</b>
1 Contexte, enjeux et acteurs .....	4
2 Objectifs .....	5
2.1 Corrections des bugs et améliorations sur les fonctionnalités existantes .....	6
2.2 Implémentation de la planification automatique .....	6
2.3 Implémentation de la planification temps réel.....	6
3 Bases méthodologiques .....	7
<b>2 Description générale</b>	<b>8</b>
1 Environnement du projet .....	8
2 Caractéristiques des utilisateurs .....	9
3 Contraintes de développement .....	9

4	Description des interfaces externes du système .....	10
4.1	Interface matériel/logiciel .....	10
4.2	Interface homme/machine .....	10
4.3	Interface logiciel/logiciel.....	10
5	Fonctionnalités du système .....	10
5.1	Ajout de fonctions de tri lors de l’affichage des ressources .....	11
5.2	Ajustement du planning selon les ressources sélectionnées.....	11
5.3	Planification automatique .....	12
6	Structure générale du système .....	13
<b>3</b>	<b>État de l’art</b> .....	<b>14</b>
1	Présentation des problèmes multi-ressources dans un hôpital.....	14
2	Les niveaux de décision dans la gestion des hôpitaux.....	15
3	Choisir la portée du modèle .....	15
3.1	Le service hospitalier concerné .....	15
3.2	Le type de patient .....	16
3.3	Les ressources .....	17
4	Choisir ce qu’on veut optimiser .....	17
5	Choisir comment optimiser.....	18
6	Appliquer et valider le modèle .....	18
<b>II</b>	<b>Développement</b> .....	<b>19</b>
<b>4</b>	<b>Analyse et conception</b> .....	<b>20</b>
1	Révision de la suppression des ressources.....	20
2	Révision de l’onglet <i>Plan de parcours</i> .....	21
3	Révision du formulaire de création de patient .....	22
4	Affichage du planning patient .....	22
5	Ajout des fonctions de tri .....	23
6	Correction des erreurs de redirection .....	23
7	Implémentation de la planification automatique.....	24
8	Génération de jeux de données pour les tests.....	24
<b>5</b>	<b>Mise en œuvre</b> .....	<b>26</b>
1	Ajout d’un avertissement lors de la suppression d’un élément .....	26
2	Révision du plan de parcours.....	27
3	Corrections sur le formulaire de création de patient.....	27
4	Fonction de tri sur les onglets de Gestion.....	28
5	Planification automatique .....	29
5.1	Fonctionnement de l’affichage graphique.....	29

5.2	Corrections sur la planification d'origine .....	29
5.3	Mise en œuvre de l'ordonnancement .....	30
<b>6</b>	<b>Validation et Tests</b>	<b>33</b>
1	Le framework Codeception .....	33
2	Test de la planification automatique.....	34
<b>7</b>	<b>Qualité de code</b>	<b>36</b>
1	Documentation .....	36
2	SonarQube .....	36
<b>8</b>	<b>Avancement du projet et suite à donner</b>	<b>38</b>
1	Fonctionnalités mises en place.....	38
2	Fonctionnalités à ajouter ou à améliorer pour la suite du projet .....	38
	<b>Bilan et conclusion</b>	<b>39</b>
	<b>Annexes</b>	<b>41</b>
<b>A</b>	<b>Fonctionnalités existantes</b>	<b>42</b>
1	Gérer les personnels .....	42
2	Gérer les ressources matérielles .....	42
3	Gérer les activités .....	42
4	Gérer les parcours.....	43
5	Gérer les plans de parcours .....	43
6	Visualiser du planning .....	43
7	Affecter un patient à un parcours .....	43
8	Visualiser et modifier le planning de toutes les ressources .....	43
<b>B</b>	<b>Modèle conceptuel de données</b>	<b>44</b>
<b>C</b>	<b>Explications des tables du modèle</b>	<b>46</b>
<b>D</b>	<b>Chiffrage du projet</b>	<b>48</b>
<b>E</b>	<b>Gestion de projet</b>	<b>49</b>
1	Trello .....	49
2	Diagramme de Gantt .....	50
3	Git et GitKraken .....	50
<b>F</b>	<b>Notice des livrables</b>	<b>51</b>
1	Livable du 9/01/2019 .....	51
2	Livable du 11/02/2019 .....	51

<b>G</b>	<b>Détails des parcours utilisés pour les tests</b>	<b>52</b>
1	Parcours P1 : Obésité sévère - diagnostic.....	52
2	Parcours P2 : Obésité sévère - Post OP.....	53
3	Parcours P3 : Obésité sévère - J+1an.....	54
<b>H</b>	<b>Guide développeur</b>	<b>55</b>
1	Installation du projet.....	55
2	Structure du projet .....	56
	<b>Bibliographie</b>	<b>60</b>



# Table des figures

## 2 Description générale

1	Diagramme de déploiement.....	9
2	Affichage du personnel sur la plateforme .....	11
3	Aperçu de la page de Planification .....	12
4	Arborescence du projet.....	13

## 4 Analyse et conception

1	Exemple de message d'avertissement en cas de suppression .....	21
2	Aperçu de l'onglet "Plan de parcours" à la reprise du projet .....	21
3	Formulaire de patient.....	22
4	Affichage du planning patient .....	23
5	Affichage d'une erreur de redirection .....	24
6	Affichage de la page de création de jeu de données .....	25

## 5 Mise en œuvre

1	Message d'avertissement en cas de suppression .....	26
2	Requêtes SQL pour récupérer les informations du plan de parcours.....	27
3	Nouvel affichage de l'onglet <i>Plan de Parcours</i> .....	27
4	Syntaxe des règles du formulaire de création de patient .....	28
5	Affichage des activités triées par nom .....	28
6	Exemple d'erreur de sélection de ressource.....	29
7	Exemple d'erreur d'affichage de contraintes .....	30
8	Affichage du type de ressource <i>Autres</i> .....	30
9	Gestion des patients et des activités à ajouter .....	31
10	Gestion des disponibilités des ressources .....	32

**6 Validation et Tests**

1	Exemple de test d'acceptance pour se connecter .....	33
2	Affichage des résultats de tests Codepetion .....	34
3	Planification vue du personnel .....	35
4	Planification vue des salles .....	35

**7 Qualité de code**

1	Fenêtre d'affichage de Sonar .....	37
---	------------------------------------	----

**E Gestion de projet**

1	Aperçu de ma page <b>Trello</b> .....	49
2	Diagramme de Gantt .....	50
3	Diagramme de Gantt .....	50

**G Détails des parcours utilisés pour les tests**

1	Parcours P1 .....	52
2	Parcours P2 .....	53
3	Parcours P3 .....	54

**H Guide développeur**

1	Arborescence du projet.....	56
2	Arborescence du répertoire source .....	57
3	Aperçu d'un contrôleur .....	57
4	Aperçu d'une fonction d'un modèle .....	58
5	Schéma d'une vue.....	58
6	Mécanisme de la vue .....	58



# Liste des tableaux

## D Chiffrage du projet

1	Chiffrage des tâches à réaliser.....	48
---	--------------------------------------	----



# Introduction

Ce rapport traite du projet Recherche et Développement réalisé au sein de l'école Polytech. Il se déroule lors la cinquième année d'étude et représente une forme de synthèse des connaissances acquises lors des années d'études précédentes. Il permet de développer et d'approfondir son savoir sur un ou plusieurs champs de compétences spécifiques, afin de devenir un spécialiste du domaine choisi dans le cadre du projet. Ce travail est mené seul, sous la supervision d'un tuteur académique et, pour certains projets, avec l'aide d'un intervenant extérieur qui fait partie des initiateurs du sujet. Il permet ainsi de développer son autonomie en menant à bien un travail des prémices jusque, dans le meilleur des cas, à la production.

J'ai décidé de me consacrer à un projet lié à un enjeu important de notre société : la gestion des patients dans un hôpital. Aujourd'hui, les hôpitaux sont l'objet de nombreux débats dans l'actualité, et l'un des points de discussion majeurs concerne la gestion des patients au sein des hôpitaux. Plusieurs problèmes peuvent être cités : un temps d'attente bien trop long pour les patients (que ce soit une fois à l'hôpital ou pour avoir un rendez-vous avec un spécialiste), des problèmes de ressource (concernant le personnel ou bien les salles à disposition) ou encore des problèmes d'ordre financier en général. Certaines de ses problématiques peuvent être étudiées selon le point de vue de l'optimisation, et la recherche opérationnelle peut permettre d'améliorer certains aspects de la gestion actuelle. Pour ma part, me pencher sur les problèmes de recherche opérationnelle en général m'intéressait, et avoir l'opportunité de réaliser un projet dans ce domaine, d'autant plus dans un secteur qui en a besoin, était une bonne occasion pour améliorer mes compétences.

Le sujet de ce travail rentre dans le cadre d'un projet visant à développer un outil de gestion temps réel de parcours de patients pour l'AP-HP (Assistance publique - Hôpitaux de Paris). L'objectif de cet outil est de gérer un ensemble de patients nécessitant plusieurs activités de soins planifiées sur une journée dans un hôpital de jour. Chaque patient doit suivre un « parcours de soins » défini par un ensemble d'activités de soins. Certaines doivent être réalisées avant d'autres (contraintes de précédences) alors que pour d'autres, l'ordre n'a pas d'importance. Il peut exister des délais d'attente à respecter entre les activités. Chaque activité de soin est caractérisée par une durée et un ensemble de ressources nécessaires pour la réaliser. Ces ressources peuvent représenter le personnel, la salle où s'effectue l'activité ou encore le matériel utilisé. Elles sont caractérisées par une quantité et un horaire où elles sont disponibles. L'idée de l'outil est de pouvoir proposer un calendrier adaptatif dans lequel les horaires des activités de soins des patients peuvent être ajustés en fonction des aléas potentiels (retard, patient absent, etc.) afin de diminuer au mieux possible le temps d'attente des patients.

Ce projet est la suite de plusieurs travaux effectués à l'école Polytech ces dernières années et qui ont abouti à une première version simple d'un outil sous forme de plateforme web. L'objectif est d'améliorer l'outil actuel en corrigeant certains problèmes et en ajoutant des fonctionnalités afin de se rapprocher d'une version finale exploitable à l'avenir. Parmi les fonctionnalités qui devront être ajoutées, on peut notamment penser à la planification automatique des activités de soin sur le calendrier ou encore l'ajout de notions de temps réel sur les éléments du calendrier.

La première partie de ce rapport sera consacrée à l'analyse préliminaire de la plateforme et à la détermination des tâches à réaliser. Elle comprend des éléments sur le contexte de la réalisation, la description de la plateforme telle qu'elle se trouvait au début du projet, un état de l'art sur les modélisations de la planification automatique des activités, et la présentation des tâches qui seront effectuées durant les semaines de projet.

La seconde partie du rapport sera dédiée aux implémentations et aux développements réalisés au cours du projet. Des éléments concernant les phases de tests et la qualité de code seront également présentés. Des documents complémentaires sont disponibles en annexe de ce rapport comme les diagrammes de Gantt, les diagrammes UML, des éléments de gestion de projet, etc. Un guide développeur destiné aux personnes qui désireront reprendre le projet à l'avenir est également présent en [Annexe H](#).

## Première partie

# Analyse préliminaire et détermination des objectifs

Cette partie correspond à la première phase du Projet Recherche et Développement, la partie Recherche. Dans le cadre de ce projet, cette partie sera une analyse générale de la plateforme existante afin de déterminer quels objectifs nous devons atteindre à l'issue de ce projet. Le premier chapitre sera consacré au contexte de la réalisation, la raison de l'initiation du projet ainsi que les acteurs concernés. Le chapitre suivant décrira la plateforme telle qu'elle se trouvait à la reprise du sujet, en détaillant la structure et les fonctionnalités qui existent. Enfin le dernier chapitre évoquera des éléments de recherche opérationnelle dans le domaine de la gestion des patients dans un hôpital.

# 1

## Contexte de la réalisation et objectifs

Ce chapitre présente le contexte du sujet et les objectifs à atteindre à l'issue du projet.

### 1 Contexte, enjeux et acteurs

La santé dans notre société est un sujet prédominant dans l'actualité. De nombreux débats existent concernant la modernisation des structures hospitalières et surtout, la réduction des dépenses dans le secteur. Cependant, ces problématiques sont très complexes à gérer car il est important de maintenir une certaine qualité d'accueil nécessaire au bon traitement des patients, néanmoins une baisse des dépenses pourrait se répercuter dans la qualité des prestations. L'un des aspects sur lesquels ce projet sera consacré est la gestion de prise de rendez-vous médicale. Les consultations sont soumises à de nombreux aléas, que ce soit côté patient comme côté médecin. Il n'est pas rare pour un patient de devoir attendre plusieurs mois avant d'avoir un rendez-vous chez un spécialiste, tout comme il est fréquent pour un médecin de se retrouver avec un patient absent lors d'un rendez-vous. Ceci entraîne de la frustration et du gaspillage de ressources en général. Des solutions dans le domaine de la recherche opérationnelle peuvent être mises en place, afin d'améliorer la gestion des ressources médicales et de réduire les temps d'attente pour les patients, que ce soit avant ou pendant la prise en charge.

Les services hospitaliers sont souvent surchargés par le nombre de patients dans l'établissement. Il est de plus en plus fréquent de voir les hôpitaux proposer des services en ambulatoire, c'est-à-dire prendre en charge un patient sur une seule journée, évitant ainsi les frais pouvant s'appliquer lorsqu'un patient dort sur place. Les dépenses liées aux hospitalisations sont parmi les plus importantes dans un établissement et de nos jours, les hôpitaux proposent même des alternatives pour que les patients ne restent pas sur place la nuit, comme par exemple des chambres d'hôtel afin de ne pas occuper une chambre pendant l'hospitalisation [2]. Un établissement, ou une partie d'établissement, proposant ce type de service est appelé "hôpital de jour".

L'idée d'implémentation d'un outil numérique dans une structure implique une réflexion sur les concepts liés l'hospitalisation d'un patient. Il est possible de définir, selon les pathologies à traiter, un parcours-patient à suivre. Un parcours-patient (ou parcours-clinique) est un plan de soins pluridisciplinaires exposant les étapes que doivent un patient pour un problème spécifique. On peut les décomposer en plusieurs activités prédéfinies, certaines devant être

réalisées avant d'autres (ce qu'on appelle une contrainte de précédence), nécessitant des ressources physiques ou matérielles (contrainte de ressources) et devant être effectuées dans un délai ou un créneau horaire précis (contrainte temporelles). La mise en place d'un système de centralisation des parcours-patient reste difficile, notamment car elle repose en priorité sur une bonne communication entre les services d'un établissement. Néanmoins, cela permettrait d'avoir potentiellement une meilleure utilisation des ressources de l'hôpital et par conséquent, un effet favorable sur les durées d'hospitalisation et les dépenses hospitalières entre autres.

L'Assistance Publique - Hôpitaux de Paris (AP-HP), dans son plan stratégique 2015 - 2019, a lancé une réflexion sur la mise en place d'une clinique ambulatoire. Avec la mutualisation de certains services au sein de différents hôpitaux de la région, le site de l'hôpital *Béclère*, situé dans la ville de Clamart dans le département des Hauts-de-Seine, obtiendra un gain de place suffisant pour proposer une telle solution. Il regroupera tous les hôpitaux de jour existants du site, avec un management centralisé et transversal à l'aide de parcours-patients. Les équipes médicales et administratives ont pris le soin de définir d'ors et déjà une vingtaine de parcours patients en adéquation avec les ressources déjà en place, et présenté une répartition type des prises en charge hebdomadaire. Celle-ci devrait en théorie permettre d'accueillir 179 patients chaque semaine, et à terme environ 1000 par mois. Ainsi, la conception d'un système d'aide à la décision pour la prise de rendez-vous serait utile pour une nouvelle structure comme celle-ci.

Le sujet a été proposé en 2015 par Lucie ROUSSEL, ingénieur en ordonnancement à l'AP-HP et a été suivi lors de plusieurs projets à Polytech : tout d'abord, lors d'un projet collectif (réalisé par les étudiants Benjamin COLMART, Jean COQUELET, Anaëlle HAMON, Jiang MING, Yan LI et Minghui ZHANG) qui a posé les bases de la modélisation et des premières fonctionnalités, un projet R&D effectué par Jean COQUELET en 2016 également basé davantage sur l'aspect gestion et optimisation des parcours, un projet R&D par Guillaume POCHET en 2017 qui a ajouté des éléments sur la gestion de calendrier et enfin un autre projet R&D réalisé par Jing YANG qui ajoute des améliorations générales. Le projet tel qu'il est développé à Polytech servira dans un premier temps en tant que *proof of concept*, et si celui-ci satisfait l'administration, il pourra entrer en production à plus grande échelle dans le futur.

La plateforme, une fois installée sur mon poste à la reprise du projet, fonctionne. Il est possible d'ajouter des patients et de leur associer un parcours-patient avec des heures de rendez-vous et des activités à suivre lors d'une journée. Cependant, de nombreux éléments sont à améliorer, des bugs à corriger dans un premier temps mais aussi des améliorations d'un point de vue graphique. Des fonctionnalités importantes restent à implémenter : d'un côté, l'ajout d'une notion de temps réel sur les activités et les rendez-vous, et de l'autre, la possibilité de faire une planification automatique tenant compte des patients et des ressources disponibles. Compte-tenu du temps alloué pour le projet, il sera difficile d'ajouter ces deux fonctionnalités importantes. Nous verrons par la suite vers quels éléments nous allons nous pencher en priorité.

## 2 Objectifs

Suite à l'affectation du sujet, les premières rencontres avec M. KERGOSIEN ont permis de fixer des objectifs principaux pour mener à bien le projet. Trois axes de développement ont été définis pour la finalisation de la plateforme : la correction de certains problèmes sur les fonctionnalités existantes, l'ajout de fonctions de planification automatique et l'implémentation de fonctionnalités permettant de planifier en temps réel.

Les trois objectifs présentés par la suite sont conséquents en terme de temps. Il sera sans doute très compliqué de réaliser les trois objectifs dans la limite de temps alloué au projet. Nous nous focaliserons ainsi sur deux des trois objectifs à suivre.



## 2.1 Corrections des bugs et améliorations sur les fonctionnalités existantes

Le premier objectif consiste à corriger certaines erreurs que l'on peut trouver sur la plateforme. À la première analyse du projet, plusieurs problèmes sont apparus, que ce soit des erreurs d'affichage ou des soucis d'ergonomie par exemple. Ainsi, le but est d'améliorer les éléments déjà existants de la plateforme afin de la rendre plus stable et de meilleure qualité. Parmi les corrections et améliorations à apporter, on retrouve la révision de la suppression de données, de l'onglet "Plan de parcours", du formulaire de création de patient, de certains éléments de la page d'accueil.

Plusieurs fonctionnalités qui n'ont pas encore été implémentées sont aussi à rajouter pour enrichir les possibilités offertes à l'utilisateur, comme l'affichage du planning pour un patient donné, l'ajout de fonctions de tri pour l'affichage des ressources, ou encore, la création de jeux de données permettant de tester plus facilement la planification.

## 2.2 Implémentation de la planification automatique

L'une des parties principales du projet consiste à l'élaboration de la planification automatique des rendez-vous médicaux. À l'heure actuelle, la planification des rendez-vous se fait de façon manuelle. Une fois les patients définis pour une journée, un calendrier et la liste des activités à réaliser sont à disposition. Par un système de "glisser-déposer", l'administrateur peut agencer les activités selon les ressources disponibles (en terme de personnels ou de matériels). Cependant, l'agencement manuel des activités peut prendre un temps considérable, car pour chaque activité, il faut prendre en compte les activités qui peuvent la précéder ou la suivre, voir si le personnel est disponible, si les salles le sont également, etc. Lorsque le nombre de patients commence à se multiplier, la planification manuelle des activités devient un véritable casse-tête.

La planification automatique des activités permettrait d'avoir un premier agencement plus facile, et par conséquent, de gagner un temps considérable lors de l'élaboration des plannings. La notion d'automatisme peut être néanmoins très large, il sera important de la définir. Cela peut aller de la planification la plus simple, sans prendre en compte certaines contraintes et que l'utilisateur pourra corriger à sa guise, à une planification complète avec des options sélectionnées (agencement des activités au plus tôt de la journée, minimisation du temps d'attente pour les patients, pour les médecins, etc.). Une donnée importante est à prendre en compte : plus le type d'automatisation que l'on souhaite est précise ou détaillée, plus la modélisation nécessaire pour trouver une solution sera complexe. Ainsi, le but sera dans un premier temps de réaliser une première automatisation la plus simple possible puis, si les conditions sont réunies, faire évoluer le modèle et le complexifier davantage.

## 2.3 Implémentation de la planification temps réel

Le dernier objectif fixé lors des premières réunions est l'ajout d'une notion de temps réel à l'application. La plateforme pour le moment est conçue pour que l'utilisateur crée un planning pour une journée à un moment particulier (le matin avant les consultations ou le soir pour les rendez-vous du lendemain). Une fois le planning acté, il est difficile de le modifier en direct si un aléa se produit (par exemple, si un rendez-vous est annulé) à cause notamment des contraintes de précedence et de durée qui peuvent exister entre les activités.

L'idée serait donc d'ajouter des fonctionnalités permettant d'ajuster le planning en temps réel. Tout comme la planification automatique, cela peut impliquer des concepts différents qu'il est nécessaire de définir. Cela peut aller d'un simple indicateur sur le calendrier pour voir où nous

en sommes dans la journée, jusqu'à un système de pointage pour chaque activité. Il est possible d'imaginer un outil pour lequel les médecins pointent à chaque début de rendez-vous puis signalent dès que le rendez-vous est terminé sur la plateforme. Ainsi, il serait possible d'ajuster les activités en temps réel en cas de retard voire même en cas d'avance sur la planning.

Se pencher sur la planification en temps réel nécessiterait une refonte de la modélisation du système, notamment en ce qui concerne les activités. Compte tenu du temps alloué au projet et aux autres objectifs à réaliser, nous avons décidé de nous focaliser en priorité sur les deux premiers objectifs. Si les évolutions avancent plus rapidement que prévu, nous pourrions songer à implémenter certains éléments. Dans un premier temps, nous nous tiendrons au chiffrage initial du projet (chiffrage disponible en [Annexe D](#)).

### 3 Bases méthodologiques

À l'issue des premières réunions de lancement du projet, nous avons avec mon encadrant établi un chiffrage du projet avec les objectifs à réaliser et une estimation du temps de travail pour chaque activité. Ce chiffrage est visible en [Annexe D](#). Une fois le chiffrage effectué, j'ai établi un diagramme de Gantt pour planifier les étapes au long du projet. Le découpage des étapes peut s'apparenter à une méthode Agile, avec un sprint représentant une étape à réaliser. Les détails concernant le découpage et le cheminement du projet sont disponibles en [Annexe E](#).

En ce qui concerne l'organisation des tâches, j'utilise l'outil de gestion de projet *Trello*. Les détails concernant l'outil et l'utilisation que j'en ai fait sont disponibles en [Annexe E](#).

Concernant la méthodologie de développement, la plateforme a été réalisée en suivant un pattern MVC. Je vais donc reprendre le code en respectant l'organisation déjà établie.

# 2

## Description générale

Ce chapitre est consacré à la description générale de la plateforme. Nous évoquerons l'environnement autour du projet, les caractéristiques des futurs utilisateurs, les fonctionnalités du système ainsi que sa structure générale.

### 1 Environnement du projet

Comme évoqué dans le [Chapitre 1](#), ce projet est la reprise de plusieurs projets successifs effectués par les étudiants de Polytech. Il a été initié lors de l'année scolaire 2015-2016 avec un projet R&D et un projet collectif en simultané. Par la suite, plusieurs projets R&D se sont imbriqués chaque année de façon successive. La première modélisation et les premières briques de code datent donc de plusieurs années maintenant.

L'application existante fonctionne sur une machine locale et est déployée via la plateforme de développement Web WAMP. Cette plateforme permet de faire fonctionner localement des scripts PHP sur machine Windows. WAMP fait le lien entre 4 composantes comme indiqué dans son acronyme :

**Windows** comme le nom du système d'exploitation qui assurera la distribution des ressources entre les composants ;

**Apache** comme le serveur Web qui va déployer l'application et répondre aux requêtes Web du navigateur ;

**MySQL** comme le nom de la base de données utilisée pour l'application ;

**PHP** comme le langage de script utilisé.

Les développements lors de ce projet ont été effectués sur ma machine personnelle, sous Windows 10, et en utilisant le navigateur *Mozilla Firefox*. Le diagramme de déploiement de la [Figure 1](#) représente les liens entre le serveur, la base de données et l'application.

L'application si elle est mise en production sera disponible dans un premier temps sur une machine à la disposition de la personne chargée de la coordination du planning au sein du service. Les évolutions sur la plateforme pourront amener à déployer l'application sur plusieurs machines en parallèle au sein de l'hôpital. Par exemple, si l'on souhaite implémenter un système de pointage en temps réel des activités, il serait intéressant d'avoir plusieurs postes à disposition pour faciliter la tâche. Ici, nous nous focaliserons sur un déploiement sur une unique machine.

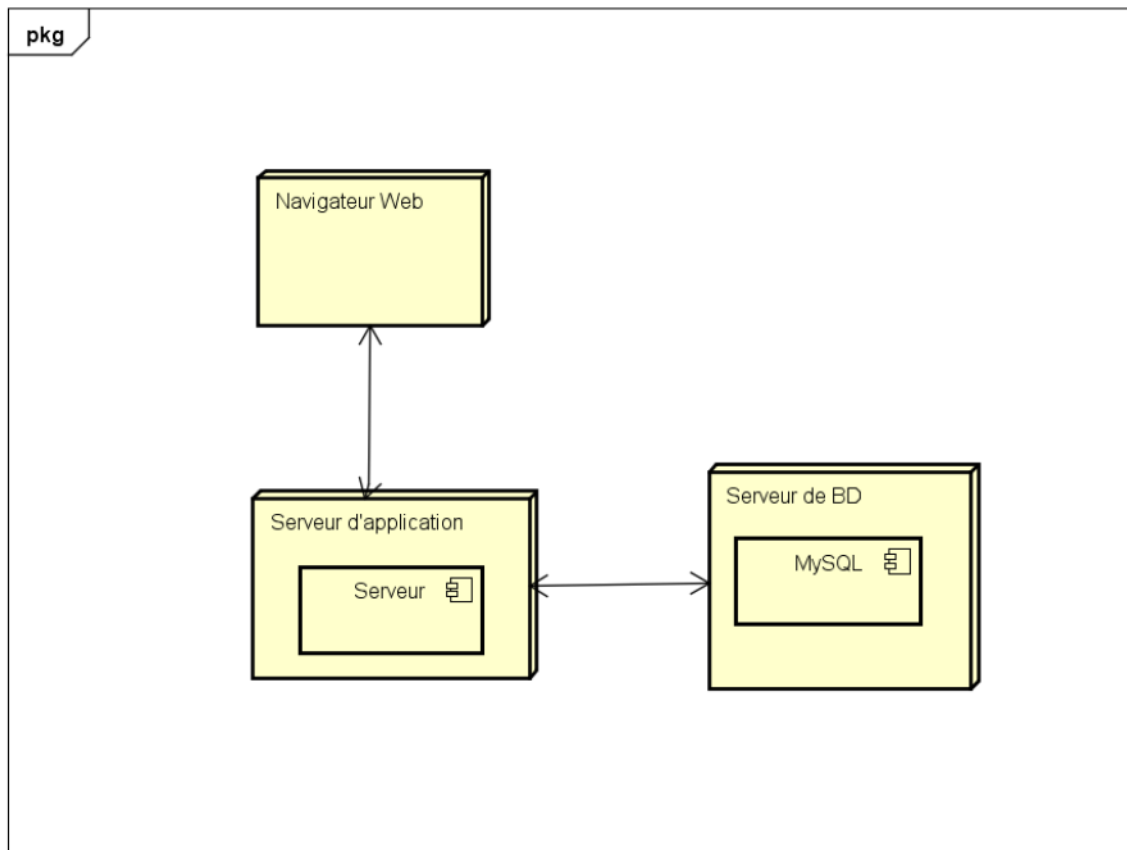


Figure 1 – Diagramme de déploiement

## 2 Caractéristiques des utilisateurs

La plateforme incorpore un système de gestion de compte afin de garantir la sécurité des données. Nous pouvons définir 2 types d'utilisateurs pour l'application.

- Le personnel de soins : ils auront accès à la majorité des fonctionnalités de la plateforme (l'ajout et la recherche de patients, l'accès à leur planning personnel et au planning des patients). Ils ne pourront pas effectuer de modification sur les rendez-vous ou sur le planning en général.
- Administrateurs et infirmières de coordination : ils disposeront de tous les droits sur l'application. Ils pourront réaliser les mêmes opérations que le personnel de soins et auront la possibilité de gérer les ressources, gérer les activités, créer et modifier les plannings, ajouter et modifier un parcours-patient, etc.

Les utilisateurs de l'application devront s'authentifier pour pouvoir accéder aux différentes fonctionnalités et selon leur grade, ils auront accès aux éléments avec lesquels ils pourront interagir.

## 3 Contraintes de développement

Étant donné que ce projet fait suite à plusieurs projets consécutifs et que la base de la plateforme a déjà été développée, il faudra utiliser les mêmes langages de programmes ainsi que les mêmes frameworks que lors des projets passés. Ainsi, les contraintes concernent les langages *HTML*, *CSS*, *Javascript*, *PHP*. La structure de l'application sera la même, à savoir une architecture MVC

(Modèle - Vue - Contrôleur) mise en place à l'aide du framework PHP *CodeIgniter*. En ce qui concerne la mise en page (CSS), nous utiliserons *Bootstrap*.

Guillaume POCHET, qui a effectué son PR&D sur le sujet lors de l'année scolaire 2016-2017, avait réalisé un état de l'art sur les bibliothèques utilisables pour mettre en place un planning interactif [4]. À la suite de cela, il avait décidé d'utiliser la bibliothèque *FullCalendar* pour constituer les plannings sur la plateforme. Nous poursuivrons avec l'utilisation de cette même bibliothèque.

De la même façon que pour les langages, il faudra reprendre la structure de la plateforme et le modèle de la base de données. Les détails concernant les structures seront dans la section suivante [Section 6](#).

## 4 Description des interfaces externes du système

### 4.1 Interface matériel/logiciel

L'application est une interface web qui fonctionne au travers d'un navigateur. Il faut donc disposer d'un ordinateur avec un navigateur web qui intègre *Javascript* pour pouvoir utiliser l'application. La présence de *Javascript* est nécessaire pour la visualisation du calendrier. Pour le moment, la plateforme est reliée une base de données qui doit être installée sur la machine et qui récupérera les différentes informations nécessaires au fonctionnement de l'application.

### 4.2 Interface homme/machine

Les utilisateurs accéderont à l'application par le biais d'un compte. L'authentification se fait à l'aide d'un nom d'utilisateur et d'un mot de passe. La création de compte est gérée par l'administrateur qui définira un type au nouveau compte créé.

On peut distinguer deux types d'utilisateurs qui ont été définis dans la [Section 2](#). Par conséquent, le type d'utilisateur définira les différents onglets à afficher ou non.

Les pages web et les onglets déjà présentes seront réutilisés lors de ce projet. Il n'y aura pas de nouvelles pages qui seront créées durant ce travail, néanmoins des modifications seront apportées aux pages actuelles, le tout en conservant l'aspect général et la charte graphique de l'application d'origine.

### 4.3 Interface logiciel/logiciel

Comme évoqué auparavant, l'application web est liée à une base de données via *MySQL*. La connexion à la base se fait au travers des classes de configurations du framework *CodeIgniter*. Les contrôleurs de notre modèle MVC vont faire la liaison entre la base et l'affichage des informations.

## 5 Fonctionnalités du système

Cette section regroupe les fonctionnalités qui vont être implémentées lors de ce projet. Les fonctionnalités déjà existantes se trouvent en [Annexe A](#). La suite présentera les fonctionnalités qui seront ajoutées lors de ce projet.

## 5.1 Ajout de fonctions de tri lors de l'affichage des ressources

Lors de la reprise du projet, sur les premiers tests, nous avons pu observer plusieurs corrections à apporter à la plateforme. L'une d'entre elles qui m'a semblé intéressante est l'ajout de fonctions de tri lors de l'affichage des ressources. Lorsque l'on souhaite consulter le personnel, les salles ou les activités par exemple, l'affichage se fait sous forme de liste comme on peut le voir sur la **Figure 2**.

Personnel de l'hôpital			
<a href="#">Ajouter une personne</a>			
Nom	Prénom	Poste	
Cheniour	Soumaya	IDE obésité	⋮
Attias	Elie	IDE	⋮
Cabinet	Selari	IDE	⋮
Pasquet	Jean-Pierre	Psychologue	⋮
Viomesnil	Vanessa	Psychologue	⋮
Guerrero	Jean-Marc	Dietéticien	⋮
Neuman	Caroline	Dietéticien	⋮
Teboul	Patrick	nutritionniste	⋮
Belhadj	Karim	nutritionniste	⋮
Jean	Roche	IDE	⋮
Mélanie	Fontaine	Psychologue	⋮

Figure 2 – Affichage du personnel sur la plateforme

Cependant, dès que le nombre d'éléments dans la liste est conséquent, il devient vite difficile de trouver l'élément que l'on souhaite, d'autant plus que ces onglets n'ont pas de fonctions de recherche d'éléments.

Réaliser des fonctions de recherche peut prendre beaucoup de temps et n'est pas une priorité majeure dans notre projet. Ainsi, le compromis pour retrouver plus facilement les éléments et ne pas perdre du temps sur des fonctionnalités non prioritaires est d'ajouter des fonctions de tris sur les onglets. L'idée serait d'avoir la possibilité de trier les éléments par nom et par spécificité selon l'onglet (tri par poste pour le personnel, tri par spécialité pour les salles, etc.).

## 5.2 Ajustement du planning selon les ressources sélectionnées

Selon le même principe que l'ajout de fonctions de tri sur les onglets de gestion, un des ajustements possibles qui améliorerait l'expérience utilisateur sur la plateforme est la possibilité d'ajuster le planning selon les ressources que l'on souhaite faire apparaître.

La mise en place du planning fait partie des fonctionnalités les plus importantes mais aussi des plus complexes de la plateforme. Sur l'onglet Planification, à la reprise du projet, toutes les ressources existantes sont affichées dans le calendrier, et ainsi on peut glisser-déposer les activités non attribuées sur la ressource que l'on souhaite. Or, le nombre de ressources présentes dans l'hôpital peut vite se multiplier, et la gestion du planning devient alors très vite difficile dans ce cas.

Rendre l'affichage plus lisible et plus facile d'utilisation pour l'utilisateur n'est néanmoins pas une mince affaire car chaque information présente sur la page est importante pour faire

le planning. Là où les améliorations peuvent se faire, c'est au niveau de la disposition des informations à l'écran. L'une des idées pour améliorer l'affichage est d'afficher uniquement les ressources que l'on souhaite faire apparaître à l'écran, c'est-à-dire afficher uniquement un type de personnel ou un type de salle précis. Cela permettrait de limiter les affichages à l'écran et de voir plus facilement les problèmes d'ordonnancement liés à un certain type de ressources. Par exemple, s'il y a un problème de gestion de salle (si une salle est réservée pour deux rendez-vous au même moment), en affichant uniquement les salles de l'hôpital, on peut voir directement où est le problème sans avoir à chercher parmi le flot d'informations que peut contenir la page principale.

### 5.3 Planification automatique

Comme évoquée dans les parties précédentes de ce rapport (notamment dans la [Section 2.2](#) (Chapitre 1)), l'élaboration du planning manuellement peut s'avérer très chronophage d'une part, car il faut glisser-déposer chaque activité du parcours pour chaque patient ce qui peut vite représenter plusieurs dizaines d'activités à agencer tout en prenant en compte chacune de leurs contraintes, et redondant d'autre part, car trouver un planning qui respecte les contraintes peut être long mais de plus ré-itérer cela tous les jours peut être éprouvant pour la personne en charge des plannings. La planification automatique va par conséquent être une fonctionnalité majeure de la plateforme.

La planification automatique peut se matérialiser de plusieurs manières sur l'application. Un bouton a déjà été ajouté sur la page de planification que l'on peut voir sur la [Figure 3](#) mais celui-ci est inactif. L'idée derrière cette implémentation réalisée lors des projets précédents est d'agencer automatiquement les activités non-attribuées sur le calendrier d'un clic sur le bouton à l'écran.

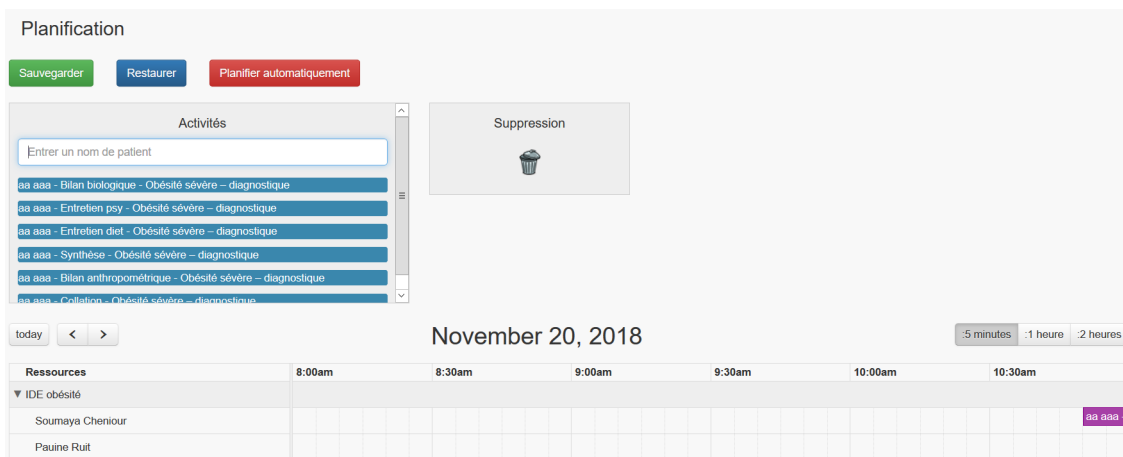


Figure 3 – Aperçu de la page de Planification

Toute la complexité de la fonctionnalité réside dans la notion que l'on souhaite donner à l'automatisme. Est-ce que l'on préfère trouver un planning qui minimise les temps d'attente des patients? Ou alors, est-il préférable de minimiser les temps d'attente entre chaque rendez-vous pour les médecins? Faut-il surcharger les salles qu'ont à *over-booker* comme on peut le trouver dans les aéroports? De nombreuses autres questions peuvent se poser à partir de ce problème.

Pour éviter les problèmes majeurs de complexité, nous allons commencer au plus simple : l'agencement patient par patient. L'idée derrière cela est d'avoir la liste des patients pris en charge le jour que l'on souhaite (comme on peut le voir pour les activités sur la [Figure 3](#)), de sélectionner un patient et de cliquer sur un bouton qui placera chacune de ses activités du jour sur le calendrier. Le placement se fera au plus simple dans un premier temps, c'est-à-dire ajuster

sur l'heure d'arrivée du patient et placer sur la première ressource disponible pour les activités. Même s'il peut y avoir des problèmes d'ordonnancement liés à cette méthode, la personne en charge pourra tout à fait ajuster l'agencement de la manière qui lui convient.

Mis en place de la sorte, même si la méthode ne donne pas un résultat parfait du premier coup, la fonctionnalité permettrait de gagner beaucoup de temps à l'utilisateur et de se débarrasser du côté redondant de l'agencement manuel. Selon l'avancement du projet, cette fonctionnalité pourra être étoffée et proposer de nouvelles options à l'utilisateur.

## 6 Structure générale du système

La structure générale du système est basée sur l'architecture MVC (Modèle - Vue - Contrôleur) par le biais du framework PHP *CodeIgniter*. On peut voir un aperçu de l'arborescence du projet sur la [Figure 4](#).

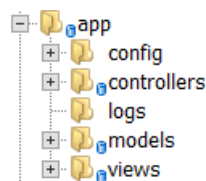


Figure 4 – Arborescence du projet

Sur l'arborescence, les 3 parties du MVC sont présentes ainsi qu'un dossier *config* qui regroupe les classes de configuration du framework et le dossier *logs* avec les classes de logger de l'application. La partie Modèle représente les données de la plateforme (patient, ressources humaines, ressources matérielles, parcours, activités, etc.). La partie Contrôleur est utilisée pour faire le lien entre les données et les différentes vues. Elle va également permettre de gérer la cohérence et l'intégrité des données. La partie Vue regroupent les pages web de l'application.

Concernant la base de données, nous reprendrons le modèle tel qu'il a été pensé lors des projets précédents. Les explications sur l'utilité des tables et le diagramme de Gantt se trouvent en annexe ([Annexe C](#) pour les explications sur les tables et [Annexe B](#) pour le diagramme). Des ajustements sur certains champs de la base seront apportées, notamment lors de la modification du formulaire de création de patient.



# 3

## État de l'art

À ce stade, le projet est avant tout un projet axé développement plutôt que recherche. La modélisation et les phases d'état de l'art sur la conception de la plateforme ont déjà été réalisées (en particulier, sur le PR&D de Jean Coquelet [1]). La phase de recherche qui va nous concerner ici sera sur la question des méthodes de planification. Comme évoquée dans les sections précédentes, il s'agit d'un problème complexe où de nombreuses questions peuvent se poser. Pour éclaircir certains aspects du problème, nous allons résumer les différentes avancées de la recherche concernant les problèmes multi-ressources dans un hôpital au travers d'un article publié sur l'*European Journal of Operating Research* au mois de mars 2018 [3]. Ici, nous allons voir cela au travers des démarches à effectuer pour réaliser un modèle permettant de résoudre les problèmes de planification.

### 1 Présentation des problèmes multi-ressources dans un hôpital

Dues aux constantes augmentations des frais hospitaliers et d'une demande accrue auprès des services de soins, les hôpitaux doivent trouver des moyens d'améliorer l'efficacité de leurs opérations. De nombreuses tentatives ont été réalisées ces dernières années pour développer de nouveaux plannings ou de nouvelles techniques d'admissions de patients. De ce fait, tout un panel de la recherche sur le sujet existe, que ce soit pour la planification de patient passant la nuit à l'hôpital ou non. Cependant, sur la planification, les recherches sont principalement limitées à un seul type de ressource ou un seul rendez-vous par patient. Dans la réalité, les patients doivent souvent effectuer de multiples diagnostics, consultations et/ou opérations pour être traités. De plus, traiter un seul type de ressource par patient ne correspond pas à la réalité car il est possible que le patient attende pour une autre ressource en même temps. De la même manière, un patient qui doit naviguer entre plusieurs services, implique nécessairement plus de ressources ce qui fait qu'utiliser un algorithme de planification limité à une ressource ajouterait du temps d'attente entre deux rendez-vous sur deux ressources différentes. Cela impacte le temps de traitement, et entraîne des effets négatifs sur le patient.

Ces dernières années, le nombre de travaux traitant le problème en considérant de multiples ressources est en augmentation. C'est un domaine relativement récent puisqu'aucune étude sur le sujet n'existait avant 1995. On les référence avec le terme anglais suivant : *multi-appointment scheduling problems in hospitals* (MASPH). Le MASPH est défini par le problème de planification des patients qui ont besoin de rendez-vous sur certaines ressources de l'hôpital et d'optimisation

des ressources de façon centralisé. Les ressources peuvent être des tests de diagnostic (scanner radio, IRM), des salles d'opérations, des médecins, etc. Chaque type de ressource peut être considéré comme un unique serveur ou comme multi-serveur. Les ressources peuvent faire partie d'un ou plusieurs services de l'hôpital.

Le concept de MASP est lié à plusieurs domaines de la recherche, dont notamment les suivantes : la prise de rendez-vous (*appointment scheduling*), le domaine de la médecine intégrée (*integrated healthcare*), les flux de patients (*patient flow*), la planification de ressources (*resource scheduling*) et la gestion des infirmiers (*nurse rostering*).

## 2 Les niveaux de décision dans la gestion des hôpitaux

Dans la littérature médicale, il existe 3 niveaux de décision : stratégique, tactique et opérationnel. Tout d'abord, le niveau stratégique concerne les décisions structurelles et s'adressent au long-terme en général. Dans le cas de la planification de patient, cela concerne le volume et la composition du personnel présent à l'hôpital, des ressources à acquérir et leurs localisations, et le nombre de patients à prendre en charge. Ensuite, au niveau tactique, les décisions prises sont des lignes directrices pour faciliter les décisions opérationnelles. C'est à ce niveau que sont allouées les capacités pour les ressources disponibles, comme par exemple les plages horaires de disponibilité d'une machine de radiologie. Enfin, les décisions opérationnelles concernent la planification jour après jour des patients et on peut le diviser en deux parties : en différé (ou *offline* c'est-à-dire les requêtes se traitent avant le jour du rendez-vous) ou en direct (*online*, les événements sont traités le jour même en direct et ne peuvent pas être prévus).

Il est clair que tous les niveaux de décision sont importants dans le cadre des MASP. Cependant, les principaux éléments de la recherche se font au niveau opérationnel, car bien que les niveaux stratégiques et tactiques sont nécessaires pour atteindre un résultat optimal, le niveau opérationnel constitue le cœur du problème de MASP.

## 3 Choisir la portée du modèle

Pour développer une méthode de planification sur de multiples rendez-vous, le premier moyen de décision est lié aux paramètres du problème, en l'occurrence le service concerné, les ressources relatives aux problèmes et le type de patient concerné.

### 3.1 Le service hospitalier concerné

Les MASP peuvent être appliqués pour de nombreux services différents dans un hôpital et par conséquent, les problèmes de planification selon les services peuvent être très différents les uns des autres. Parmi les services où peuvent s'appliquer les MASP, on en retrouve trois qui sont les plus étudiés dans la recherche.

Tout d'abord, une des applications se trouve dans les services de rééducation. Ces services traitent des patients souffrant de blessures physiques ou bien d'addictions diverses. Le traitement des patients nécessitent plusieurs spécialistes et matériels venant de nombreux domaines. La visite auprès de chacune des ressources doit être planifié avec soin, ce qui est parfois une tâche difficile impliquant de nombreux acteurs. Avec une planification manuel, sans coordination, les plannings résultant sont souvent très loin de la solution optimale du point de vue du patient. Étant donnée que la rééducation est un processus long, les patients rencontrent leurs spécialistes et utilisent les ressources à de multiples reprises. Ce qui signifie que la planification

dans les services de rééducation ne doit pas se focaliser uniquement sur l'agencement des tâches mais aussi sur les séries de rendez-vous successifs. Une autre difficulté dans le cadre de la rééducation est que certains spécialistes organisent des sessions groupées avec un horaire pré-défini. La rééducation peut concerner aussi bien les patients restants pour la nuit ou les patients en ambulatoire, avec dans les deux cas pour objectif, de terminer le traitement aussi tôt que possible. Pour les patients en ambulatoire, les chercheurs essaient principalement de programmer le plus de traitements possibles en un jour pour que le patient aille le moins possible à l'hôpital. En faisant de la sorte, la satisfaction des patients sera accrue et par la même occasion, le nombre de patients traités augmentera par rapport à une planification manuelle.

La deuxième application bien référencée par la recherche concerne la planification des tests de diagnostic. Ces tests ne prennent généralement pas beaucoup de temps, il est alors possible pour un patient d'effectuer plusieurs tests en un seul jour, ce qui permettrait aux médecins de diagnostiquer plus rapidement. De ce fait, l'objectif de la planification est de minimiser le temps de réalisation de toutes les étapes de tests. Cela peut être modéliser aisément par des méthodes de recherche opérationnelle comme l'*open shop*, le *flow shop* ou le *hybrid shop* selon les contraintes de précédence. Les tests sont suivis la plupart du temps par une consultation afin que le médecin puisse décider de la suite à donner aux traitements. Il est important de noter que tous les services de diagnostic n'utilise pas de système de rendez-vous pour planifier les patients. Certains utilisent les files d'attente pour notifier quand le patient sera admis.

Une troisième application des MASP se trouve dans les services d'oncologie. Bien qu'une partie de la recherche se base sur la planification des traitements de chimiothérapie et des radiothérapies, d'autres se sont focalisés sur la planification du processus de traitement entier, incluant les consultations avec les oncologues et/ou des phases de pré-traitements. Les recherches sont basées sur l'idée que les retards de traitement peuvent avoir un effet négatif sur la santé du patient. De ce fait, minimiser le temps de réalisation du traitement entier peut être très important.

Le nombre de services où les MASP ont été étudiés est assez limité. Cela peut s'expliquer par le fait que tous les services ne se basent pas sur une planification des patients. Par exemple, les services d'urgence utilisent davantage un modèle de file d'attente avec des priorités sur les ressources.

### 3.2 Le type de patient

Une fois que le service a été défini, le chercheur connaît le contexte du problème et quels attributs peuvent être modifiés. Cela permet de se focaliser sur le type de patient qui seront pris en charge. Trois types de patient peuvent être identifiés : les patients en ambulatoire (*outpatients*), les patients "traditionnels" c'est-à-dire qui restent la nuit à l'hôpital (*inpatients*) et les patients en urgence.

Tout d'abord, les cliniques ambulatoire traitent les patients qui ne passent pas la nuit à l'hôpital. Cela signifie que le patient rentre chez lui une fois que tous les services nécessaires lui ont été administrés. Le terme "clinique ambulatoire" peut faire référence à une clinique séparée (organisée autour d'une spécialité ou sur certaines conditions médicales) ou une sous-division d'un hôpital général pour lequel les consultations sont organisées sur certains laps de temps. Les principaux challenges qui se présentent au niveau de décision opérationnel dans la procédure de planification des patients sont les incertitudes sur les temps de services et les patients ne se présentant pas aux rendez-vous (communément appelé "no-show"). L'existence des no-shows peut s'expliquer par un oubli de la part du patient ou d'un problème de transport pour aller au rendez-vous. Pour réduire l'impact de ces problèmes, il existe des heuristiques qui, en comparaison avec un planning manuel, permettent de doubler le nombre de patients programmés par jour, entraînant des revenus à la hausse pour l'hôpital.

Ensuite, pour les patients passant la nuit à l'hôpital, des ressources supplémentaires sont à prendre en compte. La ressource principale à laquelle il faut penser est le lit. Ainsi, pour planifier les patients sur les ressources, une nouvelle contrainte sur le nombre total de lits dans le système s'ajoute. De plus, le temps de séjour d'un patient est un facteur important pour la rentabilité de l'hôpital. La plupart des travaux de recherche dans ce domaine cherche ainsi à minimiser les temps de séjour. Sachant que les patients sont déjà sur place, prendre en compte les préférences du patient pour l'heure des rendez-vous n'a pas d'intérêt. Des tentatives sur des problèmes d'hospitalisation d'une semaine sur un service de rhumatologie ont été réalisées et montrent que le nombre de patients admis peut être augmenté, entraînant de meilleures revenus pour le service.

Enfin, les patients en urgence ne font en majorité pas partie des sujets de MASPH car l'arrivée des patients n'est pas prévisible et ils doivent être traités immédiatement. Certaines exceptions existent avec des modèles qui planifient les patients dès que l'on sait quel traitement ils doivent subir. Un algorithme appliqué sur des données réelles de service d'urgence a donné des résultats améliorant les temps de réalisation des tests comparé aux temps réels.

De la cadre d'une implémentation avec tout type de patients, prendre en compte les patients en urgence est important lors de la planification des autres patients car une capacité suffisante de ressource doit rester disponible pour traiter les cas d'urgence. Si les capacités restantes ne sont pas suffisantes pour accueillir les urgences, cela entraîne un report de rendez-vous programmés qui se résulte en heures supplémentaires. Cela n'est pas désirable dans le cas des MASPH car un patient peut être attendu sur d'autres ressources par la suite, ce qui crée un cumul des retards.

### 3.3 Les ressources

Pour compléter la portée du problème, il est nécessaire de définir les ressources à prendre en compte pour modéliser au plus proche de la réalité le problème. Elles doivent être bien définies et validées car la complexité du problème en dépend fortement. Par exemple, prendre en compte les consultations avec un médecin dans le problème implique d'ajouter les disponibilités du docteur, ce qui peut faire décroître l'espace de recherche d'une solution comparé à une situation où toutes les ressources sont disponibles continuellement. Pour réduire la complexité, les chercheurs simplifient certaines suppositions, en fixant un temps de traitement pour tous les patients par exemple.

## 4 Choisir ce qu'on veut optimiser

Arrivé à ce stade, les contraintes et les paramètres du modèle sont connus du chercheur. Il est alors temps de se poser la question sur ce que l'on doit optimiser et la définition de la performance pour un hôpital. Un des points de vue est d'augmenter la satisfaction des patients. La prise en charge rapide est un élément majeur dans la satisfaction d'un patient. Bien que de nombreuses recherches partent de ce principe, un autre point de vue peut entrer en rigueur : la maximisation des profits de l'hôpital. En effet, les hôpitaux font face à de nombreuses coupes budgétaires et se doivent d'être moins dépensiers. Dans les deux cas, les MASPH ont été prouvés efficaces. Il est possible de combiner les deux en assignant des poids pour chaque fonction objectif et avec plusieurs étapes d'optimisation. De cette manière, les hôpitaux peuvent ainsi choisir ce qui leur conviennent le mieux.

## 5 Choisir comment optimiser

À ce stade, le modèle est complet. Le but maintenant est de voir quelle méthodologie est la plus appropriée pour atteindre l'objectif. Il est important de rappeler que la communication avec l'hôpital concerné est primordial pour maximiser les chances d'implémentation du modèle dans la réalité. Ainsi, au-delà d'une technique de planification, il faut avoir aussi une stratégie de planification. Comme nous l'avons vu auparavant, on peut faire deux distinctions entre une planification *online* et *offline*. D'un côté, le modèle répond immédiatement avec une date et une heure pour les rendez-vous. Cela implique que la planification se fait de façon séquentielle avec les patients obtenant un rendez-vous à leur demande. De l'autre côté, les planificateurs collectent les rendez-vous et forment une liste d'attente depuis laquelle un algorithme est appliqué pour sélectionner les patients sur la liste. Choisir la stratégie de planification n'est pas une tâche aisée car les deux stratégies impliquent un mode opératoire différent pour l'hôpital. En utilisant une liste d'attente, il faut prendre en compte le fait que les patients ne doivent pas rester trop longtemps dans la liste, ce qui causerait une baisse de satisfaction du patient. De plus, dans le cas des cliniques ambulatoires, il y a un risque que les patients non traités se présentent au service d'urgence à la place car ce service traite en général tous les patients.

Pour choisir une méthodologie de planification, plusieurs options se présentent. Certaines proposent des solutions optimales, d'autres des solutions approchées. Le dilemme ici est de trouver un équilibre entre qualité de la solution et temps de calcul. Dans le cas d'une stratégie de planification *online*, la méthode doit être appliquée à chaque nouvelle requête. Par conséquent, les temps de calcul doivent rester courts, limitant le modèle à l'utilisation d'heuristique, ce qui atteint rarement la meilleure solution. Pour le cas de la planification *offline*, le nombre de patients à prendre en compte est plus grand, la complexité en sera ainsi accrue. Tout comme la planification *online*, la recherche d'une solution optimale relève souvent d'une approche à laquelle on ne peut qu'aspirer. Les recherches traitant des méthodes optimales sont ainsi restreintes comparées aux recherches sur des solutions approchées.

## 6 Appliquer et valider le modèle

Les dernières étapes consistent à valider le modèle pour vérifier s'il s'exécute comme prévu en utilisant des benchmarks ou des données réelles provenant d'un hôpital. Pour les résultats positifs, le modèle peut être appliqué dans la réalité. Cependant, il semble y avoir une contradiction dans les résultats de la recherche entre le nombre de modèles validés et le nombre de modèles implémentés en situation réelle. Bien qu'une partie significative de modèles dans la littérature est validée, seule une infime fraction est implémentée dans la réalité. Cela pourrait s'expliquer par une implémentation qui se fait après la date de publication par exemple. Toutefois, le fait que peu de publications ont des résultats appliqués dans la pratique a des implications sur le concept de MASP en général. Comme peu de résultats sont disponibles, cela devient plus difficile de prouver que les MASP sont efficaces. Par conséquent, il devient plus difficile de convaincre les hôpitaux d'implémenter de nouvelles méthodes. Les études qui ont été implémentées en situation réelle ne reportent pas toujours les résultats de l'implémentation, car soit elle n'est pas complète, soit aucune donnée de performance n'est disponible avant l'implémentation. Des signes préliminaires positifs sont néanmoins apparus dans des travaux, comme dans les services d'oncologie par exemple. Peu d'études contrôlent les performances avant implémentation et rapportent les améliorations après coup. Néanmoins, afin d'appliquer une nouvelle méthodologie, il reste compliqué pour les autres chercheurs et praticiens de voir quelle méthode est meilleure qu'une autre en terme de performance.

## Deuxième partie

# Développement

Cette partie est consacrée aux développements réalisés sur la plateforme. Nous allons dans un premier temps évoquer les éléments d'analyse et de conception des nouvelles fonctionnalités de l'application, puis nous détaillerons les aspects de la mise en œuvre des développements.

# 4

## Analyse et conception

Dans ce chapitre, nous verrons les éléments d'analyse sur les fonctionnalités de la plateforme. Ils sont séparés en plusieurs étapes correspondant au chiffrage réalisé lors des premières réunions. La gestion de projet en général ainsi que le chiffrage se trouvent en [Annexe D](#).

Voici les phases d'analyse que nous allons voir par la suite :

- Révision de la suppression des ressources
- Révision de l'onglet *Plan de parcours*
- Révision du formulaire de création de patient
- Affichage du planning patient
- Ajout des fonctions de tri
- Correction des problèmes d'URL
- Affichage des ressources sélectionnées dans le planning
- Implémentation de la planification automatique
- Génération de jeux de données pour les tests

### 1 Révision de la suppression des ressources

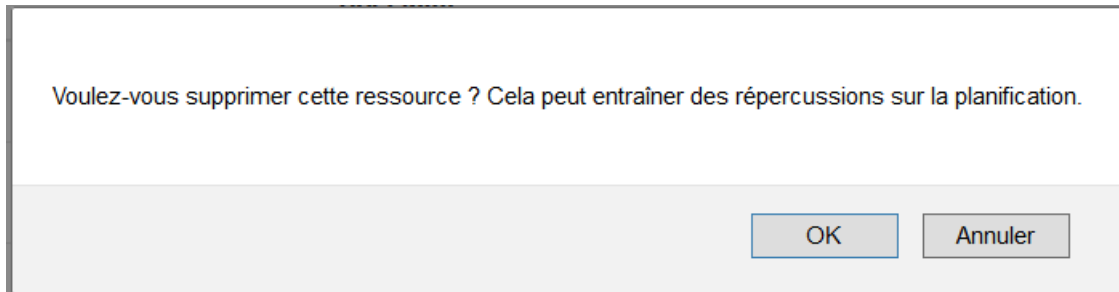
Les fonctionnalités de suppression des ressources ont été ajoutées lors du projet R&D précédent sur le sujet réalisé par Yang JING. Celles-ci sont implémentées dans les onglets de gestion de la plateforme (réunis sous l'onglet "Gérer") et permettent de supprimer de la base de données les ressources qui ne sont plus nécessaires ou qui sont devenues obsolètes au fil du temps.

L'implémentation de la suppression des ressources comporte des risques importants pour la plateforme. En effet, la suppression d'un élément peut entraîner des répercussions sur l'intégralité de l'application. Par exemple, si l'on souhaite supprimer une ressource mais que des activités sont planifiées avec cette même ressource, quelle conséquence y aura-t-il sur l'intégrité des données ? Ou encore, si on désire supprimer une activité, mais qu'un patient doit réaliser un parcours avec cette même activité, quel impact cela aura sur la planification ? Ce type de comportement sur la plateforme n'est pas souhaitable et serait même contraire au bon fonctionnement de l'application, mais une erreur de manipulation pourrait engendrer ce type d'action.

Le problème est que, à la reprise du sujet, les boutons "Supprimer" sur la plateforme ne disposent d'aucune sécurité, ce qui peut se révéler très dangereux en cas d'erreur de manipulation. Un

simple clic sur le bouton, même involontaire, entraîne la disparition immédiate de la ressource sélectionnée. Compte tenu de la dangerosité de la manœuvre sur le bon fonctionnement de la plateforme, il est nécessaire de pallier à ce problème.

L'une des solutions à minima serait de faire apparaître un message d'avertissement lors du clic sur le bouton. Ainsi, ça permettrait déjà d'éviter les clics involontaires. Un message comme affiché sur la **Figure 1** serait déjà une première étape.



**Figure 1** – Exemple de message d'avertissement en cas de suppression

Une autre solution plus coûteuse à mettre en place serait de cacher davantage les options de suppression. En effet, la suppression de ressource devrait être une fonctionnalité plutôt rare à utiliser, une ressource devenant obsolète qu'en cas de changement de parcours. L'idée serait de placer les options de suppression sur une page séparée disponible uniquement via les paramètres du compte de la personne. Bien sûr, les suppressions peuvent être réalisées uniquement par les personnes ayant les droits adéquates sur la plateforme.

## 2 Révision de l'onglet *Plan de parcours*

L'onglet "Plan de parcours" permet d'ajuster la nombre de patients qu'un parcours-patient peut accueillir par jour. La **Figure 2** représente l'affichage tel qu'il était à la reprise du sujet. La première remarque que l'on peut faire est que la notion de "Plan de parcours" semble floue. On ne comprend pas vraiment quel est l'utilité de l'onglet. Ajouter un texte permettant de comprendre la notion de "Plan de parcours" serait intéressant pour commencer.

Attribution du nombre de patients par parcours

Liste de parcours			Sauvegarder
Tous	Parcours	Jour	NB Patient
	Obésité sévère – diagnostique	1	5
	Obésité sévère – diagnostique	2	4
	Obésité sévère – diagnostique	3	6
	Obésité sévère – diagnostique	4	6
	Obésité sévère – diagnostique	5	6
	Obésité sévère – Post-op	1	10
	Obésité sévère – Post-op	2	10

**Figure 2** – Aperçu de l'onglet "Plan de parcours" à la reprise du projet

Ensuite, l'affichage des parcours se fait sous forme de liste. Pour chaque parcours, nous avons cinq lignes disposées, chacune représentant un jour de la semaine. Les parcours sont d'abord tous affichés à la suite sur l'onglet. Une liste déroulante est disponible sur le côté de la liste des parcours pour sélectionner le parcours que l'on souhaite ajuster. À première vue, l'affichage



ne semble pas très intuitif pour un non-initié. Un affichage plus clair, avec des aides pour l'utilisateur, permettrait de comprendre facilement les fonctionnalités de l'onglet.

Parmi les améliorations sur l'affichage, la mention des jours de la semaine serait intéressante. Pour le moment, comme on le voit sur la [Figure 2](#), les jours sont symbolisés par des numéros de 1 à 5. Plutôt que d'assigner un numéro, la mention des jours en toute lettre permettrait de gagner en clarté.

Le nombre de patients par jour et par parcours est disposé à droite de la liste. Cependant, il existe ce qui semble être une erreur sur le décompte. Il est possible d'ajuster le nombre avec des flèches disposées à droite du champ. Toutefois, il est impossible de dépasser 10 patients avec ce compteur, ce qui n'est pas normal. Une correction est donc à apporter sur ce champ.

### 3 Révision du formulaire de création de patient

L'ajout d'un patient dans la base de données se fait à partir de l'onglet "Création" dans le menu "Patient". À partir de l'onglet, on arrive sur un formulaire à remplir pour renseigner les données du patient. L'aspect du formulaire se trouve sur la [Figure 3](#).

Ajout d'un nouveau patient.

Nom	<input type="text"/>
Prénom	<input type="text"/>
Numéro de rue	<input type="text"/>
Rue	<input type="text"/>
Code postal	<input type="text"/>
Ville	<input type="text"/>
Pays	<input type="text"/>
Email	<input type="text"/>

Figure 3 – Formulaire de patient

Plusieurs améliorations peuvent être effectuées sur ce formulaire, au niveau de l'affichage et des données à remplir notamment. Tout d'abord, les données à remplir vont dépendre des demandes de l'hôpital et seront quoiqu'il arrive modifiables. Néanmoins, modifier le formulaire implique de modifier la structure de la table "Patient" dans la base de données également.

Les champs du formulaire ne disposent pas de formatage pré-disposé pour le champ en question, c'est-à-dire qu'il est possible, par exemple, de mettre des lettres sur le champ "Numéro de téléphone". L'ajout d'un formatage par défaut sur certains champs permettraient d'éviter les erreurs d'entrées dans la base de données (par exemple, une syntaxe d'adresse mail, un nombre défini de chiffres pour le numéro de téléphone et de sécurité sociale, etc.).

### 4 Affichage du planning patient

L'affichage du planning d'un patient est une fonctionnalité qui a été présentée dans les projets précédents. Cependant, lors du test initial de la plateforme, cette fonctionnalité ne fonctionnait pas. L'affichage de planning d'un patient consiste à afficher les activités prévues pour la journée d'un patient, avec quatre informations pour chaque activité : le nom, le personnel qui s'occupera de l'activité, l'heure de début et de fin prévue. La présentation du planning est visible par la suite sur la [Figure 4](#).

Votre Planning :

Nom de l'activité	Personnel médical	Heure début prévue	Heure fin prévue
<div>Informations</div> <p>Les dates de début et fin de chaque activité peuvent être amenées à changer.</p>			

Figure 4 – Affichage du planning patient

L’affichage sur le planning patient se fait à partir de la planification générale des activités. Une fois que les activités ont été positionnées sur le planning et que celui-ci a été enregistré, les activités doivent devenir visibles sur le planning du patient.

Des éléments supplémentaires pourrait compléter cet onglet et afficher ainsi plus d’informations à l’écran. L’idée de base est d’afficher uniquement les activités prévues pour le jour même de la consultation. Ajouter un historique des consultations passées et un aperçu des activités futures seraient intéressant, tout comme proposer un visuel sur les activités prévues pour le jour-même mais qui n’ont pas été placées dans le calendrier général.

## 5 Ajout des fonctions de tri

L’affichage des différentes ressources se fait sous forme de liste. Cependant, il peut y avoir un nombre important de ressources présentées mais il n’existe aucun tri possible pour ordonner l’affichage des entrées comme évoqué dans la [Section 5.1](#) (Chapitre 2).

Par défaut, les ressources sont agencées selon leur *id* dans la base de données, ce qui est une donnée invisible pour l’utilisateur. Le but serait d’ajouter une possibilité de tri pour chaque champ affiché sur la page, par exemple pour l’affichage du personnel, l’utilisateur pourra trier par *Nom*, *Prénom* et par *Poste* en cliquant sur les titres des colonnes correspondantes comme cela se fait sur de nombreuses applications de gestion.

## 6 Correction des erreurs de redirection

Il existe une erreur concernant la gestion des URL sur la plateforme. L’application étant une plateforme accessible via un navigateur, le parcours du site se fait au travers d’URL. Toutefois, il existe un problème au niveau de ses URL lorsque certaines actions sont effectuées. Lors de la modification d’une ressource, on la sélectionne sur l’onglet correspondant, à l’adresse `/RessourcesMat` par exemple, puis on clique sur *Modifier*. On arrive sur une page de modification avec l’URL `/modifier/id` avec l’*id* de la ressource. Une fois que l’on confirme la modification, en cliquant sur le bouton *Confirmer* de la page, on est alors redirigé vers la page des ressources en question mais avec l’adresse `/RessourcesMat/confirmModifier`, comme on peut le voir sur la [Figure 5](#).

Ce type d’erreur peut sembler anodin, mais il peut créer des dysfonctionnements sur la plateforme, notamment en cas d’actualisation de la page ou de retour arrière. La correction de ce bug peut se faire simplement dans le code en changeant les adresses de redirection des boutons de confirmation lors de la modification d’un élément.

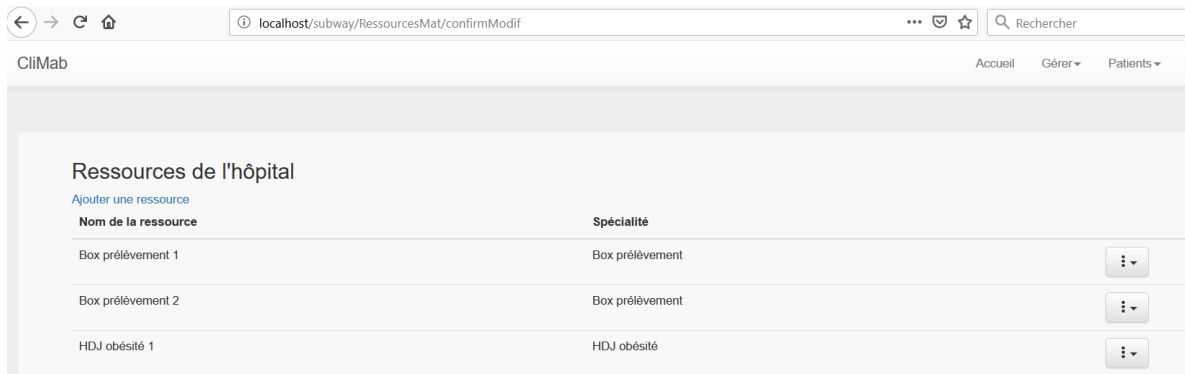


Figure 5 – Affichage d’une erreur de redirection

## 7 Implémentation de la planification automatique

L’implémentation de la planification automatique se fera en plusieurs étapes. Tout d’abord, une refonte de certains éléments graphiques sur la page de planification sera nécessaire. La Figure 3 (Chapitre 2) montre la page telle qu’elle est à l’origine du projet. La première chose que l’on remarque sur la page est la tuile centrale *Suppression*. Celle-ci permet de supprimer une activité placée sur le calendrier en faisant glisser cette activité sur la zone de la tuile. Bien que l’idée de base est intéressante, l’application est moins pratique. À première vue, un utilisateur ne peut pas savoir le fonctionnement de la tuile puisque rien n’est indiqué sur la page. Ensuite, la position centrale prépondérante de la tuile n’aide pas à rendre la page ergonomique.

Une des solutions serait de supprimer cette tuile et d’introduire un autre système de suppression. Une activité doit être placée quoiqu’il arrive, donc au lieu de faire glisser une activité que l’on souhaite enlever sur la tuile *Suppression*, avoir la possibilité de la faire glisser directement sur la liste des activités qui ne sont pas placées permettrait de rendre la tuile *Suppression* inutile, et ainsi elle pourrait être retirée de la page.

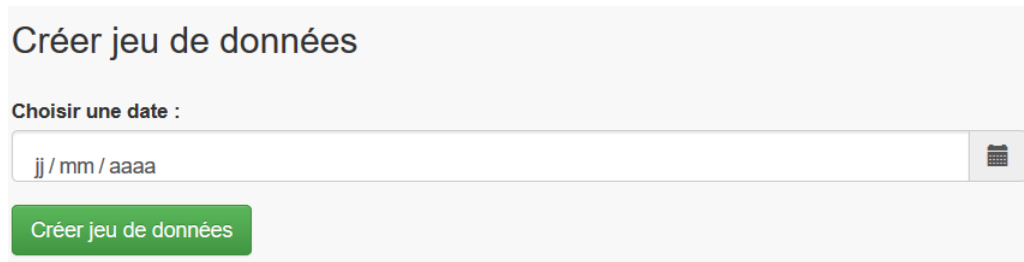
Ensuite, dans le cadre de la planification automatique, nous voulons placer les activités de la façon la plus simple possible dans un premier temps, à commencer par une affectation patient par patient. Pour ce type d’affectation, il faudra ajouter une liste de patients prévu pour la journée sur la page. Celle-ci peut être placée à côté de la liste des activités, et en cliquant sur un patient, la possibilité de l’affecter automatiquement sera affichée.

Enfin, le bouton *Planifier automatiquement* comme on le voit sur la Figure 3 (Chapitre 2), qui ne fonctionne pas à l’heure actuelle, resterait en place et permettrait d’affecter automatiquement tous les patients à toutes les ressources. Cette fonctionnalité sera implémentée une fois les autres éléments réalisées et sera étoffée selon le temps restant pour le projet.

## 8 Génération de jeux de données pour les tests

Sur la plateforme, on retrouve un onglet nommé *Créer jeu de données*. Sur cette page, on retrouve un panneau proposant de choisir une date et un bouton avec pour intitulé *Créer jeu de données* comme on le voit sur la Figure 6. Cet onglet doit servir pour les tests de la planification. Rien n’a encore été implémenté pour le moment sur cette page.

La création de jeu de données va permettre de générer des patients avec des activités prévues pour la journée sélectionnée sur l’onglet. Sans ces jeux de données, le test de la planification est fastidieux puisqu’il faut créer manuellement plusieurs patients, puis leur attribuer des parcours à chacun, ce qui peut prendre plusieurs minutes. Ainsi, ce bouton ferait office de macro pour



Créer jeu de données

Choisir une date :

jj / mm / aaaa

Créer jeu de données

**Figure 6** – *Affichage de la page de création de jeu de données*

les tests. Rendre ces tâches automatiques ferait gagner ainsi beaucoup de temps lors des tests sur la plateforme. Bien sûr, cet onglet ne doit pas être accessible à tous les utilisateurs d'une part, mais aussi ne doit pas être fonctionnel en cas de mise en production.

# 5

## Mise en œuvre

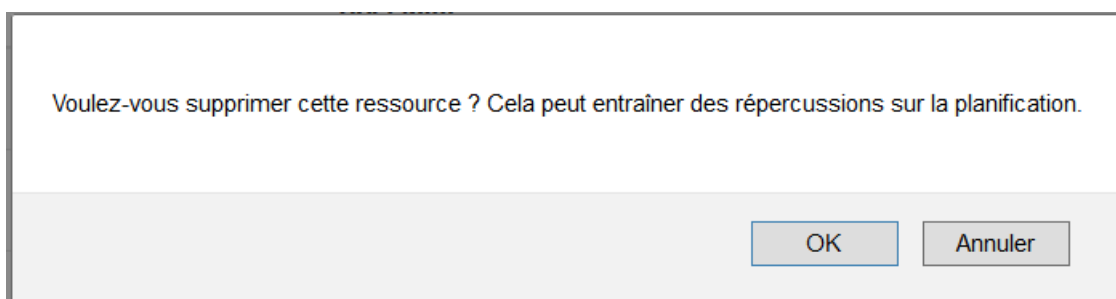
Le chapitre suivant est consacré aux mises en place réalisées lors des phases de développement conformément aux analyses effectuées dans les parties précédentes. Seuls les éléments de mise en œuvre les plus pertinents pour le rapport seront évoqués par la suite.

### 1 Ajout d'un avertissement lors de la suppression d'un élément

Conformément à ce qui était évoqué dans la [Section 1](#) (Chapitre 4) de l'analyse, l'ajout d'une fenêtre d'avertissement lors de la suppression d'une ressource a pour but d'ajouter une sécurité en cas d'erreur de manipulation. En effet, supprimer un élément peut avoir des conséquences importantes pour l'intégrité des données et la cohérence générale de l'outil.

Ajouter un message d'avertissement peut être implémenté facilement sur les différentes vues de l'application. Pour afficher un message de confirmation, il suffit d'ajouter le code suivant : `onclick="return confirm();" avec le message désiré sur le lien de l'action que l'on souhaite dans une balise HTML <a>, en l'occurrence la suppression dans notre cas. L'ajout du message va concerner les onglets de gestion de ressource (Personnel, Ressources Matérielles, Activités et Parcours) ainsi que l'onglet d'affichage des patients.`

Le message d'avertissement est similaire à celui de la [Figure 1](#).



**Figure 1** – Message d'avertissement en cas de suppression

## 2 Révision du plan de parcours

Les opérations sur l'onglet *Plan de parcours*, comme évoquées précédemment dans la [Section 2](#) (Chapitre 4), consistent à afficher les jours de la semaine en toute lettre et augmenter la limite du nombre de patient par parcours. La récupération de ces informations se fait via une requête sur la base de données.

Pour les noms de jour, la requête SQL d'origine récupère seulement l'identifiant du jour de la semaine dans la table *planparcours*. Afin d'afficher les noms en toute lettre, la requête a été modifiée en ajoutant une jointure avec la table *jour* qui comprend les noms des jours de la semaine. Les deux requêtes sont affichées sur la [Figure 2](#), avec en rouge la requête d'origine et en vert la requête modifiée.

```
$txt_sql = "SELECT PL.ID_PARCOURS, TXT_NOM, ID_JOUR, INT_NB_PATIENT
FROM planparcours PL, parcours PA
WHERE PL.ID_PARCOURS=PA.ID_PARCOURS AND PL.ID_PARCOURS=" . $id;
$txt_sql = "SELECT PL.ID_PARCOURS, TXT_NOM, TXT_JOUR, INT_NB_PATIENT
FROM planparcours PL, parcours PA, jour JO
WHERE PL.ID_PARCOURS=PA.ID_PARCOURS AND PL.ID_JOUR=JO.ID_JOUR
AND PL.ID_PARCOURS=" . $id;
```

Figure 2 – Requetes SQL pour récupérer les informations du plan de parcours

Après modification des vues dans le code, le nouvel affichage est le suivant ([Figure 3](#) en comparaison avec l'ancien affichage sur la [Figure 2](#) (Chapitre 4).

Parcours	Jour	Nb de patients
Obésité sévère – diagnostique	vendredi	6
Obésité sévère – diagnostique	lundi	24
Obésité sévère – diagnostique	mardi	4
Obésité sévère – diagnostique	mercredi	6

Figure 3 – Nouvel affichage de l'onglet Plan de Parcours

En ce qui concerne la limitation sur le nombre de patients par parcours, la modification se fait au niveau de la classe *V\_planparcours* correspondant à la vue sur notre modèle MVC. La correction à apporter se fait au niveau de la balise HTML `<td>` représentant les lignes du tableau qui s'affiche. Il est possible d'ajouter un minimum et un maximum sur le compteur, et ce dernier était situé à 10. Bien que pour dans la majorité des cas, le nombre de patients par jour et par parcours sera inférieur à 10, augmenter cette limite permet ainsi de prendre en compte toutes les exceptions.

## 3 Corrections sur le formulaire de création de patient

Les règles des champs du formulaire de création de patient comportaient certaines erreurs pouvant engendrer des données erronées. Comme évoqué dans la [Section 3](#) (Chapitre 4), les champs posant problème dans notre cas sont le numéro de rue, le numéro de téléphone fixe, le numéro de téléphone portable et le numéro de sécurité social du patient.

La définition des règles du formulaire se fait au sein du contrôleur du patient, `Patient.php`. Elle se fait simplement en utilisant la méthode `set_rules()` prédisposée du framework *CodeIgniter*. Les règles sont ensuite définies en précisant la colonne concernée en premier paramètre telles qu'afficher dans la **Figure 4**.

```
$this->form_validation->set_rules('nom-patient', 'Nom', 'trim|required|max_length[255]|alpha');
$this->form_validation->set_rules('prenom-patient', 'Prénom', 'trim|required|max_length[255]|alpha');
$this->form_validation->set_rules('num-add-patient', 'Numéro de rue', 'trim|required|max_length[5]|numeric');
$this->form_validation->set_rules('rue-add-patient', 'Rue', 'trim|required|max_length[255]');
$this->form_validation->set_rules('cp-add-patient', 'Code postal', 'trim|required|max_length[5]|numeric');
$this->form_validation->set_rules('ville-add-patient', 'Ville', 'trim|required|max_length[255]|alpha');
$this->form_validation->set_rules('pays-add-patient', 'Pays', 'trim|required|max_length[255]|alpha');
$this->form_validation->set_rules('email-patient', 'Email', 'trim|required|max_length[255]|valid_email');
$this->form_validation->set_rules('num-fixe-patient', 'Téléphone fixe', 'trim|required|exact_length[10]');
$this->form_validation->set_rules('tel-port-patient', 'Téléphone portable', 'trim|required|exact_length[10]');
$this->form_validation->set_rules('num-secu-patient', 'Numéro de sécurité sociale', 'trim|required|exact_length[15]|numeric');
$this->form_validation->set_rules('date-naiss-patient', 'Date de naissance', 'trim|required|max_length[255]');
```

**Figure 4** – Syntaxe des règles du formulaire de création de patient

Les champs numériques sont précisés par le mot-clé `numeric`, tandis les champs obligatoires sont annotés comme `required` dans les règles. Pour la gestion de la taille de l'entrée, il est possible d'utiliser `exact_length[]` pour donner une taille exacte ou encore `max_length[]` pour préciser le nombre maximal de chiffre qui peut figurer pour le champ.

## 4 Fonction de tri sur les onglets de Gestion

La mise en place des fonctions de tri sur les onglets de gestion se fait par le biais d'un script qui se déclenche lorsque l'utilisateur clique sur la colonne qu'il souhaite trier. Ce script est implémenté sur chacune des vues ayant besoin d'une fonction de tri, c'est-à-dire concernant les onglets *Activités*, *Personnels*, *Ressources Matérielles* et *Parcours*.

Le script de tri utilise les capacités de Javascript pour comparer les chaînes de caractère, notamment par l'utilisation des opérateurs `<` et `>` qui permettent d'indiquer si une chaîne se situe avant ou après une autre dans le dictionnaire, rendant l'implémentation dans le projet beaucoup plus simple. La **Figure 5** représente un exemple d'affichage des activités triées par nom d'activité.

Activités existantes		
<a href="#">Ajouter une activité</a>		
Nom Activité	Durée (en minutes)	Personnels
Administration chimio	15	IDE chimio : 1
ARM	15	
Bilan anthropométrique	60	Interne obésité : 1
Bilan biologique	20	IDE : 1

**Figure 5** – Affichage des activités triées par nom

## 5 Planification automatique

La planification automatique des événements représente la partie la plus importante des implémentations sur ce projet. Pour cette occasion, nous allons voir dans un premier temps le fonctionnement de l’affichage graphique de l’onglet *Planification*, avant de voir les différentes corrections nécessaires et de voir l’ordonnancement automatique des activités.

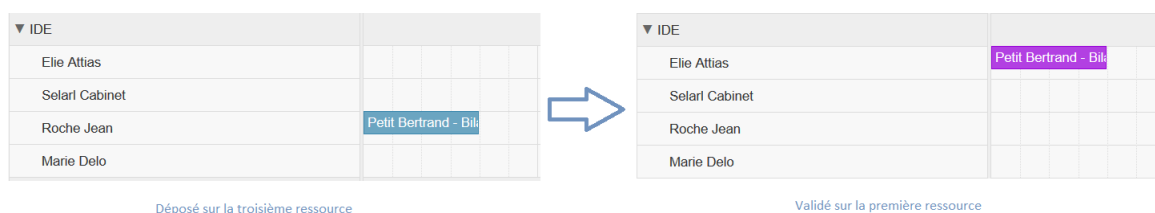
### 5.1 Fonctionnement de l’affichage graphique

L’affichage du calendrier se fait par le biais de la bibliothèque *fullCalendar*. Celle-ci permet d’obtenir un affichage dynamique des événements et des possibilités de manipulation manuelle facilitées. Les requêtes permettant d’ajouter, de supprimer ou de déplacer des événements sur le planning sont effectuées en AJAX, ce qui permet d’obtenir un rendu rapide sans passer par un rafraîchissement de la page à chaque action de l’utilisateur.

Les scripts de sauvegarde et de restauration du planning, ainsi que la planification automatique par la suite, font également appel à AJAX pour appeler les méthodes définies dans les contrôleurs. Une fois les événements placés et enregistrés en base, que ce soit via un glisser-déposer ou via un clic sur le bouton de planification automatique, les scripts font appel à une fonction de *fullCalendar* qui permet d’afficher les nouveaux événements sur le calendrier intitulée *refetchEvents*.

### 5.2 Corrections sur la planification d’origine

Avant de procéder à l’implémentation de la planification automatique des activités, plusieurs erreurs étaient à corriger. Tout d’abord, l’outil a été pensé pour que lors de l’ajout d’une activité, celle-ci soit placée sur toutes les premières ressources libres disponibles dont elle a besoin. Cependant, ce principe était utilisé pour tous les cas y compris lorsque l’on souhaite sélectionner une ressource en particulier. Cela signifiait que si une personne désirait placer une activité sur la troisième ressource alors que la première est disponible, l’activité était replacée directement sur la première ressource. Cet exemple est illustré dans la **Figure 6**.



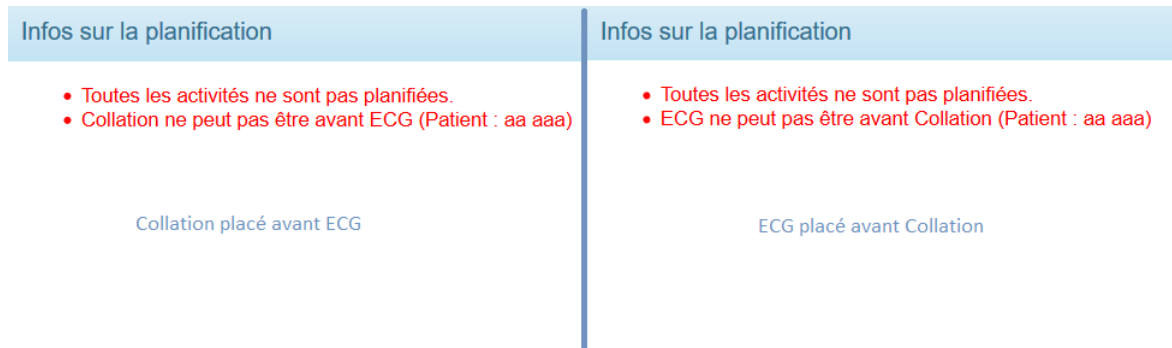
**Figure 6** – Exemple d’erreur de sélection de ressource

Pour résoudre cette erreur, j’ai implémenté une méthode intitulée *getDisponibiliteRessource()* qui permet de vérifier si la ressource sélectionnée est déjà utilisée ou non pour l’intervalle de temps indiqué. Si celle-ci n’est pas affectée à une autre activité sur cet intervalle, alors on y place l’activité.

La seconde correction concerne l’affichage des contraintes. Une vingtaine de parcours différents seront traités au sein de l’hôpital, et les activités peuvent donc avoir des contraintes de précedence différentes selon la nature du parcours. Ainsi il peut exister des cas particuliers où une activité précède une autre dans un parcours et succède à cette même activité dans un autre parcours. Le problème vient du fait que la méthode qui détecte les erreurs de placement



des activités ne prenait pas en compte le parcours pour déterminer si une activité est correctement placée par rapport à une autre. Un exemple avec la **Figure 7** montre que peu importe le placement de l'activité *ECG* par rapport à *Collation*, une erreur va être affichée, car dans le parcours *Diagnostic Obésité*, *ECG* est placé avant la collation tandis que pour un autre parcours, l'électrocardiogramme se fait après la collation.

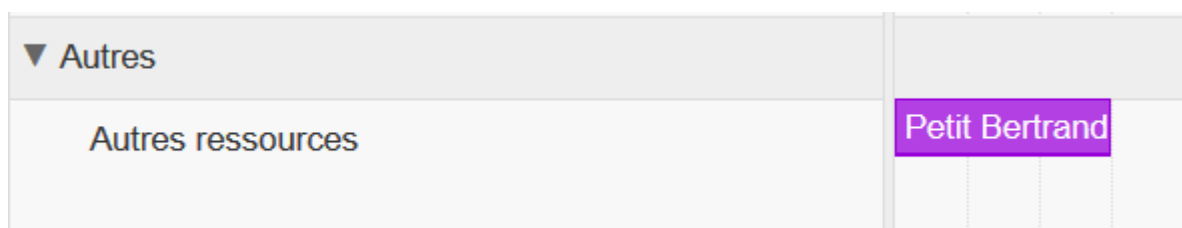


**Figure 7** – Exemple d'erreur d'affichage de contraintes

Pour corriger ce type d'erreur d'affichage, il suffit de préciser au préalable le parcours concerné dans la méthode récupérant les contraintes de précédences, c'est-à-dire `getDependancesActivites()` dans la classe *M\_parcours*.

Enfin, certaines activités n'ont besoin d'aucune ressource pour être réalisées. C'est le cas pour les collations ou les activités nécessitant du matériel en dehors de l'hôpital comme des échographies ou des scanners par exemple. Auparavant lorsque l'on faisait glisser ces activités sur le calendrier, celle-ci était placée sur la ressource où l'activité est relâchée, ce qui occupait donc une ressource sur le calendrier pour rien.

Pour pallier à cela, j'ai ajouté un nouveau type de ressource regroupant ces cas particuliers dénommé *Autres ressources*. Celles-ci ne comportent aucune contrainte temporelle, c'est-à-dire que plusieurs activités peuvent s'y trouver dans le même intervalle de temps, et lorsque que l'on place une activité sans ressource, l'évènement vient se placer directement sur ce type de ressource.



**Figure 8** – Affichage du type de ressource *Autres*

Ce nouveau type de ressource était absolument nécessaire à rajouter sans quoi la planification des activités aurait été impossible.

### 5.3 Mise en œuvre de l'ordonnancement

Pour la planification, nous sommes partis sur l'implémentation la plus simple possible, à savoir la planification patient par patient. Le but est d'avoir un ordonnancement fonctionnel avant de pouvoir par la suite, optimiser l'agencement des activités.

Le principe de cette implémentation se trouve sur l'algorithme de la **Figure 9**. Il représente la gestion des patients ainsi que des activités et de leurs précédences dans le projet. L'idée est de

vider un tableau représentant les activités du patient dans la journée au fur et à mesure du placement des différentes activités en base, tout en gérant les heures de début et de fin des activités.

```

Variables: $ActivitesPatient : le tableau des activités d'un patient
$ActivitesAjoutees : tableau des activités déjà ajoutées au calendrier;
initialisé vide
$ActivitesAAjouter : tableau des activités à ajouter dans la boucle en
cours; initialisé vide
$start : heure du début d'une activité
$end : heure de fin d'une activité

1 Pour chaque patient du jour faire
2   $start ← heure de disponibilité du patient
3   Tant que $ActivitesPatient n'est pas vide :
4     Pour chaque activité dans $ActivitesPatient faire
5       On regarde les précédences de l'activité
6       si les précédences sont dans $ActivitesAjoutees alors
7         | On l'ajoute à $ActivitesAAjouter
8       fin
9     Fin
10    Pour chaque activité dans $ActivitesAAjouter faire
11      $end ← $start + durée de l'activité
12      On l'ajoute au calendrier sur l'intervalle de temps [$start;$end]
13      $start ← heure de fin de l'activité
14      On l'ajoute à $ActivitesAjoutees
15      On la retire de $ActivitesPatient
16    Fin
17  Fin
18 Fin

```

**Algorithm 1:** Gestion des patients et des activités à ajouter

**Figure 9** – Gestion des patients et des activités à ajouter

Les difficultés que j'ai pu rencontrer durant l'implémentation de cet algorithme résidaient dans la gestion des tableaux en PHP qui ont une structure particulière et des nombreuses fonctions de gestion à prendre en main afin récupérer les bonnes informations. La gestion des dates avec les objets *DateTime* a été aussi difficile à prendre en main dans un premier temps.

Nous pouvons voir à la ligne 12 de la **Figure 9** que l'instruction indique que l'on ajoute l'activité au calendrier sur l'intervalle de temps indiqué. Le fonctionnement de l'ajout d'un évènement en base se trouve sur l'algorithme de la **Figure 10**. Le principe de la méthode reprend celui de la fonction déjà implémentée dans un projet précédent, à savoir `addEvenement()`, qui est de placer les évènements sur les premières ressources disponibles sur le calendrier, et cela pour tous les types de ressource nécessaires à la réalisation d'une activité.

Nous pouvons remarquer que la méthode prend en entrée les heures de début et de fin qualifiées "d'idéales" c'est-à-dire au plus tôt pour le patient, et retourne les heures disponibles pour la réalisation de l'activité. Pour retrouver les dates de réalisation de l'évènement, l'algorithme va vérifier dans la base si une ressource de chaque type nécessaire à l'activité est disponible sur l'intervalle de temps précisé. Si un des types a toutes ses ressources occupées, alors on décale l'intervalle de temps de 5 minutes. Le choix d'ajouter 5 minutes vient du fait qu'il s'agit de la plus petite unité de temps pour notre calendrier. Ainsi avec cette méthode, on retrouve les dates de disponibilité les plus tôt possible.

À la fin de l'exécution, l'algorithme retourne les dates à laquelle l'activité va être réaliser. Ces valeurs vont permettre d'ajuster nos heures de début des activités suivantes, comme on le voit avec la variable `$start` à la ligne 13 de l'algo de la **Figure 9**.

```

Entrées: $activite : l'activité à ajouter
$start : heure du début idéale de l'activité
$end : heure de fin idéale de l'activité
Sorties : $newStart : heure réelle de début de l'activité
$newEnd : heure réelle de fin de l'activité

1 si des ressources sont nécessaires à l'activité alors
2   Pour chaque type de ressources nécessaires faire
3     On vérifie si une ressource est disponible pour l'intervalle
       [$start;$end]
4     si une ressource du type demandé est disponible alors
5       On utilise cette ressource
6     finsi
7     sinon
8       Tant que une ressource n'est pas disponible pour l'intervalle
         [$start;$end] :
9         $start ← $start + 5 minutes
10        $end ← $end + 5 minutes
11        On vérifie si une ressource est disponible pour l'intervalle
           [$start;$end]
12      Fin
13    finsi
14    On ajoute l'évènement dans la base avec les premières ressources
       disponibles sur l'intervalle [$start;$end]
15  Fin
16 finsi
17 sinon
18   On ajoute l'évènement dans la base sur le type de ressource Autres
       sur l'intervalle [$start;$end]
19 finsi
20 retourner $start et $end

```

**Algorithm 2:** Gestion des disponibilités des ressources**Figure 10 –** Gestion des disponibilités des ressources

Le point faible de cette méthode vient du fait qu'elle favorise grandement les premiers patients de la journée. En effet ceux-ci n'auront pas de problème de disponibilité vu qu'aucune ressource n'est occupée au début de la journée, tandis que les derniers seront sur la liste risqueront d'avoir des temps d'attente considérables car très peu de ressources seront disponibles. Un autre problème avec cette implémentation est que les contraintes de délais entre les activités ne sont pas pris en compte. L'ajout de ces contraintes nécessite un soin particulier car elles peuvent impliquer des planifications impossibles. Or notre but ici est non pas de proposer une solution parfaite et optimale mais d'avoir une solution viable que l'utilisateur peut modifier par la suite s'il le souhaite.

# 6

## Validation et Tests

Ce chapitre est consacré à la validation des développements réalisés. Les tests de la plateforme sont séparés en deux parties : d'un côté les tests pour vérifier les corrections de bugs avec le framework Codeception, de l'autre des tests de charge sur la planification automatique pour voir les limites de la solution implémentée.

### 1 Le framework Codeception

Il existe de nombreux moyens pour réaliser des tests sur une plateforme web. Pour ma part, j'ai utilisé le framework *Codeception*. Il s'agit d'un framework permettant de couvrir une application entièrement car il intègre tout type de tests : les tests unitaires en se basant sur *PHPUnit*, les tests fonctionnels et les tests d'acceptances qui simulent des actions d'utilisateur lambda sur le site.

Les tests fonctionnels et les tests d'acceptances sont écrits de façon à ce qu'ils soient compréhensibles par tous. La **Figure 1** montre un exemple de tests pour vérifier si on peut se connecter sur la page de connexion.

```
public function testAuth(AcceptanceTester $I)
{
    $I->amOnPage('/');
    $I->fillField('username', 'admin');
    $I->fillField('password', 'admin');
    $I->click('Se connecter');
    $I->amOnPage('/index.php/');
}
```

**Figure 1** – Exemple de test d'acceptance pour se connecter

Les tests se lancent en invite de commande en utilisant le script `codecept run` en précisant `--steps` si l'on souhaite voir les étapes de tests. Les détails concernant la mise en place du framework se trouvent sur leur site <https://codeception.com/quickstart>.

Les résultats sont affichés comme on peut le voir sur la **Figure 2**.

Par manque de temps, peu de tests ont pu être implémentés pour vérifier la viabilité de la plateforme et des corrections qui y ont été réalisées. L'idéal pour les prochains projets serait de

```

Codeception PHP Testing Framework v2.4.1
Powered by PHPUnit 4.8.36 by Sebastian Bergmann and contributors.

a[1mAcceptance Tests (2) a[22m-----
a[35;1mAuthCest:a[39;22m Test auth
Signature: a[32mAuthCest:testAutha[39m
Test: a[32mtests\acceptance\AuthCest.php:testAutha[39m
a[33mScenario --a[39m
a[1m I a[22mam on page "/"
a[1m I a[22mfill field "username", "admin"
a[1m I a[22mfill field "password", "admin"
a[1m I a[22mclick "Se connecter"
a[1m I a[22mam on page "/index.php/"
a[32;1m PASSED a[39;22m

a[35;1mFirstCest:a[39;22m Try to test
Signature: a[32mFirstCest:tryToTesta[39m
Test: a[32mtests\acceptance\FirstCest.php:tryToTesta[39m
a[33mScenario --a[39m
a[1m I a[22mam on page "/"
a[1m I a[22msee "Se connecter"
a[32;1m PASSED a[39;22m

-----

a[1mFunctional Tests (0) a[22m-----

a[1mUnit Tests (0) a[22m-----

Time: 2.96 seconds, Memory: 19.00MB

a[30;42mOK (2 tests, 1 assertion)a[0m

```

Figure 2 – Affichage des résultats de tests Codeception

faire un ensemble de tests sur chaque fonctionnalité de la plateforme de façon automatisé en utilisant le framework, ce qui permettrait de voir si un développement n'impacte pas les autres fonctionnalités existantes.

## 2 Test de la planification automatique

Pour la planification automatique, des tests de charge ont été mis en place afin de voir si elle supporte un nombre conséquent de patient. Pour tester la planification, nous nous sommes basés sur les données fournis par l'AP-HP qui nous permettent d'avoir des informations sur les ressources matérielles à disposition et sur le nombre de patients qui peuvent être pris en charge sur une journée. Aussi, pour ses tests, un secteur précis de l'hôpital, à savoir les traitements de l'obésité, afin d'avoir un aperçu sur les partages de ressources communes à plusieurs patients.

Les parcours sont distingués par des codes : P1, P2 et P3, et certaines abréviations sur les ressources peuvent apparaître. Ceux-ci sont détaillés en [Annexe G](#).

Par conséquent, voici les données d'entrée du test :

- 9 patients disponibles de 8h à 20h : 4 sur le parcours P1, 2 sur le parcours P2 et 3 sur le parcours P3
- Le personnel est composé de : 3 IDE obésité, 4 IDE, 2 psychologues, 3 diététiciens, 3 nutritionnistes, 2 internes obésité et 1 médecin hépato.
- L'hôpital dispose des salles suivantes : 2 box prélèvement, 4 HDJ obésité et 5 bureaux CS

Avec ses données, la planification obtenue est celle présente sur les [Figure 3](#), avec une vue sur le personnel, et [Figure 4](#), avec un focus sur les salles cette fois.

Plusieurs constats peuvent être faits à partir de ces deux images. Tout d'abord, toutes les activités ont été correctement réparties sur le calendrier. Ensuite, la répartition des tâches entre certains membres du personnel semble assez disparate. D'un côté, nous avons les IDE obésité qui ont un calendrier assez similaire dans la matinée, et de l'autre nous avons deux IDE qui n'ont aucune tâche d'attribuer. Ceci s'explique par le fait que l'on sélectionne les premières ressources disponibles, ainsi la première infirmière aura vraisemblablement le plus de tâches à réaliser.

Au niveau des problèmes de planification, nous avons les trois derniers patients qui débordent sur la fin du calendrier, ce qui signifie que dans cette configuration, 6 patients peuvent rentrer

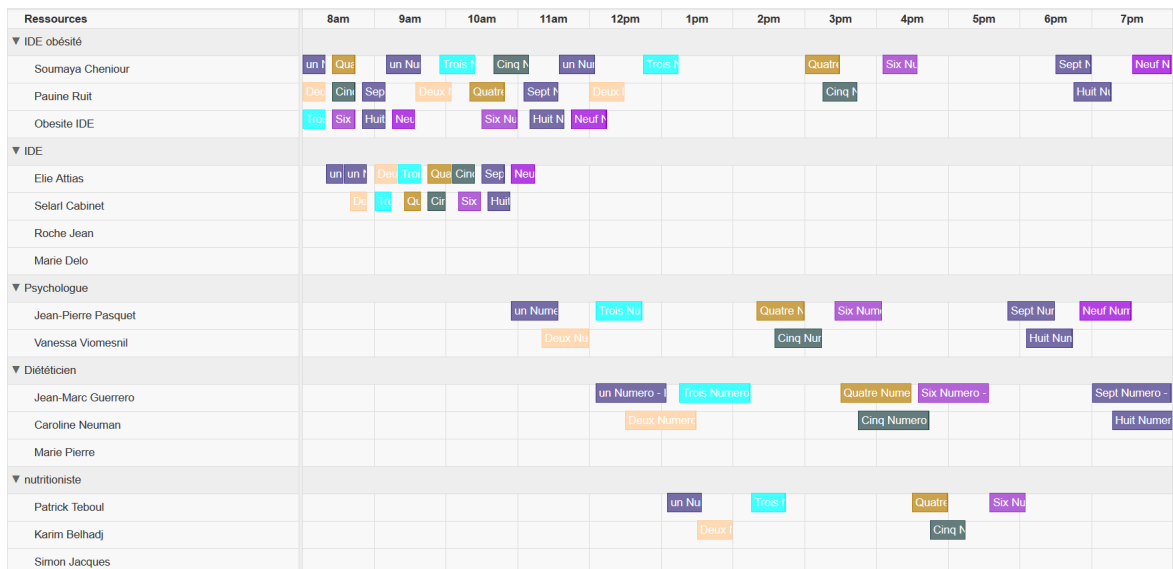


Figure 3 – Planification vue du personnel

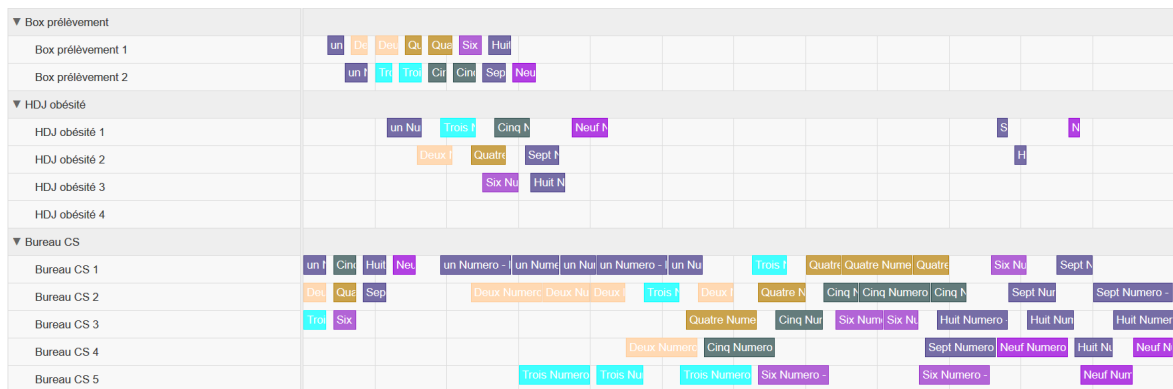


Figure 4 – Planification vue des salles

dans les délais. Cela ne laisse que très peu de marges de manœuvre pour l'hôpital puisque, d'après leurs données fournies, ils prévoient d'accueillir 6 patients sur les parcours Obésité. Un autre aspect à prendre en compte est le temps d'attente des patients qui décuple au fil de la journée. Un exemple avec le sixième patient qui commence son parcours à 8h25 le matin pour repartir à 18h05 le soir.

Enfin, parmi les problèmes de la planification, nous avons l'agencement des salles qui peut sembler assez chaotique, ou tout du moins difficilement possible dans la vie réelle. Un exemple avec le patient cinq qui passe du bureau CS 4, avant d'aller au 3 puis au 2 au final. Cela provient de la sélection des ressources pour les activités qui ne dépend pas des activités précédentes. La logique voudrait que si l'on doit effectuer deux activités dans le même type de salle consécutivement, on reste dans la même salle ce qui n'est pas le cas ici.

Ainsi, comme convenu lors des spécifications, cette planification est loin d'être parfaite et nécessiterait d'avoir de nombreux ajustements pour améliorer le rendu final. Néanmoins, l'objectif qui est de proposer une solution directe qui peut être ajustée ensuite par l'utilisateur est atteint.

# 7

## Qualité de code

Ce chapitre répertorie les éléments de qualité de code du projet au travers de deux éléments : la documentation et le framework *SonarQube*.

### 1 Documentation

Il existe de nombreux frameworks permettant de générer la documentation de son projet PHP. Pour ma part, j'ai choisi le framework *phpDocumentor*, dont les informations sont disponibles sur le site <https://docs.phpdoc.org/>. Ce framework est non seulement facile à installer mais il permet d'avoir un affichage claire du projet en un coup d'œil. La documentation du projet est disponible dans le répertoire *documentation* de l'application.

### 2 SonarQube

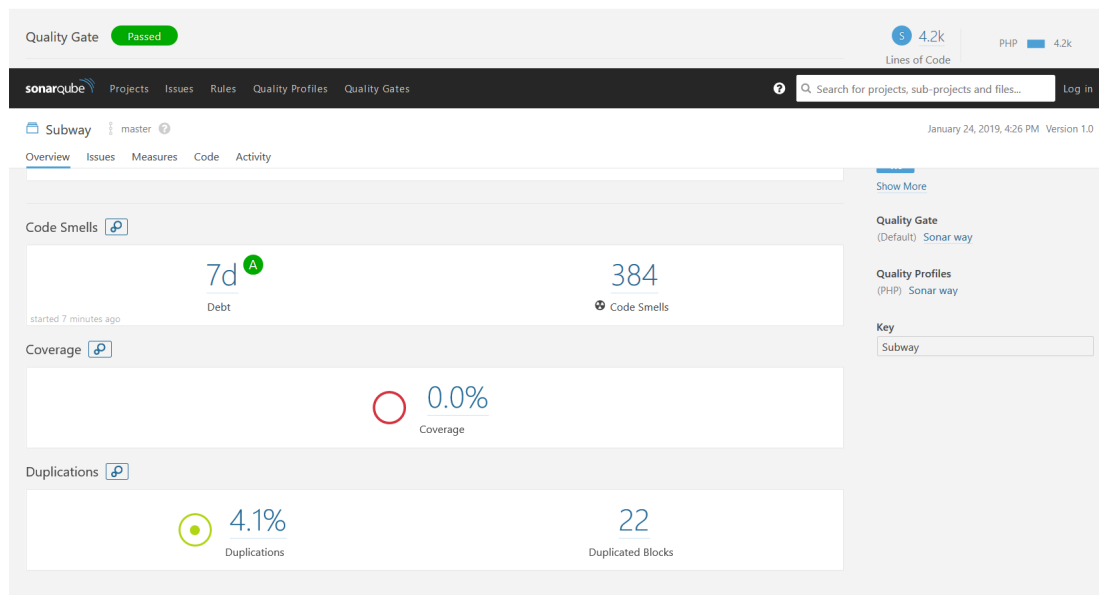
Pour le développement de la plate-forme, nous nous sommes servis de l'outil de qualité de code *SonarQube*. C'est un logiciel libre permettant de mesurer la qualité du code produit en se basant sur divers règles mises en place au préalable. Il peut détecter les bugs potentiels, les duplications de code, la couverture de code par les tests unitaires, et bien d'autres. Il est adaptable et personnalisable selon les besoins et les langages que l'on souhaite utiliser avec de nombreux plug-ins à disposition.

L'installation de SonarQube se fait très simplement. Il suffit de télécharger le programme, puis de lancer le script *StartSonar.bat*. Une instance de SonarQube sera lancée, et une fois que le message "*SonarQube is up*" s'inscrit sur l'invite de commande, nous pouvons ouvrir la page du programme sur un navigateur. Par défaut, SonarQube utilise le port 9000 sur l'hôte local.

Dès lors que le serveur *SonarQube* est opérationnel, nous pouvons lancer une analyse du projet. Le serveur va donc analyser tous les fichiers sources du projet. Pour que le serveur puisse obtenir des résultats, il est nécessaire d'installer au préalable un scanner *Sonar* correspondant au langage de programmation utilisé.

Les résultats d'une analyse sont présentés dans une fenêtre comme sur la **Figure 1**.

Comme on le voit sur l'image, *SonarQube* a détecté plus de 380 problèmes dans le code. Cependant, la plupart d'entre elles viennent des fichiers de configuration de *CodeIgniter*. On peut voir



**Figure 1** – Fenêtre d'affichage de Sonar

qu'il y a également 22 blocs de code dupliqués. *SonarQube* apporte ainsi de nombreuses informations sur les erreurs ou les mauvaises pratiques présentes dans le code avec des explications précises pour chacun des cas trouvés.



# 8

## Avancement du projet et suite à donner

Ce chapitre constitue la mise au point du projet à la fin de son échéance. Nous allons parcourir les fonctionnalités mises en place et les éléments à ajouter à l'avenir pour améliorer davantage la plateforme.

### 1 Fonctionnalités mises en place

À l'issue de ce projet, la plateforme a été améliorée et la majeure partie des objectifs fixés lors du chiffrage du projet (voir [Annexe D](#)) ont été atteints. Parmi les fonctionnalités ajoutées, on peut citer :

- La révision de la suppression des ressources
- La révision de l'onglet "Plan de parcours"
- La révision du formulaire de création de patient
- L'ajout des tris dans les onglets d'affichage
- L'implémentation de la planification automatique patient par patient

Lors du premier chiffrage, l'intention était d'aller plus loin dans les fonctionnalités, et notamment sur la partie Planification. Cependant, des retards dûs à la montée en compétence nécessaire en PHP et sur les frameworks utilisés ont réduit la marge de manœuvre sur les objectifs finaux.

Néanmoins, le but premier qui était d'implémenter une planification automatique des activités a été réalisé, ce qui permet d'envisager de nouvelles améliorations à l'avenir.

### 2 Fonctionnalités à ajouter ou à améliorer pour la suite du projet

De nombreux éléments peuvent être apportés à la plateforme afin qu'elle puisse être utilisée à l'avenir. Voici une liste non-exhaustive des ajouts qui pourraient être amenés à la plateforme :

- Implémenter des jeux de données automatisés afin de faciliter les tests sur la plateforme
- Améliorer la planification automatique en passant à des algorithmes d'ordonnancement et en prenant en compte davantage de contraintes
- Ajouter une notion temps-réel à la plateforme, avec ajustement de la planification en direct selon les événements dans l'hôpital

# Bilan et conclusion

## Bilan de la première partie

À l'issue de cette première partie, nous avons pu observer les tenants et les aboutissants du projet. La plateforme comporte déjà de nombreuses fonctionnalités et l'objectif principal qui était de réaliser un outil de gestion de parcours patients a été atteint lors des précédents projets. Ici, le sujet concerne les nombreuses améliorations possibles sur la plateforme. Bien que la planification fonctionne, l'application n'est pas utilisable en production à cause d'une part, des problèmes d'ergonomie généraux, et d'autre part, de la planification qui devient vite impossible lorsque les données sont trop conséquentes.

Parmi les améliorations à effectuer, il y a dans un premier temps des améliorations générales sur la plateforme (sur l'affichage, l'ergonomie, l'expérience utilisateur en général) et dans un second temps, la planification automatique des activités. Celle-ci nécessitera la majorité du temps de développement dans la seconde partie du projet.

La suite du projet sera principalement consacrée aux développements des solutions évoquées dans les chapitres précédents. Certaines corrections ont déjà été réalisées à l'issue de cette partie, affectant l'affichage en général (corrections des fautes d'orthographe, révision de la suppression des ressources comme évoquée dans la [Section 1](#) (Chapitre 4), affichage du planning patient comme présenté dans la [Section 4](#) (Chapitre 4)). Les développements principaux, la planification automatique notamment, seront réalisées par la suite.

## Bilan de la seconde partie

La seconde partie de ce projet aura été orientée développement plus que jamais. L'implémentation de la planification aura pris une majeure partie du temps dédié au projet dû à la complexité de la gestion de tous les facteurs, le code d'une part avec les algorithmes d'ordonnancement à penser, l'aspect graphique avec *fullCalendar* et la base de données avec la gestion des différentes ressources à prendre en compte.

Néanmoins, l'objectif qui était de proposer une première solution respectant les contraintes principales, en particulier les précédences des activités entre elles, a été atteint. Le temps n'a pas permis d'étoffer davantage l'aspect ordonnancement en tant que tel, mais j'espère toutefois que ce rapport permette d'éclaircir certaines pistes pour le futur de l'outil.

## Conclusions

Ce projet de Recherche et Développement m'aura beaucoup apporté dans de nombreux domaines de compétence. Sur un aspect purement technique dans un premier temps, j'ai pu découvrir ou re-découvrir certaines technologies web que je ne connaissais pas ou très peu. Sur l'aspect de l'évolutivité également, reprendre un projet provenant de plusieurs personnes différentes au fil des années n'est jamais quelque chose d'aisé. Chacun a son style de développement et sa vision de voir les choses, démêler les différents aspects des développements était une expérience très intéressante pour moi et qui m'apportera à coup sûr à l'avenir.

Ce projet peut encore beaucoup évoluer s'il trouve repreneur dans les années à venir. Si une suite est donnée, je souhaite bon courage à mon successeur en espérant que ce projet lui apporte de la réussite.

## Annexes

# A

## Fonctionnalités existantes

L'application a été réalisée par des étudiants au cours de l'année scolaire 2015-2016 puis sur les années suivantes lors d'un projet collectif et trois projets Recherche et Développement. La plateforme comporte les fonctionnalités suivantes : la gestion du personnel, des ressources matérielles, des activités, des parcours et des plans de parcours. Autre les fonctionnalités de gestion, la plateforme peut aussi créer et rechercher les patients, leur affecter des parcours et planifier les activités.

### 1 Gérer les personnels

Chaque personnel a une fonctionnalité dans l'hôpital, l'administrateur peut les modifier, supprimer et les rendre indisponible pour une période convenue.

- Modifier un personnel
- Supprimer un personnel
- Mettre un personnel en indisponibilité

### 2 Gérer les ressources matérielles

Chaque ressource matérielle a un nom et une spécialité. Ces ressources correspondent aux salles de l'hôpital (les bureaux de consultation, les salles de prélèvements, etc.).

- Modifier une ressource matérielle
- Supprimer une ressource matérielle

### 3 Gérer les activités

Chaque activité doit avoir un nom et une durée, elle sera faite dans une certaine salle et par un ou plusieurs types de personnel.

- Ajouter une nouvelle activité
- Modifier les activités existantes
- Supprimer les activités qui n'existent plus

## 4 Gérer les parcours

Un parcours se compose par plusieurs activités. Certaines activités peuvent avoir des contraintes de précedence avec d'autres. Lors de la création d'un parcours, l'agencement des activités a donc un impact important sur la planification. Pour créer un parcours, il est nécessaire de créer au préalable les activités de ce parcours si celles-ci n'ont pas encore été créées.

- Ajouter un nouveau parcours
- Modifier un parcours
- Supprimer un parcours
- Visualiser un parcours

## 5 Gérer les plans de parcours

Dans un hôpital de jour, la capacité d'accueil les patients est limitée. Cette donnée est renseignée dans l'onglet "Plan de parcours" et peut être ajustée selon les parcours et les jours de la semaine.

## 6 Visualiser du planning

Une fonctionnalité "Afficher planning du patient" est disponible et permet de visualiser l'heure de début et de fin de la prochaine activité du patient dans son parcours. L'affichage est restreint au jour de consultation de la plateforme (exemple : si on est le 5 du mois, on ne peut pas voir les rendez-vous planifiés pour le lendemain). Cette visualisation est accessible auprès de l'administrateur, du personnel de soins et également du patient.

## 7 Affecter un patient à un parcours

Un patient peut être affecté à un parcours en précisant les disponibilités de ce dernier. Celles-ci seront renseignées sous formes d'horaires (exemple : 8h-17h). Cette fonctionnalité est accessible à l'administrateur et aux personnels de soins.

L'administrateur ou l'infirmière de coordination peut affecter un patient à un parcours pour une journée donnée et a la possibilité de visualiser également le nombre de patients déjà affectés sur la journée.

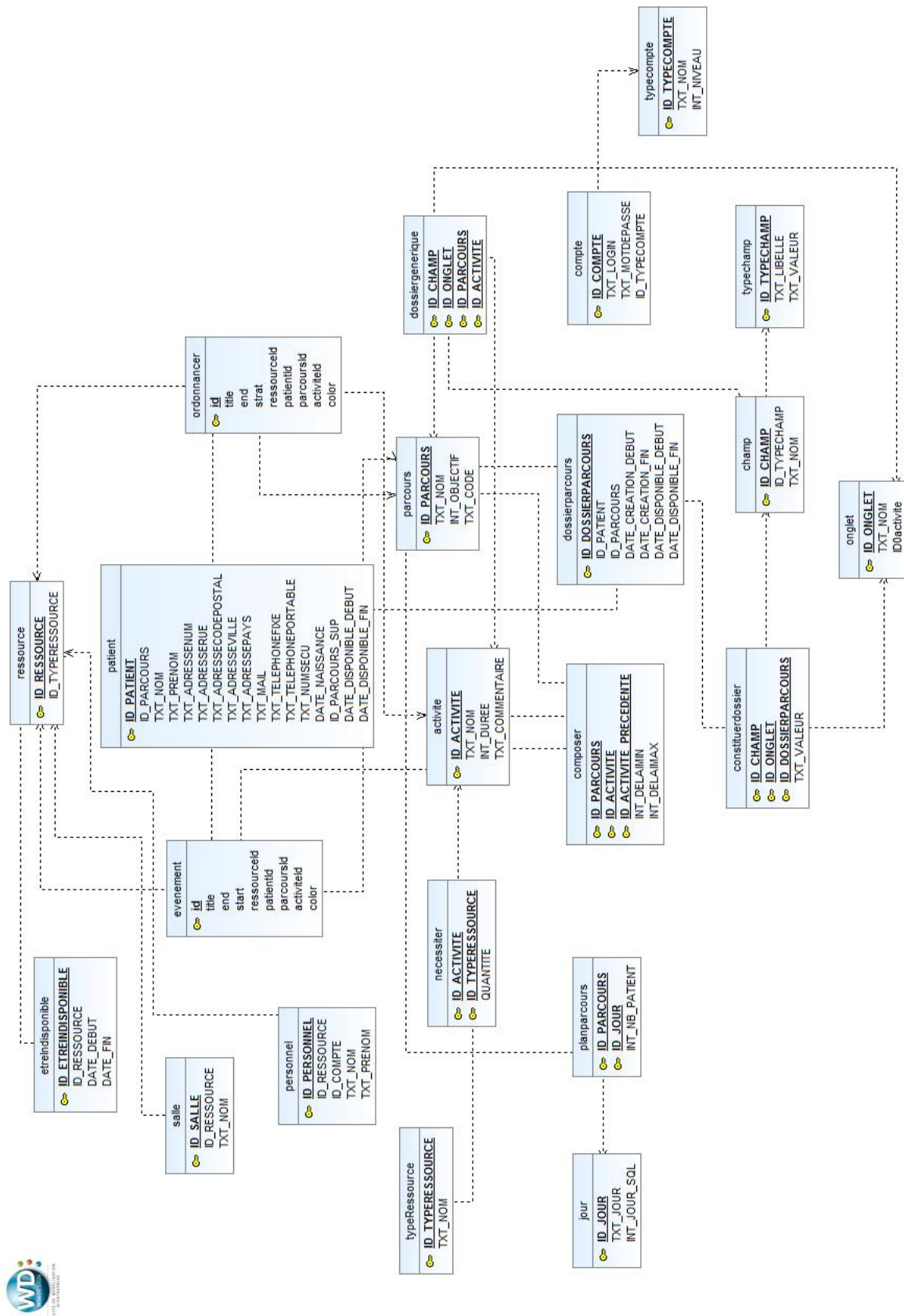
Lors d'un clic sur le bouton "Affecter", l'administrateur sera redirigé vers une page où il pourra choisir le patient à affecter pour la date choisie précédemment.

## 8 Visualiser et modifier le planning de toutes les ressources

L'administrateur peut visualiser et modifier en temps réel le planning de toutes les ressources et également d'une ressource en particulier. La modification peut s'effectuer de façon manuelle (en déplaçant les activités). L'administrateur doit pouvoir planifier automatiquement le planning de ses ressources pour une journée donnée. La planification automatique est l'un des sujets de ce projet et est évoquée dans les chapitres précédents. Ces fonctionnalités ne sont accessibles que pour l'administrateur. Les différentes interfaces ont été étudiées dans les projets précédents et la gestion du calendrier utilise le framework *fullCalendar*.

**B**

## Modèle conceptuel de données





# C

## Explications des tables du modèle

La section suivante détaille les éléments présents dans le modèle de la base de données. Les tables et la modélisation en général ont été réalisées lors des projets précédents et seront réutilisées pour le projet actuel.

**Activité** : Table regroupant l'ensemble des informations concernant une activité. C'est cet élément qui constitue les parcours. Il est possible qu'une activité puisse se réaliser uniquement dans le cas où d'autres activités ont été effectuées au préalable. Cette notion de dépendance dépend du parcours en cours. Les liens entre les activités et la notion de précedence seront évoqués via la table **composer**. En plus de cela, une activité a besoin de ressources. Ce lien se fait par la table **nécessiter**.

**Champ** : Table contenant l'ensemble des champs qu'il est possible d'ajouter dans un onglet d'un dossier *Parcours*. À chaque champ est lié un type de champ, que nous verrons par ailleurs avec la table **typechamp**.

**Composer** : Table permettant de faire le lien entre un parcours et une activité. Par l'intermédiaire de cette table, nous pouvons dire en fonction d'un parcours et d'une activité s'il y a des besoins en termes de précedence. Chaque ligne, dans cette table, a pour signification : « Pour l'activité A dans le parcours P, il faut avoir réalisé l'activité B avant, et ce dans un délai compris entre *delaiMin* et *delaiMax*. ». Il est important de noter qu'il est possible de mettre à « null », l'id de l'activité précédente s'il n'y a aucune contrainte.

**Compte** : Table regroupant l'ensemble des comptes qu'ils soient des comptes patient ou des comptes de type ressource médicale.

**Constituerdossier** : Table permettant de faire le lien entre un dossier *parcours* et les informations qui le constituent. En effet, nous retrouvons pour chaque dossier *parcours* et pour chaque onglet dans ce dernier, la valeur des champs le composant.

**Dossiergenerique** : Table définissant, pour un parcours, les onglets et les champs que tous les dossiers parcours doivent avoir impérativement.

**Dossierparcours** : Table renseignant les informations génériques d'un dossier parcours. Soit le patient associé, le parcours, dates de création et de dernière modification.

**Etreindisponible** : Table regroupant l'ensemble des indisponibilités pour une ressource. Cette indisponibilité est caractérisée par une date de début et de fin, acceptant toutes les deux le renseignement de l'heure.

**Jour** : Table représentant les jours de la semaine, ainsi que leurs index sous MySQL. Cette table va nous servir à établir les plans de parcours, c'est-à-dire le nombre de patients pouvant être acceptés par jour et par parcours.

- Necessiter** : Table permettant de renseigner les types de ressources requises pour une activité, ainsi que la quantité nécessaire.
- Onglet** : Table des onglets disponibles pour constituer un dossier parcours.
- Parcours** : Table décrivant un parcours de façon générale.
- Personnel** : Table contenant l'ensemble de personnel médical de l'établissement. Chaque personne de l'hôpital a un compte, et est considérée comme une ressource.
- Planparcours** : Table regroupant l'ensemble des objectifs concernant le nombre de patients pour un parcours pour un jour donné.
- Ressource** : Table faisant le lien entre la table **typeressource** et **personnel** ou matériel. Ce lien sera expliqué plus en détail avec la table **typeressource**.
- Typechamp** : Table regroupant les différents types de champs qu'il est possible d'ajouter dans un dossier parcours. Elle contient également le code HTML des composants, permettant ainsi une mise en page en accord avec les autres éléments des pages.
- Typecompte** : Table utilisée pour la gestion des droits.
- Typeressource** : Table contenant les types de ressource d'un point de vue activité. En effet, une activité peut avoir besoin d'un type de ressource bien caractéristique (ex : IDE obésité). C'est pourquoi nous avons un double niveau de type de ressource. Un concernant les activités (**typeressource**) et un second plus d'un point de vue logique générale (**personnel, matériel**).
- Ordonnancer** : Table de fait de notre système. C'est la table la plus importante. Chaque ligne veut dire : « Pour le patient P qui fait le parcours Pa à la date D, il a besoin de la ressource R pour faire l'activité A de start à end. Cette table contient la planification réalisée de manière manuelle ou automatique.
- Evènement** : Table identique à la table **Ordonnancer**. Elle a le même but que la table **ordonnancer** mais cette table contient la planification en cours. Cette table permet de pouvoir sauvegarder ou restaurer la planification en fonction des modifications que l'utilisateur a effectuées.

# D

## Chiffrage du projet

Tâches à réaliser	Nombre de jours
Révision de la suppression des ressources	1
Révision de l'onglet "Plan de Parcours"	1
Révision du formulaire de création de patient	2
Révision de la page d'accueil	1
Affichage du planning patient	5
Ajout des tris dans les onglets d'affichage	1
Correction des problèmes d'URL	2
Affichage des ressources sélectionnées et test planning	10
Implémentation de la planification automatique	20
Génération des jeux de données	3
Rédaction du rapport	15
Préparation des soutenances	3
<b>Total</b>	<b>64</b>

**Table 1** – *Chiffrage des tâches à réaliser*

# E

## Gestion de projet

La bonne gestion d'un projet est un élément crucial dans la réussite d'un travail, d'autant plus pour ce PR&D car il est réalisé sur un temps plus restreint et se termine en janvier au lieu d'avril. Pour la gestion de projet, j'ai utilisé plusieurs outils divers pour gérer les tâches à réaliser au mieux. Celui qui m'a le plus servi est *Trello*.

### 1 Trello

**Trello** est un outil de gestion de projet en ligne, lancé en septembre 2011, et inspiré par la méthode *Kanban* de l'entreprise Toyota. Il est basé sur une organisation des projets en planches listant des cartes, chacune représentant des tâches. Les cartes sont assignables à des utilisateurs et sont mobiles d'une planche à l'autre, traduisant leur avancement. La version de base est gratuite, et c'est celle-ci que j'ai utilisé pour ordonner mon projet. J'ai principalement conservé les 3 sections disponibles par défaut : les tâches à effectuer (*To do*), en cours de réalisation (*Doing*) et terminées (*Done*).

La **Figure 1** présente un aperçu des tâches de mon projet sur **Trello**.

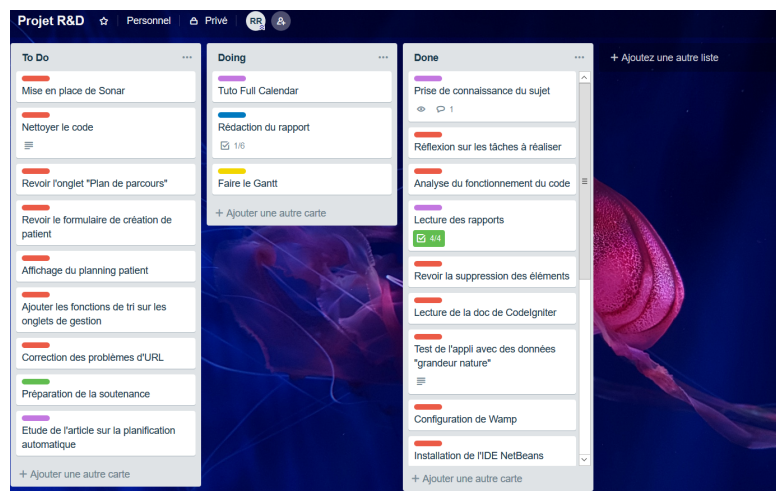


Figure 1 – Aperçu de ma page Trello

## 2 Diagramme de Gantt

Pour avoir un aperçu des tâches à réaliser dans le temps, j'ai réalisé un diagramme de Gantt avec l'utilitaire *GanttProject* visible par la suite.

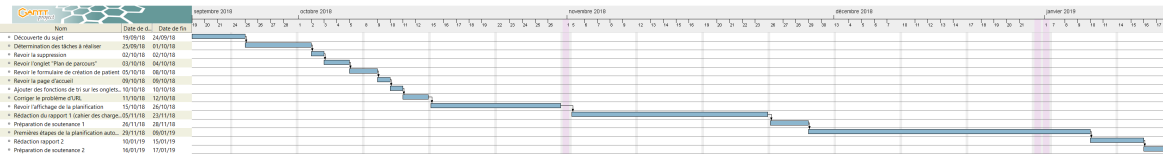


Figure 2 – Diagramme de Gantt

Comme on le voit sur le diagramme, le projet va être séparé en deux parties : une consacrée aux corrections sur la plateforme et aux rédactions, tandis que la deuxième sera dédiée à la programmation de la planification automatique.

## 3 Git et GitKraken

Pour le versionning de la plateforme et du rapport, j'utilise le logiciel de gestion de versions **git**. Les sources du projet sont enregistrées sur la plateforme **GitHub**. L'application est disponible au téléchargement à l'adresse <https://github.com/RomainR37/ParcoursPatient>.

Pour aider au versionning, j'utilise **GitKraken** qui propose une interface graphique de l'état du projet avec Git. La **Figure 3** montre un aperçu de l'interface de **GitKraken** pour mon projet.

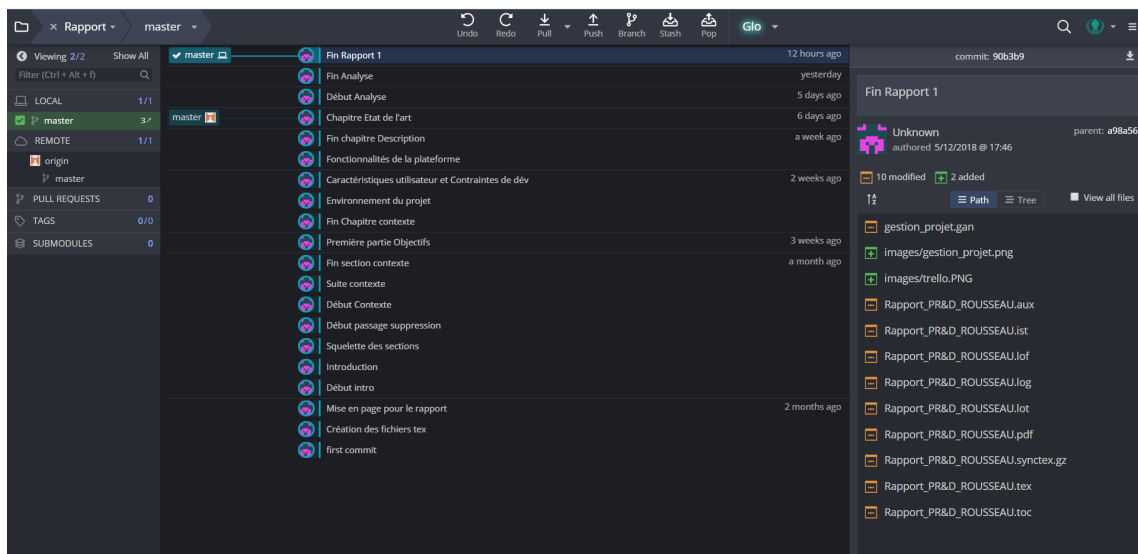


Figure 3 – Diagramme de Gantt

# F

## Notice des livrables

Cette partie regroupe les notices des deux livrables réalisés lors du projet : le premier le 9 janvier 2019 et le second à la fin du projet, c'est-à-dire le 11 février 2019.

### 1 Livable du 9/01/2019

- Ajout d'un message d'avertissement lors de la suppression d'une ressource. Concerne les pages de l'onglet Gérer (Personnels, Ressources Matérielles, Activités, Parcours)
- Corrections sur l'onglet Plan de Parcours : affichage des jours de la semaine sous forme de texte (lundi, mardi, etc.) plutôt que par des chiffres, correction de la limite du nombre de patients affecté (limité à 10 auparavant, jusqu'à 99 maintenant)
- Ajout de règles supplémentaires sur le formulaire de création de patients : Aucune restriction sur les numéros de rue → Numéro de rue limité à 5 chiffres, caractère numérique obligatoire, Aucune restriction sur les numéros de téléphone → l'entrée doit contenir 10 chiffres exactement, Numéro de sécurité sociale maximum 15 chiffres → Numéro de sécurité sociale exactement 15 chiffres.
- Fonction de tri dans les onglets Gérer : en cliquant sur les en-têtes des colonnes, la colonne correspondante est triée par ordre alphabétique. En cliquant à nouveau, la colonne est dans l'ordre alphabétique inversé. Les colonnes affectées sont les plus pertinentes (Nom, prénom, fonction, etc.)
- Correction de fautes d'orthographe diverses

### 2 Livable du 11/02/2019

- Ajout de la planification automatique. Les événements sont ajoutés patient par patient, avec la première ressource disponible. Les contraintes de délais minimum et maximum entre les activités ne sont pas prises en compte.
- Nettoyage du code et génération d'une nouvelle documentation

# G

## Détails des parcours utilisés pour les tests

3 parcours-patients ont été utilisés lors des tests de la planification automatique. Les voici présentés dans cette annexe.

Chaque parcours dispose d'un code pour les reconnaître plus facilement : P1, P2 et P3. Plusieurs abréviations sont utilisées, parmi lesquelles :

**IDE** : infirmier diplômé d'état

**ECG** : électrocardiogramme

**Bureau CS** : bureau consultation

### 1 Parcours P1 : Obésité sévère - diagnostic

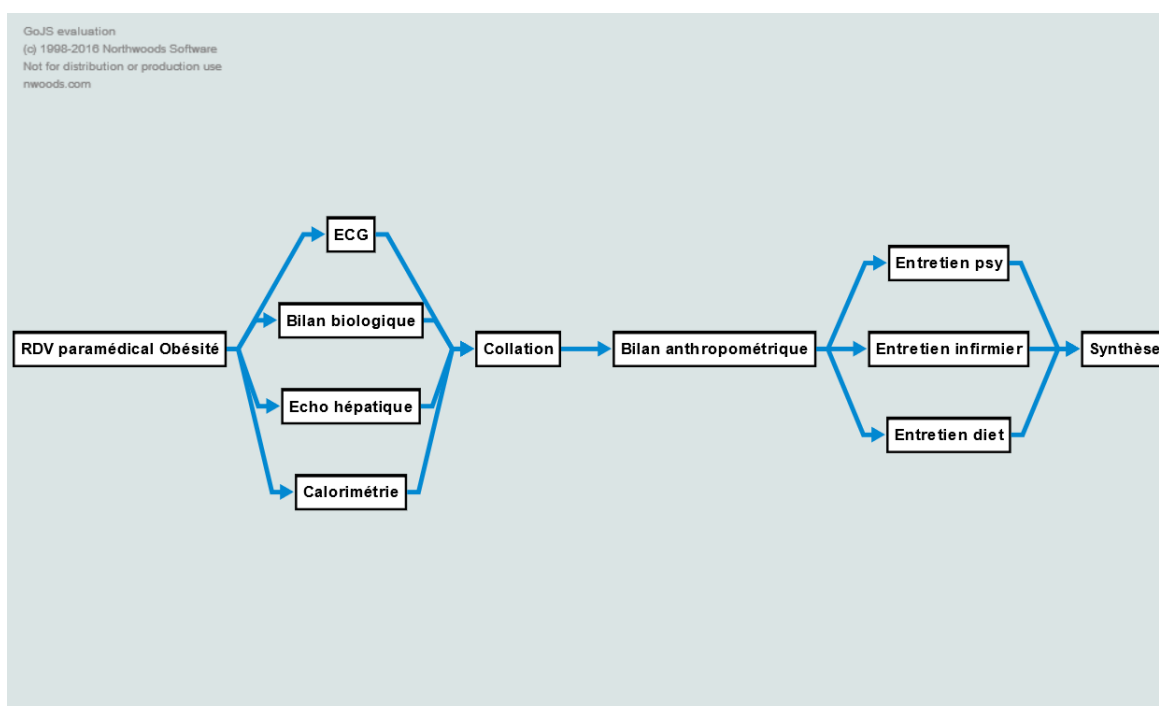


Figure 1 – Parcours P1

**Activités du parcours**

**RDV Paramédical Obésité** : 20 minutes, nécessite 1 IDE Obésité et 1 Bureau CS

**ECG** : 15 minutes, nécessite 1 IDE et 1 Box Prélèvement

**Bilan biologique** : 20 minutes, nécessite 1 IDE et 1 Box Prélèvement

**Echo hépatique** : 15 minutes

**Calorimétrie** : 30 minutes, nécessite 1 IDE Obésité et 1 HDJ obésité

**Collation** : 15 minutes

**Bilan anthropométrique** : 1 heure, nécessite 1 interne obésité et 1 Bureau CS

**Entretien psy** : 40 minutes, nécessite 1 psychologue et 1 Bureau CS

**Entretien infirmier** : 30 minutes, nécessite 1 IDE obésité et 1 Bureau CS

**Entretien diet** : 1 heure, nécessite 1 diététicien et 1 Bureau CS

**Synthèse** : 30 minutes, nécessite 1 nutritionniste et 1 Bureau CS

## 2 Parcours P2 : Obésité sévère - Post OP

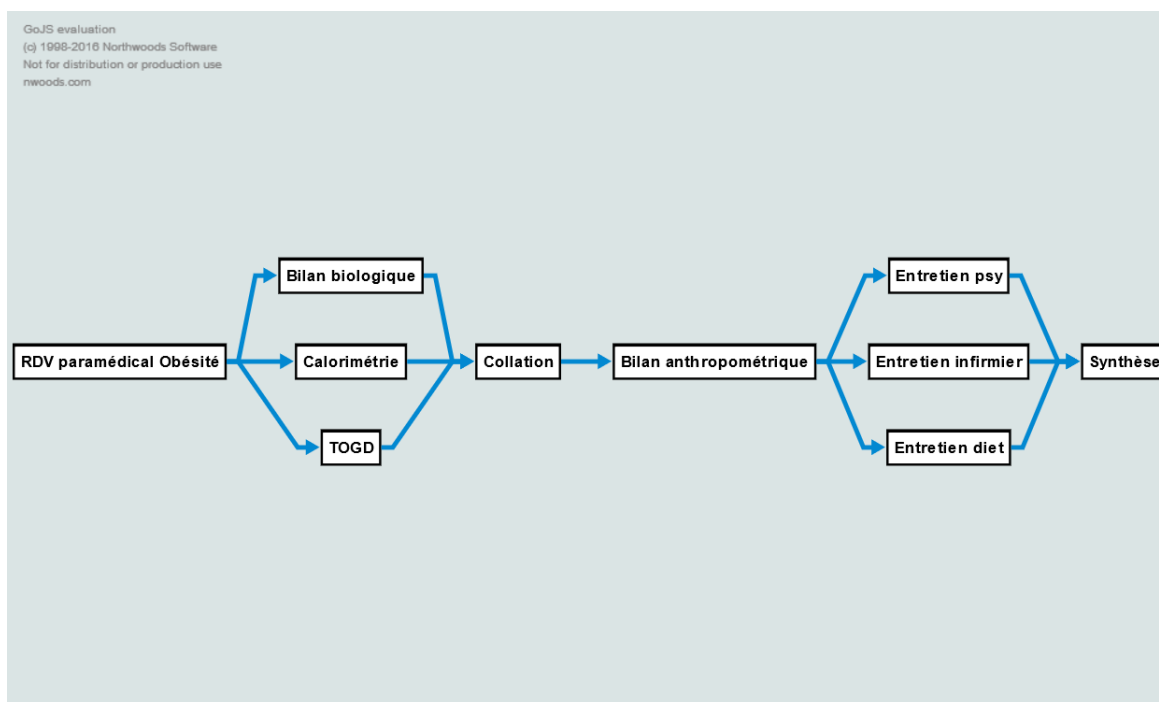


Figure 2 – Parcours P2

**Activités du parcours**

**RDV Paramédical Obésité** : 20 minutes, nécessite 1 IDE Obésité et 1 Bureau CS

**Bilan biologique** : 20 minutes, nécessite 1 IDE et 1 Box Prélèvement

**TOGD** : 20 minutes

**Collation** : 15 minutes

**Bilan anthropométrique** : 1 heure, nécessite 1 interne obésité et 1 Bureau CS

**Entretien psy** : 40 minutes, nécessite 1 psychologue et 1 Bureau CS

**Entretien infirmier** : 30 minutes, nécessite 1 IDE obésité et 1 Bureau CS

**Entretien diet** : 1 heure, nécessite 1 diététicien et 1 Bureau CS

**Synthèse** : 30 minutes, nécessite 1 nutritionniste et 1 Bureau CS



### 3 Parcours P3 : Obésité sévère - J+1an

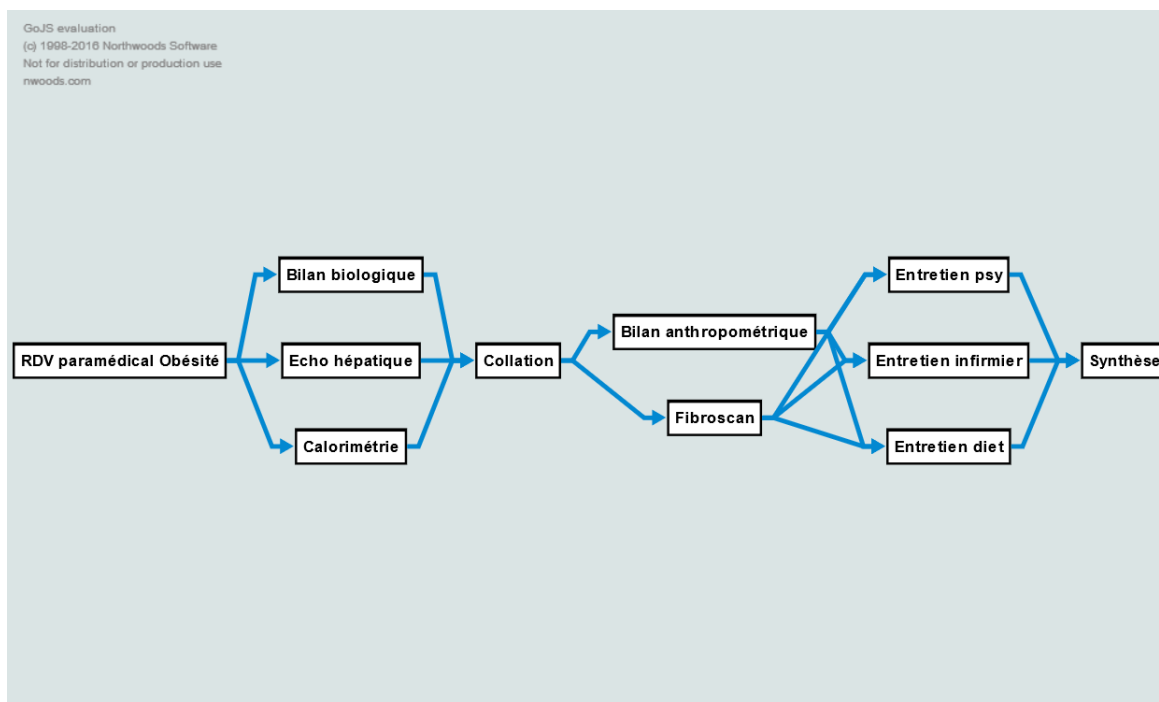


Figure 3 – Parcours P3

#### Activités du parcours

**RDV Paramédical Obésité** : 20 minutes, nécessite 1 IDE Obésité et 1 Bureau CS

**Bilan biologique** : 20 minutes, nécessite 1 IDE et 1 Box Prélèvement

**Echo hépatique** : 15 minutes

**Calorimétrie** : 30 minutes, nécessite 1 IDE Obésité et 1 HDJ obésité

**Collation** : 15 minutes

**Bilan anthropométrique** : 1 heure, nécessite 1 interne obésité et 1 Bureau CS

**Fibroscan** : 10 minutes, nécessite 1 médecin Hépatite et 1 HDJ obésité

**Entretien psy** : 40 minutes, nécessite 1 psychologue et 1 Bureau CS

**Entretien infirmier** : 30 minutes, nécessite 1 IDE obésité et 1 Bureau CS

**Entretien diet** : 1 heure, nécessite 1 diététicien et 1 Bureau CS

**Synthèse** : 30 minutes, nécessite 1 nutritionniste et 1 Bureau CS

# H

## Guide développeur

Cette annexe est destinée aux développeurs souhaitant prendre en main la suite du projet. Ce guide va regrouper les instructions permettant d'installer le projet avec l'environnement requis. Ce guide est inspiré de celui de Yang Jing effectué lors du projet précédent [5].

### 1 Installation du projet

#### Mise en place de l'environnement de développement

L'application web a été implémentée sous l'IDE *netbeans*. La base de données utilisée est MySQL par l'intermédiaire de *WAMP*

#### Prérequis

Les différentes sources de l'application sont disponibles sur GitHub à l'adresse suivante : <https://github.com/RomainR37/ParcoursPatient>

Pour pouvoir utiliser l'application, il faut avoir un serveur web et d'un serveur de base de données sur sa machine de préférence *WAMP*.

#### Déploiement

Pour déployer l'application, il suffit de réaliser ces différentes étapes :

- Copier toutes les sources dans un dossier *www* (*WAMP* par exemple)
- Exécuter le script « *subway.sql* » dans *PhpMyAdmin* afin de créer et d'ajouter des données à la base de données
- Aller sur l'adresse suivant : `http://localhost/subway/`.

Auparavant, pensez à changer le mot de passe d'accès à MySQL dans le fichier *database.php* dans le répertoire *config*.

Pour avoir accès à l'intégralité de la plateforme, le nom d'utilisateur est "admin" avec pour mot de passe "admin".

## 2 Structure du projet

L'application web a été développée en utilisant les technologies suivantes :

- PHP
- HTML, CSS
- JavaScript

Pour le code PHP, nous avons utilisé un Framework PHP (codeIgniter) basé sur le modèle MVC (Modèle - Vue - Contrôleur). L'utilité de ce Framework est de pouvoir séparer les données des différentes vues permettant l'affichage de ces données. Concernant le CSS, nous avons également utilisé bootstrap afin d'avoir un rendu agréable sans avoir besoin de grandes connaissances dans le domaine.

Voici l'architecture générale de notre application web :

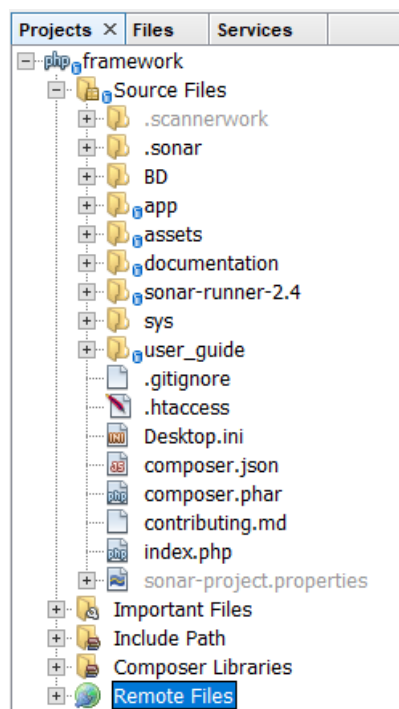


Figure 1 – Arborescence du projet

Nous nous plaçons dans le répertoire *source files* qui est le répertoire qui nous intéresse le plus. Dans ce répertoire, nous trouvons un dossier BD qui contient le script à exécuter permettant l'ajout de données tests afin d'utiliser l'application sans devoir créer ses propres données soi-même.

Le répertoire « assets » contient tous les fichiers images, javascript, css nécessaire à notre application. Le répertoire documentation contient toute la documentation nécessaire à la compréhension du code. Les répertoires « sys » et « user\_guide » ne nous intéressent pas forcément. Le premier contient tous les fichiers nécessaires à l'utilisation du Framework *codeIgniter*. Le répertoire qui nous intéresse le plus est le répertoire « app » qui contient l'intégralité du code de notre application.

Dans ce répertoire nous trouvons 4 sous répertoires :

- Config (répertoire de configuration du projet comme la base de données etc.)
- Controllers (tous les contrôleurs de notre application)
- Models (toutes les données de notre application : voir diagramme de classe)

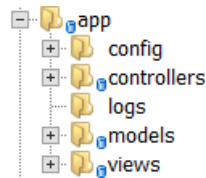


Figure 2 – Arborescence du répertoire source

— Views (les vues permettant l’affichage des données)

## Contrôleurs

Tous les contrôleurs héritent d’une classe *CI\_Controller* (contrôleur de base du Framework codeIgniter). Les contrôleurs permettant de faire le lien entre les données et les vues. Les contrôleurs permettent dans la majeure partie des cas de faire appel à un modèle qui lui-même exécute une requête à la base de données et le contrôleur envoie ces données à la vue qui se charge de les afficher. Nous allons prendre un exemple pour voir l’utilisation d’un contrôleur.

```

class Activites extends CI_Controller {

    function __construct() {
        parent::__construct();
        if ($this->session->userdata("username") === null)
            redirect('/Auth', 'refresh');
        if ($this->session->userdata("level") === "1")
            redirect('/AffichageSejour', 'refresh');
    }

    /**
     * Affichage la liste des différentes activités
     *
     * La méthode récupérer la liste des activités
     * et envoie ces données sur la page V_activite.
     * Cette page se charge d'afficher les données
     * \param      Aucun
     */
    public function index() {
        $this->load->model('M_Activite');
        $data = array();
        $data['activite'] = $this->M_Activite->getAllActivites();
        $data['chemin'] = '/activite/V_activite';
        $this->load->view('/V_generale', $data);
    }
}
  
```

Figure 3 – Aperçu d’un contrôleur

Dans l’exemple ci-dessus, nous prenons le cas du contrôleur « Activites » et la méthode « index » permettant d’afficher toutes les activités sur une page. La méthode charge le modèle nécessaire à la récupération des différentes activités, nous faisons appel à la méthode « getAllActivites » du modèle « M\_Activite » permettant de récupérer toute la liste des activités. Ensuite, nous créons un tableau data permettant de récupérer le résultat de la méthode « getAllActivites » ainsi que le chemin de la vue correspondante.

Ensuite, notre contrôleur charge la vue principale (voir dans la partie explication des vues) en lui donnant également le chemin de la vue « V\_activite » ainsi que les données provenant du modèle. Chaque contrôleur et chaque méthode sont basés sur cette structure.

## Modèles

Les modèles exécutent uniquement des requêtes à la base de données. Un exemple avec la Figure 4.

```

public function supprActivite($id) {
    $txt_sql = "DELETE FROM activite
                WHERE id_activite = " . $id;
    $query = $this->db->query($txt_sql);
    $sql = "DELETE FROM onglet
            WHERE id_onglet = " . $id;
    $query = $this->db->query($sql);
}

```

Figure 4 – Aperçu d’une fonction d’un modèle

## Vues

Les vues de notre application sont dans le dossier « views ». Chaque vue de notre application dépend d’une vue qu’on appelle vue générale (V\_generale). C’est dans la vue générale que l’on ajoute le menu, le pied de page ainsi que tous les fichiers js et css nécessaires au fonctionnement de notre application.

Le schéma de la structure d’une vue se trouve à la Figure 5.

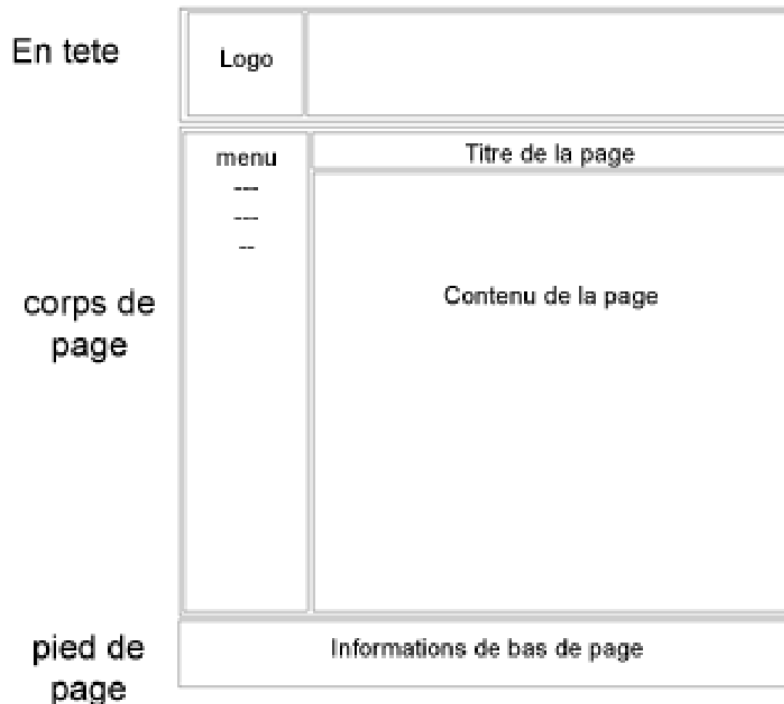


Figure 5 – Schéma d’une vue

Dans notre application, chaque vue contient la vue générale, c’est uniquement la partie contenue de la page qui est modifiée.

Voici le code expliquant ce mécanisme :

```

<?php
$this->view('/menu/V_menu');
$this->view($chemin);
?>

```

Figure 6 – Mécanisme de la vue

La variable « \$chemin » contient le chemin de la vue permettant d’afficher les données en

fonction des actions de l'utilisateur. C'est pour cela que nous devons définir une variable chemin dans nos différentes méthodes de nos contrôleurs.

## Bibliographie

- [1] Jean COQUELET. *Gestion et Optimisation des Parcours Patient*. fr. Rapp. tech., p. 62.
- [2] *Le CHU de Tours a lancé son service d'hôtel hospitalier*. fr. URL : <http://www.info-tours.fr/articles/tours/2018/07/30/9333/le-chu-de-tours-a-lance-son-service-d-hotel-hospitalier/> (visité le 17/10/2018).
- [3] Joren MARYNISSEN et Erik DEMEULEMEESTER. « Literature review on multi-appointment scheduling problems in hospitals ». en. In : *European Journal of Operational Research* 272.2 (jan. 2019), p. 407–419. ISSN : 03772217. DOI : [10.1016/j.ejor.2018.03.001](https://doi.org/10.1016/j.ejor.2018.03.001). URL : <https://linkinghub.elsevier.com/retrieve/pii/S0377221718302108> (visité le 22/11/2018).
- [4] Guillaume POCHET. *Outil de gestion de parcours patients dans un hôpital de jour*. fr. Rapp. tech., p. 62.
- [5] Jing YANG. *Outil de gestion de parcours patient dans un hôpital de jour*. fr. Rapp. tech., p. 56.

# Outil de gestion de parcours patient dans un hôpital de jour

## Résumé

Les hôpitaux sont l'objet de nombreux débats dans l'actualité, et l'un des points de discussion principal est la gestion des patients au sein des établissements. Un projet de développement d'une plateforme permettant de planifier les rendez-vous des patients au sein d'un hôpital de jour a été réalisé lors des années précédentes par des étudiants de l'école. La plateforme permet de gérer les patients, de leur assigner ce qu'on appelle un parcours patient, et d'agencer leur planning selon les ressources disponibles. Je vais reprendre ce projet pour corriger les divers problèmes présents et pour implémenter la planification automatique des activités des patients. Ajouter une planification automatique implique de nombreuses questions qui seront évoquées dans ce rapport.

## Mots-clés

plateforme, web, gestion, patients, planification automatique

## Abstract

Hospitals are subject to numerous talks in the news, and one of the main point is about the scheduling of patients. The creation of a platform allowing users to schedule patients activities in the hospital was made by student of the school in the years before. The application allows to add patients and resources, to assign them a course to follow, and to schedule with the resources available. I will continue this project by first, correct some bugs in the platform and to add automatic scheduling. The automatic scheduling implies lots of questions which will be answered in this report.

## Keywords

platform, web, management, patients, automatic scheduling