Rapport de Stage

Bikay, Phnom Penh, Cambodge

Tuteurs entreprise
Kamaroudin SOS
Mehdi TOUBA

Tutrice université
Marie-Emilie VOGE

du 1er Avril au 31 août

par Antonin Durey


BIKAY


Université
Lille1
Sciences et Technologies

Remerciements

Je tiens tout d'abord à remercier ma tutrice université, Marie-Emilie VOGE, pour son accompagnement semaine après semaine et pour ses conseils tout au long du stage qui m'auront été précieux.

De très grands remerciements s'imposent également à Kama et Mehdi, qui m'auront permis de venir au Cambodge, qui m'auront aidé et épaulé pour que je me sente très à l'aise rapidement. De plus, ils m'auront laissé avoir de grandes responsabilités dans le projet sur lequel j'ai travaillé, aussi bien au niveau des fonctionnalités, mais également avec le client avec qui j'ai eu des contacts directs et réguliers. Cela m'a permis d'en apprendre énormément et je leur en suis très reconnaissant.

Enfin, je souhaite remercier très chaleureusement tous mes collègues cambodgiens et français : Maly, Sokunthea, Cheayden, Chamnap, Vannarith, Phalla, Vitou, Barang, Vuthy, Ya, Mesa, Kimsla, Cédric, Kama, Mehdi, Charles, Pierre et Thomas. J'ai passé de très bons moments avec eux, au travail, au badminton, au football ou ailleurs et souhaite leur dire à tous un très grand merci.

Sommaire

Introduction

Pour mon stage de fin de Licence, j'ai en priorité recherché des stages concernant la programmation web, domaine que je connaissais très peu mais qui m'intéressait beaucoup. Dans un premier temps, mes recherches se sont orientées sur un stage en France, mais par la suite, j'ai étendu celles-ci à l'étranger. C'est ainsi que j'ai trouvé une offre de stage pour du développement web au Cambodge pour une durée de 5 mois. Intéressé par le projet et curieux à propos du pays, j'ai décidé de me lancer.

J'ai ainsi rejoint BiKay, petite entreprise informatique qui développe des applications et sites internet. Le projet sur lequel j'ai travaillé était un projet d'application consistant à s'occuper en interne de location de résidences ou de chambres d'hôtels. L'entreprise cliente utilisait auparavant un logiciel de gestion généraliste de gestion locative. La ligne directrice de ce projet a donc été de coller au plus près des besoins du client, tout en réalisant le projet en moins de 5 mois pour qu'il puisse être livré début septembre.

Dans une première partie, je présenterai le Cambodge, pays dans lequel j'ai vécu pendant 5 mois. J'introduirai également BiKay, l'entreprise dans laquelle j'ai effectué mon stage, pour finalement parler du projet qui a nécessité un certain temps d'analyse. Dans une seconde partie, je présenterai les différentes étapes de mon stage, en commençant par l'apprentissage des langages et outils nécessaires à la réalisation de l'application. Dans cette partie, je parlerai également du développement des différentes fonctionnalités qui m'ont été confiées. Je concluerai en parlant de mon expérience au Cambodge, des compétences acquises et de l'évolution de mon projet professionnel.

Première partie :
Du Cambodge au projet

1.1) Le Cambodge

Le Cambodge est un petit pays d'Asie du sud-est, situé entre la Thaïlande, le Vietnam et le Laos. Il est peuplé d'environ 15 millions d'habitants et a une superficie d'environ 180 000 kilomètres carré, soit environ le tiers de la France.

Historiquement, le pays est connu pour les temples d'Angkor, construits entre le VIII^{ème} et le XIII^{ème} siècle. Dédiés à la famille des souverains de l'époque, ils sont au nombre de plusieurs centaines. Le plus connu d'entre tous est Angkor Wat, qui se trouve même sur le drapeau cambodgien présenté ci-contre.

Illustration 1: Drapeau cambodgien

Le Cambodge est aussi connu en France parce qu'il a fait parti de l'Indochine française, depuis la seconde moitié du XIX^{ème} siècle jusque les années 50, époque de l'indépendance. Le pays fut alors dirigé par le roi Norodom Sihanouk, qui s'est fait chasser du pouvoir en 1970. À la même époque débuta la guerre civile qui mis aux prises le gouvernement désormais pro-américain et les révolutionnaires khmers rouges.

Les khmers rouges furent victorieux en 1975 et plongèrent le pays dans un désastre humain sans précédent : destruction des systèmes scolaires et médicaux, traque des élites et des ennemis du régime, déplacement de la population dans les campagnes pour y construire rizières, barrages, systèmes d'irrigation, etc.

En moins de 4 ans, 3 millions de personnes décédèrent, soit un quart de la population de l'époque. Les khmers rouges furent chassés par l'armée vietnamienne en 1979, mais la guerre civile continua jusque la fin les années 90.

Aujourd'hui, le Cambodge est un pays pauvre, mais avec un fort taux de croissance depuis une dizaine d'années. C'est dans ce contexte que se situe l'informatique, domaine peu développé, mais ayant un potentiel énorme.

1.2) BiKay

BiKay est une entreprise d'expertise et de développement informatique. Elle est axée sur le développement de sites et d'applications web.

1.2.1) Historique

La société Prolepsys est fondée en février 2010 avec l'assistance décisionnelle comme colonne vertébrale de son activité. Les premiers projets sont réalisés par une petite équipe de consultants et se concentrent principalement sur le potentiel du marché local.

En 2011, Prolepsys met en place son premier projet de système d'analyse financière pour GEFCO, le leader mondial dans le secteur logistique. Dans le même temps, et grâce à ses origines françaises, Prolepsys développe son réseau en France et propose d'externaliser le développement et la maintenance de sites pour agrandir son panel de clients : Danone, Evian, Sonuts, Centiflor...

Le Ministère de l'Education, de la Jeunesse et des Sports cambodgien choisit Prolepsys pour développer leur système de gestion et leur base de données centralisée. Ce système sera à terme déployé dans toutes les écoles publiques de toutes les provinces. Ce projet, commencé en 2012, durera 2 ans.

Dans le même temps, l'entreprise chinoise Dumex (agroalimentaire) choisit Prolepsys pour maintenir leur système de simulation de ventes.

Illustration 2: Logo de BiKay

Au début 2015, Prolepsys devient BiKay System & Consulting. Depuis sa création, l'entreprise a connu en moyenne 20 % de croissance annuelle et est passée de 3 à 18 employés.

1.2.2) Domaines d'action

BiKay se concentre aujourd'hui sur le développement web et le développement d'applications mobiles. Elle propose d'autres services comme l'E-marketing, l'expertise ou l'extraction de données. Enfin, elle délivre des formations sur différentes technologies, comme les scripts shell, Wordpress, etc.

1.3) Le projet

1.3.1) Présentation générale

L'entreprise cliente est une entreprise dont l'objectif est d'accompagner les propriétaires de résidences locatives dans la gestion courante de leurs installations, ainsi que dans la commercialisation de leurs biens auprès d'acteurs commerciaux, comme des tours opérateurs, des offices de tourisme, des comités d'entreprise, etc… Elle agit pour le compte de divers propriétaires et se substitue à eux pour effectuer tout ou une partie des opérations courantes des bâtiments. Son portefeuille client est composé de résidences de tourisme ayant différentes caractéristiques : hôtels, appartements, chalets, résidence de mer, de montagne, de ville, ou autre.

Elle opérait son exploitation à l'aide de l'application Resalys qui est une application généraliste pour faire de la gestion de résidences touristiques. Les contraintes imposées par cette application forçaient le client à adapter son modèle aux exigences de l'application. Celui-ci n'utilisait donc qu'un nombre restreint de fonctionnalités car nombre d'entre elles étaient difficilement utilisables en l'état dans le cadre des spécificités d'exploitation de l'entreprise.

La mission consistait en la mise en place d'une nouvelle application qui répondrait aux exigeances du client. Ce dernier a estimé que le temps imparti pour la réalisation de ce projet ne permettait pas de construire une application en partant de zéro. Nous devions ainsi nous baser sur l'application HotelDruid (HD), application open source qui correspondait le mieux à ses demandes.

L'application demandée par le client devait bien sûr contenir des fonctionnalités « de base », pour la plupart incluses dans HD comme la gestion des réservations ou des chambres. Cependant, il était nécessaire qu'elle s'adapte au maximum à ses besoins. Ainsi, d'autres fonctionnalités plus précises et plus difficiles à mettre en œuvre ont été incluses dans le projet.

Dans un soucis pratique et pour cause de confidentialité, la société pour laquelle le projet a été réalisé sera nommée société **Azerty** dans ce rapport.

1.3.2) Détails des fonctionnalités

La liste des fonctionnalités qui nous a été demandée se trouve ci-dessous. Celles-ci ont été triées dans un ordre qui permet facilement la compréhension et les liens entre les différentes tâches. Ainsi, les réservations se trouvent vers la fin de cette liste alors que la logique aurait voulu qu'elles soient placées en premier, étant le cœur de l'application.

- Gestion des résidences et des Unités d'Hébergements (UH). Les UH représentent les chambres d'hôtels, chalets, maisons et autres biens pouvant être loués. Ce menu comprend également divers critères de différenciation pour les résidences et les UH. Il fallait aussi gérer les actions gouvernantes, qui représentent un type de ménage à effectuer. Enfin, ce menu comprenait de pouvoir bloquer des UH pour une durée déterminée, pour cause de travaux, rénovation, inondation, etc.

- Gestion des conventions avec les différents partenaires et types de partenaires. Une convention représente un contrat entre un partenaire et la société, prévoyant notamment les les UH concernées, les conditions de vente, d'annulation ou de paiement.

- Création des fiches clients, fiches partenaires et fiches propriétaires.

- Quantification et tarification. Cela concerne les UH, mais aussi les prestations. La tarification des UH devait se faire en fonction d'un nombre de nuits et d'une période de l'année, ce qui a nécessité plusieurs fonctionnalités au préalable.

- Gestion des réservations. Hormis les actions gouvernantes, cette fonctionnalité se sert de toutes les fonctionnalités listées précédemment.

- Gestion complète des courriers, de la création des modèles à l'envoi automatique.

- Gestion des notes PMS, qui sont des notes de frais adressées aux clients, lors de l'utilisation d'une prestation, d'une consommation au bar/restaurant, etc.

- Gestion comptable de l'application : création des échéances de règlement, édition des factures, affichages des règlements créditeurs et débiteurs...

- Affichages récapitulatifs sous forme de tableaux pour la comptabilité et de plannings pour les réservations et l'occupation des UH.

1.3.3) Planification et répartition

La livraison était prévue pour septembre 2015. L'objectif était de passer entre 10 et 12 semaines sur le développement, celui-ci devant s'achever début juillet. Ainsi, les mois de juillet et d'août devaient nous permettre de corriger l'application et de l'adapter selon les modifications que le client demanderait. De plus, nous devions rédiger un minimum de documentation pour que les futurs développeurs qui reprendraient notre code comprennent certains points délicats.

Initialement, nous étions 3 pour développer ce projet : Mehdi, développeur chez BiKay depuis quelques mois, Thomas, étudiant en 3ème année à Epitech et moi-même.

Mehdi a tout d'abord étudié le projet avec le client avant notre arrivée. Il était prévu qu'il s'occupe des conventions, des notes PMS, d'une partie de la comptabilité et de certains écrans récapitulatifs.

Thomas avait pour tâches initiales les courriers, fiches clients, partenaires et propriétaires, les réservations et les plannings.

Pour ma part, je devais m'occuper de la gestion des résidences, des UH et de leur blocages des actions gouvernantes, des calendriers tarifaires, de la tarification des prestations, des rubriques de ventes et des UHs, de l'autre partie de la comptabilité et de certains affichages statistiques ou récapitulatifs.

Pour mieux comprendre les relations entre les fonctionnalités, vous pouvez consulter l'*Annexe 1 : Récapitulatif des fonctionnalités – début du projet* qui reprend les fonctionnalités et le développeur devant en avoir eu la charge.

Deuxième partie : Développement de l'application

2.1) Prise en main

2.1.1) Serveur Web

L'application que nous allions développer est une application web, domaine dans lequel j'avais des compétences limitées. Ainsi, mes premiers jours chez BiKay ont consisté en une remise à niveau dans ce domaine.

Le mécanisme d'un serveur Web est très simple et peut se résumer en 4 étapes, grâce au schéma ci-dessous :

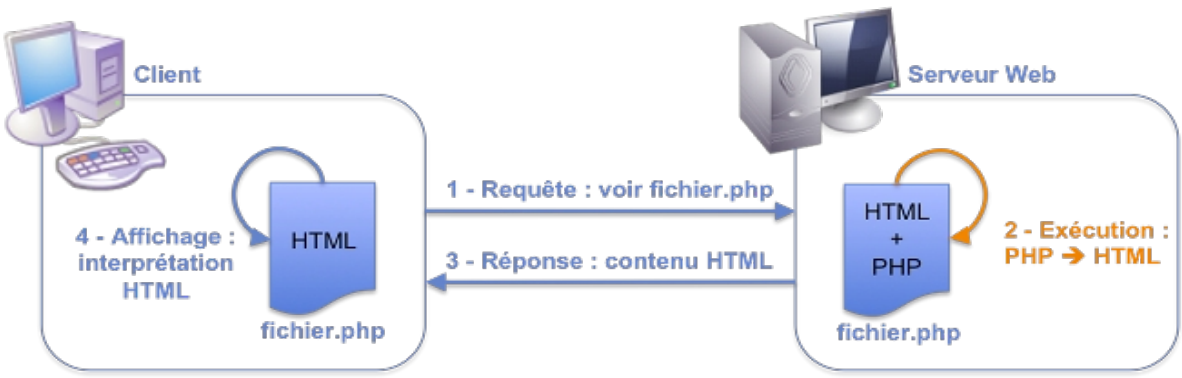


Illustration 3: Schéma représentant le fonctionnement d'un serveur Web

1 – Le client envoie une requête au serveur web HTTP (*HyperText Transfer Protocol*).
2 – Lorsque la requête arrive au serveur, celui-ci analyse le fichier demandé. Si celui-ci doit être interprété, le serveur l'interprète et génère la page HTML (*HyperText Markup Language*) associée.
3 – Le serveur envoie la page HTML générée.
4 – Le client reçoit la page HTML puis son navigateur l'interprète.

Le serveur web est composé d'un système d'exploitation, un serveur HTTP, un langage serveur et un SGBD (*Système de Gestion de Base de Données*). Dans notre cas, nous nous sommes servis d'une plateforme LAMP :

- Linux : système d'exploitation
- Apache : serveur HTTP
- MySQL : SGBD
- PHP : langage serveur

2.1.2) PHP

J'ai rapidement commencé à apprendre le langage PHP (acronyme signifiant *PHP Hypertext Preprocessor*) et le framework Laravel qui allaient être les technologies principales que j'allais utiliser pendant mon stage. Durant mon cursus, je n'ai travaillé que sur une seule technologie web - le Java EE (*Java Enterprise Edition*) - et cela pendant un temps très court. L'apprentissage du PHP m'a fait retravailler les bases que j'avais acquises en DUT (*Diplôme Universitaire de Technologie*) lors de l'étude du Java EE. Quant à Laravel, cela a été une découverte totale.

PHP est un langage de programmation impératif orienté objet. Il est principalement utilisé pour produire des pages web dynamiques en utilisant un serveur HTTP comme présenté ci-dessus. Néanmoins, il peut aussi être utilisé comme n'importe quel langage s'il est interprété de manière locale. Développé à la fin des années 90, le langage est aujourd'hui très répandu à travers le monde et est considéré comme la base de création des sites dynamiques[1].

Le principal intérêt de PHP est donc de générer des pages en fonction des données envoyées par le client. Ainsi, lorsque ce dernier naviguera sur Internet, le PHP permettra, grâce à des données susceptibles de l'identifier, de générer du contenu approprié : nom, prénom, email, mot de passe, achats en cours sur internet, etc...

Illustration 4: Logo de PHP

Le code PHP s'écrit dans la balise `<?php ?>`. A l'intérieur de celle-ci, la syntaxe du PHP est très similaire à d'autres langages comme le C, à l'exception faite qu'il n'y a pas besoin de déclarer le type d'une variable. Voici ci-dessous un exemple simple de code PHP.

```
<?php if (strtolower($_POST['lang']) == 'fr')?>
   <p>Vous parlez français !</p>

<?php elseif (strtolower($_POST['lang']) == 'en')?>
   <p>You speak English !</p>

<?php else ?>
   <p>Je ne vois pas quelle est votre langue !</p>
```

Néanmoins, utiliser le PHP seul dans des pages HTML devient rapidement contraignant et très lourd. Il est donc essentiel d'utiliser un bon framework[2] PHP pour automatiser un maximum de tâches et bénéficier d'un moteur de template (voir plus loin). Pour le projet Azerty, il a été décidé de se servir du framework Laravel.

1 Voir glossaire
2 Voir glossaire

2.1.3) Laravel

Laravel est un framework PHP respectant le modèle MVC[1] et entièrement développé en programmation orientée objet. Bien que créé en 2011, il est l'un des frameworks les plus utilisés en PHP. Il fournit des fonctionnalités dans plusieurs domaines. Ceux-ci sont présentés les uns à la suite des autres et pour chacun d'eux, l'équivalent Laravel est donné :

- Une interface en ligne de commande : elle permet de saisir des commandes particulières et uniques au framework. Celle de Laravel s'intitule *Artisan* et permet de créer des modèles, contrôleurs, de lancer les migrations de la BDD, etc.

- Relation avec la base de données (BDD) : Laravel se sert du mapping objet-relationnel. Ce système crée l'illusion d'une base de données orientée objet à partir d'une base de données relationnelle. Dans Laravel, la base de données orientée objet est constituée de modèles qui sont des classes reliées entre elles. Une présentation plus détaillée est effectuée plus loin dans ce rapport.

- *Migration* de BDD : Cela permet de facilement créer, modifier ou supprimer des tables dans une BDD. Avec Laravel, chaque migration correspond à une classe[2] ayant une syntaxe propre. Laravel contient également un moyen de remplir facilement une base de données avec des données factices, appelé *Seeder*.

- Un système de gestionnaire de dépendances : celui de Laravel s'appelle *composer*.

- Un système de template : celui-ci permet principalement de distinguer les éléments de traitement de ceux d'affichage. Cela permet ainsi d'avoir un code plus clair et plus lisible. Chez Laravel, le moteur de template s'appelle blade. Il permet l'héritage de templates et l'existence de sections pour avoir des parties de page similaires. Le code ci-dessous est le même code que cité dans la section PHP, mais en se servant de blade. Cela permet de se rendre compte de la plus grande lisiblité et maniabilité du code lors de l'utilisation d'un moteur de template.

```
@if (strtolower($_POST['lang']) == 'fr')
    <p>Vous parlez français !</p>
@elseif (strtolower($_POST['lang']) == 'en')
    <p>You speak English !</p>
@else
    <p>Je ne vois pas quelle est votre langue !</p>
@endif
```

Les fichiers se servant de blade ont l'extension `.blade.php`. Lorsque le serveur va détecter un fichier ayant l'extension `.blade.php`, il va tout d'abord convertir ces fichiers en fichiers `.php` avec uniquement la syntaxe du PHP.

1 Voir glossaire
2 Voir glossaire

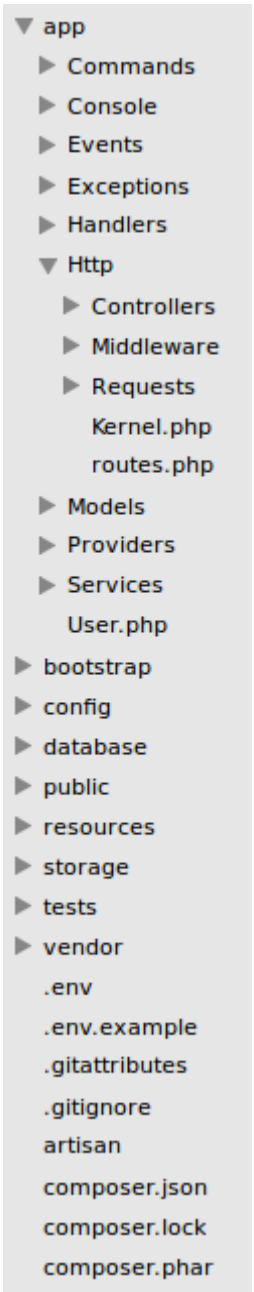


Illustration 5: Arborescence d'un projet Laravel

Lors de la création d'un projet Laravel, une arborescence est créée. Chaque dossier et chaque sous-dossier correspond à une partie bien spécifique de l'application. Ci-dessous sont listés les dossiers selon un projet *Laravel 5.0*, avec ce qu'ils contiennent.

- `app` : contient un ensemble de dossiers représentant les `commandes`, `évènements`, `exceptions`, `handlers`, `services` ou encore `providers`.

Ce sous dossier contient les `contrôleurs`, `middleware`, `modèles` et le fichier `routes.php` qui indique l'ensemble des URL autorisées par l'application.

- `bootstrap` : contient les fichiers de Bootstrap (voir plus loin).

- `config` : contient des fichiers de configuration relatifs à l'application, comme la connexion avec la BDD, des éléments d'authentification, des chemins par défaut, etc.

- `database` : contient les migrations et le seeder.

- `public` : contient les fichiers JavaScript, CSS et les images.

- `resources` : contains certains fichiers Bootstrap, les fichiers de langues pour la traduction et l'ensemble des vues créées par les développeurs/intégrateurs.

- `storage` : vues générées par Laravel. Elles ne contiennent donc plus que la syntaxe native de PHP.

- `vendor` : fichiers générés automatiquement en fonction de dépendances, des migrations...

À la racine du projet se situe le fichier `.env`. C'est le fichier qui indique les données de connexion à MySQL. Il est ignoré par `git` grâce au fichier `.gitignore` et permet donc d'avoir un nom d'utilisateur et un mot de passe propre pour chaque développeur.

On trouve également à cet endroit les fichiers concernant `composer`.

2.1.4) JavaScript

Le JavaScript est comme son nom l'indique un langage de programmation de scripts. Il est majoritairement utilisé conjointement avec les pages web HTML. Il n'est ni compilé, ni précompilé, mais interprété. Cela signifie qu'il est nécessaire d'avoir un interpréteur pour lire et réaliser les actions demandées. Pour le JavaScript, l'intepréteur est compris dans le navigateur. C'est un langage côté client, ce qui signifie que les scripts sont exécutés par le navigateur du client.

Ce langage permet de dynamiser une page HTML, en ajoutant des interactions avec l'utilisateur ou des animations telles que masquer/afficher du texte, faire défiler des images, créer des infobulles.

Le JavaScript se place entre les balises `<script></script>`. Bien que pas obligatoire, il est préférable de placer le code JavaScript d'une page à la fin de celle-ci, pour permettre dans un premier temps le chargement des éléments HTML et CSS.

```
<span id='click-me'>Cliquez moi!</span>

<script>
	var element = document.getElementById('click-me');

	element.addEventListener('click', function(){
		alert('Vous m'avez cliqué !');
	});
</script>
```

Ci-dessus se trouve un court exemple de code JavaScript. Il affiche un message `Vous m'avez cliqué !` lorsque l'on clique sur l'élément ayant l'id `click-me`.

Parmi les technologies que contient le JavaScript, l'AJAX (*Asynchronous JavaScript and XML*) est l'une des plus intéressantes. C'est un ensemble de technologies pour exécuter des fonctions côté serveur, sans que cela ne nécessite le moindre rechargement visible de la page. Ainsi, l'AJAX permet de transmettre des données à un serveur web et d'en recevoir en retour, en utilisant des technologies de formatage de données, comme le XML ou le JSON (respectivement *Extensible Markup Language* et *JavaScript Object Notation*).

Comme tout langage, le JavaScript possède de nombreuses librairies qui permettent de faciliter son utilisation. Pour développer le projet, nous nous sommes servis de la librairie JQuery.

2.1.5) JQuery

JQuery est une bibliothèque JavaScript pour faciliter l'écriture de scripts. Elle permet notamment de parcourir et de modifier très facilement les éléments HTML, de gérer plus facilement les évènements ou encore de simplifier l'utilisation des requêtes AJAX.

Selon moi, l'aspect le plus utile et le plus puissant est la présence des sélecteurs CSS. Ces derniers permettent de sélectionner des éléments HTML très précis. Une fois l'élément/les éléments sélectionné(s), il est possible d'y lier des évènements, de modifier leur contenu HTML, leurs attributs ou leur comportement. Ci-dessous se trouvent plusieurs exemples pour bien comprendre la puissance et l'intérêt de ces sélecteurs.

```
/* Tous les éléments de type <select> */
$("select")

/* L'élément ayant l'id table */
$("#table")

/* Les <tr> de l'élément ayant l'id table */
$("#table tr")

/* Les éléments ayant la classe input et contenus dans les <tr> de
l'élément ayant l'id table */
$("#table tr .input")

/* Les éléments ayant la classe input contenus dans le troisième
<td> du cinquième <tr> de l'élément ayant l'id paiement et étant
de type table */
$("table#paiement tr:eq(5) td:eq(3) .input")
```

2.1.6) Bootstrap

Bootstrap est un framework front-end élégant et intuitif utilisé pour le développement web. C'est un ensemble d'outils qui contient des éléments HTML et CSS (*Cascading Style Sheets*), des formulaires, boutons et outils de navigation. Il est développé depuis 2010 et est placé sous licence open source depuis 2011.

D'après mon utilisation, le principal intérêt de Bootstrap réside dans le fait que l'agencement des composants HTML peut facilement être géré, peu importe la plateforme sur laquelle le site est utilisé. En effet, Bootstrap possède un système de grille qui « découpe » l'élément parent en 12 colonnes de taille égale. Ce système permet ensuite de spécifier la taille , en colonnes, de chaque élément, pour chaque taille d'écran – extra small, small, medium ou large.

L'*Annexe 2 : Présentation de Bootstrap* montre plusieurs composants disponibles avec Boostrap, et prend l'exemple de l'affichage d'un même code avec un écran de taille moyenne et avec un écran très petit.

2.1.7) Git

J'ai retrouvé l'incontournable `git` pour travailler en collaboration avec mes collègues. Mes connaissances de ce précieux outil se sont ainsi grandement améliorées et je suis désormais très à l'aise avec les commandes tel que `git branch`, `git checkout` ou `git stash`.

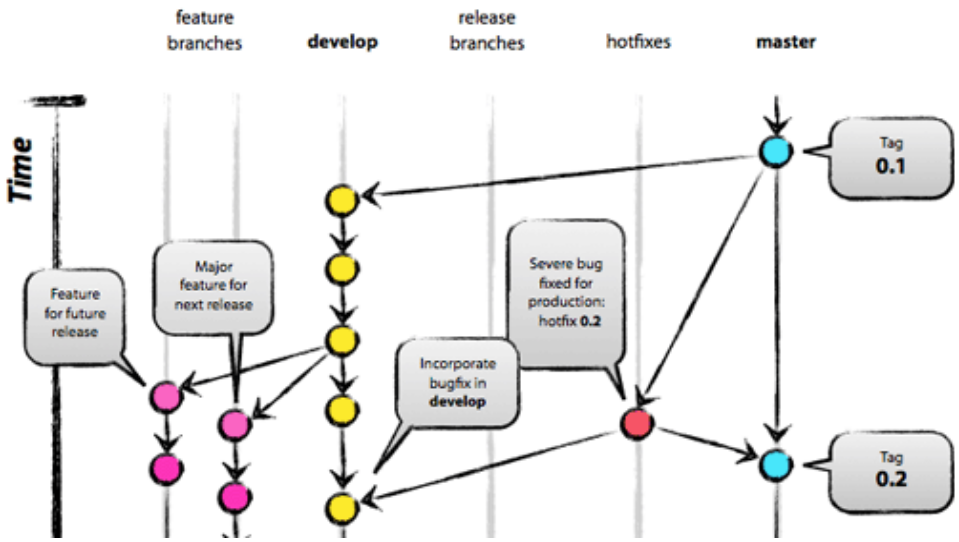


Illustration 6: Différentes branches sur un même projet

L'*Illustration 6* ci dessus présente l'intérêt d'avoir plusieurs branches sur un même projet. Cela permet d'avoir des fonctionnalités pouvant être tester en toute sécurité, et également de corriger des problèmes en production sans mélanger les codes des autres branches.

2.2) Base de données

En même temps que mon apprentissage des éléments techniques, j'ai été chargé d'une première tâche consistant à créer et remplir la base de données. J'ai ainsi fait la connaissance de *MySQL Workbench*.

Ce dernier est un logiciel de création et d'édition de BDD qui s'utilise via une interface graphique ergonomique et intuitive. Pour utiliser toutes ses fonctionnalités, il doit normalement être connecté à une BDD. Dans notre cas, nous nous en sommes servis uniquement pour visualiser l'ensemble des tables et leurs relations. Pour le reste, c'est-à-dire la création des tables et leur remplissage avec des données, nous avons utilisé les migrations et le seeder de Laravel.

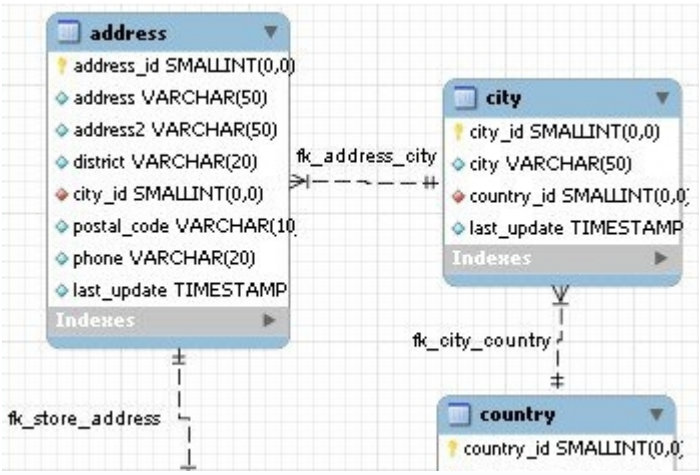


Illustration 7: Exemple d'utilisation de MySQL Workbench

L'image ci-contre montre un exemple de quelques tables créées avec *MySQL Workbench*. On peut y voir également les champs, les clefs primaires, les relations, etc.

En utilisant les tables et relations que Mehdi avait créées à partir de *MySQL Workbench*, j'ai créé l'ensemble des migrations. À l'époque, la BDD contenait une quarantaine de tables.

Nous avons fait le choix de minimiser les relations entre les tables au niveau de la BDD. En effet, cela aurait tout d'abord pris plus de temps à être créé et entretenu. De plus, les relations en cascade auraient pu poser problème lors de la suppression ou l'insertion de données. Ainsi, nous avons modélisé les relations au niveau de la BDD objet de Laravel, dont je reparlerai plus tard.

Ma tâche suivante a été de remplir ces tables avec des données factices pour pouvoir tester rapidement les premières fonctionnalités. J'ai pris la plupart de ces données dans les fichiers que le client nous avait donnés.

Ces deux tâches m'ont permis d'avoir très rapidement une connaissance détaillée de la BDD et ainsi d'entrevoir la structure globale de l'application. J'ai eu par la suite d'innombrables modifications à faire sur les migrations et le seeder au fur et à mesure que notre connaissance du projet augmentait. En effet, nous n'avions pas de cahier des charges mais une unique note de cadrage d'une dizaine de pages. Cela a entraîné beaucoup d'interrogations car une majorité de fonctionnalités était décrite assez superficiellement. Au fur et à mesure que le client nous expliquait plus en détail celles-ci, il a fallu modifier les tables et leur relations.

Lorsque j'ai eu fini ces tâches, l'environnement technique était prêt et la première analyse complétée. Nous avons alors pu commencer le développement.

2.3) Paramétrage des résidences

Il était prévu que nous nous servions d'*HotelDruid* comme support de développement. Cela aurait dû nous permettre de gagner du temps sur le développement des modules « courants » pour nous attarder sur les fonctionnalités plus spécifiques.

Après l'avoir étuidé, nous ne nous en sommes finalement pas servi, pour plusieurs raisons : front-end peu ergonomique et intuitif, back-end en italien, etc…

Le paramétrage des résidences consistait à créer un ensemble de CRUD. Un CRUD (*Create Read Update Delete*) représente l'ensemble des actions de base concernant les données dans une BDD. Lors de la création d'un contrôleur avec Laravel, celui-ci contient déjà les bonnes méthodes pour créer un CRUD. Celles-ci sont au nombre de 7.

- `index` : renvoie vers une page HTML listant les différentes informations de la table concernée. Cette page contient les liens pour créer, éditer, afficher ou supprimer une donnée.
- `create` : renvoie vers une page HTML permettant de créer une donnée en BDD. La page HTML renverra vers la méthode `store`.
- `store` : contient le code PHP pour insérer une donnée en BDD.
- `show` : renvoie vers une page HTML pour afficher une donnée.
- `edit` : renvoie vers une page HTML contenant un formulaire pour modifier une donnée déjà existante en BDD. La page HTML renverra vers la méthode `update`.
- `update` : contient le code PHP pour modifier une donnée en BDD.
- `destroy` : contient le code PHP pour supprimer une donnée de la BDD.

Ainsi, chacun des éléments concernant le paramétrage des résidences et des UH était un CRUD. Cela concernait les résidences et les UH mais aussi des éléments pouvant servir de critères de différenciation. En effet, le client a demandé à ce que les résidences aient également un thème (ville, montagne...) et une catégorie. De la même manière, les UH devaient être différentiables selon un type, un groupe et une caractéristique.

L'écran index du CRUD concernant les UH se trouve ci-dessous. On retrouve 2 entrées en BDD, et pour chacune d'elles, les différents boutons pour les actions de base.

Illustration 8: Vue 'index' concernant les UH

C'est à cette occasion que j'ai réellement découvert la BDD objet utilisée par Laravel. Chacun des modèles est une classe, possédant une série d'attributs et de fonctions. Ces dernières permettent d'établir des relations entre les modèles, au moyen de mots-clés telles que `belongsTo`, `hasOne`, `hasMany`, `belongsToMany`, `morphTo`. Les noms de ces fonctions donnent une idée des relations entre les tables – 1.1, 1.N ou N.N. Ainsi, en appelant ces fonctions, on obtient directement les objets souhaités.

J'ai ensuite travaillé sur les blocages d'UH. Un blocage consiste à empêcher la location d'une UH pour une durée déterminée, pour cause de travaux, inondations, ou autre… Un tableau récapitulatif, présenté ci-dessous, liste les différents blocages. En dessous de celui-ci, un court formulaire permet d'en rajouter. Peu utiles sous cette forme, ces informations se trouvent également sur l'un des plannings, élément dont je reparlerai plus tard dans ce rapport.

Numero d'UH	Type d'UH	Groupe d'UH	Blocages	Description	Action
BI-01	Appartement 2 pièces - 4 personnes	Etage 1	10-07-2015 / 13-07-2015 Raison : Mise hors service d'hébergements	Fuite d'eau	Supprimer
BI-02	Appartement 2 pièces - 4 personnes	Etage 1	13-07-2015 / 15-07-2015 Raison : Mise hors service d'hébergements	Problème Ecoulement	Supprimer
			10-07-2015 / 13-07-2015 Raison : Mise hors service d'hébergements	Fuite d'eau	Supprimer

Illustration 9: Tableau récapitulatif des blocages d'UH

Pour clôturer ce module, j'ai développé la fonctionnalité concernant les actions gouvernantes. Il s'agit de types de ménages à effectuer, selon plusieurs paramètres. Même si un CRUD a été prévu pour pouvoir en créer dans le futur, il n'y a actuellement que 2 types d'actions gouvernantes : la recouche et la mise à blanc :

- une recouche est un ménage à effectuer tous les jours tant que la chambre est occupée : changement des serviettes, nettoyage du sol…
- la mise à blanc concerne les chambres quittées dans la matinée : il est donc nécessaire de changer les draps, nettoyer la chambre complètement...

Ces premières fonctionnalités ont nécessité environ 3 semaines de programmation. C'est une durée conséquente car je maîtrisais encore mal Laravel et ai ainsi pris du temps sur des tâches simples. Lorsque j'ai eu fini sur ces tâches « de base », j'ai entamé une autre partie de l'application : la quantification et la tarification.

2.4) Quantification et tarification

2.4.1) Calendrier tarifaire

Selon la note de cadrage du projet, un calendrier tarifaire est un intervalle de date. Il permet de pouvoir paramétrer des prix en fonction de dates. Comme ces intervalles de dates peuvent être très courts (une semaine, un jour), la solution retenue a été de permettre de créer dynamiquement des calendriers tarifaires. Cela permet d'en insérer plusieurs à la fois en BDD.

Pour programmer cette fonctionnalité, j'ai pour la première fois utilisé le JavaScript et son côté dynamique pour insérer du code HTML dans une page. Le résultat de cette fonctionnalité, montré ci-dessous, contient un bouton pour ajouter dynamiquement une ligne et un bouton *Corbeille* à la fin de chaque ligne pour supprimer celle-ci. Lors d'un clic sur un champ d'insertion, un calendrier apparaît pour permettre plus facilement la saisie d'une date.

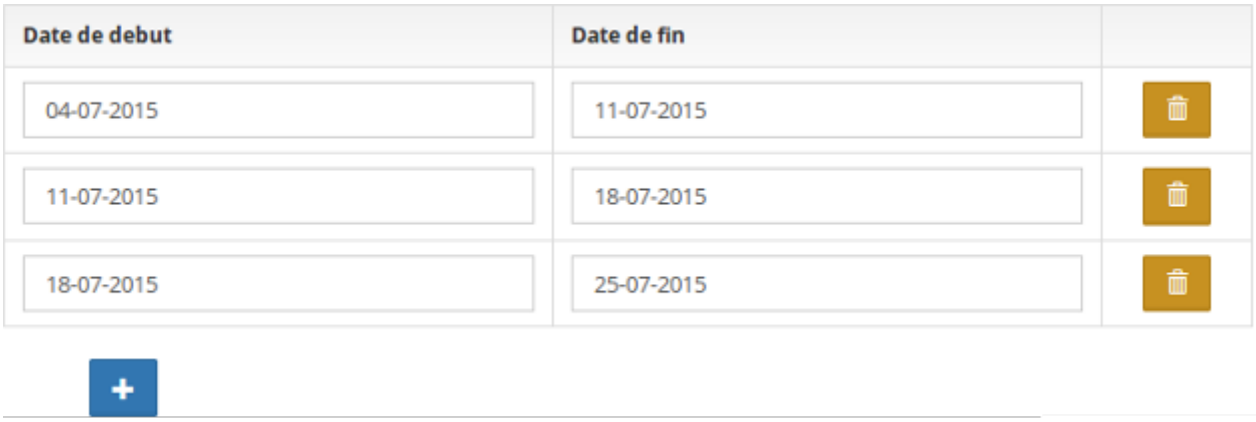


Illustration 10: Saisie dynamique des calendriers tarifaires

Après avoir créé plusieurs dizaines de calendriers tarifaires pour une résidence, il devient difficile pour l'utilisateur de vérifier rapidement si un calendrier tarifaire existe. J'ai donc ajouté un court formulaire en haut de page pour saisir une résidence, une date de début et une date de fin. Cela permet dans un premier temps de vérifier l'existence de calendriers.

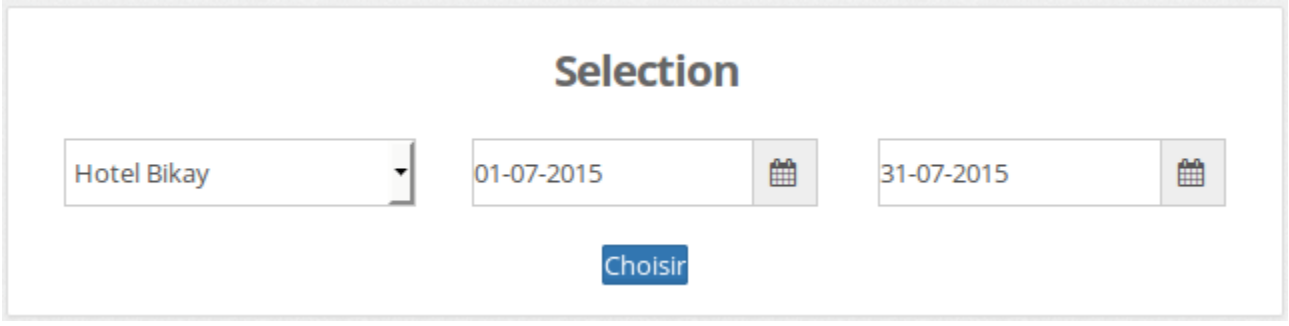


Illustration 11: Formulaire de sélection et d'affichage des calendriers tarifaires

En étudiant de plus près l'application Resalys, je me suis rendu compte qu'elle possédait un générateur automatique de calendriers tarifaire. J'ai reproduit un générateur semblable qui permet de créer des entrées en BDD à partir d'un intervalle de date et d'un nombre de jours.

La validation de ce formulaire avec une date de début égale au 05-09-2015, une date de fin égale au 26-09-2015 et un nombre de jours égal à 7 créera 3 calendriers tarifaires : du 5 au 12, du 12 au 19 et du 19 au 26 septembre.

Dans un premier temps, les dates étaient au format `Y-m-d` (Année – mois – jour → 2015-09-20), qui est le format dans lequel les dates sont stockées en BDD. Néanmoins, le client nous a rapidement demandé de changer ce format pour que les dates soient au format `d-m-Y` (jour – mois – année → 20-09-2015). En conséquence, il a fallu formater toutes les dates venant de la BDD et également les dates lors de tests SQL.

2.4.2) Prestations

J'ai ensuite programmé la fonctionnalité concernant les prestations. Une prestation représente un service ou un produit pouvant être loué et consommé pendant une réservation. Cela peut concerner des choses aussi variées que les fauteuils roulants, les places de parking, les salles de conférence... La première partie a concerné un CRUD expédié assez rapidement. La seconde portait sur la quantification et la tarification des prestations pour chaque résidence. En effet, une résidence possède plusieurs prestations et pour chacune, il fallait pouvoir saisir sa quantité, c'est-à-dire son nombre disponible et son prix.

J'ai ainsi mis en place un tableau à 2 dimensions, présenté ci-dessous, où les calendriers tarifaires et les prestations sont à gauche. Pour chaque intervalle de date et pour chaque prestation, il est ainsi possible d'entrer le nombre de prestations disponibles, leur prix par jour et par heure.

Calendrier	Prestation	Quantite	Prix a la journee	Prix a l'heure
01-08-2015 / 08-08-2015	Fauteuil roulant		€	€
	Place de parking		€	€
08-08-2015 / 15-08-2015	Fauteuil roulant		€	€

Illustration 12: Tarification des prestations

Un court formulaire, assez identique à celui des calendriers tarifaires, permet tout d'abord de sélectionner une résidence, un intervalle de date et une liste des prestations à quantifier et tarifer. Cela permet à l'utilisateur de quantifier et tarifer uniquement les prestations disponibles dans une résidence donnée, sur un intervalle de date.

Après en avoir discuté avec le client, ce dernier nous a expliqué qu'il existait 2 types de prestations : les prestations de base et les prestations complémentaires, que l'on peut appeler également prestations fixes et prestations non fixes.

Les premières sont les prestations de type Bed & Breakfast, demi pension, pension

complète, etc. Elle représentent le type et le nombre de couverts sur une réservation. Une seule prestation de ce type est possible à la réservation, où l'on multiplie son prix par le nombre de nuits et le nombre de personnes. Elles se nomment prestations fixes car leur prix ne varient pas selon les calendriers tarifaires.

À l'inverse, les prestations non fixes représentent tous les autres types de prestations. Il peut s'agir de choses multiples et variées comme une place de parking, un lit bébé, un fauteuil roulant ou une salle de conférence. Elles sont appelées prestations non fixes car leur prix dépend des calendriers tarifaires. Ce sont des prestations de ce type qui sont affichées et tarifées sur l'*Illustration 12*.

Il était également prévu de prévoir un CRUD et une tarification concernant des rubriques de vente. Sans trop comprendre ce que cela représentait, j'ai développé ces fonctionnalités en approximativement une semaine. J'ai malheureusement compris par la suite que les rubriques de vente et les prestations représentaient les mêmes choses. La seule différence est que le terme de prestation est utilisée lorsque son utilisation fait l'objet d'une tarification. J'ai ainsi perdu une semaine de travail.

2.4.3) Type de période et affectations des prix

Avant de pouvoir associer un prix à une UH, il a fallu créer une autre fonctionnalité de paramétrage : les types de périodes. Un type de période représente un nombre de jours pour lesquels un prix peut être attribué. Le client a demandé à ce que ce nombre de jours soit toujours égal à 1, 3 ou 7, mais à laisser la possibilité aux utilisateurs de saisir d'autres types de période. Cette fonctionnalité a été développée sous la forme d'un autre CRUD.

Il a ensuite été question de tarifer les UH. La tarification d'une UH dépendait de trois paramètres : son type, le type de période et le calendrier tarifaire. J'ai ainsi conçu un tableau à 3 dimensions pour cette fonctionnalité, ressemblant à celui utilisé pour tarifer les prestations non fixes. Ce tableau se trouve ci-dessous.

Calendrier	Periode	Appartement 2 pièces - 4 personnes	Appartement 3 pièces - 6 personnes	Appartement 4 pièces - 8 personnes
27-06-2015 / 04-07-2015	1 nuit	5 €	12 €	18 €
	3 nuits	5 €	78 €	14 €
04-07-2015 / 11-07-2015	1 nuit	18 €	35 €	29 €
	3 nuits	75 €	28 €	64 €

Illustration 13: Tarification des UH

Comme pour les calendriers tarifaires et les prestations, un court formulaire a été ajouté pour permettre une tarification sur une période sélectionnée et avec les types de période voulus.

Cette série d'écrans et de contrôleurs concernant la quantification et tarification a été très instructive puisqu'elle m'a fait manipuler en profondeur le JavaScript et JQuery. Comme précisé plus haut, j'ai vraiment été impressionné par la puissance des sélecteurs utilisés par JQuery, qui permettent un contrôle très fin et très précis sur les éléments et leurs évènements.

À partir de cette étape, nous avons pu avoir une réunion hebdomadaire avec le client pour lui présenter les nouvelles fonctionnalités et discuter avec lui des modifications et des corrections qu'il souhaitait. À cette époque, Mehdi avait du retard sur la fonctionnalité des conventions et moi un peu d'avance. J'ai donc récupéré le module des notes PMS qu'il devait initialement développer.

Dans le même temps, l'équipe s'est agrandie avec l'accueil d'un nouveau stagiaire, Charles, en 4è année à l'INSA. Il a pris en charge les écrans statistiques et récapitulatifs dont j'avais initialement la responsabilité, puis l'envoi de mails.

2.5) Éléments comptables et écrans récapitulatifs

2.5.1) Taxes

Les taxes de l'application représentent l'ensemble des taxes qui doivent être prises en compte dans le calcul d'un prix. La TVA (*Taxe sur la Valeur Ajoutée*) est donc bien sûr représentée, mais également les taxes concernant la location d'UH, comme la taxe de séjour - différente selon les villes - ou la taxe additionnelle départementale.

Un premier problème s'est posé rapidement, celui de saisir le montant de la taxe. Celui-ci peut-être de 2 sortes : soit un pourcentage (TVA), soit une valeur fixe (taxe de séjour). Il a ainsi fallu faire des vérifications pour vérifier que l'utilisateur saisissait une et une seule valeur.

Un second problème, beaucoup plus complexe, nous a été expliqué par le client : la taxe additionnelle départementale est une taxe ayant un pourcentage s'appliquant sur la taxe de séjour.

On peut prendre l'exemple d'une résidence avec une taxe de séjour de 10€ et une taxe additionnelle départementale de 10 %. Si l'on prend une réservation de 3 jours pour 2 personnes, on multiplie tout d'abord le montant de la taxe de séjour par le nombre de nuits et le nombre de personnes, ce qui donne 10*3*2 = 60€. Par la suite, on applique le pourcentage de la taxe additionnelle départementale sur ce montant, soit 10 %*60€ = 6€.

Cela a déjà demandé un certain temps d'adaptation du code déjà existant, mais également pour vérifier tous les cas – création d'une taxe, édition à partir d'un pourcentage, édition à partir d'une valeur fixe…

La tâche suivante a été de rattacher des taxes à des résidences - et donc à des réservations - pour calculer dynamiquement les prix des réservations de cette résidence. Pour cela, les taxes ont été reliées à des résidences au moyen d'une table pivot[1] en back-end[2] et d'un sélecteur spécial en front-end,[3] montré ci-contre.

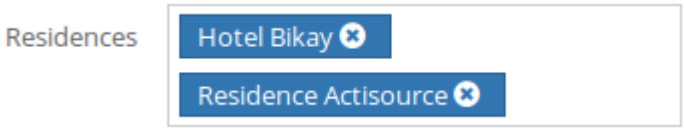


Illustration 14: Sélecteur spécial pour choisir plusieurs résidences à la fois

De plus, nous ne pouvions pas supprimer les taxes, ou les modifier car il fallait toujours pouvoir faire des calculs comptables sur des réservations remontant plusieurs mois en arrière. Nous avons ainsi mis en place le soft-delete. Cela consiste à ne pas supprimer une entrée de la BDD, mais à la considérer comme non présente lors d'affichages récapitulatifs des taxes, comme si la taxe n'était plus active. Dans notre application, lorsque l'utilisateur édite une taxe, celle-ci est soft-deleted et une nouvelle taxe est créée pour la remplacer. Combiné à la table pivot, ce mécanisme assure l'accès aux taxes d'une réservation, même supprimées avec un soft-delete.

2.5.2) Notes PMS

Une note PMS (*Promisory Management System*) est une note de frais créée lors d'une consommation d'une prestation, d'un service ou d'un produit. Elle englobe ainsi les repas des restaurants, les consommations des bars et bien d'autres choses.

La première étape a consisté en un CRUD assez simple qui liait une UH, une prestation, un montant et une description. Très rapidement, il a fallu renseigner à quelle réservation la note était reliée, pour simplifier les traitements dans d'autres fonctionnalités. Cela a été fait à l'aide d'une requête AJAX. Celle-ci est exécutée lorsque l'utilisateur renseigne l'UH et se contente de regarder si une réservation existe pour une UH donnée pour la date du jour.

Le client nous a ensuite demandé de pouvoir lier non pas une, mais plusieurs prestations à une note. Nous devions également afficher les prix TTC et HT, tout en affichant le montant de la TVA. Toutes ces informations étant stockées dans une prestation, il fallait également ne pas pouvoir les modifier dans la fonctionnalité des notes pour éviter des erreurs.

De plus, comme mentionné précédemment, une note peut concerner des repas au restaurant ou des consommations au bar. Ces éléments ne sont pas considérés comme des prestations et il a ainsi fallu le prendre en compte.

En prenant en compte tous ces éléments, le CRUD est devenu assez complexe . Dans celui-ci, on peut rajouter des prestations dynamiquement et tous les calculs se font de manière automatique. Un exemple de cette fonctionnalité se trouve page suivante.

1 Voir glossaire
2 Voir glossaire
3 Voir glossaire

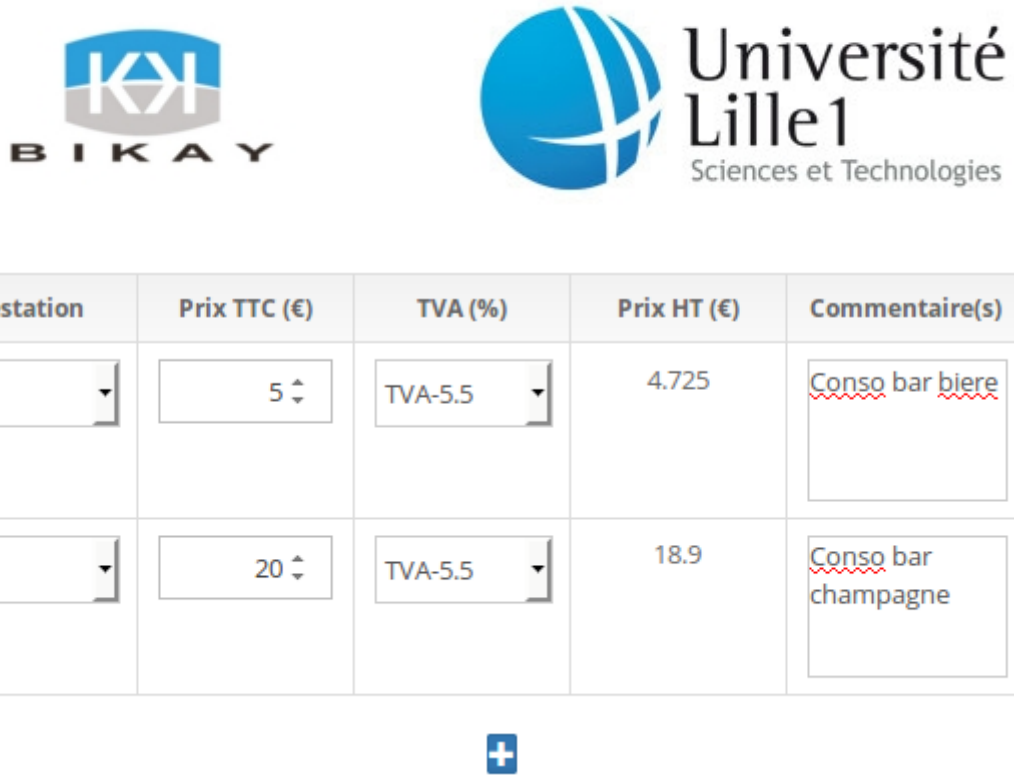


Illustration 15: Note avec deux consommations

Dans l'*Illustration 15*, la note contient deux consommations au bar. Comme ces dernières ne sont pas représentées sous la forme d'une prestation, il faut saisir les montants à la main, et choisir la TVA parmi une liste déroulante.

La dernière étape a été de prendre en compte les conventions. En effet, si une note PMS est reliée à une UH, elle est donc liée à une réservation. De ce fait, il est possible de trouver le partenaire qui a pris cette réservation et donc sa convention. Dans celle-ci, il est possible de renseigner une liste de prestations qui sera payée par le partenaire lors d'une réservation. Ainsi, il a fallu prévoir qu'un partenaire puisse payer une partie d'une note PMS, selon les prestations apparaissant dans cette note.

Cette étape a de nouveau complexifié davantage le code existant et donné l'affichage ci-dessus, qui affiche les différents montants à payer par l'occupant de la chambre et par le partenaire s'il existe.

Illustration 16: Différents champs de formulaire pour renseigner le montant devant être payé par l'occupant et par le partenaire

Cette fonctionnalité a demandé un temps très conséquent de développement, notamment parce que le client nous précisait les choses au fur et à mesure. Bien que ne travaillant pas uniquement sur ce module à cette époque, le développement de cette fonctionnalité s'est étalé sur environ un mois.

Lors d'une réunion, le client nous a expliqué que les deux tâches principales étaient la prise de réservation et le planning les concernant. Comme Thomas travaillait sur les réservations et devait ensuite faire les plannings, il nous a semblé judicieux que l'un de nous récupère la fonctionnalité concernant les plannings pour la livrer au client le plus vite possible. Étant le moins en retard, c'est à nouveau moi qui ai développé cette fonctionnalité.

2.5.3) Plannings

Les plannings, qui sont au nombre de 3, sont des écrans récapitulatifs sous forme de calendrier, en affichant à l'écran différentes informations de manière intuitive et ergonomique.

Ces 3 calendriers représentent des informations propres à chaque résidence :

- le premier représente la disponibilité selon le type d'UH,
- le second représente les disponibilités des prestations,
- le troisième liste les UH, les blocages d'UH et les réservations selon un code couleur bien précis. Ce dernier devait également comporter d'autres fonctionnalités, comme le changement d'UH d'une réservation par un *drag-and-drop*.

La principale difficulté de cette fonctionnalité a été de trouver un plugin[1] qui proposait déjà l'affichage sous forme de planning. J'ai dans un premier temps étudié le planning fourni par *SmartAdmin*, qui est l'outil d'administration de sites dont nous nous sommes servis.

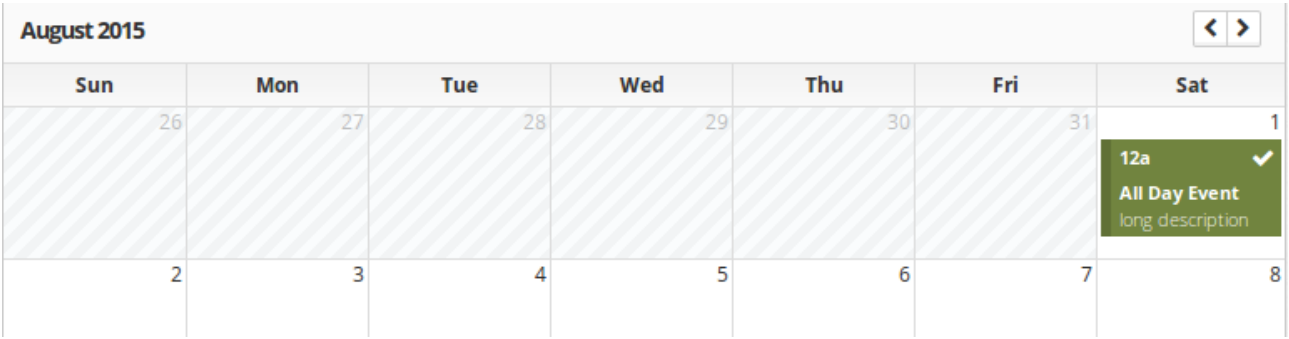


Illustration 17: Planning de SmartAdmin

Malheureusement, il n'était pas possible de le configurer comme je le souhaitais, notamment au niveau du nombre de jours affichés par ligne. J'ai rencontré ce problème sur beaucoup de plugins que j'ai testés. Finalement, j'ai trouvé un plugin nommé *FullCalendar Scheduler* que je pouvais suffisamment paramétrer pour obtenir un visuel presque identique à celui demandé par le client.

Dans un premier temps, il m'a donc fallu paramétrer l'échelle de temps pour que celle-ci se situe en haut du planning. Je devais aussi laisser le choix du nombre de jours affichés à l'écran - 30 jours, 90 jours ou autre - ainsi que la possibilité de naviguer sur l'échelle de temps. Cette première partie a été assez facile car les nombreuses options de *FullCalendar Scheduler* permettaient un paramétrage précis.

1 Voir glossaire

J'ai ensuite ajouté des critères qui permettaient de ranger les éléments dans une sorte de tableau. Ces critères se situent à gauche du calendrier. Selon les termes de *FullCalendar Scheduler*, les critères s'appellent des ressources et les éléments sont des évènements. Dans le cas du planning de réservation, ces ressources sont les UH. Ainsi, pour chaque UH, on peut y associer les réservations et les blocages sous forme d'évènements.

L'ensemble des ressources et évènements peuvent être construits de plusieurs manières. J'ai choisi d'appeler une méthode PHP pour renvoyer les listes correspondantes formatées en JSON. Ce choix m'a permis d'effectuer les traitements nécessaires côté serveur et donc de ne renvoyer que les informations utiles.

L'*Illustration 18* montre le planning dynamique. On y voit à gauche les différentes UH d'une résidence et en haut les jours. Les réservations se trouvent au centre et sont ainsi associées à une UH sur un intervalle de date.

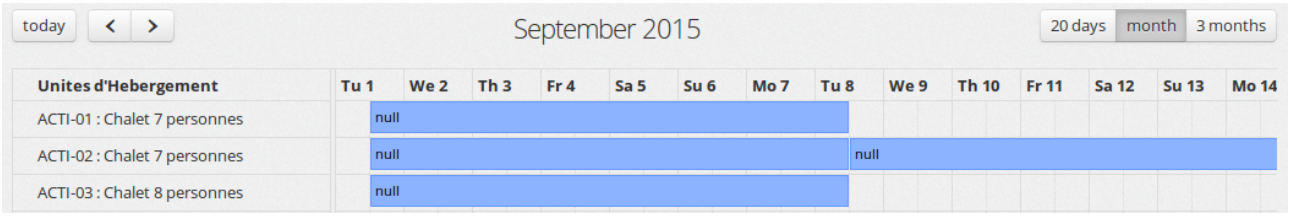


Illustration 18: Planning dynamique

L'*Illustration 19* présente le planning des disponibilités par type d'hébergement. Il se lit de la manière suivante: pour la nuit du 5 au 6, il y a 0 chalet 8 personnes disponibles sur un total de 1 chalet présent dans cette résidence. Comme les nombres négatifs l'indiquent, il peut y avoir des disponibilités négatives. La raison de ce comportement est expliquée plus loin dans ce rapport.

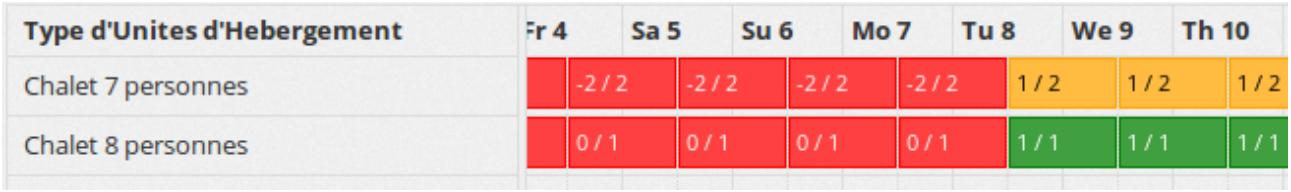


Illustration 19: Planning des disponibilités par type d'UH

2.5.4) Compte de taxes et export comptable

Ma dernière tâche a été de réaliser un export comptable. En effet, il était prévu que le client exporte des données comptables et financières selon un format bien particulier. Bien que n'ayant pas d'informations sur l'utilisation future de ces données, je me doutais grâce à mes connaissances venant de DUT que cela servirait à construire des bilans et des comptes de résultats.

La description du fichier par le client a imposé des modifications conséquentes en BDD. Ces modifications ont pris place dans tous les menus faisant état d'un paiement quelconque. Cela concernait donc bien sûr les réservations, mais également les notes PMS. Dans ces différents menus, lors de l'encaissement d'un montant, il a fallu insérer des données supplémentaires en BDD pour retenir qu'un montant A concernait le paiement d'une taxe B/d'une prestation C, etc. On peut analyser ci-dessous un exemple complet d'un règlement d'une note PMS, en deux fois :

- la ligne orange représente la facture, avec le montant total à payer, la date d'échéance...
- les deux lignes jaunes représentent deux choses différentes. La première ligne jaune (au débit) correspond à l'encaissement du paiement et vient "équilibrer" le total des lignes blanches qui sont inscrites au crédit. La seconde ligne vient "annuler" la dette du payeur par l'enregistrement du paiement de sa facture.
- les lignes blanches représentent les données stockées dans la nouvelle table. Elles vont être créditées du montant leur correspondant : 9.45€ de montant HT pour la prestation bed & breakfast, 0.55€ de TVA sur cette prestation, 10.395€ pour le montant HT de la prestation demi-pension et 0.605€ de TVA sur cette prestation. La sommes de ces 4 montants correspond bien au montant dû et payé par le client.

Si la note PMS avait été réglée en plusieurs fois, il y aurait eu plusieurs paires de lignes jaunes et plusieurs quatuors de lignes blanches mais toujours une seule ligne orange.

Num Transac	Num Note	Date Transac	Num compte	Nom Etab	Type Transac	Num Dossier	Num piece	Occupant	Descriptif generique	Mnt debit	Mnt credit	Dte echeance	Num resa
12	4	20-08-2015		Hotel Bikay		0			Collectif Client	21		20-08-2015	0
12	4	20-08-2015	CB	Hotel Bikay	Solde	0			Collectif Client	21		20-08-2015	0
12	4	20-08-2015		Hotel Bikay	SOLDE	0			Collectif Client		21	20-08-2015	0
12	4	20-08-2015	71040	Hotel Bikay	SOLDE	0			TVA-5.5		0.55	20-08-2015	0
12	4	20-08-2015		Hotel Bikay	SOLDE	0			Bed and Breakfast		9.45	20-08-2015	0
12	4	20-08-2015	71040	Hotel Bikay	SOLDE	0			TVA-5.5		0.605	20-08-2015	0
12	4	20-08-2015		Hotel Bikay	SOLDE	0			Demi pension		10.395	20-08-2015	0

On peut remarquer que certaines cases de certaines lignes ne sont pas remplies. En effet, le tableau doit prendre en compte tous les cas possible de paiement. Dans le cas pris en exemple, à savoir le paiement d'une note PMS, les informations de la réservation peuvent ne pas être connues.

Lorsque j'ai eu fini cette fonctionnalité, j'ai achevé et complété toutes les tâches qui m'avaient été confiées. Bien sûr, il a fallu retravailler plusieurs modules pour y mener des corrections ou des modifications. Dans le même temps, j'ai repris la fonctionnalité des réservations commencée par Thomas et qui finissait son stage fin juillet.

2.6) Poursuite des réservations et phase de tests

2.6.1) Reprise des réservations

La fonctionnalité concernant les réservations a été développée par Thomas. Malheureusement, son stage se finissait fin juillet et la fonctionnalité n'était pas achevée. Il m'a donc expliqué ce qu'il avait fait, comment il l'avait fait et ce qu'il restait à faire pour que Mehdi et moi puissions terminer ce module dans de bonnes conditions.

La prise de réservation représente le module central de l'application. Pour pouvoir prendre une réservation, il faut bien sûr des résidences et des UH, mais également des conventions liées à des partenaires, des prestations, des taxes, etc.

Le principe de base de la prise d'une réservation est le suivant : dans un premier temps, il faut sélectionner les dates de la réservation, puis le partenaire ou le client individuel avec lequel la réservation s'effectue. Lorsque ces différentes informations sont renseignées, la convention correspondante est chargée en AJAX. Il est alors possible de sélectionner une ou plusieurs UH à insérer dans la réservation. De plus, l'opérateur peut renseigner la prestation de base pour chaque UH et une liste de prestations complémentaires. Chacune de ces prestations non fixes peut être reliée à une UH en particulier, ou non.

Pour mieux comprendre, voici un exemple complet, étape par étape :

- la société SNCF prend contact avec Azerty pour obtenir une réservation du 15 au 31 décembre 2015 dans la résidence A.
- En chargeant la convention de ce partenaire avec les dates demandées, l'opérateur prenant la réservation propose à la SNCF l'ensemble des UH présents dans la résidence et disponibles selon les termes de la convention établie précédemment.
- La SNCF choisit un certain nombre d'UH et la prestation de base pour chacune des UH.
- Enfin, elle choisit une liste de prestations à inclure dans la réservation.

L'ensemble des informations concernant les paiements des UH et des prestations - dates, payeur… - les conditions d'annulation et de remboursement est contenue dans la convention avec laquelle la réservation a été prise.

J'ai tout d'abord amélioré le côté graphique de cette fonctionnalité, puis ai procédé à de multiples corrections au fur et à mesure que le client ou nous même remarquions des anomalies. La sous fonctionnalité concernant les prestations et leur paiement comportait plusieurs erreurs. J'ai donc passé plusieurs jours à faire toutes les vérifications, à faire de nombreuses corrections et modifications, à prendre en compte les données des conventions et bien sûr à prendre en compte les remarques du client.

2.6.2) Délogement externe

Dès le début du projet, le client nous avait précisé qu'il était possible de forcer une réservation : lorsqu'une UH n'est plus disponible pour une période donnée, l'opérateur prenant la réservation peut quand même décider de prendre la réservation. Bien évidemment, cela pose un problème de disponibilité à l'arrivée des clients. Pour y faire face, le client a inclus dans la note de cadrage une fonctionnalité intitulée délogement externe.

Le délogement externe est une technique consistant à déplacer une réservation ou une partie d'une réservation d'une UH vers une autre dans le cas où plusieurs réservations prendraient place aux mêmes dates sur une même UH. Il y a 3 cas possible de délogement dans l'application :

- délogement d'une réservation vers une UH appartenant à une résidence d'Azerty. Il suffit de changer la résidence si besoin, l'UH et de laisser la possibilité de changer le prix.
- délogement d'une réservation vers une résidence n'appartenant pas à Azerty et l'acompte a déjà été payé. Dans ce cas, c'est toujours Azerty qui percevra les paiements et qui se chargera de les transmettre à la société possédant la résidence qui accueille la réservation.
- délogement d'une réservation vers une résidence n'appartenant pas à Azerty et l'acompte n'a pas déjà été payé. Azerty ne prend plus en charge les paiements et tout est transféré à la société possédant la résidence qui accueille la réservation. Dans le système, la réservation est passée à un nouveau status, les échéances de règlement et les prestations de cette réservation sont tout simplement supprimées.

J'ai développé cette fonctionnalité lorsque je n'avais pas de modifications à faire sur le module de réservation et y ai donc travaillé par intermittence durant une semaine.

Les 10 derniers jours de stage ont été l'occasion de corriger à nouveau de multiples bugs, erreurs, comportements faux ou fautes d'orthographe dans l'ensemble de l'application et particulièrement sur le module des réservations.

2.7) Bilan

L'application est devenue très complexe au fil du temps. Alors que la note de cadrage était parfois vague, le client nous a détaillé par la suite chacune des fonctionnalités. Comme mentionné précédemment, cela a entraîné de très nombreux changements en BDD. De plus, l'estimation des durées pour réaliser les tâches s'est retrouvée la plupart du temps sous-estimée. C'est la raison pour laquelle la phase de test a été beaucoup plus courte que prévu et pour laquelle nous avons dû nous réorganiser plusieurs fois.

L'*Annexe 2 : Récapitulatif des fonctionnalités – fin du projet* présente toutes les fonctionnalités et les différentes tâches des développeurs. En comparant avec l'*Annexe 1*, il est facile de remarquer que les tâches n'ont pas toujours été réalisées par le développeur qui en avait initialement la responsabilité.

De plus, même lorsque nous pensions comprendre certains points, il nous est arrivé de nous tromper et de devoir modifier des éléments parce que nous avions mal compris les explications du client. J'ai à ce titre noté l'inconvénient de communiquer par Internet et de ne pas avoir d'explications en face à face.

Comme lors de tout apprentisage, j'ai rencontré pas mal de problème technique que j'ai parfois eu du mal à résoudre. Ne sachant pas comment formuler mon problème, Internet m'a parfois été peu utile et j'ai régulièrement demandé de l'aide à mes collègues durant les premières semaines pour me dépanner.

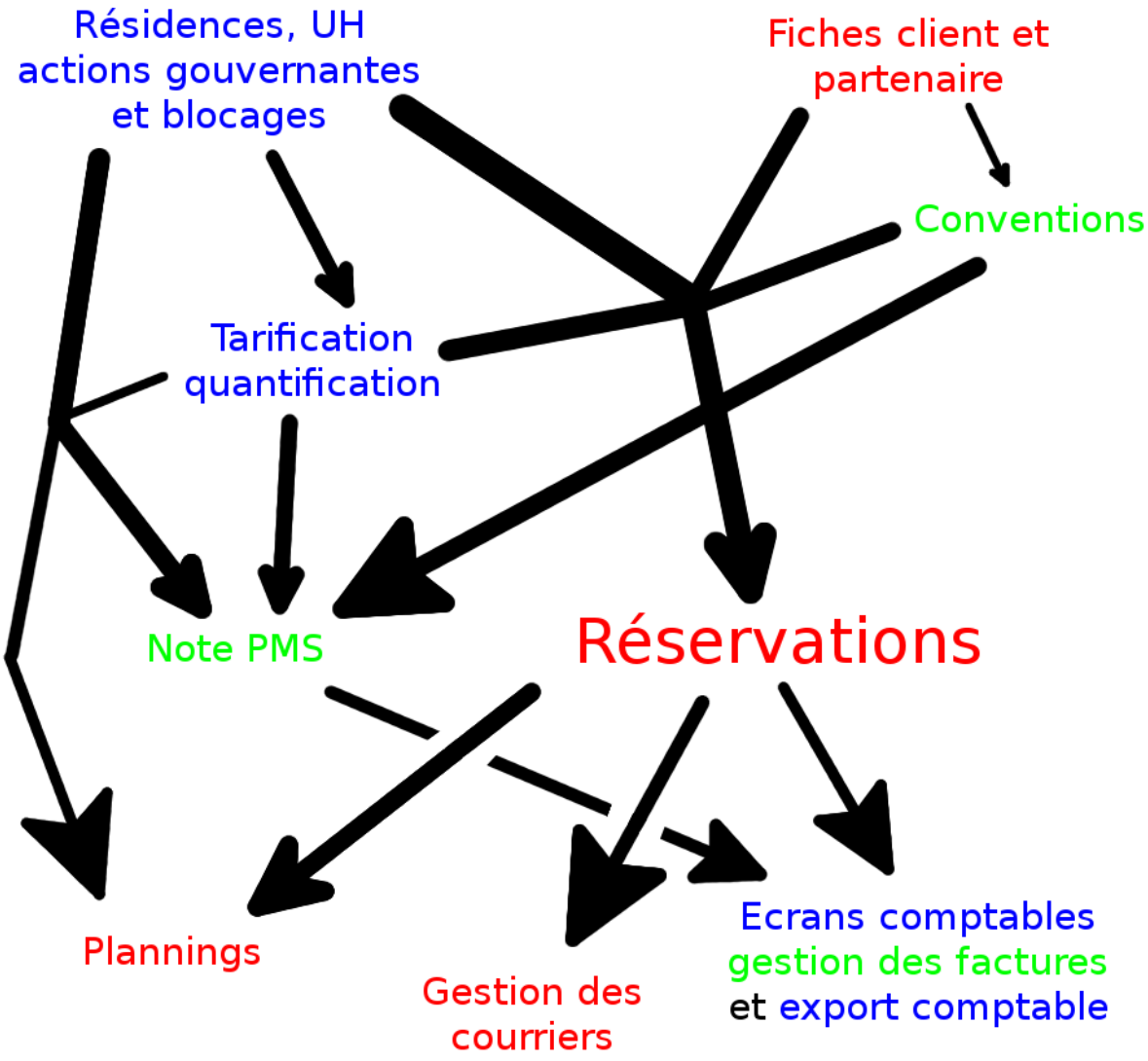
Conclusion

Mon expérience au Cambodge a été très enrichissante. J'y ai découvert une nouvelle culture et un autre mode de vie. Dans l'entreprise BiKay, j'ai travaillé sur un projet de gestion de résidences locatives et d'hébergements dans une équipe de 4 personnes. Au sein de l'application, j'ai développé les fonctionnalités concernant les résidences, les plannings ou encore la comptabilité.

Le stage a été une très bonne expérience puisque je souhaitais découvrir la programmation web. Cela m'a permis de découvrir de nombreuses technologies qui m'étaient inconnues comme PHP, JavaScript ou JQuery. De plus, j'ai été très impliqué dans les différentes phases du projet comme la modélisation ou la relation avec le client, et ai pu avoir un bon aperçu de chacune d'entre elles.

Au fur et à mesure que le projet avançait, je me suis senti à l'aise avec les différentes technologies sur lesquelles j'ai travaillé. Je souhaite dorénavant approfondir mes connaissances en programmation web en travaillant avec d'autres langages, frameworks ou librairies. J'envisage désormais de travailler dans ce domaine au début de ma carrière professionnelle avant de bifurquer vers l'administration de base de données.

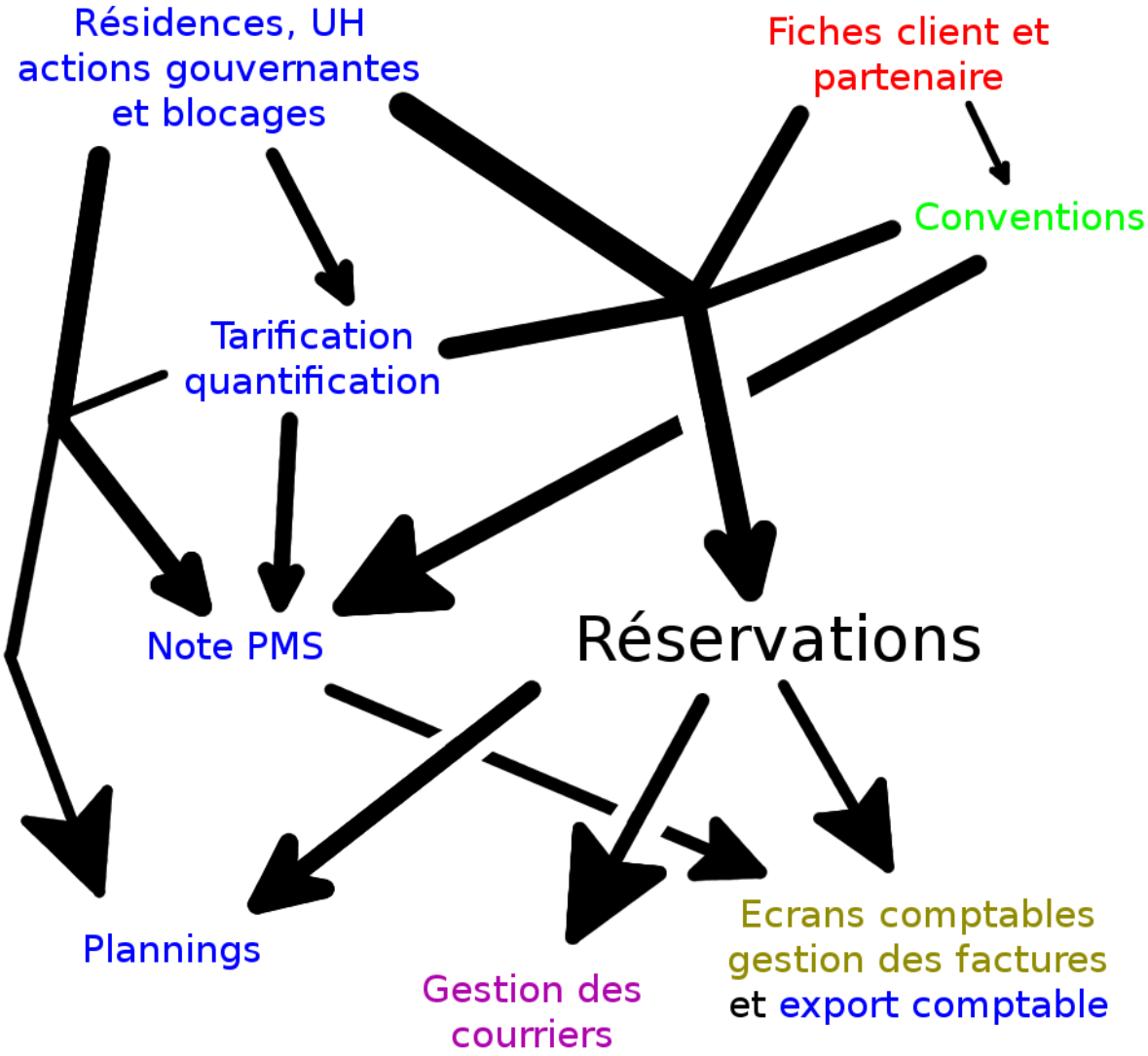
Annexe n°1 :
Récapitulatif des fonctionnalités



Cette annexe présente les différentes fonctionnalités et leurs dépendances. Par exemple, pour pouvoir réaliser le module de gestion des courriers, il fallait d'abord que les fonctionnalités concernant les réservations soient réalisées.

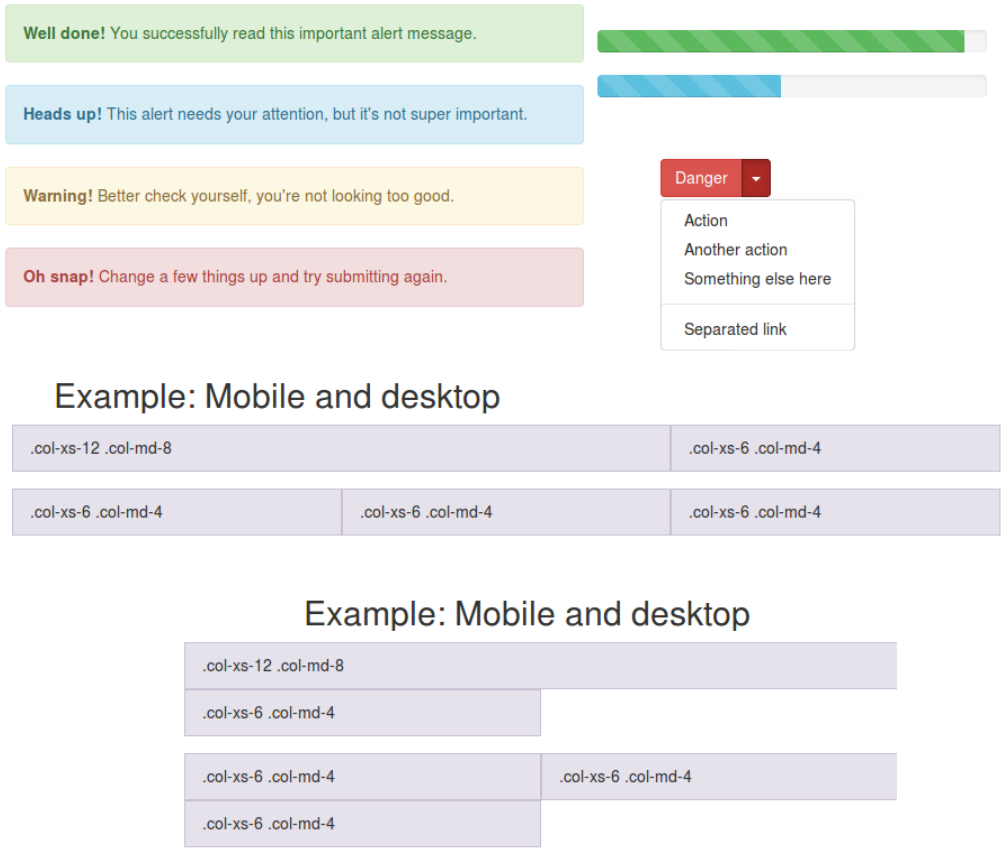
Les fonctionnalités en **vert** sont celles dont Mehdi avait initialement la charge. Celles en **rouge** sont celles de Thomas, tandis que j'avais la charge des modules représentés en **bleu**.

Annexe n°2 : Récapitulatif des fonctionnalités – fin du projet



Comme mentionné dans ce rapport, j'ai eu la chance d'être en avance sur plusieurs de mes tâches. C'est ainsi que j'ai développé en plus les fonctionnalité concernant les notes PMS et les planning. Les écrans comptables dont j'avais la charge ont été réalisé par Charles, qui a ensuite repris la fonctionnalité des courriers, non finie par Thomas. Enfin, les réservations n'étaient également pas finies par Thomas. Mehdi et moi avons travaillé sur ce module pour le compléter.

Annexe n°3 :
Présentation de Bootstrap



En première partie d'annexe se trouvent plusieurs composants inclus avec Bootstrap : des messages d'informations, d'avertissements ou d'erreurs, des barres de chargement ou des menus déroulants.

La deuxième partie présente le système de grille de Bootstrap : grâce à ce système, tout élément HTML est découpé en 12 colonnes. Par la suite, il est possible de spécifier la taille de chacun des éléments fils en nombre de colonnes. Cela permet de gérer facilement la taille des éléments HTML. Le second avantage de cette technique vient du fait que le nombre de colonnes peut être différent selon la taille de l'écran :

- dans le premier *Example : Mobile and desktop*, le navigateur est affiché sur un écran de taille moyenne. Ainsi, ce sont les *col-md* qui sont pris en compte pour l'affichage (*md* pour medium).

- dans le second *Example : Mobile and desktop*, le navigateur est affiché sur un écran de très petit taille, comme un smartphone. Dans ce cas, ce sont les *col-xs* qui tons pris en compte pour afficher les éléments (*xs* pour extra-small).

Glossaire

Back-end: Partie de l'application ou du programme s'occupant des traitements des données

Classe : Représente l'ensemble des propriétés d'un objet. Contient des attributs représentant leurs états et des méthodes pour modéliser leur comportement.

Framework : Ensemble de composants logiciels qui sert à structurer à créer les fondations d'une application.

Front-end : Partie de l'application ou du programme s'occupant des affichages à l'écran

MVC : (Modèle Vue Contrôleur). Patron de conception qui consiste à séparer les données (modèle), l'affichage (vues) et les actions entre ces 2 entités (contrôleur).

Plugin : Paquet qui complète un logiciel ou un programme en lui ajoutant des fonctionnalités supplémentaires.

Site dynamique : contrairement à un site statique, un site dynamique est généré à la demande en fonction d'informations diverses et variées : heure, lieu, données d'authentification…

Table pivot: Table permettant reliant 2 autres tables au moyen de relations 1,N.

Bibliographie

Site internets :

https://openclassrooms.com/
Nommé auparavent *lesiteduzero.com*. Site internet sur l'informatique et la programmation avec tutoriels et forums.

php.net
Site officiel de PHP, avec la documentation, les dernières versions, etc. Je m'en suis beaucoup servi pendant mon stage, ne connaissant pas du tout le PHP.

laravel.com
Site officiel de Laravel, avec la documentation, l'API, etc.

https://api.jquery.com/
Site de l'API de JQuery, très utile pour apprendre les sélecteurs et toutes leurs possibilités.

stackoverflow.com
"Question and answer site for professional and enthusiast programmers". Mon site favori pour me débloquer pendant mon stage.

www.w3schools.com
Site d'information pour l'apprentissage du développement web. Contient des tutoriels pour le HTML, CSS, JavaScript, PHP...