République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A/Mira de Béjaia
Faculté des Sciences Exactes
Département d'Informatique

MEMOIRE DE MASTER PROFESSIONNEL

En
Informatique

Option
Génie Logiciel

Thème

Conception et réalisation d'un module de gestion de temps avancé : Planning, Shifts, Pointage et Remplacements

Présenté par : M. MANSOURI Djoudi & M. KASSA Karim

Soutenu le 19 Septembre 2022 devant le jury composé de :

Président	**Mr A. AKILAL**	**M.A.A**	**U. A/Mira Béjaia.**
Examinateur	**Mr M. MOHAMMEDI**	**M.C.A**	**U. A/Mira Béjaia.**
Encadrante	**Mme N. YESSAD**	**M.C.B**	**U. A/Mira Béjaia.**
Co-Encadrante	**Mme M. SALI**	**M.O.A**	**Tech Instinct.**

Béjaia, Septembre 2022.

Remerciements

Au terme de ce projet de fin d'études, nous tenons à remercier en premier lieu le bon dieu de nous avoir donné force, foie, courage et patience pour réaliser ce travail.
Nous tenons aussi à remercier le CEO de Tech Instinct M. BELATTAF Youcef pour nous avoir accordé son entière confiance en nous offrant cette merveilleuse opportunité, sans quoi ce travail n'aurait probablement pas vu le jour, ainsi qu'un suivi personnalisé et une aide et expertise précieuse durant toutes les étapes du processus de développement.
Nous remercions notre encadrante de mémoire Mme. YESSAD Nawal, de nous avoir encadrée et conseillée tout au long du projet.
Nous voulons remercier également notre co-encadrante de mémoire Mme SALI Mina, responsable MOA chez Tech Instinct pour sa disponibilité, aide et conseils.
Nos remerciements s'adressent aussi à tous les membres de Tech Instinct qui ont su se rendre disponibles pour nous prêter main forte que ça soit en répondant à nos questions, suivre notre travail ou juste pour leurs soutien et encouragements (nous pensons plus particulièrement à Samy, Said, Mehani, Arezki et Braham).
Nos remerciements vont pareillement aux membres du jury pour avoir accepté d'examiner notre travail et de l'enrichir par leurs propositions.
Ainsi qu'à toutes les personnes qui ont contribués à la réalisation de notre projet de près ou de loin, et qui nous ont inspirés lors de la rédaction de ce mémoire.

*M. **MANSOURI Djoudi***
*M. **KASSA Karim***

Dédicaces

Je dédie cet humble travail tout d'abord à mes parents pour leurs indulgence et soutien indéfectible à mon égard, et pour tout les moyens qu'ils ont mis à ma disposition afin d'atteindre mes objectifs.
Je le dédie aussi à ma grand-mère, qui représente pour moi la personnification de la gentillesse et de la générosité.
A ma petite soeur et mon petit frère pour leurs inlassables encouragements.
Mais aussi à mes amis, cousins et toute personne ayant positivement impactée mon existence jusqu'à présent, mais aussi dans le futur...

*M. **MANSOURI Djoudi***

Dédicaces

A mes très chers parents pour tout les sacrifices à mon égard, leur patience, et leur confiance en moi. Que ce modeste travail soit un témoignage de ma profonde affection à leurs égards.
A mon très cher petit frère et mes cheres petites soeurs pour leurs encouragements et confiance.
A mes très chers amis (Mazigh, Yanis, Farid, Ali, Khelaf, Joseph, Billal) et sans oublier tout les autres.
A toute ma famille grand-parents, oncles et tantes, cousins et cousines.

*M. **KASSA Karim***

Table des matières

Table des figures

Liste des tableaux

Introduction générale

Une entreprise peut être définie comme une unité de production juridiquement autonome dont l'objectif est de produire des biens et/ou des services à destination de personnes physiques ou morales afin d'en tirer un bénéfice [1].

Cette dernière s'appuie sur un bon nombre de ressources, parmi elles : les ressources financières, matérielles, immatérielles, informationnelles et humaines afin d'accomplir les objectifs qu'elle s'est fixée. Ici ce qui nous intéresse le plus sont les Ressources Humaines (RH) et plus particulièrement le domaine de la Gestion des Ressources Humaines (GRH), qui est l'ensemble des pratiques mises en œuvre pour administrer, mobiliser et développer les ressources humaines impliquées dans l'activité d'une organisation donnée (dans notre cas, l'entreprise) [2].

Parmi les principales fonctions qu'assure la GRH, figure la gestion du temps des Ressources Humaines, qui consiste entre autres en la bonne répartition des horaires de travail des employés au sein d'une entreprise, grâce à la réalisation d'un planning (emploi du temps) global qui regroupe tous les employés et équipes d'employés, ou bien individuel, qui permettra à un employé de consulter et de voir uniquement ses horaires. La gestion du temps au sein des Ressources Humaines est cruciale, car elle permet de garantir le bon fonctionnement d'une entreprise, l'usage optimal des ressources et d'éviter au mieux les périodes d'oisiveté, car comme on dit souvent, le temps c'est de l'argent !

La problématique abordée réside dans le manque fonctionnel de la solution actuelle, qui ne réponds pas à tous les besoins des entreprises concernant les cas pratiques rencontrés quotidiennement au niveau de la planification et répartition des horaires de travail des employés.

Le but de ce projet sera de concevoir et de réaliser un module de plannings avancés qui va intégrer la notion de Shifts (sessions de travail) et de cycles, qui va par la suite être implémenté au sein du SIRH (Système d'information de gestion des ressources humaines) RH-Partner afin de l'enrichir et de le compléter davantage.

La décomposition de ce document se fera en 3 chapitres :

• Le premier chapitre jouera le rôle d'introduction du projet, il présentera le contexte, la problématique et l'entreprise d'accueil. Au sein de ce chapitre nous étalerons aussi les objectifs du projet, ainsi que la méthodologie de développement employée dans le but de les réaliser.

• Le deuxième chapitre quant à lui sera orienté conception, on y fera l'identification et l'analyse des besoins fonctionnels et non fonctionnels, pour y extraire les acteurs, les cas d'utilisations détaillés et modéliser les différents diagrammes associés (cas d'utilisation, séquence et classe), ainsi que la présentation du modèle relationnel de bases de données.

• Le troisième chapitre portera sur la réalisation du projet, dans lequel nous décrirons l'environnement de développement, les différentes technologies utilisées, l'architecture de l'application ainsi que sa présentation.

Chapitre 1

Cahier des charges et Méthodologie de Développement

1.1 Introduction

Pour commencer, lors de ce premier chapitre, nous allons décrire le contexte de notre projet, en présentant dans un premier temps l'entreprise concernée (Tech Instinct) qui nous a accueilli en stage, dans le but de résoudre la problématique qui sera abordée en second lieu. Pour à la fin conclure en parlant de la méthodologie de développement adoptée pour mener à bien le projet.

1.2 Contexte du projet

Le temps est une ressource inestimable, aussi bien pour les employeurs que les employés, et pour cette raison il se doit d'être géré de manière ferme, méticuleuse et rigoureuse. C'est bien dans ce cadre qu'entre en jeu la gestion du temps appliquée aux ressources humaines au sein de l'entreprise, qui a pour but de bien répartir les heures de travail sur un planning, d'organiser les différentes activités et de les assigner aux travailleurs.

Afin d'optimiser et de faciliter ce processus laborieux, les entreprises font usages de tous les moyens qui sont à leurs portée, et notamment des technologies de l'information et de la communication, et donc par extension des systèmes d'information (SI). D'où la naissance du concept des SIRH : Systèmes d'Information adaptés aux Ressources Humaines, ces derniers contiennent tous les outils nécessaires à la bonne gestion des ressources humaines, organisés sous forme de modules jouant un rôle bien spécifique, et celui qui nous intéresse le plus particulièrement ici est le module de gestion de temps, qui aura pour fonctionnalités principales : la consultation et la gestion des plannings et des shifts, ainsi que l'usage des données du pointage au sein de ce dernier.

1.3 Tech Instinct

Tech Instinct est un cabinet de conseil et de réalisation IT fondé en juillet 2018 et basé à Béjaïa, Algérie.

FIGURE 1.1 – Logo de Tech Instinct

Le domaine d'expertise de Tech Instinct se situe dans l'architecture des SI, du développement d'APIs, de plateformes SAAS et d'applications Web et Mobiles. La boite compte un nombre conséquent d'employés et traite avec beaucoup de clients étrangers, les prestations qu'elle fournit pourraient rivalisent avec ce qui se fait de mieux sur le plan international.
De Tech Instinct est née Business Platforms, une société de développement de logiciels formée aussi par les mêmes équipes, et comme son nom l'indique, Business Platforms a surtout pour but d'alimenter le marché national en solutions orientées pour les professionnels de différents domaines.

(Techniquement nous sommes stagiaires chez Tech Instinct, mais le produit que nous nous apprêtons à développer sera toujours pour le compte de Business Platforms.)

1.4 RH-Partner

RH-Partner est un SIRH 100% Algérien en mode SAAS (déployé entièrement sur le Cloud), initialement développé et édité par Business Platforms.

RH-Partner simplifie la gestion des ressources humaines et s'adresse aux professionnels du métier, à savoir les gérants, gestionnaires des ressources humaines, assistants administratifs et DRH.

De par son interface intuitive et ergonomique, RH-Partner est facile à prendre en main et offre la meilleure expérience utilisateur possible, ainsi qu'une fiabilité et des performances à toutes épreuves [3].

1.5 Étude de l'existant

L'organisme d'acceuil et la plateforme ayant été présentés, découvrons et étudions à présent dans cette section ce qui a été mis en place.

RH-Partner contient déjà un module de gestion du temps qui est divisé en trois sections :

• La section "Pointage", qui permet pour un employé de pointer sa présence, et à

l'administrateur d'avoir une vue d'ensemble des pointages des employés et groupes d'employés, les affectations et de les visualiser par mois. L'administrateur a aussi la possibilité de modifier, importer et exporter les données de pointage.

• La section "Planning", qui tout comme son nom l'indique permet à un employé de visualiser les plannings, et à un administrateur d'en créer en ajoutant un nom, et les jours de travail auxquels les horaires seront assignés. Un onglet pour référencer les jours fériés est aussi présent, ainsi qu'un autre pour gérer les pointages selon les plannings.

• La section "Zone de pointage", cette dernière sert à l'administrateur de point de création des modèles QR Code, qui seront par la suite scannés par les employés dans le but de marquer leurs présences au sein du système de pointage.

1.6 Problématique abordée

La problématique qui se pose est purement d'ordre fonctionnelle, la solution actuelle ne réponds pas à tous les besoins de l'entreprise en termes de fonctionnalités, et n'arrive pas à satisfaire tout les cas rencontrés au niveau de la planification des horaires de travail des employés.

Le module a été présenté comme un module de gestion du temps très simple, peut-être même trop simple, étant trop axé sur la partie pointage, et délaissant un peu la partie emploi du temps.

De plus, ce dernier pourrait faire usage d'une interface plus appropriée pour représenter au mieux les plannings.

1.7 Objectifs

La finalité du projet consiste à réaliser un module de gestion du temps, comprenant des concepts tels que les shifts, les cycles, la gestion de plannings et remplacements, ce même module est à intégrer à un SIRH déjà existant (Rh-Partner) et ce dans le but de l'améliorer et de répondre aux besoins les plus exigeants des entreprises en termes de Gestion des Ressources Humaines.

Ceci étant dit, les objectifs principaux de ce projet sont :

- La conception d'un Planning qui répondra à une gestion d'employés et/ou d'équipes d'employés.
- L'intégration d'un système de rotations à travers le concept de Cycles.
- Simplifier et augmenter en fiabilité la comparaison des pointages avec les Shifts associés.
- Proposer une meilleure gestion des remplacements au sein des Shifts.

Une fois ce module finalisé et intégré à la plateforme, nous avons comme attentes qu'il sera utilisé pleinement par les entreprises visées, à savoir les cliniques, les hôtels, les restaurants, les chaînes de supermachés et autres établissements démontrant un certain besoin de gérer les sessions de travail de leurs employés et équipes d'employés de manière agile et dynamique.

1.8 Méthodologie de développement

Une méthodologie de développement désigne le processus emprunté par une équipe de développement dans le but de mener à bien un projet donné.

Cette méthodologie ou juste méthode veille à la bonne conception, réalisation et amélioration d'un système logiciel, et ce suivant de rigoureux critères de qualité et de conformité, le tout en minimisant les ressources humaines, matérielles et temporelles employées.

Cette dernière permet aussi l'optimisation des livrables ainsi que la détection prématurée d'erreurs, pour les corriger avant qu'elles ne causent préjudice aux différentes parties prenantes du projet, ce qui pourrait produire des coûts considérables si elles étaient détectées tardivement.

Pour conclure cette section, quelques méthodes de développements connues : Agile, Lean, Itérative, DevOps, Cascade... etc. Et en ce qui nous concerne, après concertation on en a convenu que le processus de développement le plus adéquat serait Agile, et plus particulièrement SCRUM.

1.8.1 La méthode Agile

Dans un monde où le logiciel est quasiment partout, où il a prit une ampleur démesurée, devenant de plus en plus complexe et nécessitant une croissance multidimensionnelle, les méthodes de développement dites classiques ne sont pas toujours adéquates pour répondre aux exigences de certains produits. C'est là que les méthodes Agiles interviennent, elles permettent un développement totalement personnalisé, adaptatif, avec des livraisons précoces et un cycle de vie flexible.

Les méthodes Agiles ont pour géniteurs plusieurs grands noms du développement de logiciels, parmi eux figurent Kent Beck, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, James Grenning, Martin Fowler, Jim Highsmith, Andrew Hunt... et plein d'autres. Ces derniers ont signés un manifeste en Février 2001, ce même manifeste repose sur 4 valeurs qui sont décrites de la sorte :

- Le travail d'équipe et la bonne communication prime sur l'individualisme et l'efficacité des outils.

- Mieux vaut un logiciel fonctionnel qu'une documentation complète.

- Il est toujours préférable de collaborer avec un client et de le faire participer au développement de son produit que de lui faire signer un contrat.

- Rester attentif aux changements au lieu de se fier à la plannification.

Ainsi que 12 principes, que voici :

I - La priorité absolue est de satisfaire le client à travers des livraisons précoces et continues du logiciel.

II - Il faut toujours acceuillir les changements liés aux besoins, même tard dans le développement. Car la méthode Agile permet au client de rester compétitif à travers ces changement.

III - Les livraisons se doivent d'être faites le plus fréquemment possible.

IV - Les développeurs et les commerciaux doivent travailler côte à côte quotidiennement tout au long du projet.

V - Il est impératif de s'entourer d'éléments motivés pour conrétiser le projet, il faudra toutesfois aussi leurs fournir l'environnement de travail adéquat, et leurs faire confiance pour le reste.

VI - La manière la plus efficace de faire passer des informations, de transmettre le message à et au sein de l'équipe reste les conversations en face à face.

VII - Un logiciel qui marche est la principale mesure du progrès.

VIII - Les processus Agiles favorisent le développement durable. Les sponsors, les développeurs et les utilisateurs devraient être capables de maintenir un certain rythme constant de manière indéfini.

IX - Prêter continuellement attention à l'excellence du côté technique et à la bonne architecture valorise la méthode Agile.

X - La simplicité permet de minimiser les efforts en évitant de se perdre dans des détails inutiles.

XI - Les meilleures conceptions, designs et architectures émergent d'équipes auto-organisées.

XII - Les équipes devraient méditer sur comment devenir plus efficaces à intervales réguliers, en ajustant leurs habitudes et méthodes.

Veillez noter toutesfois qu'Agile est avant tout une philosophie, donc cet ensemble de valeurs et principes ne sont guère des règles à suivre ou commandements gravés sur le marbre, elles restent toujours ouvertes aux interprétations et perfectionnements [4].

1.8.2 Scrum

Scrum est l'un des Framework de développement Agiles les plus populaires, pour ne pas dire le plus populaire. Il est défini par ses créateurs comme un « cadre de travail holistique itératif qui se concentre sur les buts communs en livrant de manière productive et créative des produits de la plus grande valeur possible » [5].
Scrum vient tout droit du domaine du Rugby, c'est un terme purement anglophone qui veut dire "mêlée", il décrit le processus métaphoriquement en comparant l'équipe de développement à celle du sport en question, où les joueurs forment un bloc et essaient de progresser jusqu'à l'en-but en faisant des passes pour marquer un essai. De la même manière, les différents acteurs de la méthode

Scrum avancent main dans la main à travers des boîtes de temps appelées "Sprints", l'objectif final étant de livrer une version ou une fonctionnalité du produit au terme de celles-ci.

1.8.2.1 Acteurs du processus Scrum

Dans cette sous-section nous aborderons en détail les différents acteurs qui composent et intéragissent entre eux au sein de la méthode Scrum :

a) Le Product Owner : Plus connu en tant que "Directeur du produit", ses responsabilités sont de réaliser et de maintenir le cahier des charges, en déterminant avec précision les besoins des clients qu'il représente entre autres, et en assignant les priorités liées au projet.

b) Le Scrum Master : Il est un peu le protecteur et le métronome de l'équipe, il a pour objectifs de faciliter et d'améliorer la productivité globale de celle-ci en veillant à ce que le cadre de travail soit agréable, fluide et dépourvu de distractions et autres entraves extérieures. Le Scrum Master n'est hiérarchiquement pas supérieur aux autres membres de l'équipe, et n'a aucune relation directe avec les clients.

c) Les développeurs : Ils sont le fer de lance de l'équipe Scrum, ils sont chargés de mener à bien les différentes missions et tâches du projet, et ont pour responsabilité de délivrer une version du produit au bout de chaque Sprint.

1.8.2.2 Entités et évènements liés à Scrum

Cette sous-section énumère et définit les différentes caractéristiques et évènements liés à Scrum :
I) Le Product Backlog : Il représente la liste des exigences, fonctionnalités, améliorations et correctifs à intégrer au produit, cette liste est perpétuellement sujette aux changements.

II) Le Sprint : Pour faire simple, c'est une itération, qui s'étend en pratique entre 2 et 4 semaines. Au cours duquelle l'équipe a pour mission de concrétiser les objectifs en termes de fonctionnalités définies au tout début de celle-ci dans ce qu'on appelle un Sprint Backlog.

III) Le Sprint Backlog : Le Sprint Backlog est un document créée pendant la planification du Sprint qui contient une liste des tâches à réaliser au cours de ce dernier.

IV) La Planification d'un Sprint : La Planification du Sprint se fait à travers une réunion avec le Product Owner, qui a pour objectif de définir les priorités de réalisation des fonctionnalités au sein d'un Product Backlog.

V) Le Daily Scrum : Le Daily Scrum est un outil de communication, en prenant 10 à 15 minutes quotidiennement, chaque membre de l'équipe, tour à tour, décrit l'état d'avancement de ses tâches : ce qui a été réalisé le jour précédent et ce qui va être réalisé dans la journée, tout en réalisant aussi l'identification et la résolution des problèmes auxquels ils se sont heurtés, tout ça dans le but de déduire l'état d'avancement du sprint.

VI) Le Sprint Review : Le Sprint Review a pour utilité de dresser un bilan du Sprint et de l'exposer au Product Owner dans l'optique de recevoir ses feedbacks concernant les fonctionnalités achevées.

VII) Le Sprint Retrospective : Vient après le Sprint Review, et est une réunion axée sur comme son nom l'indique : La rétrospective. Cette dernière se concentre sur ce qui peut être mieux fait, amélioré, perfectionné par rapport au Sprint précédent.

VIII) Le Burndown Chart : Le Burndown Chart est en quelque sorte un graphique décrivant l'évolution du projet, généralement dans l'axe horizontal on a les jours, et au niveau de l'axe vertical les points de complexité restant à chaque nouvelle journée. A partir de ces deux axes est formé un tracé qui décrémente au fil du temps qui révèle la quantité de travail restante.

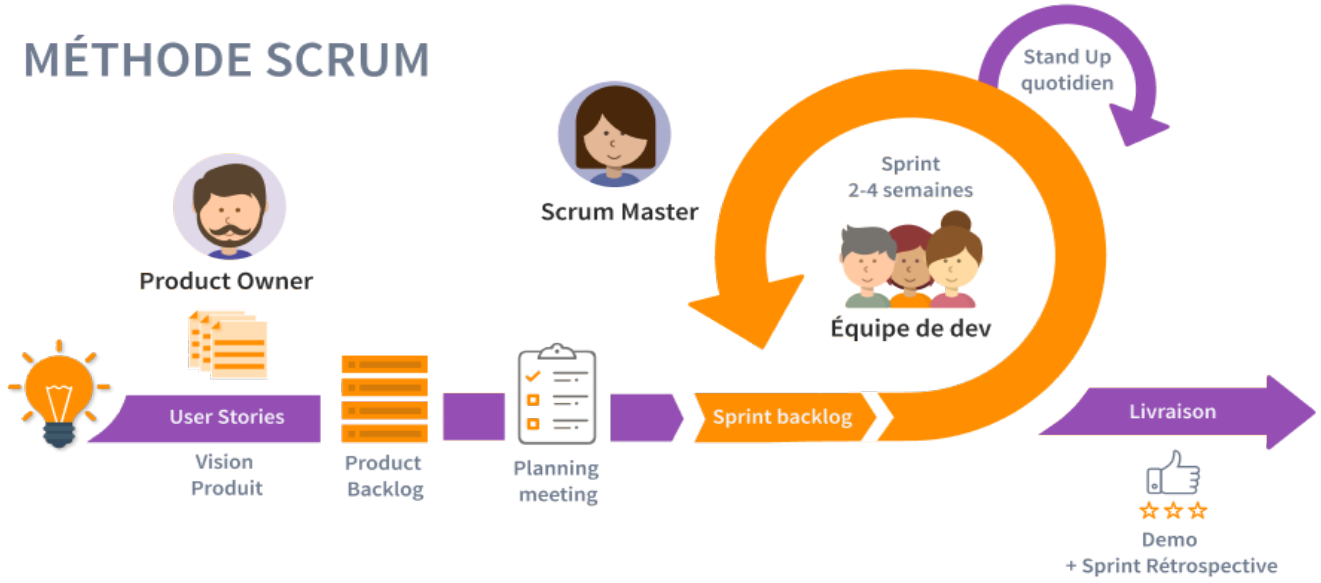

FIGURE 1.2 – Diagramme illustrant le fonctionnement de Scrum

1.9 Conclusion

Dans ce premier chapitre, nous avons eu l'occasion d'établir une présentation générale du projet : Son contexte, la problématique et ses objectifs.
Après cela nous avons aussi parlé de la méthodologie de développement qui sera utilisée pour assurer la bonne conception et réalisation de ce dernier.

Le deuxième chapitre rentrera quant à lui dans le vif du sujet, et étalera la spécification et l'analyse des besoins, ainsi que la conception détaillée de la solution.

Chapitre 2

Analyse des besoins et Conception

2.1 Introduction

Dans le précédent chapitre nous avons exposé la problématique et la solution préconisée. Pour ce qui en est de celui-ci il sera consacré à l'analyse des besoins et à la conception.

D'abord on aura la phase d'analyse, où l'on va commencer par identifier les besoins fonctionnels et non fonctionnels du système, à part desquels on va extraire les différents cas d'utilisations.

Ensuite, au sein de la phase de conception, nous allons illustrer quelques diagrammes de séquence, pour au final établir le diagramme de classes qui sera par la suite traduit au relationnel.

2.2 Spécification des besoins

2.2.1 Besoins fonctionnels

Un besoin fonctionnel est un besoin spécifiant le comportement du système, en d'autres mots c'est ce que le système peut nous permettre de faire.
Dans cette section nous allons présenter sous forme de tableau les différents besoins du projet ainsi que les fonctionnalités qui répondront à ces derniers.

Besoins	**Fonctionnalités**
Module composé de plusieurs petits espaces qui serviront à faciliter la gestion des shifts des employés et des équipes.	Espace de gestion des employés
	Espace de gestion des équipes
	Espace de gestion des plages horaires
	Espace de gestion des cycles
	Espace de gestion des shifts
	Interface de visualisation du planning global et par employé (grâce à la fonction de recherche)
	Intégration des shifts au planning.
	Options pour gérer les remplacements.
Faire usage des données de pointage pour valider un shift.	Les données de pointage seront récupérées pour les comparer avec le planning et notamment les shifts auxquels les employés ont été assignés.
	Affecter en conséquence la paie.

Tableau 2.1 – Besoins fonctionnels

2.2.2 Besoins non fonctionnels

Les besoins non fonctionnel sont les caratéristiques du système, en matière de performance et autres besoins incontournables tels que la sécurité etc. Dans cette section nous avons les différents besoins non fonctionnels ainsi que leurs descriptions toujours sous forme de tableau

Dénomination	Description
Robustesse	Les données de pointage doivent être précises et en concordance avec celles des shifts pour que la somme horaire communiquée à la paie soit celle attendue sans défaillances.
Utilisabilité	L'interface doit être intuitive et facile à manipuler par l'employé et surtout l'administrateur.
Sécurité	Les communications client-serveur doivent être sécurisés et confidentielles vis-à-vis d'autres employés.
Maintenabilité	Les différents fragments de code liés à ces nouvelles fonctionnalités doivent être faciles à éditer, maintenir et tester.
Scalabilité	L'application doit être en mesure de gérer un gros volume de shifts, plannings et salariés.
Rendement	L'application doit accomplir une exécution assez conséquente vis-à-vis du temps et des ressources.
Documentation	Fourniture d'un document pour l'architecture globale et aux différentes unités du module.
Portabilité	L'application devrait être accessible sur tout type de plateformes.

Tableau 2.2 – Besoins non fonctionnels

2.3 Identification des acteurs

Acteurs	Description
Employé	Cet acteur aura accès à certaines fonctionnalités du module, il pourra pointer sa présence, communiquer sa disponibilité et consulter son planning.
Administrateur	Cet acteur aura accès à toutes les fonctionnalités du module, il pourra notamment gérer les plannings, les pointages de présenceset les remplacements.

Tableau 2.3 – Identification des acteurs

2.4 Identification des cas d'utilisation

Un cas d'utilisation est une unité discrète d'interaction entre l'utilisateur et le système.

Cas d'utilisation				**Acteurs**
Module des Shifts	Accéder à l'espace du module des Shifts	Lister les employés		Employé/ Administrateur
		Lister les équipes		Employé/ Administrateur
		Lister les plages horaires		Employé/ Administrateur
		Lister les cycles		Employé/ Administrateur
		Lister les shifts		Employé/ Administrateur
		Visualiser son planning		Employé/ Administrateur
		Communiquer ses disponibilités à l'administrateur		Employé/ Administrateur
	Gérer le module des Shifts	Gérer Employés	Créer un employé	Administrateur
			Modifier un employé	Administrateur
			Archiver un employé	Administrateur
		Gérer Equipes	Créer une équipe	Administrateur
			Modifier une équipe	Administrateur
			Supprimer une équipe	Administrateur
		Gérer Plages Horaires	Créer une plage horaire	Administrateur
			Modifier une plage horaire	Administrateur
			Supprimer une plage horaire	Administrateur
		Gérer Cycles	Créer un cycle	Administrateur
			Modifier un cycle	Administrateur
			Supprimer un cycle	Administrateur
		Gérer Shifts	Créer un shift	Administrateur
			Modifier un shift	Administrateur
			Supprimer un shift	Administrateur
		Ajouter un shift au planning (générer le planning)		Administrateur
		Gérer les remplacements au sein d'un shift		Administrateur
Utiliser le pointage au sein des shifts		Comparer les données de pointage aux shifts associés		Administrateur
		Valider les données de présence et les transmettre au module de paie		Administrateur

Tableau 2.4 – Identification des cas d'utilisation

2.5 Description détaillée des cas d'utilisation

2.5.1 Cas d'utilisation «Gestion du module des shifts»

Objectif : Accéder à l'espace du module des shifts afin de gérer les shifts des équipes d'employés.

Accès à l'espace du module des shifts

Pré-conditions : être employé de l'entreprise et être authentifié dans le système.

Scénario nominal :

1. L'employé accède à l'espace du modules des shifts, au sein duquel il est acceuilli par le planning global.
   Cas du listage :
2. L'employé, en naviguant vers la section adéquate a la possibilité de lister et parcourir les employés, équipes, plages horaires, cycles et shifts disponibles.
   Cas de la visualisation :
3. Un employé peut visualiser son planning pour obtenir plus de détails sur ses journées de travail et ses horaires.
4. Il peut utiliser la fonction de recherche pour visualiser un planning bien précis.
   Cas de la transmission des disponibilités à l'administrateur :
5. Un employé peut communiquer ses disponibilités à l'administration, qui par la suite va décider si elle va les prendre en considération ou non.

Gérer le module des Shifts

Pré-conditions : être authentifié en tant qu'administrateur du système.

Scénario nominal :

Cas de la création

1. L'administrateur accède à l'espace du modules des shifts.
2. L'administrateur, dépendamment de la section CRUD dans laquelle il se trouve (Employés, Equipes, Plages Horaires, Cycles ou Shifts), il peut cliquer sur le bouton en haut à gauche pour ouvrir une boite de dialogue qui représente un formulaire à remplir pour créer l'instance associée à l'entité de son choix.

3. Une boite de dialogue s'affiche, dans laquelle se trouve un formulaire différent à remplir pour chaque section CRUD.
4. L'administrateur valide et envoie le formulaire pour finaliser la création.

Cas de la modification :

3. L'administrateur clique sur les 3 points horizontaux à droite au sein de la ligne abritant l'instance qu'il souhaite modifier.
4. Un petit menu s'ouvre dans lequelle l'option modifier se propose à lui, il clique dessus.
5. Une boite de dialogue s'affiche, dans laquelle se trouve un formulaire identique à celui de la création, mais à la différence près que les champs ne sont pas vides mais contiennent les informations stockées au sein de l'instance.
6. L'administrateur modifie les champs qu'il souhaite modifier puis valide les modifications.

Cas de la suppression

3. L'administrateur clique sur les 3 points horizontaux à droite au sein de la ligne abritant l'instance qu'il souhaite supprimer.
4. Un petit menu s'ouvre dans lequelle l'option supprimer se propose à lui, il clique dessus.
5. Une boite de dialogue s'ouvre, demandant à l'administrateur de confirmer la suppression de l'instance concernée.

Cas de la gestion des remplacements au sein d'un shift :

Pré-conditions : réception d'une notification d'absence à un shift de la part d'un employé, et disponibilité de l'employé qui va le remplacer.

1. L'administrateur reçoit une notification d'absence à un futur shift.
2. L'administrateur consulte le shift en question.
3. L'administrateur assigne au shift un remplaçant parmi les employés libres et le valide.

Sénario alternatif :

1. L'administrateur consulte l'état d'un shift et remarque qu'un employé affecté à ce shift a signalé son absence.
2. L'administrateur assigne un remplaçant au shift et le valide.

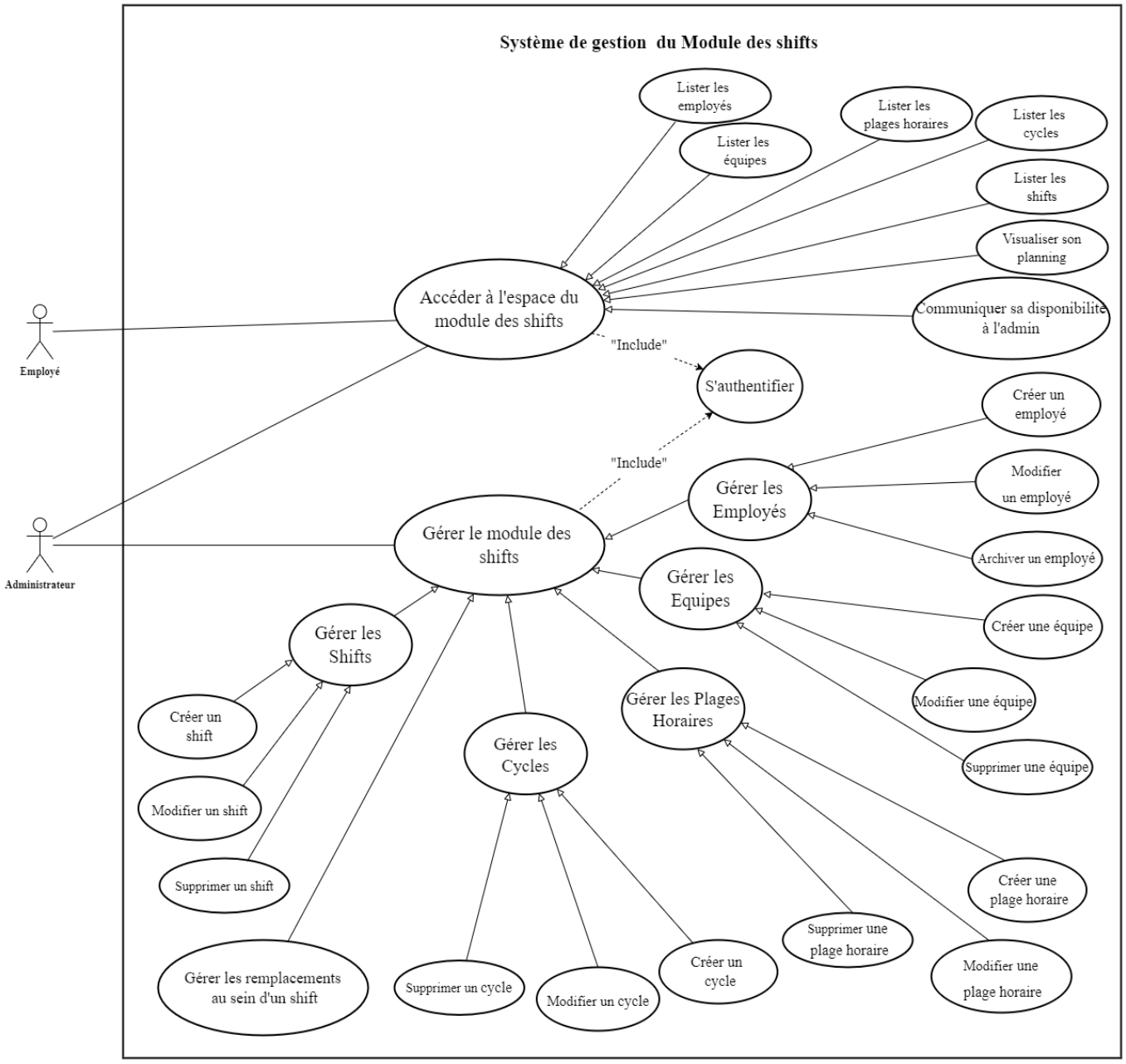

FIGURE 2.1 – Diagramme de cas d'utilisation gestion géneralisée du module des shifts

2.5.2 Cas d'utilisation «Utiliser le pointage au sein des shifts.»

Cas d'utilisation « Usage des données de pointage au sein des shifts ».

Objectif : Utiliser les données des pointages et les comparer au planning afin de générer un état.

Pré-conditions : être administrateur de l'entreprise et être authentifié dans le système en tant que tel.

Scénario nominal :

1. L'administrateur accède à l'espace des données de pointage au sein des shifts.

Cas de comparaison des données de pointage aux shifts assignés :

2. L'administrateur clique sur l'option de récupération des pointages.
3. Le système récupère les données à partir du module de pointage.
4. Le système compare les pointages aux shifts.
5. Le système retourne les shifts complétés et ceux qui ne l'ont pas été.
6. L'administrateur peut à présent choisir d'affecter la paie suivant les résultats de l'opération précédente.

Cas de la validation des données de présence et leur transmission au module de paie :

7. L'administrateur valide les données de présence et les transmet au module de paie.

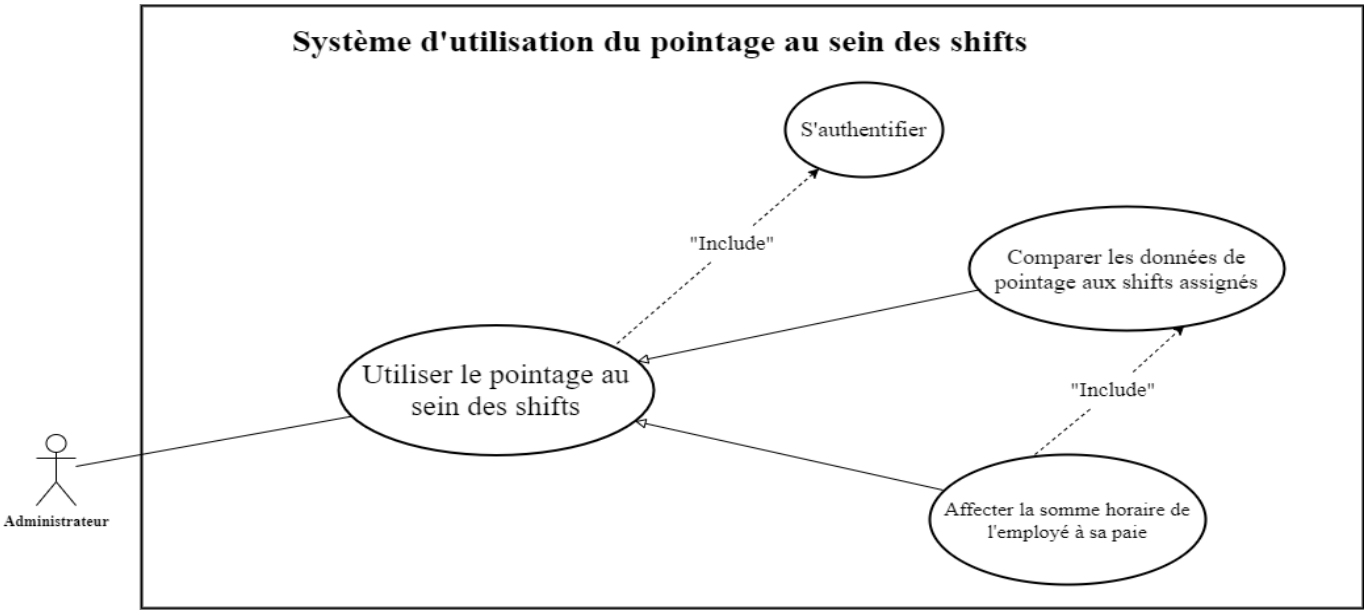

FIGURE 2.2 – Diagramme de cas d'utilisation utiliser le pointage au sein des shifts

2.6 Diagrammes de séquences

Les diagrammes de séquences sont des diagrammes qui décrivent les interactions entre les objets du système tout en indiquant l'ordre et la chronologie des échanges [6]. Dans cette section seront présentés quelques les diagrammes de séquences essentiels :

2.6.1 Diagramme de séquence du cas d'utilisation créer un shift :

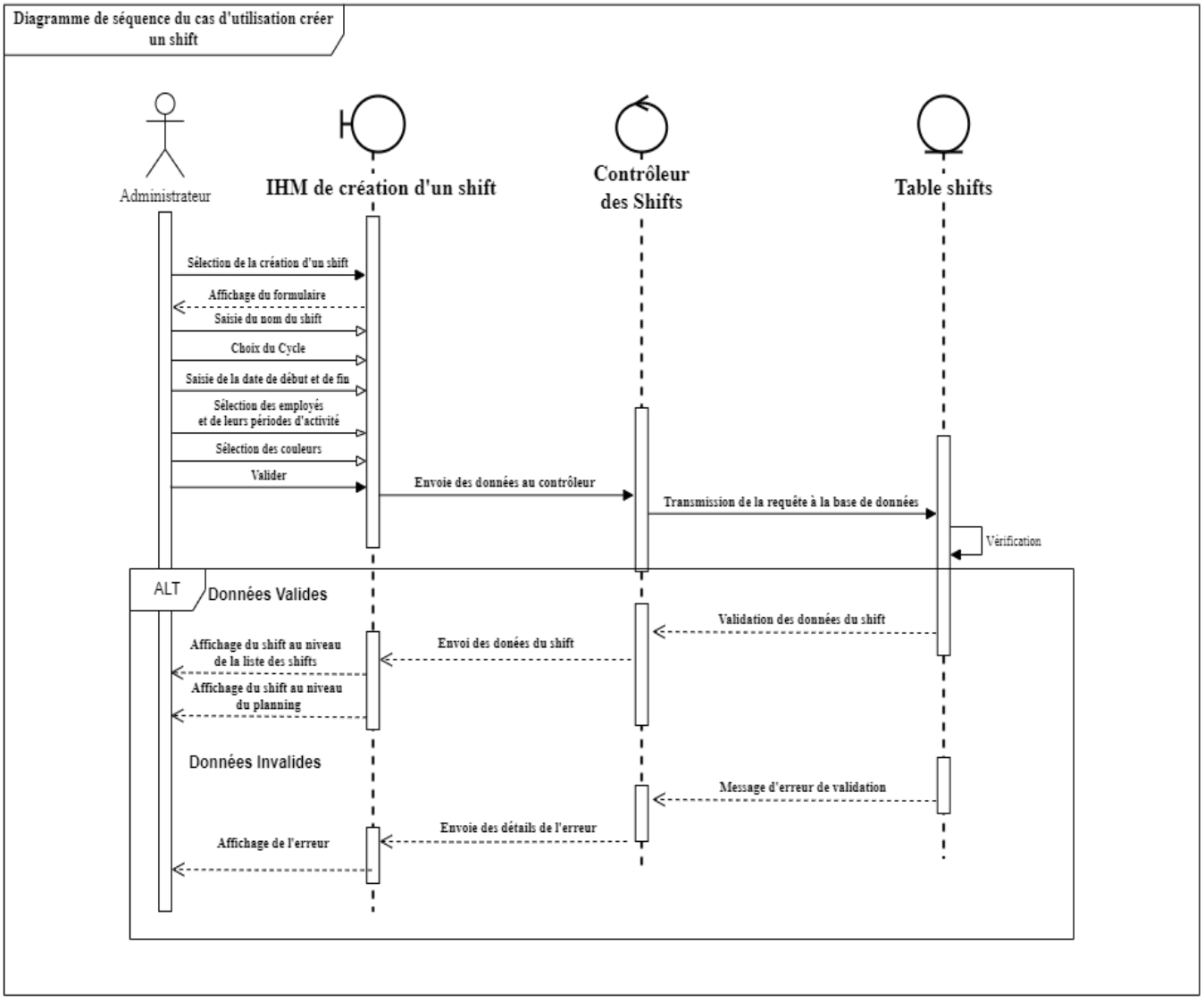

FIGURE 2.3 – Diagramme de séquence du cas d'utilisation créer un shift

2.6.2 Diagramme de séquence du cas d'utilisation notifier l'administrateur de sa disponibilité :

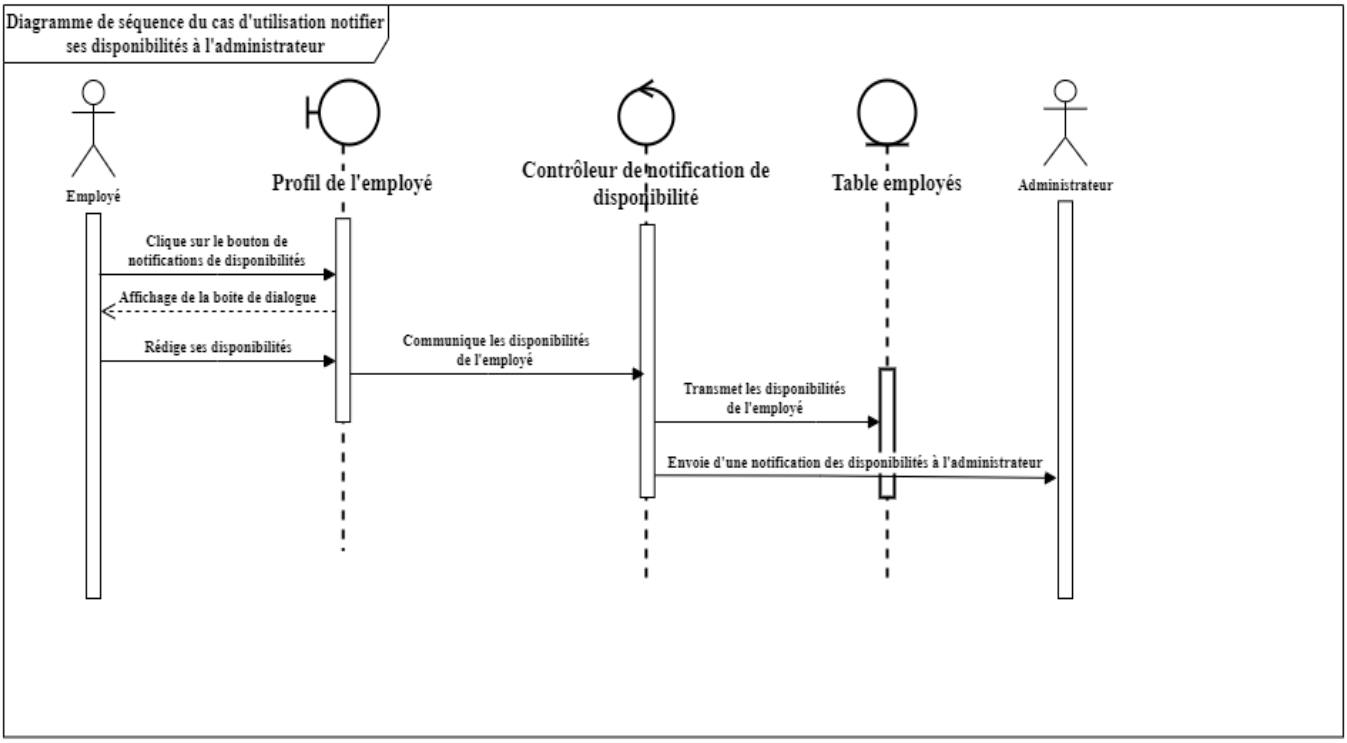

FIGURE 2.4 – Diagramme de séquence du cas d'utilisation notifier l'administrateur de sa disponibilité

2.6.3 Diagramme de séquence du cas d'utilisation gérer les remplacements au sein d'un shift :

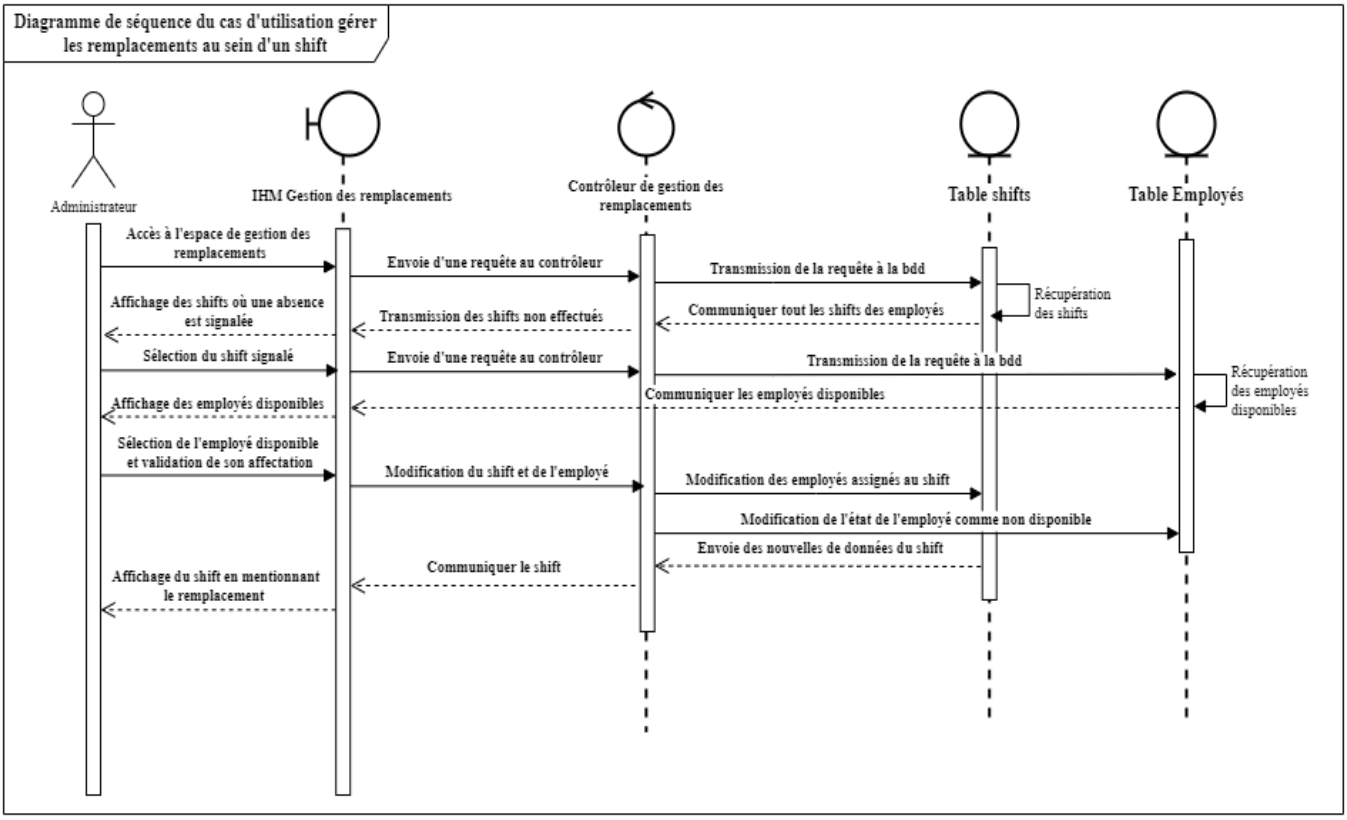

FIGURE 2.5 – Diagramme de séquence du cas d'utilisation gérer les remplacements

2.7 Diagramme de classes

Les diagrammes de classes sont l'un des plus importants éléments pour la modélisation UML. En fait, ce graphique est une description statique du système en cours de développement. Cette représentation est centrée sur le concept de classe, association. Chaque classe est décrite par les données et les traitements dans lesquels elle réside. Le traitement est réalisé par des opérations, sauf que les détails de ce dernier ne sont pas reflétés au sein du diagramme [7].

2.7.1 Dictionnaire de données :

Classes	Attributs	Type	Description	Méthodes associées
Employé	infos	Objet	Toutes les informations relatives à un employé, tel que l'identifiant,nom ..etc. Ces données existent déjà dans l'application mise en place	lister_employes()
				lister_equipes()
				lister_plages_horaires()
				lister_cycles()
				lister_shifts()
	vlm_horaire	Réel	Volume horaire hebdomadaire de l'employé.	visualiser_planning()
				commu_dispo()
Equipe	nom	Chaine	Nom de l'équipe	
	employes	Tableau	Les employés assignés à l'équipe	
Administrateur	infos	Objet	Le même attribut que pour l'employé.	Creer_employe()
	badge_admin	Booléen	Cet attribut prouve que cet utilisateur est un administrateur du système.	Modifier_employe()
				Archiver_employe()
				Creer_equipe()
				Modifier_equipe()
				Suppr_equipe()
				Creer_plage_horaire()
				Modifier_plage_horaire()
				Suppr_plage_horaire()
				Creer_cycle()
				Modifier_cycle()
				Suppr_cycle()
				Creer_shift()
				Modifier_shift()
				Suppr_shift()
				Gerer_rempl()
				Duplq_shift()
Shift	nom	Chaine	Nom du shift.	
	cycle	Objet	Cycle relié au Shift.	
	employees	Tableau	Les employés (équipes) assignés au shift.	
	date_debut	Date	La date de début du shift.	
	date_fin	Date	La date de fin du shift.	
	clr_principale	Chaine	Couleur de fond du shfit	
	clr_secondaire	Chaine	Couleur de la bordure du shfit	
Cycle	nom	Chaine	Nom du cycle.	
	jour_cycle	Tableau	Les jours qui constituent le shift	
JoursCycle	code_jour	Chaine	Le code assigné au jour, pouvant être un simple numéro ou bien le nom complet de la journée.	
	plage_horaire	Objet	La plage horaire assignée au jour	
PlageHoraire	nom_plg_hrr	Chaine	Nom de la plage horaire.	
	heure_debut	Temps	L'heure du début la plage horaire.	
	heure_fin	Temps	L'heure ou termine la plage horaire.	
	duree	Temps	Durée de la plage horaire	
Planning	shifts	Tableau	L'ensemble des shifts que va afficher le planning	
	jours_feries	Tableau	Liste des jours fériés.	

Tableau 2.5 – Dictionnaire de données du diagramme de classes

2.7.2 Définition des différentes entités du diagramme :

Cette section définit les différentes entités (hormis l'employé et l'administrateur qui sont des acteurs ayants étés définis bien plus haut au sein de la section identification des acteurs), qui sont au nombre de 6 :

Équipe : Une équipe est un ensemble d'employé soit chargé d'une même fonction ou bien œuvre en collaboration afin de réaliser un certain objectif dans l'entreprise.

Les plages horaires : Elles définissent avec exactitude les heures durant lesquelles les employés se doivent d'exercer leurs fonctions au sein de leurs lieux de travail, une plage horaire a une heure de début et une heure de fin, et une journée d'un cycle peut avoir plusieurs plages horaires, par exemple : Plage 1 (matin) va de 8h à 12h, Plage 2 (soir) s'étend de 13h à 17h, ...etc. Et le temps entre deux plages représente une pause.

Les jours d'un cycle : Ils sont la composante essentielle d'un cycle, dans la vie réelle ils représentent les journées de la semaine ou du mois ou de l'année, peu importe, ils peuvent porter des noms (par exemple, dans le système hebdomadaire : Dimanche, Lundi, Mardi, ...etc), ou bien des numéros pour le système mensuel ou personnalisé, ce qui est pratique dans les cycles de travail où les jours de repos ne sont pas fixes. Ces journées comportent des plages horaires.

Le cycle : Un cycle dans ce contexte-là est une entité associée à un ou plusieurs shifts et fait office de répétition de ces derniers dans le temps, le but étant de préciser les jours durant lesquels un employé (ou une équipe) donné(e) devrait être présent(e) sur le poste, ainsi que ses journées de repos. Un cycle contient donc des jours.

Le Shift : Les shifts représentent simplement les sessions de travail des employés, ils constituent une manière efficace de répartir et de visualiser le volume horaire des travailleurs au sein d'un planning donné, qu'il soit individuel ou bien global. Ces derniers peuvent être reliés à un cycle afin de bien gérer la répétition fréquente des horaires ainsi que les systèmes de rotation.

Le planning : Le planning est le contenant général de toutes les entités citées plus haut, il abrite en son sein tous les Shifts ainsi que leurs détails (équipes affectées, cycles, plages horaires, ...etc). Il sert de représentation globale (vue) du système.

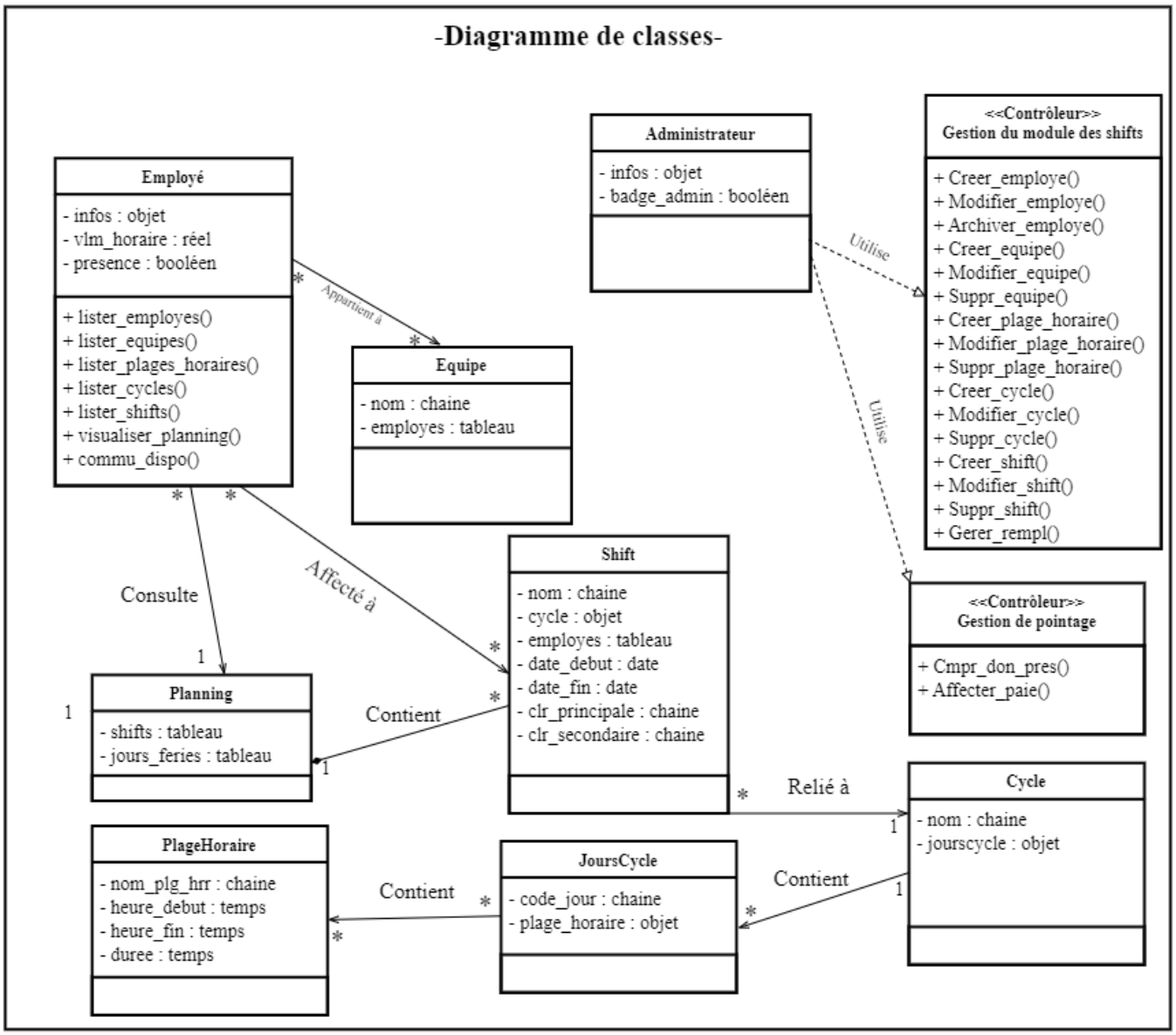

FIGURE 2.6 – Diagramme de classes

2.7.3 Le modèle relationnel

Pour passer du diagramme de classes au modèle relationnel on applique quatre règles qui sont les suivantes :

— **Règle 01 :** Association un-a-plusieurs implique l'ajout d'un attribut de type clé étrangère dans la relation de l'association. L'attribut porte le nom de la clé primaire de la relation père de l'association.

— **Règle 02 :** Association un-a-un implique l'ajout d'un attribut de type clé étrangère dans la relation dérivée de l'entité ayant la cardinalité minimale égale à zéro. Dans le cas de UML, il faut ajouter un attribut clé étrangère dans la relation dérivée de la classe ayant la

multiplicité minimale égale à un. L'attribut porte le nom de la clé primaire de la relation dérivée de l'entité (classe) connectée à l'association..

— **Règle 03 :** Association N-M on ajoute une table qui aura comme clé primaire la combinaison des deux clés étrangères des parties prenantes a l'association [8].

On appliquant ces règles on aura le modèle suivant :

— **adminsitrateur** ($\underline{infos}, badge\_admin$)
— **cycle** ($\underline{id}, nom$)
— **shift** ($\underline{id}, nom, date\_debut\_shift, date\_fin\_shift, \#cycle\_id, clr\_principale, clr\_secondaire$)
— **employe** ($\underline{infos}, vlm_h oraire$)
— **employe_shift** ($\underline{id}, date\_debut, date\_fin, \#infos.employe\_id, \#shift\_id$)
— **equipe** ($\underline{id}, nom$)
— **equipe_shift** ($\underline{id}, date\_debut, date\_fin, \#equipe\_id, \#shift\_id$)
— **jour_cycle** ($\underline{id}, code\_jour, \#cycle\_id$)
— **plage_horaire** ($\underline{id}, nom\_plg\_hrr, heure\_debut, heure\_fin, duree$)
— **rel_employe__equipe** ($\underline{\#equipe\_id, \#infos.employe\_id}$)
— **rel_plage_horaire__jour_cycle** ($\underline{\#jour\_cycle\_id, \#plage\_horaire\_id}$)

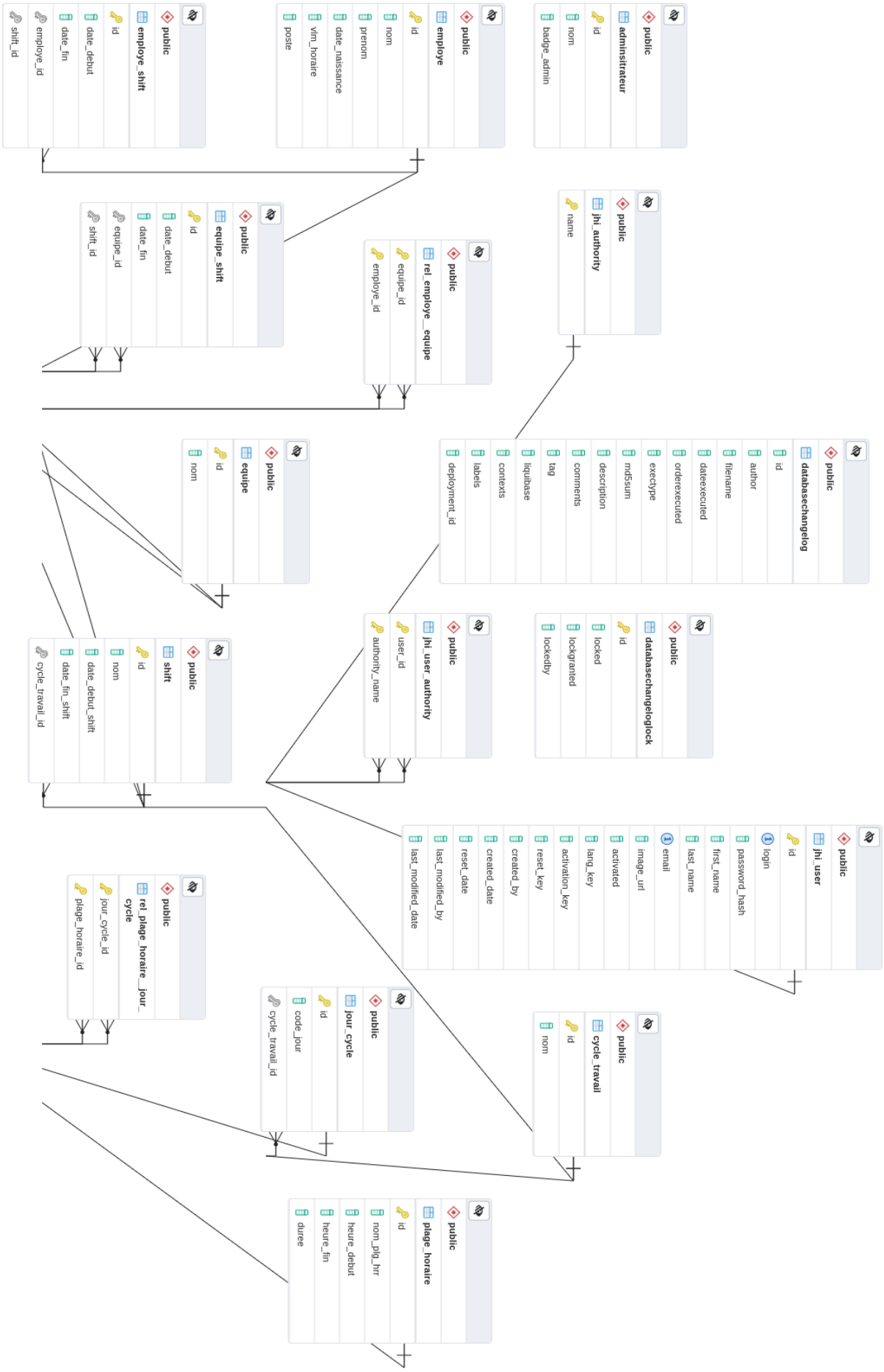

FIGURE 2.7 – Modèle relationnel

2.8 Conclusion

Pour conclure ce chapitre, récapitulons ce qui a été fait. Tout d'abord nous avons spécifié les besoins fonctionnels et non fonctionnels, par la suite nous avons identifié les acteurs et leurs cas d'utilisation, puis nous avons décrit de manière détaillée ces cas d'utilisation que nous avons illustré grâce à différents diagrammes (de cas d'utilisation toujours, ainsi que de séquence), pour après finir en nous aidant d'un dictionnaire de données préalablement formé par constituer un diagramme de classe, qui au final sera traduit au format relationnel, représenté par le diagramme du même nom.

Chapitre 3

Implémentation

3.1 Introduction

Dans ce dernier chapitre, nous allons mettre à l'évidence les moyens techniques que nous avons utilisé afin d'implémenter ce module. Nous exposerons les plateformes de collaboration, ainsi que les environnements de développements accompagnés des différentes technologies employés dans le cadre du projet, puis nous présenterons l'architecture du système dans les deux phases développement et production. Et pour finir nous ferons une présentation des interfaces pertinentes de l'application.

3.2 Plateformes de collaboration

3.2.1 GitLab

GitLab est une plateforme de développement qui booste la productivité des équipes, en réunissant l'ensemble des outils nécessaires pour la planification de projets, la gestion du code source, l'approche en CI/CD (intégration continue et distribution continue) afin de créer des logiciels et de les tester dans les meilleurs délais [9].

3.2.2 Asana

Asana est une plateforme d'organisation de projet, qui regroupe toutes les taches de l'équipe, tout en facilitant la communication entre les membres et leurs planifications dans un seul espace, ce qui permet de faciliter la collaboration, jusqu'à l'achèvement du projet [10].

3.2.3 Figma

Figma est une plateforme de collaboration conçu pour les designers et développeurs, afin de travailler en équipe sur la création/édition de designs et prototypes en temps réel, et tout ça dans le Cloud, elle permet aussi d'intégrer de nouvelles recrues dans le processus de design [11].

3.3 Environnements de Développement Intégrés (IDE)

3.3.1 IntelliJ IDEA

Un IDE intelligent et sensible au contexte qui permet de travailler avec diverses applications en Java et d'autres langages JVM tels que Kotlin, Scala et Groovy. De plus, IntelliJ IDEA Ultimate vous aide à développer des applications Web complètes avec ses puissants outils intégrés, la prise en charge de JavaScript et des technologies associées, et la prise en charge avancée des frameworks populaires tels que Spring, Spring Boot, Jakarta EE, etc [12].

3.3.2 Visual Studio Code

VSCode est un puissant éditeur de code source léger conçu par Microsoft, qui inclus un support pour JavaScript, Type Script, et NodeJS. Il est le plus populaires des éditeurs de code source à cause de son riche écosystème en extensions pour le contrôle de version de Git et pour plusieurs autres langages tels que Python, C, C++ et autres [13].

3.4 Technologies employées

3.4.1 En Backend

3.4.1.1 Java

Java est un langage de programmation et une plate-forme informatique créées par Sun Microsystems en 1995, parmi les plus populaires des langages de programmation utilisés par plus de 10 million de développeurs dans presque chaque pays au monde, « il existe plus de 51 milliards de JVM lancés par les développeurs autour du monde » (Oracle Java) la version actuelle est 18 mais la version stable est la 17 [14].

3.4.1.2 Spring

Spring est un framework de développement d'applications Java qui apporte Spring Security, SpringMVC, Spring Data et de nombreuses autres fonctionnalités. Ces frameworks sont conçus pour faciliter la tâche du développeur. Malheureusement, leur mise en œuvre est assez compliquée par des fichiers de configuration XML toujours plus complexes et une gestion des dépendances fastidieuse. C'est en réponse à cette préoccupation que le projet Spring Boot a vu le jour.
Spring Boot est un sous-projet de Spring qui vise à rendre Spring plus facile à utiliser en éliminant plusieurs étapes de configuration. L'objectif de Spring Boot est de permettre aux développeurs de se concentrer sur les tâches techniques, et non sur les tâches de configuration, de déploiement, etc.

Cela permet un gain de temps et de productivité considérable (avec Spring Boot, démarrer un projet n-tiers est très facile) [15].

3.4.1.3 Jhipster

Jhipster est un générateur Yeoman qui permet de créer des applications spring boot et Angular/React , soit des architectures monolithiques ou bien des micro service avec toutes les fonctionnalités prédéfinies [16].

3.4.1.4 PostgreSQL

PostgreSQL est un systèm de base de données objet-relationnel open source il étend du langage SQL (Structured Query language), il combine plusieurs fonctionnalités qui permettent de stocker et mettre en échelle de complexes charges de données. Son origine remonte à 1986 où il était un projet de l'Université de Californie [17].

3.4.2 En Frontend

3.4.2.1 TypeScript

Type Script est un langage qui représente une surcouche typée de JavaScript, conçu pour améliorer et sécuriser le code JavaScript. Il ajoute des règles sur comment différents types de valeurs peuvent être utilisés [18].

3.4.2.2 Angular 13

Angular est un framework de développement basé sur TypeScript pour concevoir des applications SPA (Single Page Application) sophistiquées, basées sur la hiérarchie des composants et intégré avec les meilleures librairies pour les meilleures fonctionnalités tel que le routing, la gestion des formulaires, la communication client-serveur. Et d'autres outils qui aide au développement, tests et mise à jour du code [19].

3.4.2.3 HTML

HTML signifie "Hypertext Markup Language" traduit par "Langage de balises pour l'hypertexte". Il est utilisé pour créer et représenter le contenu des pages Web et leur structure. D'autres technologies sont utilisées avec HTML pour décrire la présentation d'une page (CSS) et/ou sa fonctionnalité interactive (JavaScript). La version actuelle est la 5 [20].

3.4.2.4 CSS

Le CSS (Cascading Style Sheet), Les feuilles de style en cascade est l'un des principaux langages du Web ouvert, est un standard du W3C, permettant d'ajouter du style (polices, couleurs, espacement, etc) à des documents Web. La version actuelle est la 3 [21].

3.4.2.5 SCSS (SASS)

SASS (Syntactically Awesome Style Sheet) est en quelques sortes un sur-ensemble du CSS, en plus de ce qui est déjà présent au sein du CSS, le SCSS regorge de bien plus de fonctionnalités avancées telles que l'emploi de variables, de fonctions, de mixins et autres règles imbriquées... etc [22].

3.5 Architecture du système

Afin de réaliser ce système, on a opté pour l'architecture d'une application monolithique dans la phase de développement, une fois qu'elle répond aux besoins, on l'ajoutera sous forme de microservice à RH Partner, et ceci serait expliqué en détails ci-dessous.

— Nous avons opté pour le framework Angular afin de concevoir l'application Frontend, communiquant avec notre application Backend réalisée avec Springboot via des requêtes HTTP, celles-ci comprennent des données qui seront stockées sur notre base de données Postgresql.

3.5.1 Architecture utilisée en développement

Nous avons utilisé en développement une architecture monolithique, pour ce qui est de l'API en Backend nous l'avons déployé sur Heroku, afin de connecter le Backend au Frontend, tout ceci en la mettant constamment à jour pour exploiter et tester les nouvelles fonctionnalités ajoutées.

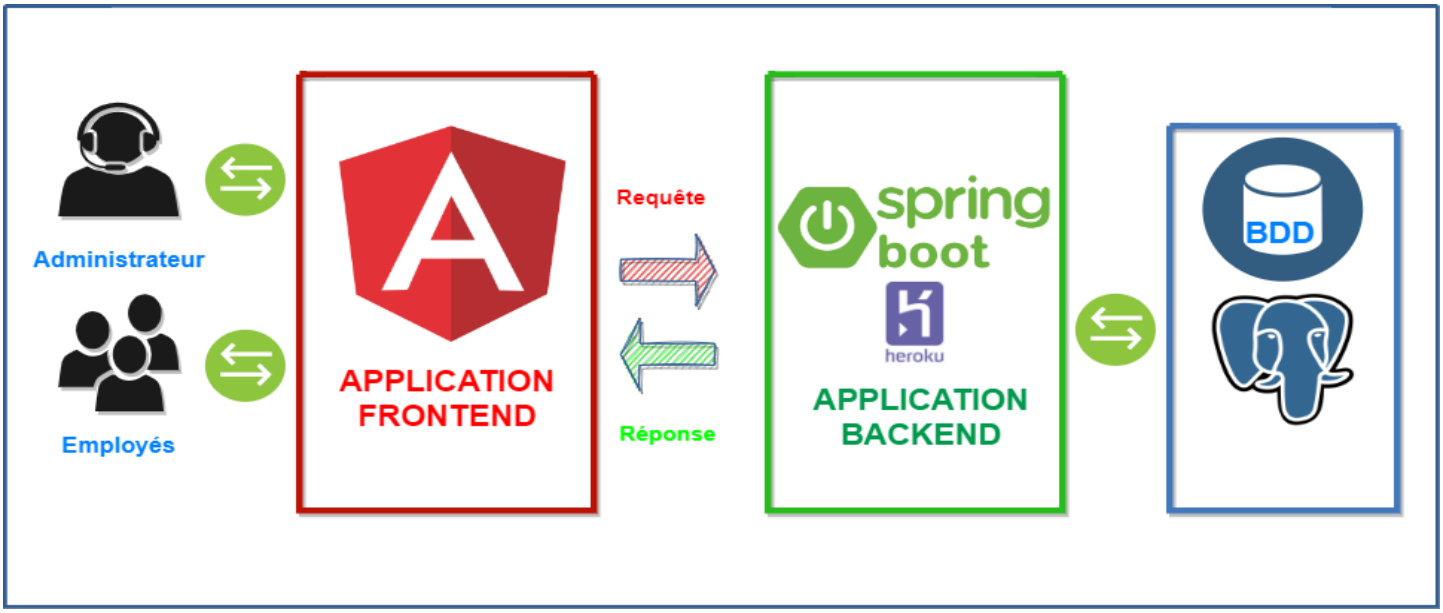

FIGURE 3.1 – Architecture du développement

3.5.2 Architecture Microservices

L'architecture est déja mise en place pour l'application RH Partner, on aura juste à convertir notre application en microservice qui sera par la suite ajoutée au registre et sans oublier bien sûr d'ajouter la partie frontend au niveau du gateway.

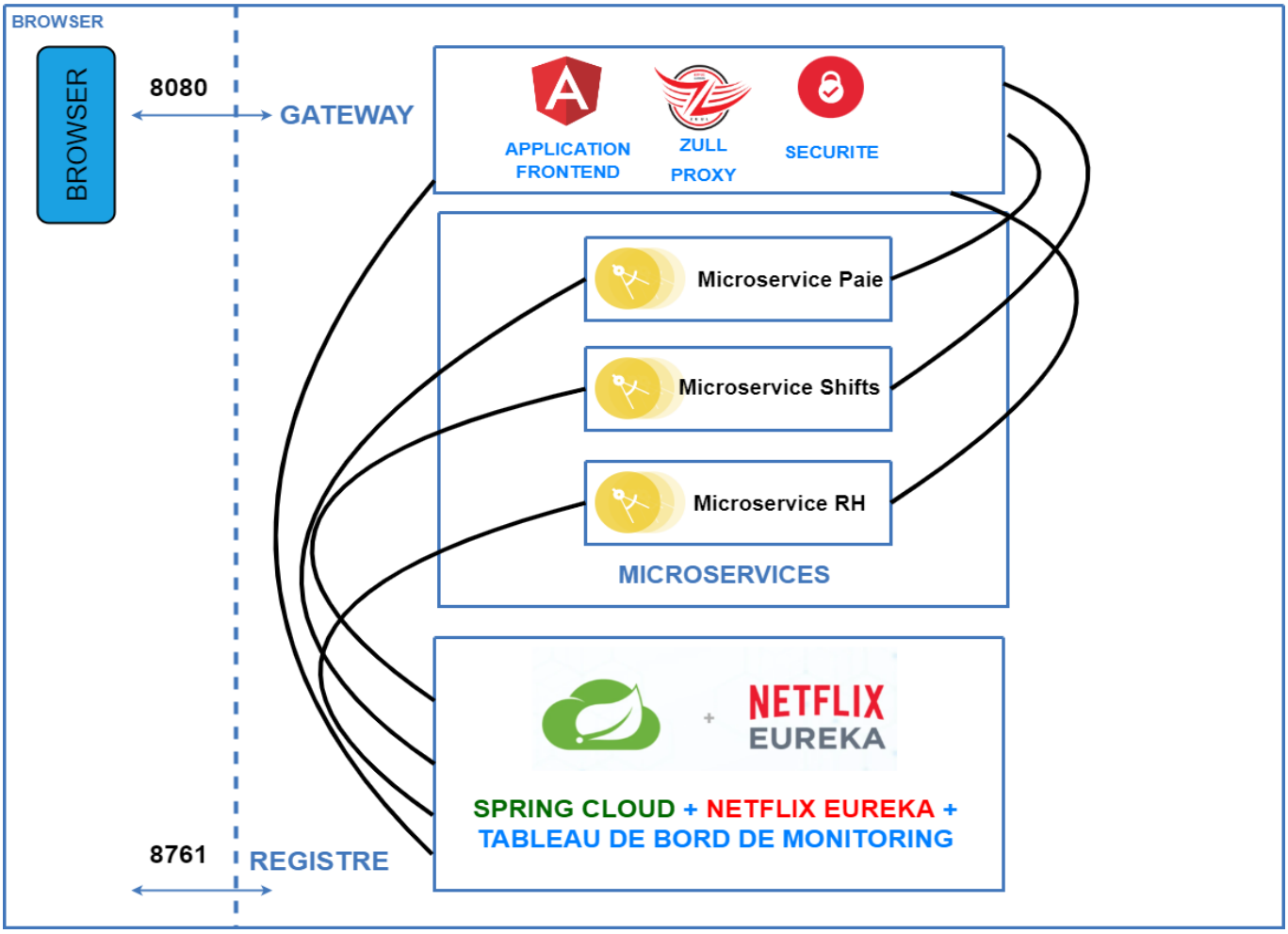

FIGURE 3.2 – Architecture microservices

3.6 Présentation des interfaces graphiques

Notre application comporte plusieurs interfaces graphiques permettant la bonne gestion des employés et de leur shifts, certaines de ces interfaces seront utlisées par l'employé alors que d'autre seront reservées pour l'administrateur du système. Nous allons mettre à l'évidence les interfaces les plus pertinentes.

3.6.1 Espace de gestion d'employés

Dans cet espace nous pourrons ajouter, modifier ou supprimer un employé

3.6.1.1 Lister les employés

Dans cette interface nous pourrons voir la liste des employés et les équipes auxquelles ils sont affectés.

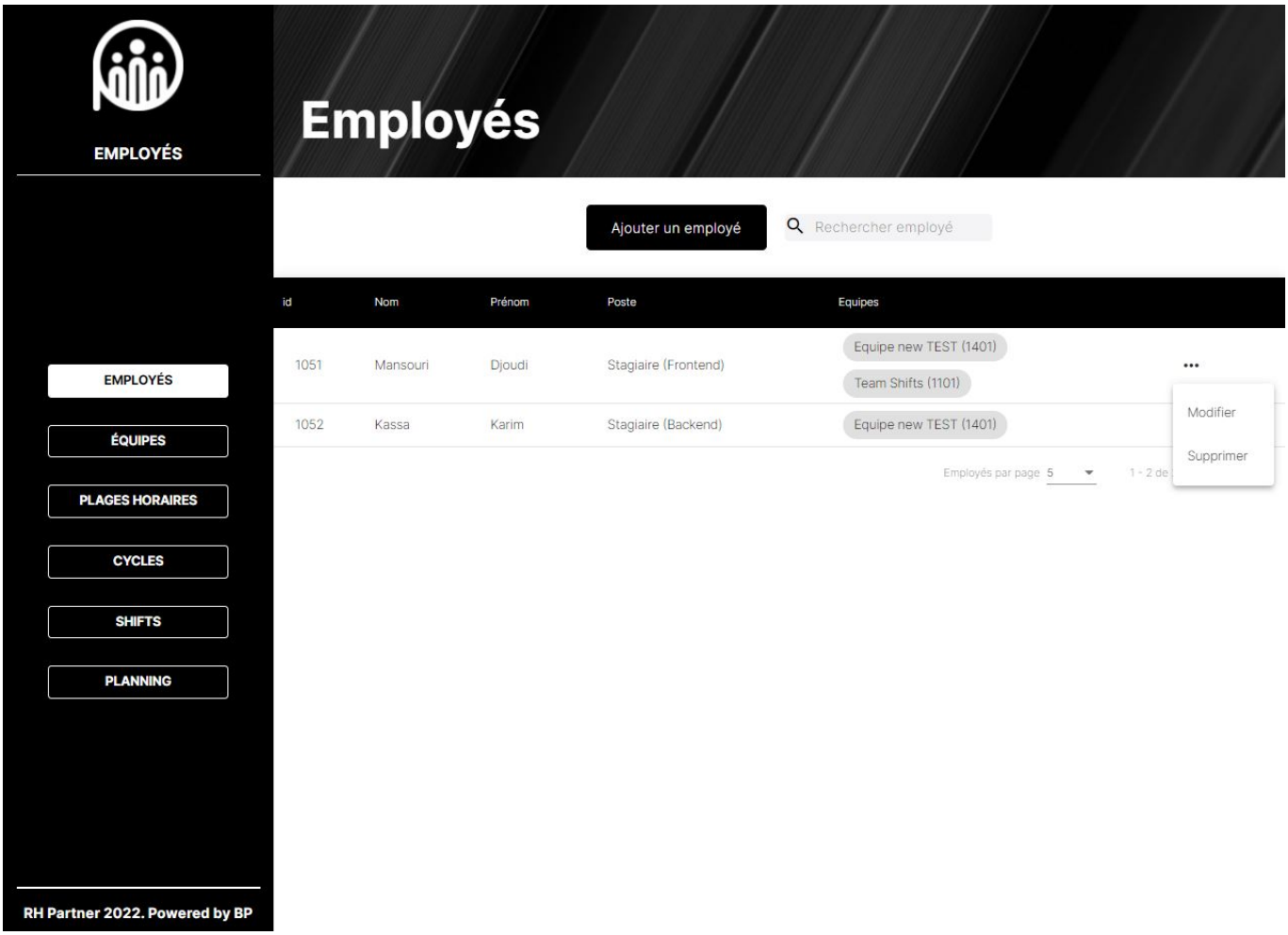

FIGURE 3.3 – Interface espace employés

3.6.1.2 Ajouter des employés

Dans cette interface nous aurons la possibilité d'ajouter des employés et leur assigner des équipes.

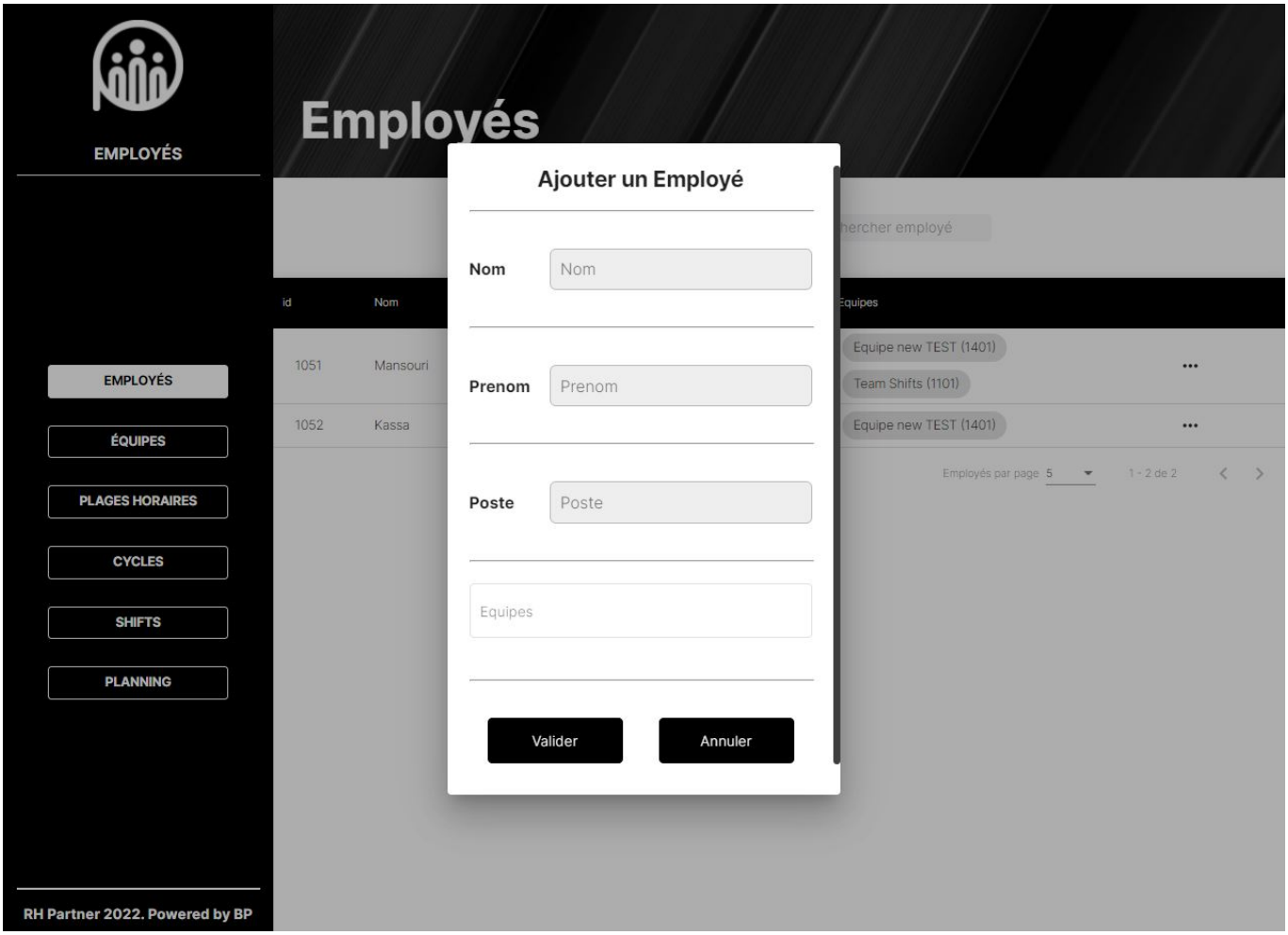

FIGURE 3.4 – Interface ajouter un employé

3.6.1.3 Modifier des employés

Dans cette interface nous pourrons modifier un employé et ses équipes respectives.

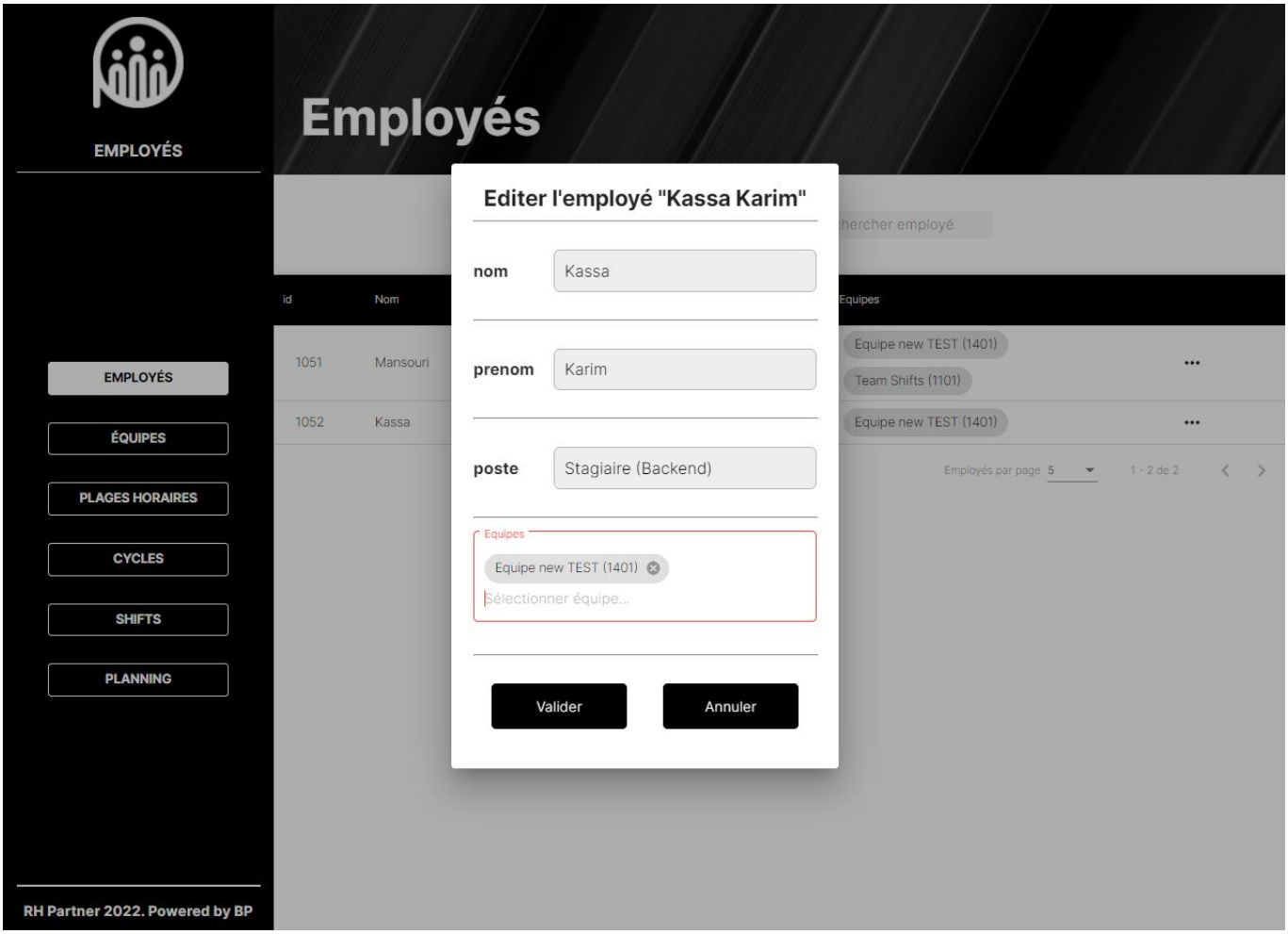

FIGURE 3.5 – Interface modifier un employé

3.6.1.4 Supprimer des employés

Dans cette interface nous pourrons supprimer un employé.

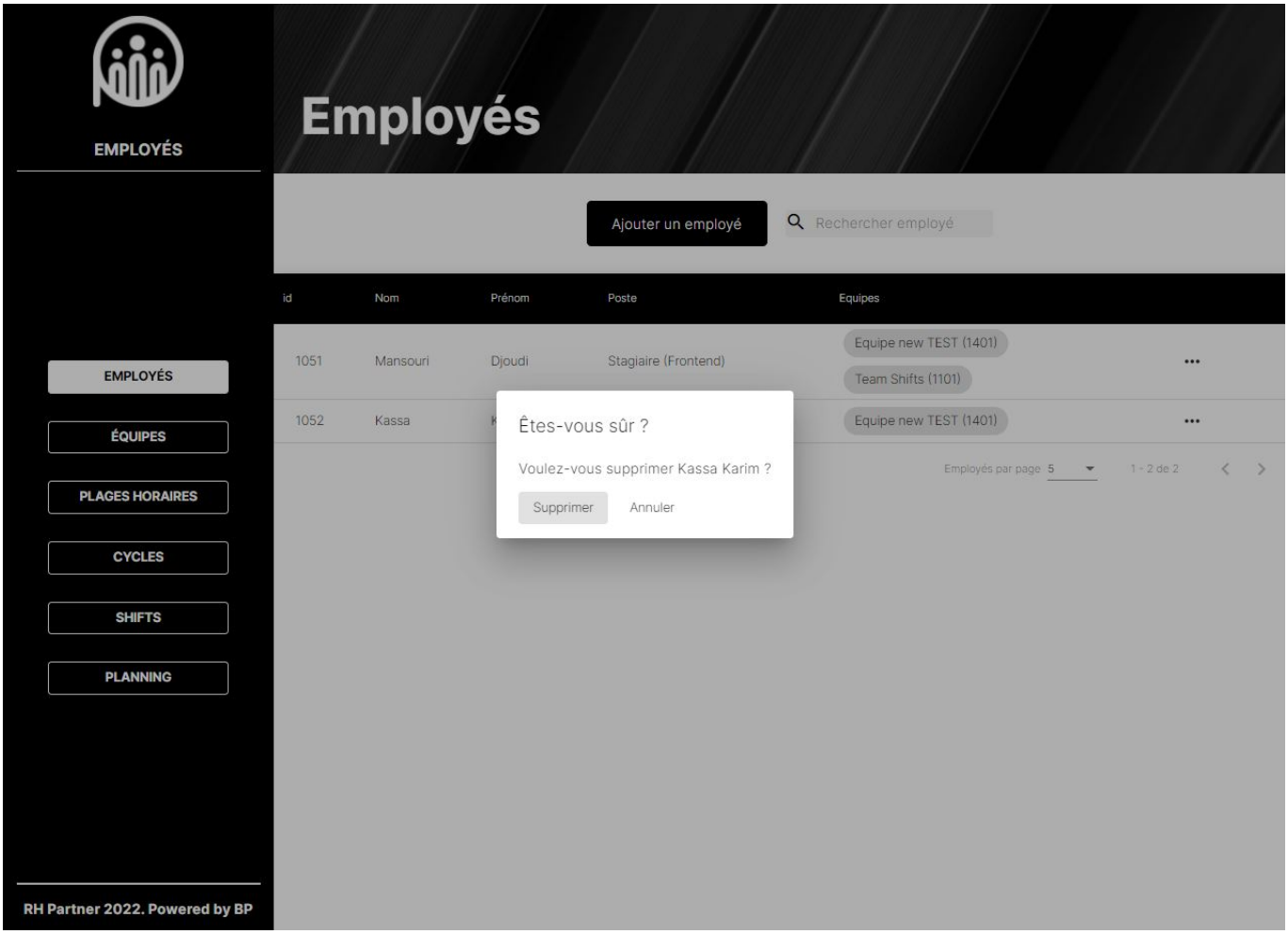

FIGURE 3.6 – Interface supprimer un employé

3.6.2 Espace de gestion de plages horaires

Dans cet espace nous aurons la possibilité d'ajouter, modifier ou supprimer une plage horaire.

3.6.2.1 Lister les plages horaires

Dans cette interface nous pourrons voir la liste des plages horaires

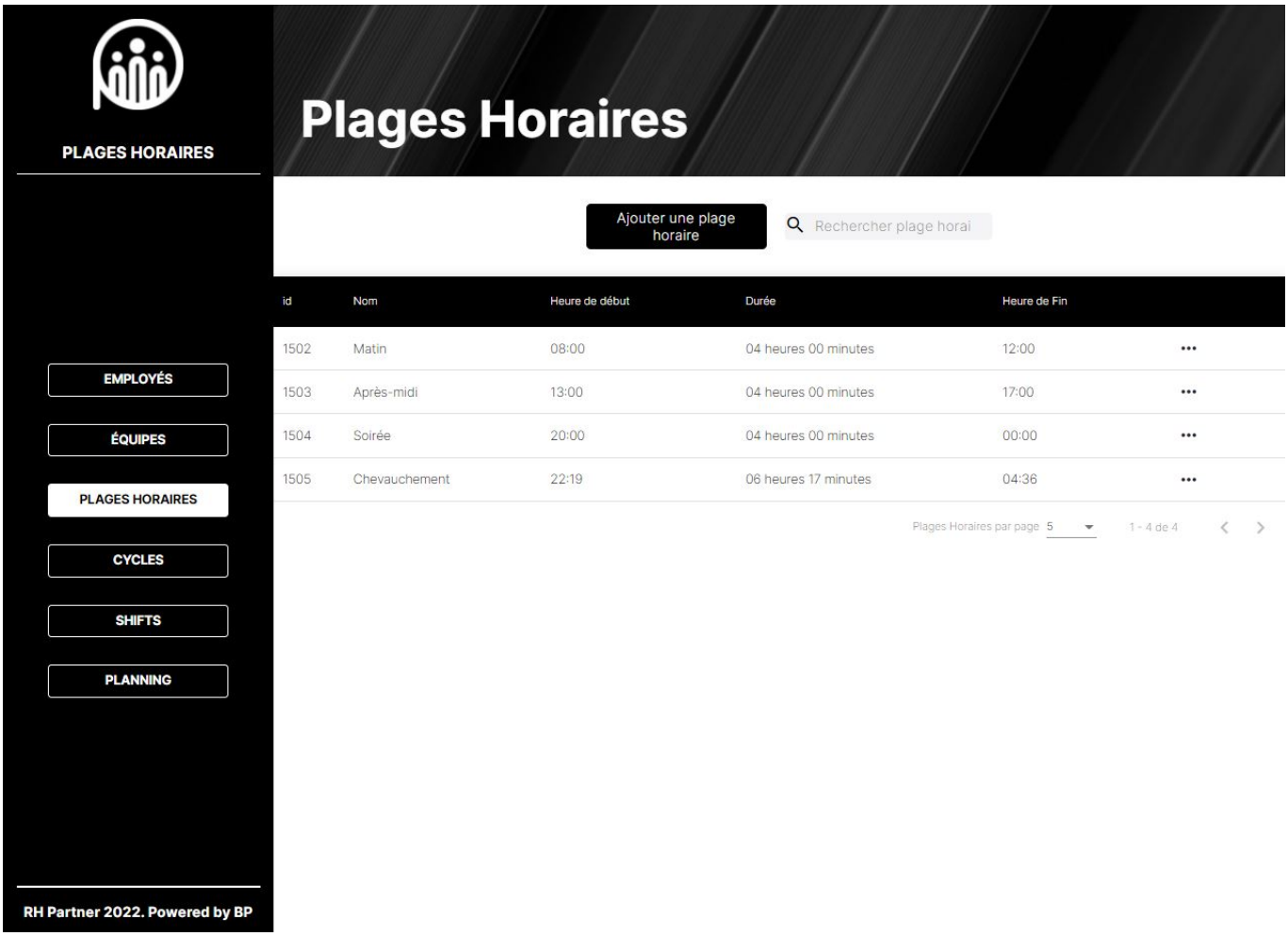

FIGURE 3.7 – Interface espace plages horaires

3.6.2.2 Ajouter des plages horaires

Dans cette interface nous pourrons ajouter des plages horaires.

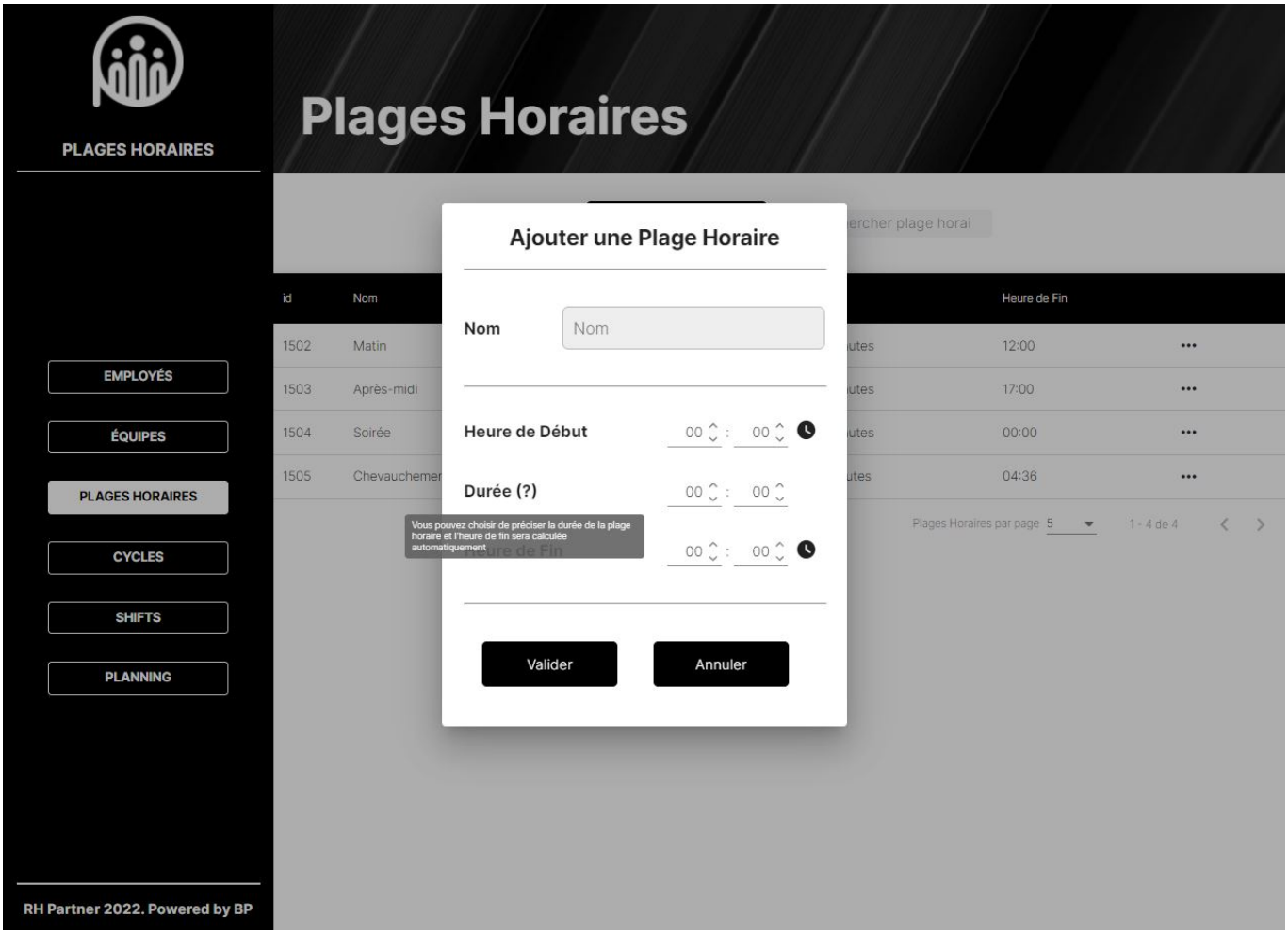

FIGURE 3.8 – Interface ajouter une plage horaire

3.6.2.3 Modifier une plage horaire

Dans cette interface nous pourrons modifier une plage horaire.

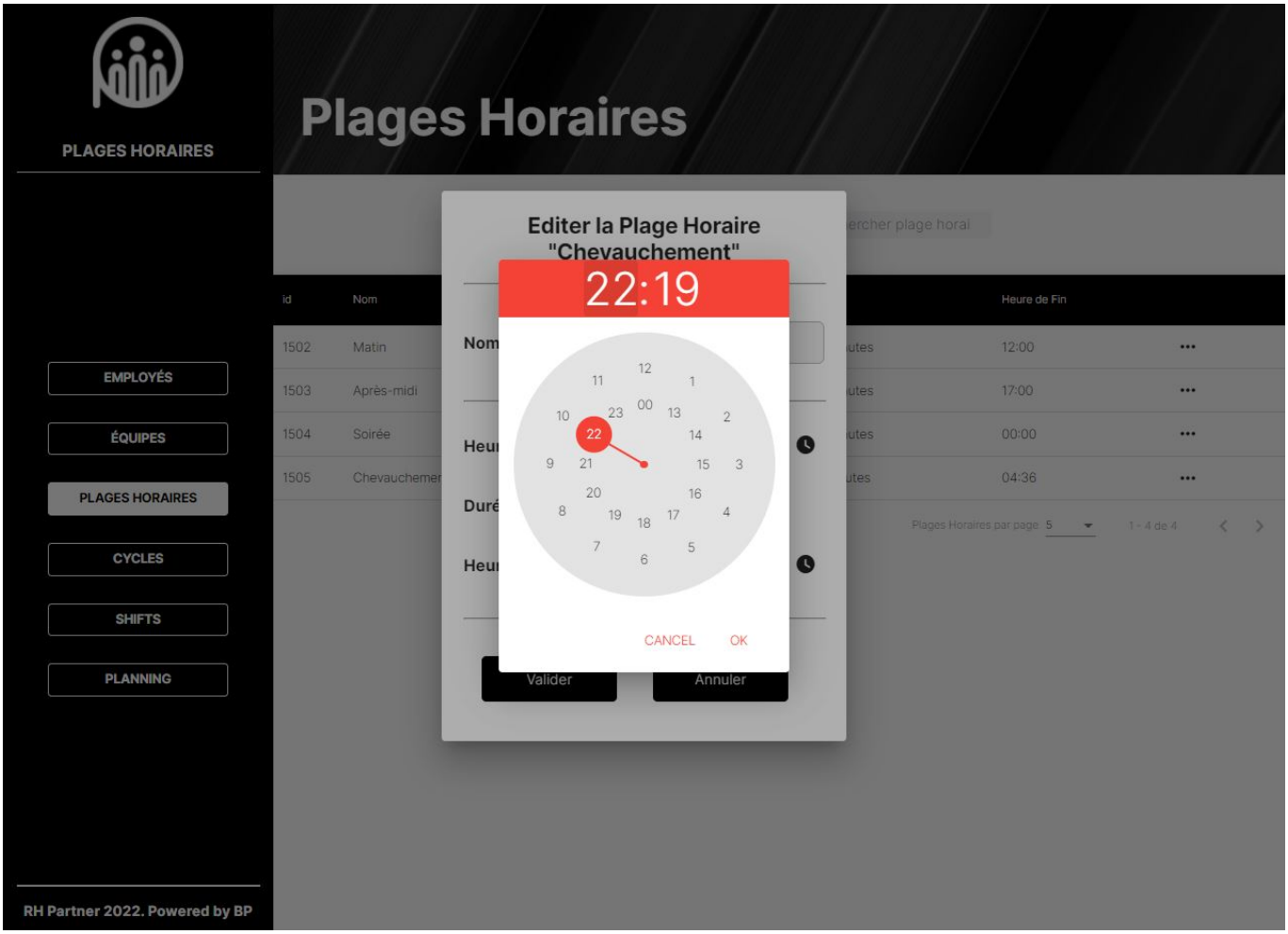

FIGURE 3.9 – Interface modifier une plage horaire

3.6.3 Espace des cycles

Dans cet espace nous aurons la possibilité d'ajouter, modifier ou supprimer un cycle.

3.6.3.1 Ajouter un cycle

Dans cette interface nous pourrons ajouter un cycle, auquel on introduira des jours tout en leurs affectant des plages horaires. Si on n'assigne pas au jour au moins une plage horaire, il sera considéré comme un jour de repos.

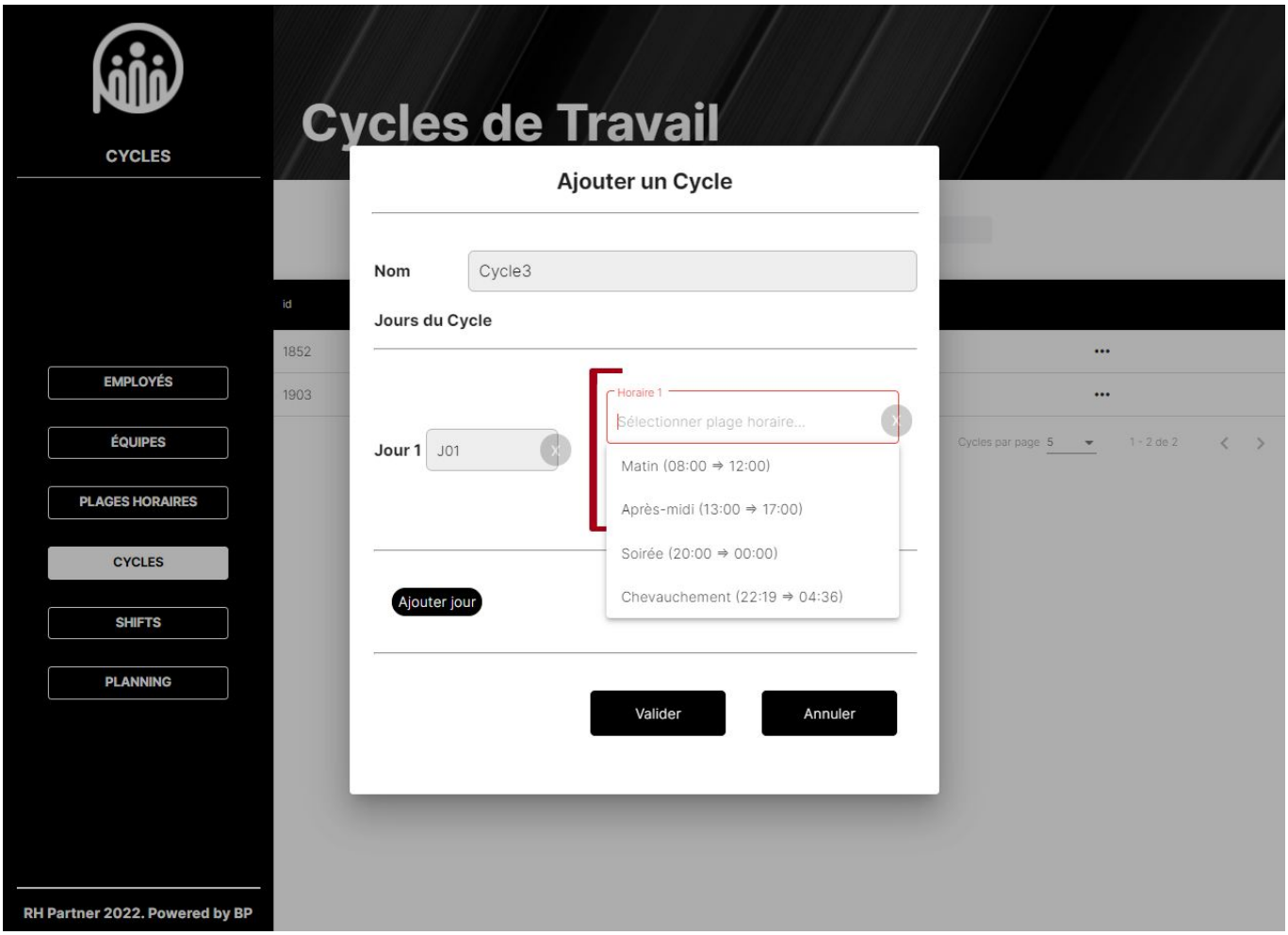

FIGURE 3.10 – Interface ajout d'un cycle

La capture suivante illustre le message d'aide affiché lors du survolement du bouton d'ajout d'horaires qui explique le fonctionnement des jours de repos.

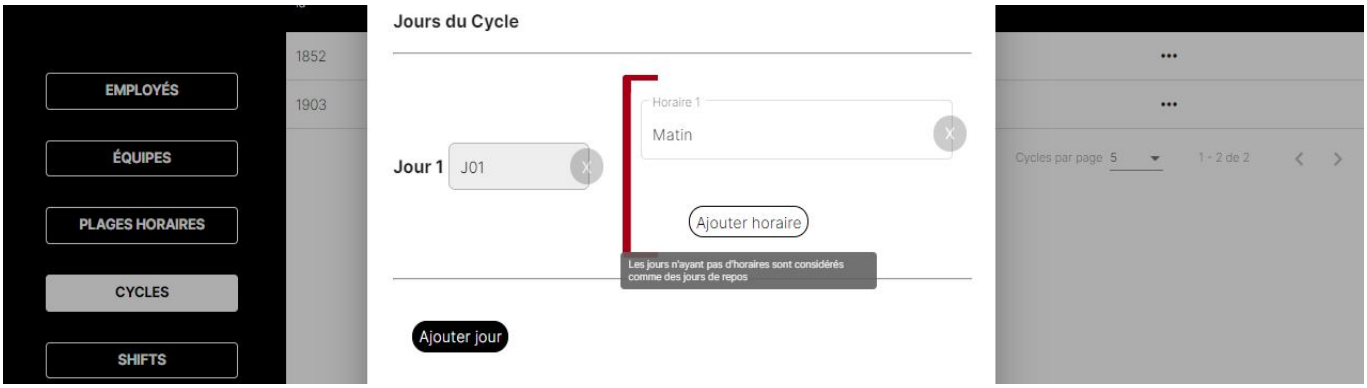

FIGURE 3.11 – Interface spécification d'un jour de repos

3.6.4 Espace des shifts

Dans cet espace nous aurons la possibilité d'ajouter, modifier ou supprimer un shift.

3.6.4.1 Ajouter un shift

Pour ajouter un shift, on spécifie le cycle à adopter et une periode d'activité.

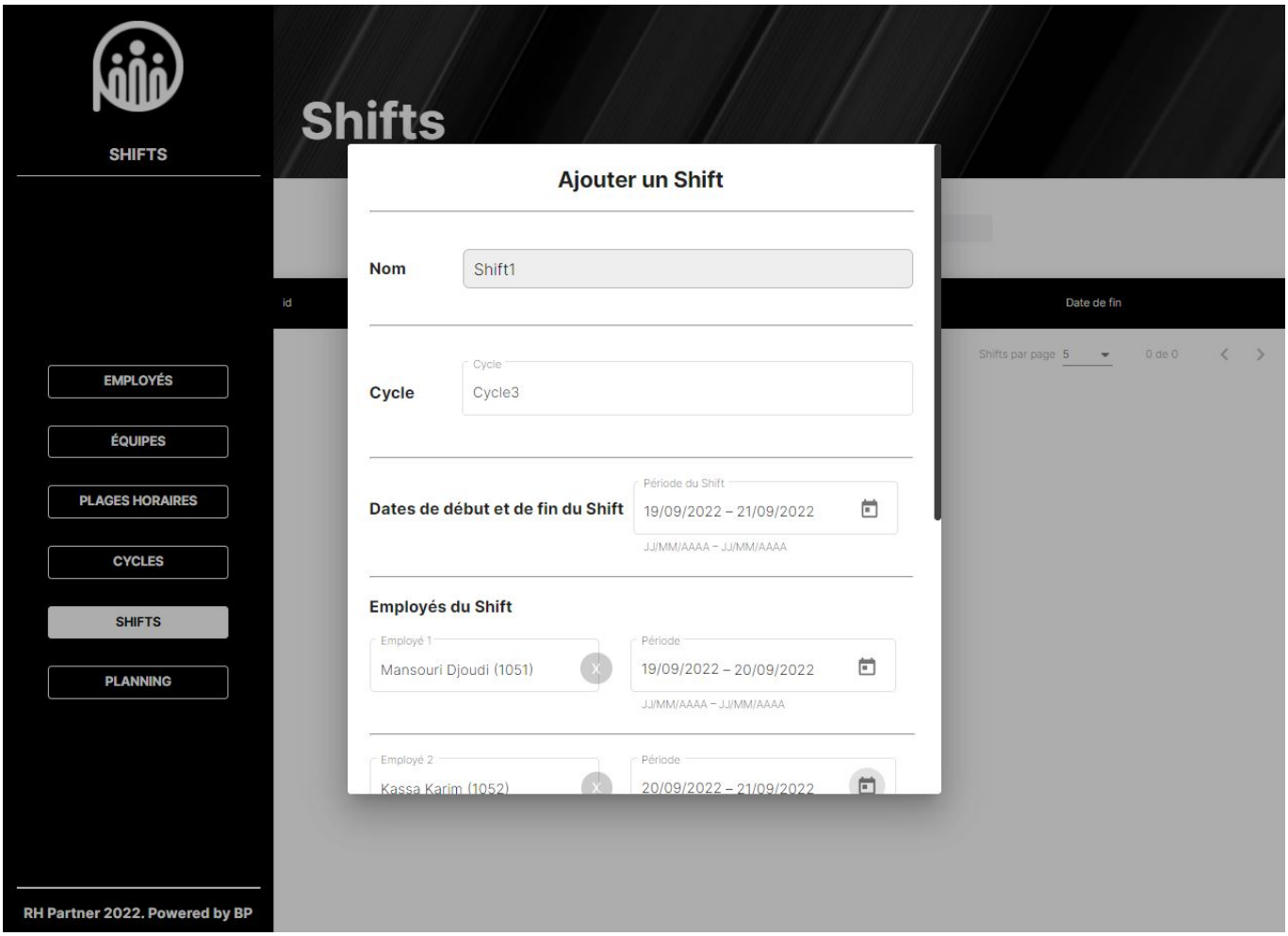

FIGURE 3.12 – Interface ajout d'un shift première partie

Dans cette deuxième partie, on assigne les employés avec une certaine période. Et quelques préférences de couleurs à afficher au planning.

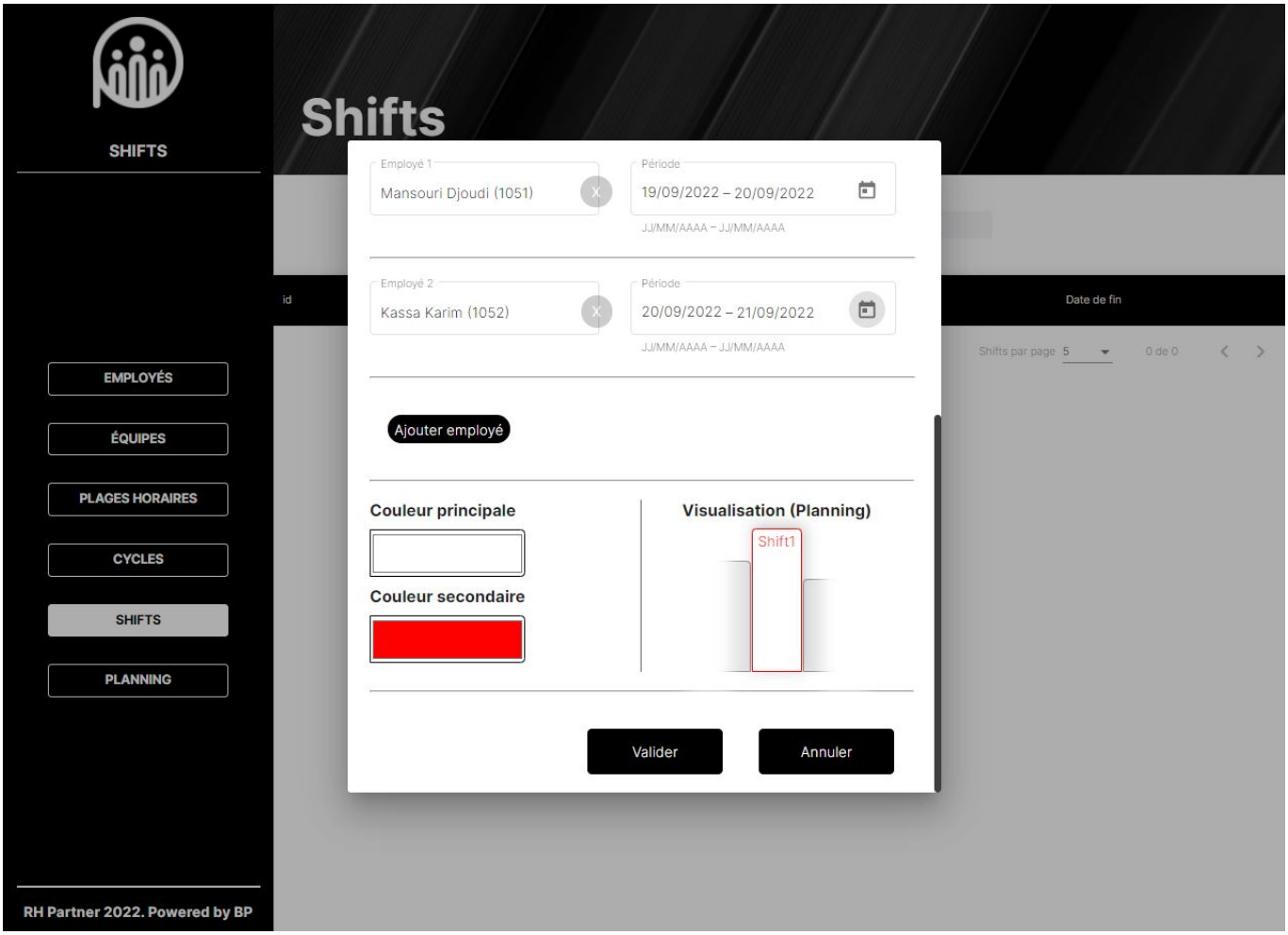

FIGURE 3.13 – Interface ajout d'un shift deuxième partie

3.6.5 Espace du planning

Pour finir, dans l'espace planning, nous avons pensé qu'il serait généré à la volée en récupérant les instances et en les affichant durant la période choisi.

3.6.5.1 Planning vue hebdomadaire

En survolant le shifts, on affiche les détails du shifts (le cycle (jours et horaires), les employés affectés et la periode du shift). Mais on peux aussi clairement voir que les shifts montrent la période d'activité des employés en leurs sein, comme le montrerons les 2 exemples ci-dessous.

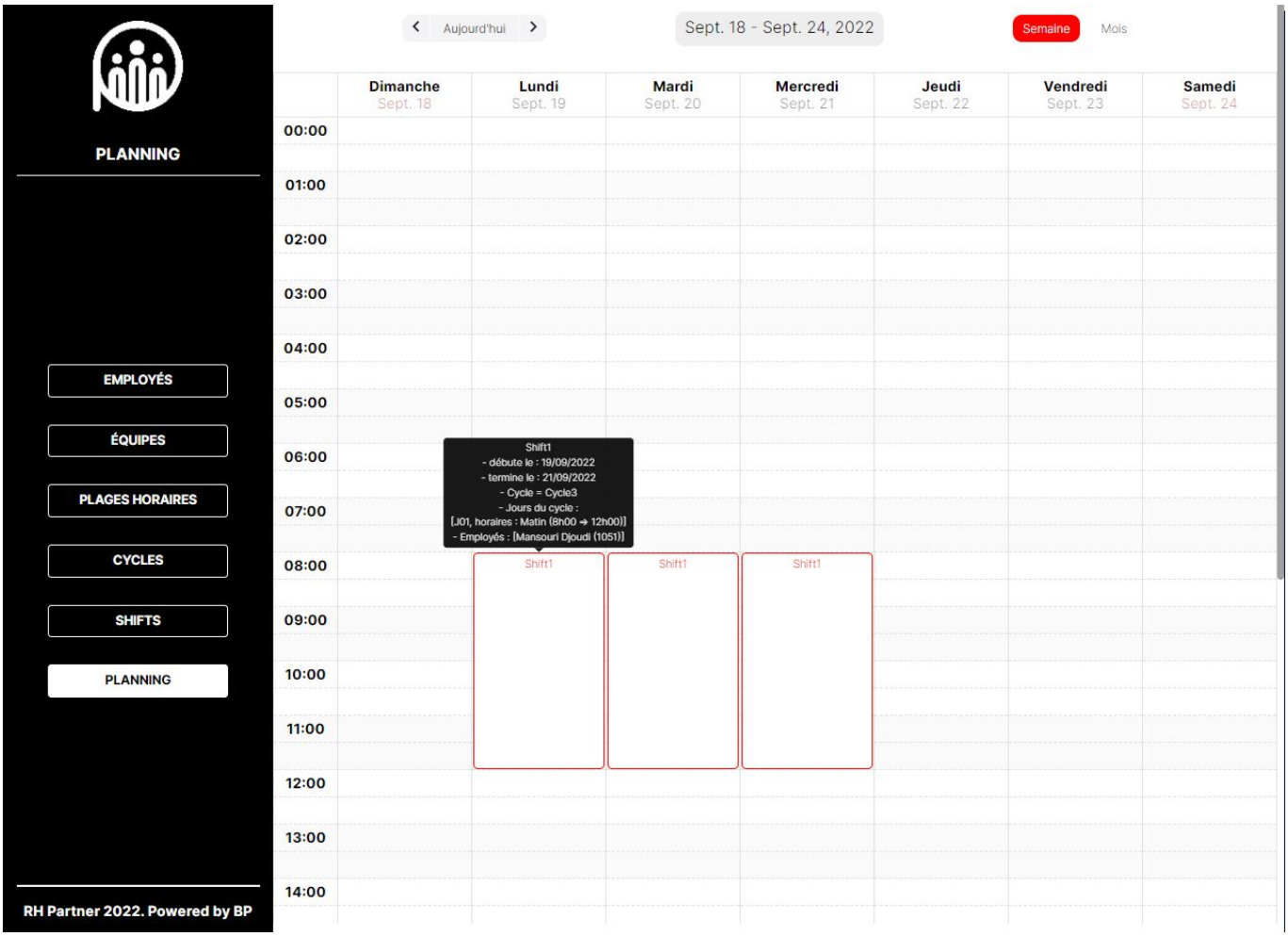

FIGURE 3.14 – Interface planning vue hebdomadaire

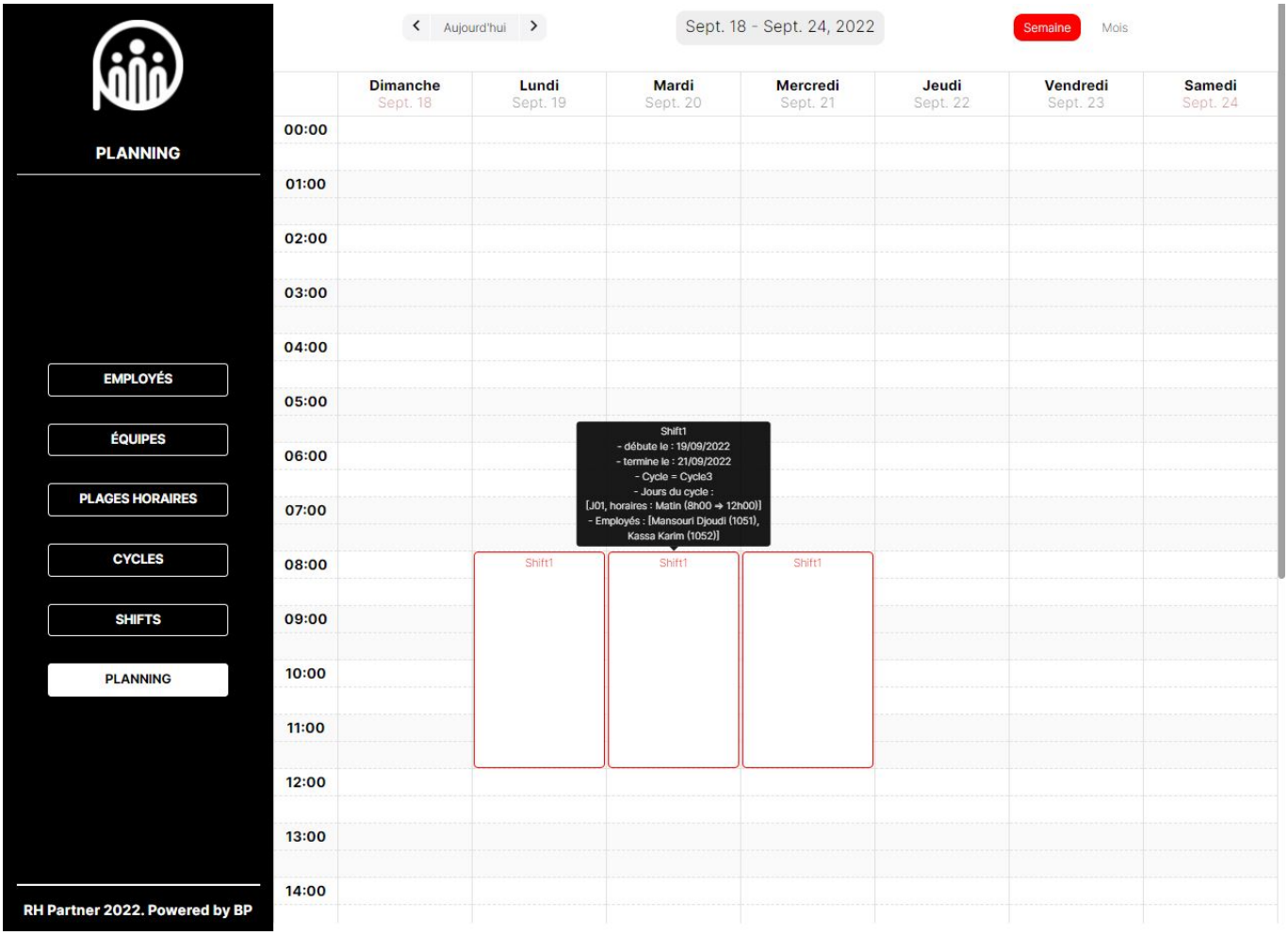

FIGURE 3.15 – Interface planning vue hebdomadaire

3.6.5.2 Planning vue mensuelle

En choisissant la vue mensuelle, on peut voir les détails des shifts en cliquant sur le jour voulu.

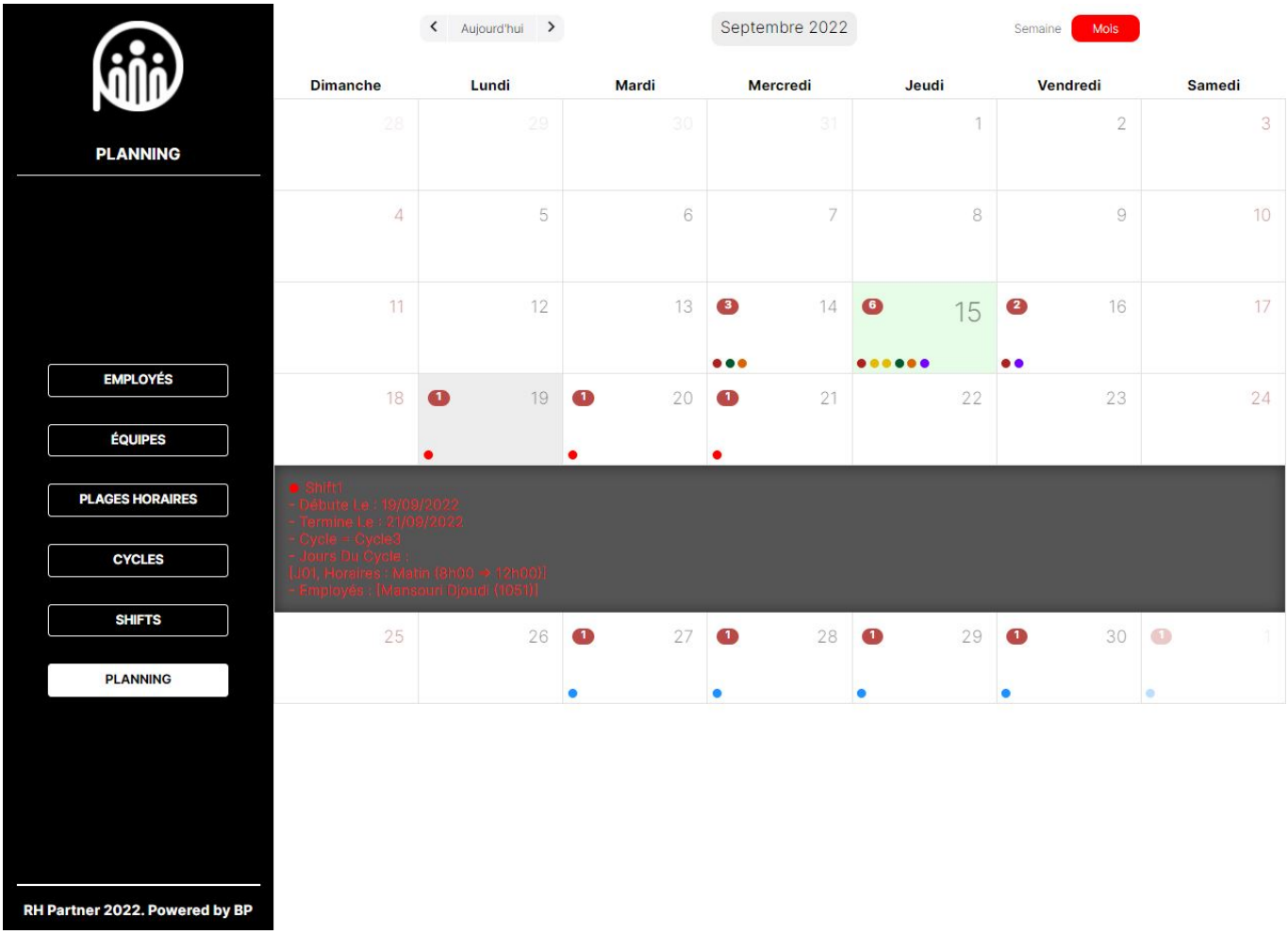

FIGURE 3.16 – Interface planning vue mensuelle

3.7 Conclusion

Dans ce dernier chapitre, nous avons présenté les différentes technologies, plateformes, et librairies que nous avons utilisé pour la réalisation de ce module. On a ensuite présenté l'architecture de notre système mis en place. Et en dernier lieu, on a exposé les différentes interfaces de notre application et leur fonctionnement.

Conclusion et perspectives

Ce travail a été réalisé dans le cadre de notre projet de fin de cycle Master en Génie logiciel. Il consiste en l'élaboration d'un module de gestion de temps avancé. Ce système est destiné aux entreprises voulant avoir un système de gestion d'horaires flexible afin de déminuer le taux d'erreurs liés à la gestion des heures de travail des employés.

En suivant un processus de développement bien détaillé, nous avons pu spécifier et analyser les besoins des utilisateurs et identifier les acteurs et les cas d'utilisation. Une fois la phase de conception terminée, nous avons pu générer tous les diagrammes et la documentation dont nous avons besoin pour commencer à créer notre application. Ainsi, nous avons pu utiliser différents outils et plateformes (GitLab, Postman, Springboot, Angular, Postgresql, etc.) pour implémenter notre solution, nous avons adopté une identité graphique unique tout en réutilisant le logo de RH-Partner.

L'application permet non seulement de gérer les employés mais aussi de gérer les éléments clés des shifts (cycles, jours et plages horaires) afin de générer un planning à la volée, qui change en fonctions des modifications apportées à ces derniers, la gestion se fera par l'administrateur qui pourra gérer les employés ainsi que leurs shifts, et d'avoir une base de données contenant toutes les informations importantes au sujet de leurs plannings.

Malgré le manque de temps, nous avons quand-même pu réaliser plusieurs aspects de notre système pouvant et devant être améliorés.

En guise de perspectives, nous voulons enrichir notre application avec d'autres fonctionnalités telles que la génération d'un planning automatique en spécifiant les contraintes de la durée de travail hebdomadaire et mensuelle.

Bibliographie

[1] Qu'est ce qu'une entreprise : définition. https://agicap.com/fr/article/definition-entreprise, (Consulté le 18 Juin 2022).

[2] Gestion des ressources humaines. https://fr.wikipedia.org/wiki/Gestion_des_ressources_humaines, (Consulté le 20 Juin 2022).

[3] Un sirh simplifié tout en un. https://rh-partner.com/solution/, (Consulté le 20 Juin 2022).

[4] Remember : Agile values are not rules. https://www.productplan.com/glossary/agile-values/, (Consulté le 21 Juin 2022).

[5] La méthode scrum par eric peyrot. https://www.linkedin.com/pulse/la-méthode-scrum-eric-peyrot, (Consulté le 20 Juin 2022).

[6] Pascal Roques. *Modéliser un application web*. Number 4ème Edition, EYROLLES.

[7] J.Joseph Gabay and D.David Gabay. *UML 2 Analyse et conception - Mise en oeuvre guidée avec études de cas : Mise en oeuvre guidée avec études de cas*. DUNOD, dunod edition, 4 2008.

[8] Bouchlaghem S, Cherifi F, Khanouche F, Maouche L, and Zebboudj S. *Application Web JAVA EE pour la gestion d'un laboratoire de recherche scientifique,*. Univérsité A/Mira Béjaia, licence en informatique générale edition, 2014.

[9] GitLab est une plateforme DevOps complète proposée sous la forme d'une application unique. https://about.gitlab.com/fr-fr/, (Consulté le 15 Juin 2022).

[10] Gérez en ligne le travail, les projets et les tâches de votre équipe •. https://asana.com/fr, (Consulté le 15 Juin 2022).

[11] À propos de figma, l'outil de design à interface collaborative. https://www.figma.com/fr/about/.

[12] Intellij idea : l'ide java performant et ergonomique de. https://www.jetbrains.com/fr-fr/idea/.

[13] Documentation for visual studio code. https://code.visualstudio.com/docs.

[14] Java platform evolution - dev.java. https://dev.java/evolution/#anchor_1.

[15] B. Bertrand Nguimgo. *Spring Boot par la pratique : Développer les services Rest avec Spring-Boot et Spring-RestTemplate (French Edition)*. Les Éditions du Net, 6 2018.

[16] D.K. Deepu Sasidharan and S.K Sendil Kumar N. *Full Stack Development with JHipster : Build modern web applications and microservices with Spring and Angular*. Packt Publishing, 3 2018.

[17] Postgresql : About. `https://www.postgresql.org/about/`.

[18] Le point de départ pour apprendre typescript. `https://www.typescriptlang.org/fr/docs/`.

[19] Angular. `https://angular.io/docs`.

[20] Html (hypertext markup language) | mdn. `https://developer.mozilla.org/fr/docs/Web/HTML`.

[21] Feuilles de style en cascade (css). `https://www.w3.org/Style/CSS/Overview.fr.html`.

[22] Quelle est la différence entre css et scss ? `https://fr.acervolima.com/quelle-est-la-difference-entre-css-et-scss`.

RÉSUMÉ

Dans ce mémoire, qui a pour but final l'obtention du diplôme de Master professionnel en Génie Logiciel, nous avons abordé le thème de la conception et réalisation d'un module de gestion des shifts à intégrer par la suite au sein du système mère qui n'est autre que le SIRH de Business Platforms : Rh-Partner. La solution que nous proposons consiste en une application Frontend à part entière (côté client) qui va venir récupérer et stocker des données au sein d'un microservice Backend (application côté serveur). Pour ce qui est de la conception, nous nous sommes aidés de la méthodologie de développement AGILE, plus particulièrement SCRUM, et d'UML pour modéliser nos diagrammes. En ce qui concerne la réalisation, elle a été faite sous Spring Boot et JAVA en Backend et avec Angular (Typescript) au niveau Frontend, nous avons aussi fait usage de plusieurs autres librairies et services tiers pour faciliter et compléter le développement.

Mots clés : Shifts ; Cycles ; Planning ; Ressources Humaines ; Gestion ; Employés ; Web ; SCRUM ; Angular ; Spring Boot

ABSTRACT

In this report, which has as its final goal the obtaining of a professional Master degree in Software Engineering, we have approached the theme of the design and implementation of a shift management module that will be integrated later on within the main system, which is none other than the Business Platforms HRIS : Rh-Partner. The solution we propose consists of a fully-fledged Frontend application (client side) which will retrieve and store data within a Backend microservice (server side application). In terms of design, we used the AGILE development methodology, to be precise SCRUM, and UML to model our diagrams. Regarding the implementation, it was done under Spring Boot and JAVA for the Backend and with Angular (Typescript) at the Frontend level, we also made use of several other libraries and third-party services to facilitate and complete the development.

Key words : Shifts ; Cycles ; Schedules ; Human Ressources ; Management ; Employees ; Web ; SCRUM ; Angular ; Spring Boot