Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Filière Télécommunication - Internet et Communication

PROJET DE SEMESTRE 5
2019-2020

BerraRegion

Création d'une extension client basée sur Yoplanning

Auteur

Pittet Raphael
Rumo Célestin

Superviseur

Ayer Serge
Scheurer Rudolph

Version 1.0

Fribourg, Le 30 janvier 2020

Hes·so
Haute Ecole Spécialisée
de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts
Western Switzerland

Version	Auteur	Description des changements	Date
0.1	Raphaël	Analyse du logiciel de réservation	22.10.2019
0.2	Raphaël	Analyse du système actuel	24.10.2019
0.3	Raphaël	Analyse de L'ERP	25.10.2019
0.35	Célestin	Analyse de PSP + début analyse du logiciel comptable	25.10.2019
0.36	Célestin	2.4 : Ajout contraintes et libertés + 2.4.2 : Modification + 2.5.2 : Agrandissement	30.10.2019
0.37	Célestin	Correction de l'orthographe et de la syntaxe du rapport	02.11.2019
0.38	Célestin	2.1.1 : changement figure 1 + 2.1.2 : Changement figure 2	03.11.2019
0.39	Célestin	2.6, 2.7 : Ajout	03.11.2019
0.40	Raphaël	2.1.2 : ajout Diagramme	12.11.2019
0.41	Raphaël	2.1 : restructuration	17.11.2019
0.42	Raphaël	2.4 : restructuration	20.11.2019
0.43	Célestin	2.3.2, 2.3.4, 2.4.5 et 2.7 : Modifications	25.11.2019
0.44	Raphaël	2.5.* : Modifications	25.11.2019
0.43	Célestin	2.9 : Ajout	26.11.2019
0.5	Raphaël	3.0, 3.1 : Ajout	27.11.2019
0.51	Raphaël	3.1.1 , 3.1.2 : Ajout	28.11.2019
0.51	Célestin	3.1.3 , 3.1.4 : Ajout	28.11.2019
0.52	Célestin	2.3 , 2.4 : Modification	03.12.2019
0.53	Célestin et Raphaël	3.1 : Modification, 3.2 : Ajout	04.12.2019
0.60	Raphaël	4 - 4.1.5 : Ajout	09.12.2019
0.61	Célestin	4.1.7-9 : Ajout	09.12.2019
0.62	Raphaël	4.1.6 : Ajout	15.12.2019
0.63	Raphaël	4.2-4 : Ajout	03.01.2020
0.64	Raphaël	4.2.5-10 : Ajout	25.01.2020
0.65	Raphaël	4.4.1-3 : Ajout	26.01.2020
0.70	Célestin	4.3.1-3 : Ajout	26.01.2020
0.71	Célestin	4.3.4-8 : Ajout	27.01.2020
0.72	Raphaël	5-5.1 : Ajout	27.01.2020
0.80	Raphaël	6 : Ajout	28.01.2020
0.82	Célestin	4.3.8 : complétion	28.01.2020
0.82	Célestin	4.3.9 : Ajout	28.01.2020
0.82	Célestin	5.1-5.3-5.4 : Ajout	28.01.2020
0.9	Raphaël	Correction de l'orthographe	28.01.2020
0.99	Célestin et Raphaël	Correction générale	29.01.2020
1.0	Célestin et Raphaël	8 : Ajout	29.01.2020

Table des matières

1 Introduction **5**

2 Analyse **5**
2.1 Situation actuelle 5
2.1.1 Yoplanning sales online plugin 7
2.1.2 Logiciel de réservations 8
2.1.3 Logiciel de comptabilité : 10
2.2 Situation idéale 11
2.3 Moyens de paiement 11
2.3.1 Analyse de la situation en Suisse 11
2.3.2 Solutions de paiement online 12
2.3.3 Calculs de coûts pour la Berra 15
2.3.4 Matrice de décision du Paiement Service Provider 17
2.3.5 Solution de point of sale 18
2.4 Logiciel de réservation 20
2.4.1 Critères 20
2.4.2 Yoplanning 22
2.4.3 SimplyBook.me 23
2.4.4 Bookeo 23
2.4.5 Prise de décision du logiciel de réservation 24
2.5 ERP 24
2.5.1 Critères 25
2.5.2 Odoo 26
2.5.3 Dolibarr 27
2.5.4 Tryton 28
2.5.5 Matrice de décision de L'ERP 28
2.6 Logiciel de comptabilité 29
2.6.1 Les critères 29
2.6.2 Les logiciels 30
2.7 Problèmes rencontrés 31
2.8 Conclusion analyse 31
2.9 Priorisation pour la suite du projet 31

3 Conception **32**
3.1 Workpackages 32
3.1.1 Workpackage 1 - Mise en place de l'environnement 32
3.1.2 Workpackage 2 - Authentification 33
3.1.3 Workpackage 3 - Traitement des données de YoPlanning à Odoo 34
3.1.4 Workpackage 4 - l'affichage des données dans le compte client 35
3.2 Planification 36

4 Développement **37**
4.1 Workpackage 1 - Mise en place de l'environnement 37
4.1.1 Equipement 38

4.1.2 Jalon 1 - Installation de la base de données 38
4.1.3 Jalon 1 - Installation du server Odoo 38
4.1.4 Jalon 1 - Configuration d'Odoo 40
4.1.5 Jalon 1 - Remarque 41
4.1.6 Jalon 2 - Analyse des modules disponibles 42
4.1.7 Jalon 2 - Analyse de la structure des données d'Odoo 44
4.1.8 Jalon 3 - Mise en place d'un serveur web 45
4.1.9 Jalon 3 - Installation de Wordpress 47
4.1.10 Jalon 3 - Importation d'une copie du site internet de l'ESS la Berra 47
4.2 Workpackage 2 - Authentification 49
4.2.1 Jalon 4 - Mise en place 49
4.2.2 Jalon 4 - Installation de l'environnement local 49
4.2.3 Jalon 4 - Diagrammes de séquence 49
4.2.4 Jalon 4 - Connection via l'API 52
4.2.5 Jalon 4 - Mise en place du formulaire d'inscription 53
4.2.6 Jalon 4 - Création des nouveaux utilisateurs via l'API 54
4.2.7 Jalon 5 - Conception des formulaires 56
4.2.8 Jalon 5 - Mise en place du frontend 58
4.2.9 Jalons 4 et 5 - Ajout d'une page personnalisée sur WordPress 58
4.2.10 Jalons 4 et 5 - Tests 65
4.3 Workpackage 3 - Traitement des données de YoPlanning à Odoo . . . 67
4.3.1 Préparation de l'environnement de travail 67
4.3.2 Jalon 6 - l'API de YoPlanning [43] est testée pour voir les données reçues 69
4.3.3 Jalon 6 - le problème de la synchronisation entre les logiciels est expliqué 70
4.3.4 Jalon 7 - Le choix de la technologie utilisée est décrit 71
4.3.5 Jalon 7 - Première importation des données présentes dans YoPlanning 73
4.3.6 Jalon 7 - Gestion de la synchronisation entre les deux systèmes 73
4.3.7 Jalon 8 - Test 76
4.3.8 Workpackage 3 - problèmes 78
4.3.9 Workpackage 3 - améliorations du code 79
4.4 Workpackage 4 - Affichage des données dans le compte client 80
4.4.1 Jalon 9 - Analyse des modèles 81
4.4.2 Jalon 10 - Diagrammes de séquence 81
4.4.3 Jalon 10 - Mockup de l'affichage des paiements 82
4.4.4 Jalon 11 - Réalisation des requêtes en Backend 83
4.4.5 Jalon 11 - Réalisation du frontend 84
4.4.6 Jalons 10 et 11 - Intégration de la page à WordPress 85
4.4.7 Jalon 10 et 11 - Tests 86

5 Résultats et améliorations possibles 88
5.1 Résultat 88
5.2 Création de module Odoo 89

5.3 Plugin WordPress 89
5.4 Mise en place d'un site internet 89
5.5 Meilleure collaboration avec les partenaires 89

6 Conclusion **91**

Déclarations d'honneur **92**

7 Versions **93**

Glossaire **99**

Acronymes **100**

1 Introduction

Dans le cadre du développement des activités dans la région de la Berra, les différents partenaires recherchent une solution de réservation en ligne prenant en charge

- les comptes clients
- les ventes, remboursements, bons et rabais
- liaisons avec logiciel de comptabilité pour toutes les transactions

La solution utilisée actuellement par l'école de ski (Yoplanning)[1] ne comprend malheureusement pas toutes ces fonctionnalités et ne permet pas de mettre à exécution le plan marketing[2] élaboré car les fonctionnalités requises sont absentes. C'est pourquoi, nous avons été mandatés par BerraRegion pour de mettre en place une solution répondant aux besoins du groupe. Dans ce travail, nous devrons définir quels outils sont les plus adéquats pour la mise en place d'un tel système.
Après avoir réalisé cette tâche importante, nous passerons au développement d'une solution respectant les contraintes et objectifs contenus dans le cahier des charges.

2 Analyse

Une première phase d'analyse est demandée afin de connaître spécifiquement l'environnement en place actuellement ainsi que les données qui seront exploitées. Cette section est divisée en plusieurs sous-chapitres, c'est-à-dire un premier chapitre pour l'analyse de la situation actuelle et celle souhaitée. Ensuite les analyses porteront sur la partie du logiciel de réservation, l'ERP[3], et les deux derniers chapitres seront destinés au logiciel de comptabilité et à l'outil de paiement.

2.1 Situation actuelle

La figure 1 est un diagramme de composants de la solution actuelle. La seule interaction que le client a sur le site internet du groupe ESS la Berra, est la possibilité de faire une réservation par le biais du Plugin Yoplanning.
Contrairement au client, l'administrateur a énormément de tâches à faire. Tous les acteurs présents sur la figure 1 mis à part le client et l'employé sont des tâches que l'administrateur réalise actuellement. La plupart de celles-ci sont redondantes et représentent une charge de travail inutile et par conséquent une perte de temps.

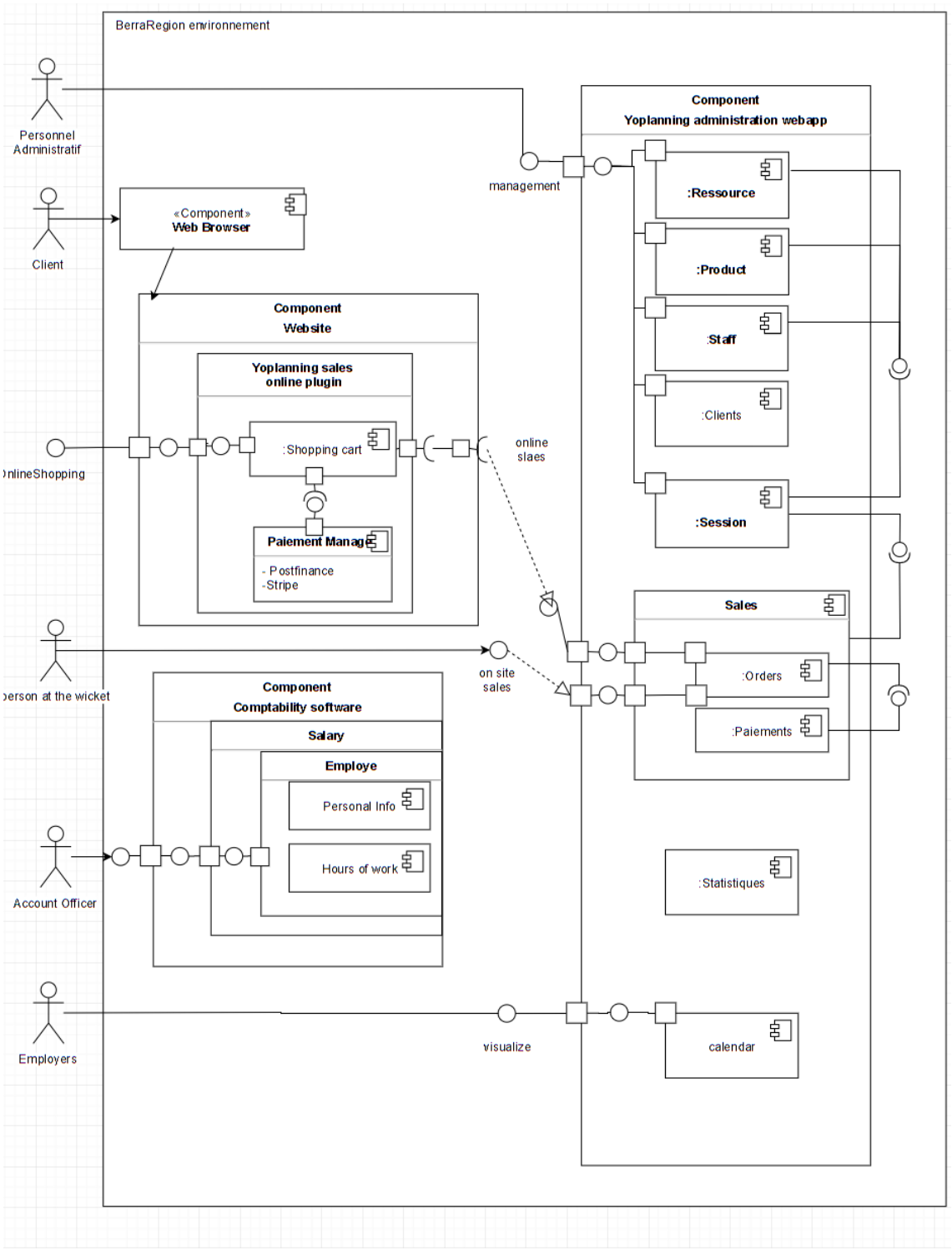


FIGURE 1 – Situation actuelle du groupe BerraRégion

2.1.1 Yoplanning sales online plugin

Le Plugin Yoplanning présent sur le site internet de BerraRégion, est fourni par yoplanning. Il a été créé afin de faciliter et d'accélérer l'intégration et la mise à disposition des produits sur le site de leurs clients.

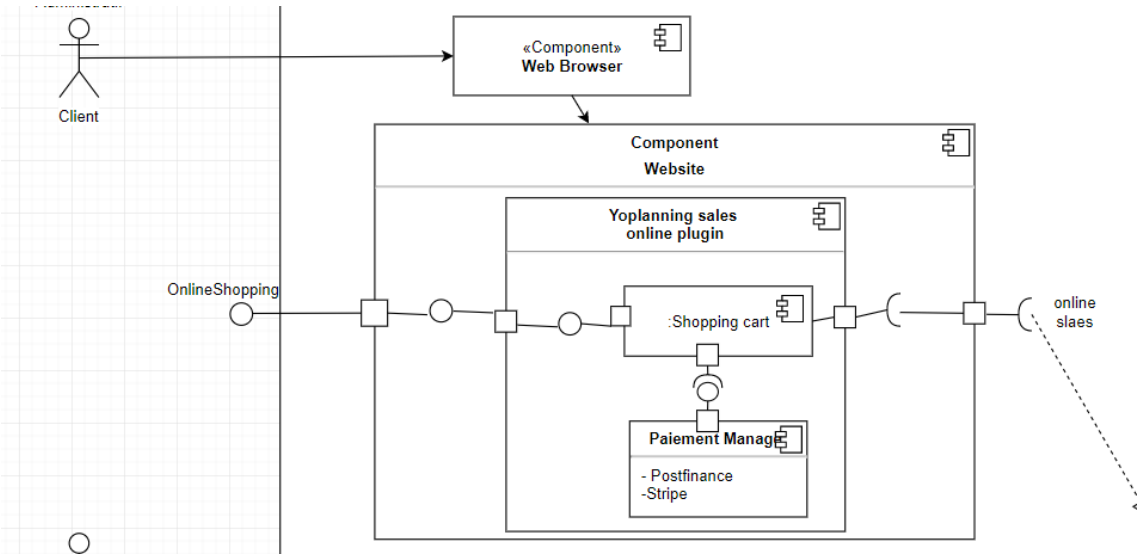


FIGURE 2 – Plugin Yoplanning

Grâce au plugin, l'administrateur n'a pas besoin de coder lui-même des pages internet afin de mettre à disposition les leçons qu'il a créées sur la webApp Yoplanning. De plus, la mise en page est aussi intégrée dans le plugin, ainsi l'administrateur n'a vraiment plus qu'à ajouter le module sur son propre site. Le plugin intègre aussi un manager de paiement afin de permettre à l'administrateur de choisir quel PSP entre PostFinance et Stripe il veut utiliser.

Le Plugin fonctionne grâce à une API de type RESTful publique fournie par Yoplanning[4]. Toutes les fonctions sont donc réalisables sans passer par le plugin à l'exception du manager de paiement qui lui est directement intégré dans le plugin.

Actuellement, comme cité plus haut l'ESS la Berra utilise ce plugin ; cependant, le design de celui-ci ne plaît pas au groupe BerraRégion. C'est pourquoi, il souhaite que l'intégration des diverses leçons ne soit plus faite par le biais du plugin, afin de pouvoir les présenter à leur manière.

Stripe est le moyen de paiement utilisé actuellement par l'ESS la Berra. Cette solution est très coûteuse mais permet un remboursement des clients automatisé. En effet même si Yoplanning permet l'intégration de PostFinance, la fonction de remboursement automatisé n'est malheureusement pas encore disponible avec celui-ci.

2.1.2 Logiciel de réservations

Le logiciel de réservation utilisé actuellement s'appelle Yoplanning, il s'agit d'une webapp disponible après l'obtention d'un abonnement payant. Celui-ci est spécialisé pour fournir une solution correspondant au besoin d'école de ski. Mais il convient aussi aux entreprises ayant uniquement besoin de gérer des sessions sans condition particulière.

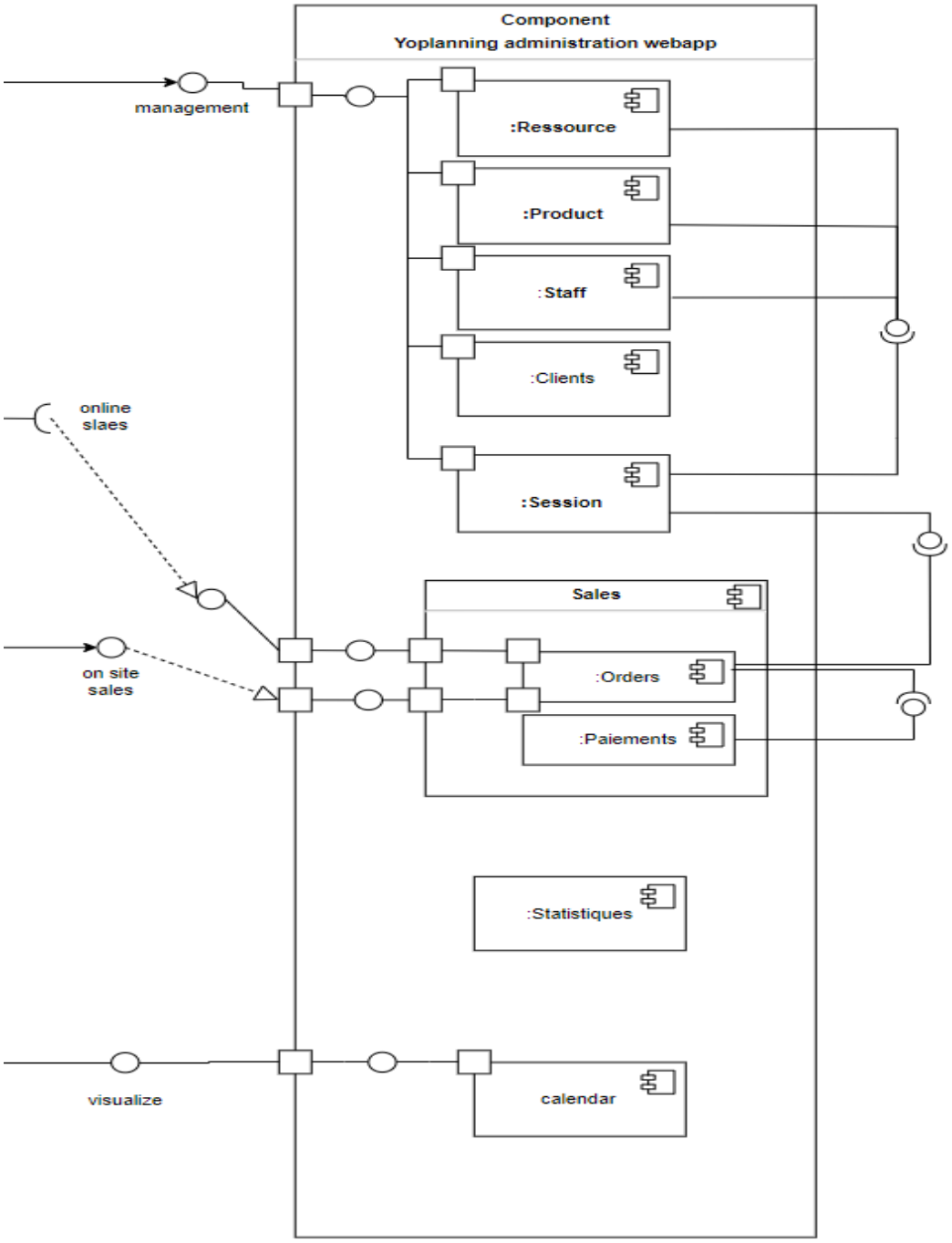


FIGURE 3 – Plugin Yoplanning

Actuellement l'administrateur utilise la webapp pour :

- **Créer des produits :** Les produit sont les squelettes d'une session, ils permettent de définir un format de session particulier.
- **Créer des ressources :** Les ressources sont des forfaits spéciaux comme par exemple un abonnement pour 5 demi-jours.
- **Gérer les employés :** l'administrateur peut actuellement ajouter et modifier un employé, assigner des attributs (par ex : jardin des neiges, bilingue, etc.). Ainsi que leur assigner des sessions.
- **Voir les clients :** Actuellement, l'administrateur n'a qu'une liste sans mémoire des clients qui ont commandé sur le site, ainsi la gestion des clients (par exemple : bon de réduction en fonction des dépenses sur la plateforme) n'est pas disponible.
- **Créer des sessions :** Les sessions sont des instances de produits qui sont agendées et disponibles pour les clients sur le site.
- **Gérer les ventes :** la partie gestion des ventes se divise en deux sous composants :
  - **Commandes :** Dans cette partie, sont listées toutes les réservations qui ont été faites, qu'elles soient déjà payées ou non.
  - **Paiements :** Dans le composant paiement, il y a uniquement les réservations qui ont été payées. L'administrateur a la possibilité de rembourser le client **(car il utilise le PSP Stripe*)**.
- **Voir les statistiques :** La webapp fournit un espace dédié aux statistiques, cependant celles-ci n'ont pas les filtres que le mandant cherche et de ce fait ne sont pas utilisées.
- **Visualiser le planning :** Actuellement, les employés qui ont été ajoutés par l'administrateur, ont la possibilité de se connecter à la webapp pour visualiser leur plan d'occupation.

*Le mandant utilise actuellement Stripe comme solution de paiement pour ses clients. Si un autre moyen de paiement est utilisé la procédure de remboursement est beaucoup plus longue. Tout d'abord, il faut retrouver le paiement, ensuite, à partir de celui-ci il faut chercher le client et ainsi le contacter par email. Dans le mail l'administrateur demande l'IBAN du client afin de faire le paiement de remboursement (cette transaction coûte de l'argent ainsi, chaque remboursement représente une perte d'argent et de temps).

2.1.3 Logiciel de comptabilité :

Le logiciel de comptabilité est Winbiz, un logiciel très utilisé en suisse. Il est utilisé pour entrer les heures des employés à la fin de mois et ainsi générer des fiches salaires pour ceux-ci. Cette tâche est actuellement faite par l'administrateur et représente une vraie perte de temps. L'administrateur doit dans un premier temps générer une feuille Excel qu'il doit tenir à jour durant le mois afin de pouvoir recopier le contenu de celle-ci dans le logiciel de comptabilité.

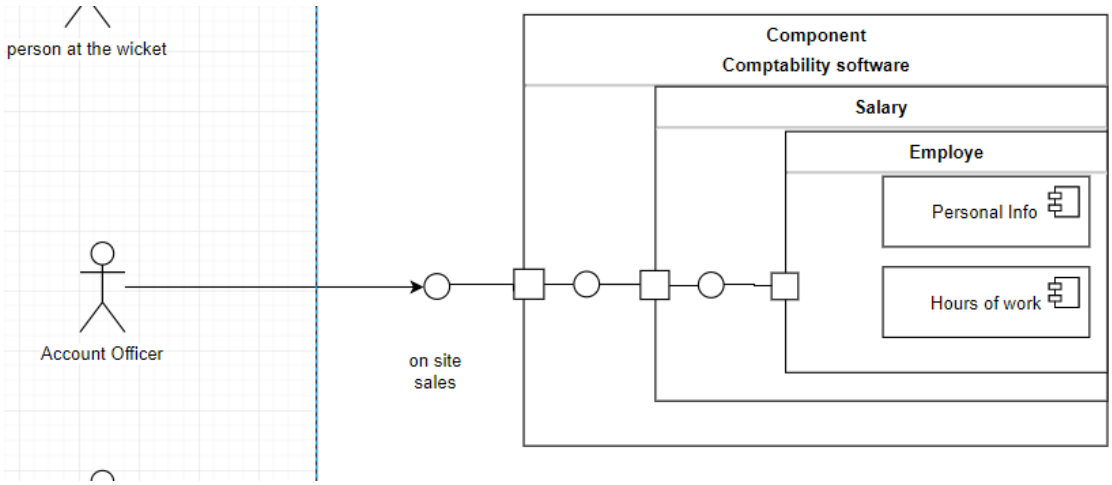


FIGURE 4 – Plugin Yoplanning

De plus lorsqu'un nouvel employé intègre le groupe, l'administrateur doit écrire plusieurs fois les mêmes informations sur les diverses plateformes. Il doit d'abord ajouter l'employé sur la webapp Yoplanning, ensuite il doit aussi l'ajouter sur sa feuille Excel pour la notation des heures, et encore l'ajouter dans le logiciel de comptabilité.

2.2 Situation idéale

Dans un idéal, un seul logiciel suffit à répondre au besoins du projet. Biensûr, il existe des logiciels très complets qui ont la capacité de le faire. SAP[5] fait partie de ceux-ci. SAP est un ERP très utilisé par les grosses entreprise comme American Airlines, Microsoft france, ou aussi la Poste. Cependant, le prix d'une licence utilisateur peut varier entre 2500.- et 5000.- suivant les modules implémentés.
Une telle solution n'est pas envisageable pour le projet, premièrement car celle-ci ne respecte pas la contrainte de prix, mais de plus le groupe BerraRegion n'est pas assez développé pour avoir besoin d'un tel progiciel.

Le but est donc de travailler avec différents composants et de les lier afin de n'avoir pas à retranscrire tout type de données. Le diagramme de composant de la solution souhaitée est disponible dans les annexes[6] car elle est trop grande et ne peut donc pas être imprimée directement dans le rapport.
Afin d'expliquer l'objectif final, chaque partie de ce diagramme de composants sera insérée dans les parties adéquates.

2.3 Moyens de paiement

Les moyens de paiement sont les systèmes qui sont mis en place afin de permettre aux clients de finaliser l'achat de produit(s), que cela soit fait online en achetant directement sur le site internet de l'école de ski ou au bureau de manière physique avec un terminal de paiement ou en cash. Deux catégories de paiement sont donc à analyser :

1. Solution de paiement online
2. Solution de Point of sale

Contraintes

- La solution de paiement online doit être simple pour la mise en place
- Les coûts de mise en place doivent être en dessous de 1000 CHF
- La solution doit proposer des moyens de paiement sans devoir passer de contrats supplémentaires avec tous les fournisseurs de cartes
- La connexion du point of sale doit pouvoir être intégrée directement au logiciel de réservation
- La carte PostFinance doit être acceptée pour le paiement online et pour le point of sale

Libertés La solution de paiement online peut être fournie séparement de la solution de Point of sale.

2.3.1 Analyse de la situation en Suisse

Afin de réaliser une analyse pertinente, il est important de prendre en compte les habitudes de paiement des suisses. Les résultats de l'enquête de la banque nationale suisse à ce sujet sont exploités afin de définir les moyens de paiement à offrir afin de

satisfaire la population hélvètique. Il est intéressant de se pencher sur les résultats mis en avant par le communiqué de presse [7] de la BNS (et ces quelques points suivants) :

— Le choix du moyen de paiement est influencé par le montant à payer. En dessous de 50 CHF, le numéraire ainsi que les applications de paiement sont utilisés. Entre 50 et 200 CHF la carte de débit est priviligiée tandis qu'à partir de 200 CHF la carte de crédit est préférée.
— Les applications ou le paiement sans contact sont très peu utilisés. Plus de 50 % des interrogés ont dit vouloir continuer à payer en liquide aussi souvent qu'aujourd'hui.
— La carte de débit (Maestro ou PostFinance) est de loin le moyen de paiement sans numéraire le plus utilisé en Suisse. (22%).
— Les personnes de plus de 55 ans ou à faible revenu privilégient le numéraire tandis que les ménages avec un revenu élevé et les personnes âgées entre 15-34 ans privilégient plus souvent les paiements par carte.

La répartition des transactions pour les achats en ligne a été analysée et comparée par moyens et lieux de paiement dans le rapport de la BNS [8].

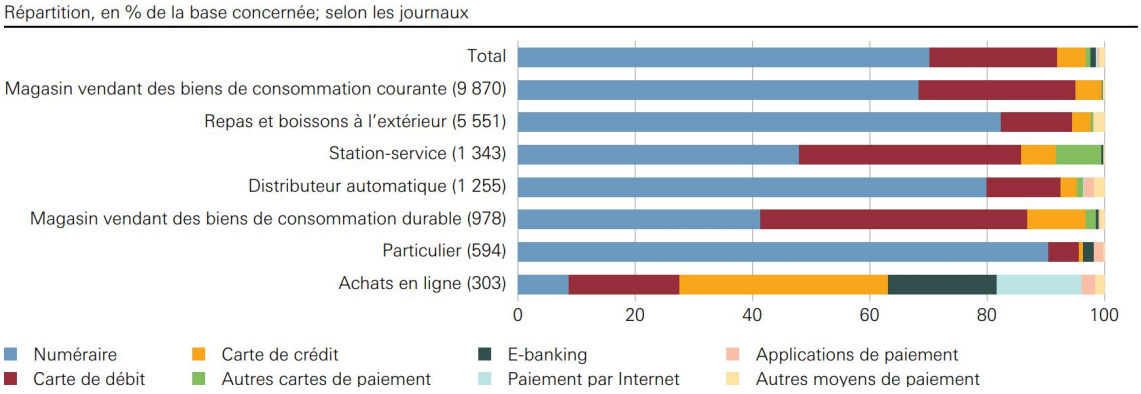


FIGURE 5 – Graphique représentant la répartition des transactions par moyen et lieu de paiement

2.3.2 Solutions de paiement online

La figure 5 permet de mettre en avant les différents moyens de paiement utilisés en Suisse. Ainsi nous pouvons définir les moyens de paiement à implémenter afin de proposer une solution pour la plus grande majorité des clients suisses sans que ceux-ci doivent changer leurs habitudes.

Afin de pouvoir proposer une boutique en ligne, il faut faire appel à un intermédiaire. Il s'agit du Payement Service Provider. Ces Payement Service Provider (PSP) intègrent tous les moyens de paiement usuels à une boutique en ligne. En Suisse, il existe trois principaux Payement Service Provider :

1. DataTrans
2. PostFinance E-payment

3. SIX Payments

Etant donné que le mandant utilise actuellement Stripe, il est important de l'inclure dans la comparaison de solutions de paiement online afin de justifier le changement de fournisseur.

DataTrans
Datatrans est le plus grand fournisseur en Suisse et peut être directement intégré ou alors implémenté dans le site internet via l'API. C'est également la solution préférée des grandes entreprises.
Cette solution est très complète mais coûte très cher à mettre en place : 975 CHF pour l'installation de base + 200 CHF par mode de paiement (minimum 6 dans la situation actuelle) pour un coût de 1200 CHF. L'inconvénient est que le prestataire de service doit passer plusieurs contrats avec les acquéreurs des modes de paiement souhaités et cela rend l'action compliquée.
Il ne respecte pas les contraintes de coûts.

PostFinance E-payment
Postfinance est le numéro 1 en Suisse en matière d'e-finance avec plus d'un million de clients. Il propose tous les modes de paiement couramment utilisés en Suisse tels que PostFinance, Twint, Visa et Mastercard. Il faut cependant passer un contrat supplémentaire avec un acquéreur pour pouvoir réaliser les paiements autres que par PostFinance card. Il y a des frais fixes d'installation ainsi que des frais mensuels.

SIX Payments
SIX payments est un fournisseur de paiement et également un acquéreur qui propose d'autres fournisseurs. Il propose tous les moyens de paiement souhaités. Un gros inconvénient est son coût fixe mensuel.

Stripe
Stripe permet les paiements par carte de crédit et il est un facilitateur de paiement, c'est pour cela qu'il n'y a pas besoin de passer de contrat avec des émetteurs de carte de crédit. Il n'y a pas de frais mensuels ni de frais d'installation. En revanche, il ne propose pas de moyens de paiement régionaux tel que les cartes de PostFinance ou Twint. Cela en diminue l'attrait pour la Suisse. Ainsi, il ne respecte pas les contraintes.

Schéma de paiement
Afin de clarifier la situation, voici un schéma des différentes intéractions qui ont lieu lors d'un paiement en ligne entre les différents acteurs. Nous avons donc besoin d'un payment service provider ainsi que de certains acquéreurs afin de pouvoir proposer les différents moyens de paiement que nous souhaitons.

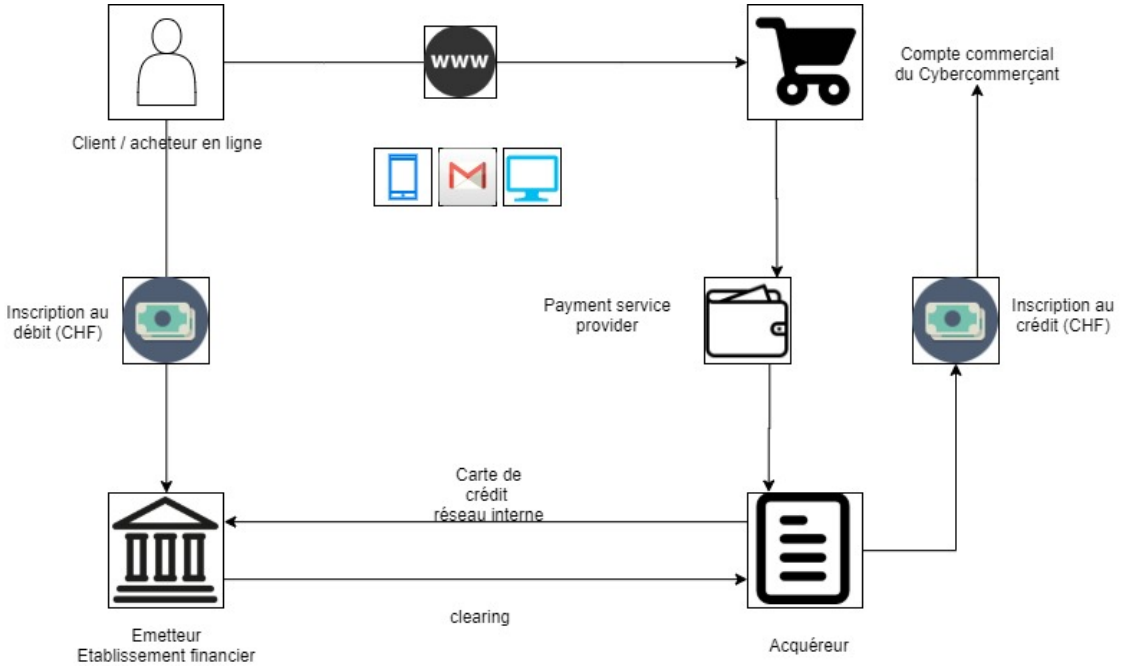


FIGURE 6 – Schéma des interconnections pour un paiement en ligne

Clearing ou compensation en français est un mécanisme permettant à des banques et des institutions financières, membres d'une réunion (la chambre de compensation)) de régler les montants dus et de recevoir les actifs correspondant aux opérations faites en leur nom propre ou pour leurs clients. De fait, une transaction, matérialisée par l'achat d'une part, la vente d'autre part, a toujours un débiteur et un créditeur. [9]

Respect des contraintes

Suite à l'analyse des différents acteurs permettant de faire de l'e-commerce, il apparaît qu'il est impossible de respecter toutes les contraintes. En effet, pour n'avoir pas besoin de réaliser des contrats avec plusieurs acquéreurs différents, le choix se dirigerait vers un facilitateur de paiement tel que Stripe ou SIX payments. Mais Stripe ne propose que des paiements par carte de crédit en suisse.[10] Avec SIX payments, il est également nécessaire de faire un autre contrat avec PostFinance [11] afin d'accepter les cartes de PostFinance.

Redéfinition des contraintes

Suite à ce constat, il est obligatoire de redéfinir les contraintes qui doivent réellement être respectées ainsi que celles qui sont abandonnées. Il a été décidé que les contraintes à respecter étaient les contraintes en terme de coût ainsi que les contraintes en terme de moyens de paiement disponibles.

2.3.3 Calculs de coûts pour la Berra

L'objectif est de proposer le fournisseur de paiement le plus adapté au niveau des coûts et des fonctionnalités disponibles. Pour cela, des critères au niveau des besoins, de la complexité de mise en place, du budget, du nombre de transactions ainsi que de la somme des transactions effectuées par mois doivent être mis en place.
Les valeurs de calculs afin d'estimer le coût de chaque Payement Service Provider sont les valeurs récupérées de l'école suisse de ski la Berra.

	oct.18		nov.18		déc.18	
	Chiffre d'affaire	nb transactions	Chiffre d'affaire	nb transactions	Chiffre d'affaire	nb transactions
STRIPE	3 751.50	8	24 307.00	120	23 567.40	122
VIREMENT					3 590.00	
CB	700.00				2 883.00	
CASH	2 185.00				2 742.00	

	janv.19		févr.19		mars.19	
	Chiffre d'affaire	nb transactions	Chiffre d'affaire	nb transactions	Chiffre d'affaire	nb transactions
STRIPE	44 907.00	357	22 409.00	175	1 127.00	7
VIREMENT	3 874.00		2 297.00		455.00	
CB	29 164.00		15 952.00		4 828.00	
CASH	26 152.00		14 331.00		6 092.00	

FIGURE 7 – Tableau des rentrées d'argent du mandant durant l'hiver

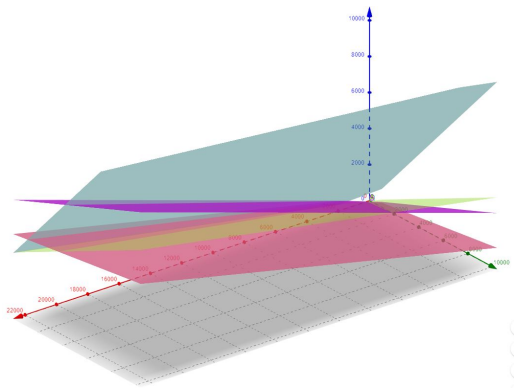
A partir de ces valeurs, il est possible de calculer les coûts pour le mandant par rapport aux différents Paiement Service Provider cités auparavant. Les variables importantes sont le chiffre d'affaires ainsi que le nombre de transactions mensuelles. La figure 8 montre qu'avec un volume d'argent et un nombre de transactions définis

	Coûts				
Colonne1	PostFinance Startu	PostFinance Pr	SIX	Stripe	Datatrans
coût mensuel (CHF)	15	39.5	152	0	
comission (%)	1.6	1.6	2	2.9	0
frais fixe par transaction	1	0.38	0.1	0.3	5
coût de mise en place (CHF)	249	495	0	0	2175
coût total 1ère année (CHF)	3132.1024	3187.2624	0	0	
coût total 1ère année à partir de la 2 ème année (CHF)	2883.1024	2692.2624	3806.008	3 718.70	3910

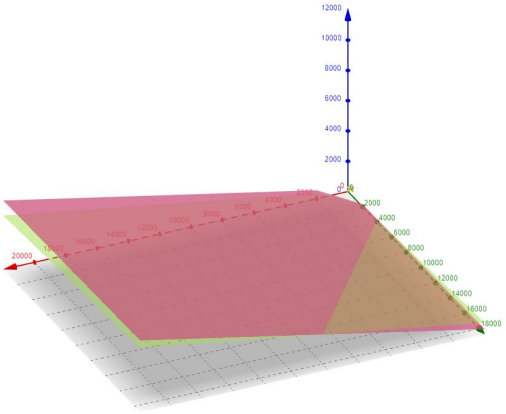
FIGURE 8 – Coûts pour la Berra par Paiement Service Provider

par rapport à l'année précédente, la solution la plus avantageuse est PostFinance Pro.

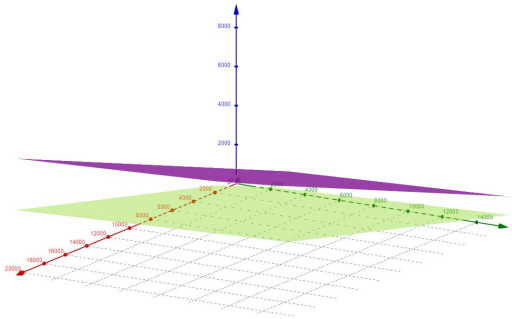
Il est également intéressant de savoir à partir de quel volume mensuel il serait bénéfique de changer de système. Pour cette comparaison ainsi que pour la précédente, l'unique métrique prise en compte est le coût brut de la solution. Le coût humain ainsi que la complexité pour la mise en place sont pris en compte dans une autre variable pour la comparaison.



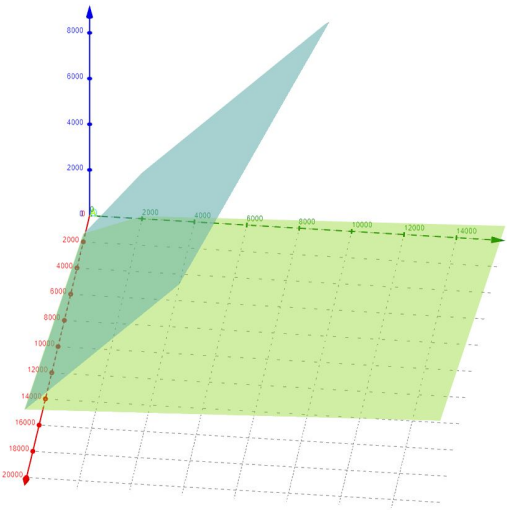
(a) Coûts de tous les PSP analysés, PostFinanceStartup en bleu, Stripe en violet, PostFinancePro en jaune et SIX en rouge



(b) Comparaison entre PostFinance Pro(vert) et SIX(rouge)



(c) Comparaison entre PostFinance Pro(vert) et Stripe(violet)



(d) Comparaison entre PostFinance Pro(vert) et StartUp(bleu)

FIGURE 9 – Graphique 3D représentant les coûts(axe vertical bleu) des différentes solutions disponibles en fonction du nombre de transaction(axe horizontal vert) et du volume d'argent pour une durée d'un mois(axe horizontal rouge)

PostFinance Pro est donc la solution la moins chère en terme de coût. Il est apparent dans la figure b que pour une entreprise recevant beaucoup de commandes online d'un petit montant, il serait préférable de se diriger vers SIX, mais ce n'est pas notre cas.

2.3.4 Matrice de décision du Paiement Service Provider

Il est important de prendre en compte les autres métriques liées aux différents Paiement Service Provider.

Les critères sont les suivants :

1. Coûts
2. API
3. Mise en place
4. Gestion des contrats
5. Intégration
6. Moyens de paiement

Critères	**Coeficient**	**P Pro**	**P StartUp**	**SIX**
Coûts	**3**	5	1	4
API	**2**	4	4	4
Mise en place	**2**	2	2	3
Gestion des contrats	**1**	1	1	1
Intégration	**2**	5	5	4
Moyens de paiement	**3**	5	5	5
Total		**53**	**41**	**50**

TABLE 1 – Matrice de décision du moyen de paiement

Critères de sélection

1. Comme premier critère de sélection, il y a les coûts du PSP. Il y a deux types de coûts qu'il est important de différencier
   (a) coûts de mise en oeuvre
   les coûts de mise en place sont les coûts uniques que le client doit payer lors de la mise en place du PSP. Ce coût diffère selon le PSP.
   (b) coûts d'exploitation
   les coûts d'exploitation sont les coûts mensuels que le client doit payer par mois afin de pouvoir utiliser le PSP. Ces coûts varient en général en fonction du nombre de transactions et de la quantité de transaction. Il y a en général un coût fixe mensuel de base.
2. comme second critère de sélection, les moyens de paiements à disposition. En effet, il est important d'offrir le plus de moyens de paiement possibles afin de pouvoir faciliter l'achat des clients avec leur moyen de paiement favori.

Contrainte écartée
La solution doit proposer des moyens de paiement sans devoir passer de contrats supplémentaires avec tous les fournisseurs de cartes, la contrainte a été écartée car il aurait fallu d'utiliser un facilitateur de paiement pour n'avoir qu'un seul contrat. Or, aucun facilitateur de paiement n'offre de moyens de paiements suisses tels que PostFinance par facture ou Twint.

Choix du PSP
PostFinance pro est le payment service provider choisi. Il est le moins cher et il propose plusieurs autres moyens de paiement sans frais supplémentaires. Il propose le paiement sur facture, ce que ne fait pas SIX payments. Comme dit plus tôt, les paiements par facture sont très utilisés en suisse et le mandant le constate également. C'est pourquoi, il est indispensable de pouvoir proposer ce moyen de paiement.
Pour pouvoir rendre disponible les paiements par carte de crédit, il faut réaliser un contrat avec un acquéreur (Concardis ou SIX payments). Pour le moyen de paiement par facture, il faut réaliser un contrat avec Swissbilling qui est un acquéreur rendant possible le paiement sur facture.
Avec swissbilling, le produit de la vente est versé sur le compte du marchand avant la réception du paiement du client. L'intégralité du risque de défaut et la gestion des débiteurs est supportée et gérée par Swissbilling.

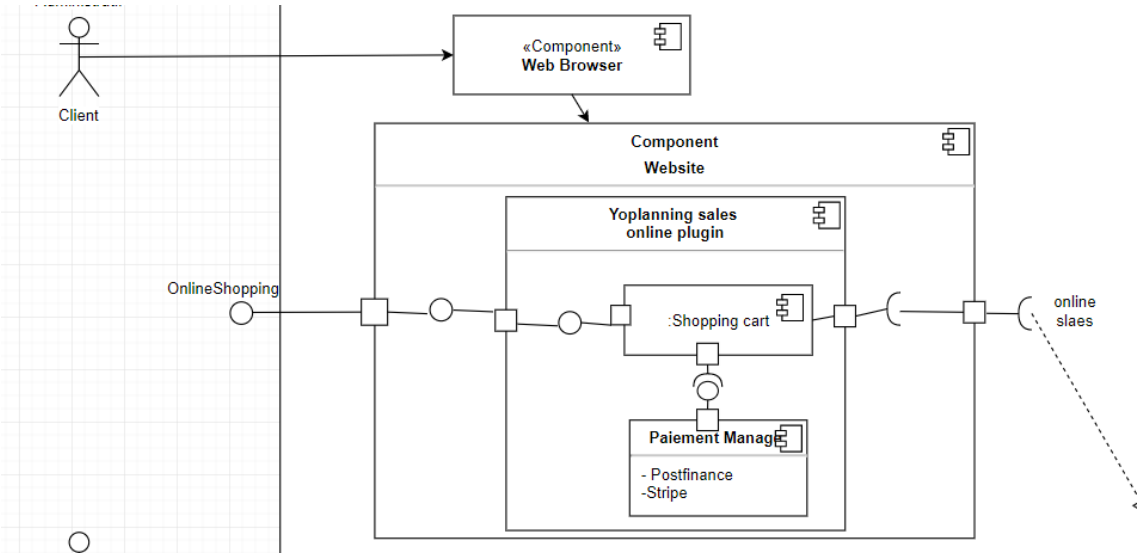


FIGURE 10 – Intégration de PostFinance dans le plugin YoPlanning

2.3.5 Solution de point of sale

Il existe plusieurs solutions mobile Point Of Sale (mPOS). L'objectif est de permettre aux clients de payer par carte de crédit au guichet de l'école de ski. Actuellement, la solution utilisée est Sumup. Cette solution n'est pas liée et le personnel administratif entre manuellement le montant de la transaction. L'objectif est de lier via API le logiciel de réservation et le POS afin d'éviter toutes erreurs humaines. Lors de l'année précédente les frais lié à l'utilisation d'un POS s'est élevé à 872.59 CHF.11

Nom	Montant (CHF)	Pourcentage	Nom2	Montant (CHF
Chiffre d'affaire (débit)	36119	**64.78**	frais carte débit (1.5%)	440.04
Chiffre d'affaire (crédit)	19638	**35.22**	frais carte crédit (2.5%)	432.55
total entrée	55757	100	frais total	**872.59**

FIGURE 11 – Frais lié à Sumup

Contraintes

— La solution POS doit offrir une API permettant de lier le lecteur de carte avec le logiciel de réservation.
— La solution ne doit pas être vendue avec un coût de base ou mensuel à moins que cela ne revienne à moins cher.
— Le logiciel doit être disponible en suisse et en français.

Square
Square est la solution référence pour les mPOS qui a été créé en 2009 aux Etats-Unis. Il propose différents produits qui permettent d'accepter des paiments online, des paiements dans une application, des paiement sur un point de vente ainsi qu'un suivi des commandes. Il est possible de l'utiliser uniquement dans les pays anglophones. C'est pourquoi cette solution ne peut pas être retenue pour le projet.

Sumup
Sumup est un des deux leader sur le marché des mPOS en Europe. La société anglaise a été créée en 2012 et est actullement présente dans 31 pays différents essentiellement en Europe. L'entreprise ne facture pas de coût mensuel et l'achat d'un terminal de paiement coûte 69 CHF. La commission pour chaque transaction est de 1.5 % pour les cartes de débit et de 2.5 % pour les cartes de crédit. Cette solution est donc très compétitive.

myPos
myPOS [12] est une entreprise qui offre des solutions de paiement intégrées et abordables, qui changent la façon dont les entreprises acceptent les paiements par carte sur tous les canaux - en magasin, en ligne et via des appareils mobiles. C'est le concurrent principal de Sumup en matière de mobile Point Of Sale (mPOS). L'entreprise ne facture pas de coût mensuel et l'achat d'un terminal de paiement coûte 34 CHF. La commission pour chaque transaction est de 1,5 % pour les cartes de débit et de 2,5 % pour les cartes de crédit. Cette solution est donc très compétitive. Une API est à disposition pour les paiements sur un point de vente.

Conclusion
Sumup et myPos sont donc très compétitifs et avec les mêmes taux de commissions. Etant donné que le mandant possède actuellement déjà le matériel Sumup, il a été

décidé de rester avec cette solution. Sumup a une API qui permet de connecter directement sa caisse au terminal en cas de besoin afin d'automatiser les transactions.

2.4 Logiciel de réservation

L'utilisation d'un logiciel de réservation dispenserait de devoir coder un grand nombre de fonctions. Cependant si un logiciel de ce genre est implémenté, il est très important que celui-ci fournisse une API complète permettant de réaliser tous les liens utiles au projet. Les critères et contraintes permettant de faire un choix sont listés dans le sous-chapitre suivant.

Le logiciel de réservation doit aussi être compatible avec le PSP PostFinance. Cette contrainte risque de fortement réduire le nombre de prétendant. En effet PostFinance est utilisé uniquement en suisse, ainsi la plupart des logiciels préfèrent être compatibles avec d'autre moyens de paiement tel que Stripe, ou Six Payments qui eux sont compatibles en Europe mais aussi en Suisse.

2.4.1 Critères

Actuellement, ESS la Berra utilise un logiciel de réservation nommé YoPlaning. Ce logiciel a beaucoup de fonctionnalités très intéressantes et est utilisé par l'école de ski, car il est spécialisé dans la création de cours de ski avec plusieurs niveaux de difficulté et un système de classes lié à ceux-ci. Le fait que ce logiciel soit déjà connu par le personnel joue un rôle important dans la prise de décision.

LA figure ci-dessous représente ce à quoi devrait ressembler le logiciel de réservation pour le projet.

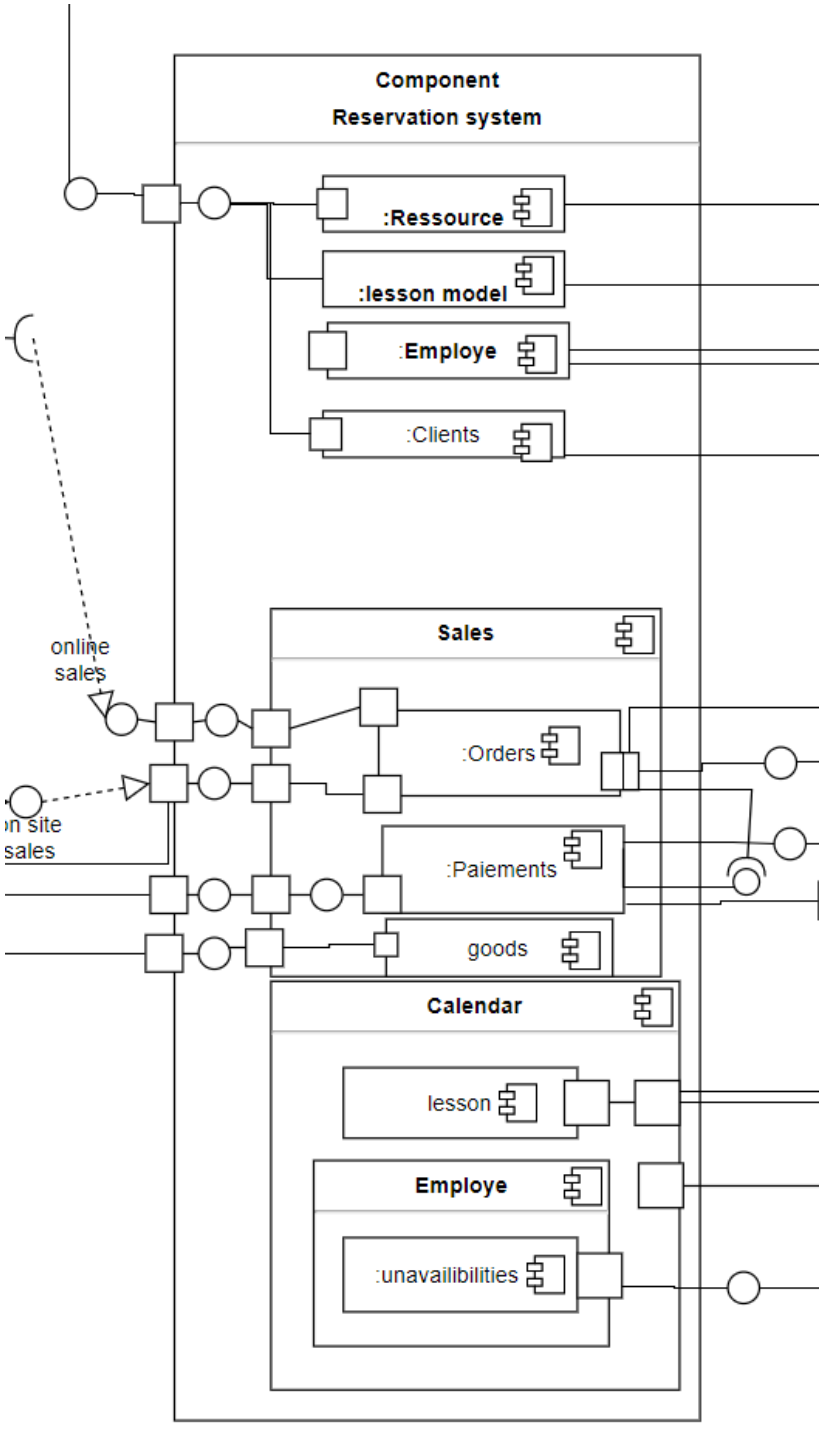


FIGURE 12 – Logiciel de réservation de la situation idéal

Afin de clarifier les besoins lié au logiciel de réservations, les divers contraintes sont listées ci-dessous. Le but étant de centraliser la gestion de l'entreprise en un seul point, les fonctions disponibles dans le logiciel de réservation doivent pouvoir être réalisées en externe, c'est à dire au travers d'une API. C'est pour cette raison que la contrainte d'API est présente dans la liste correspondante.

Contraintes

— Le logiciel ne doit pas être plus cher que la solution actuelle (Yoplanning).
— L'API doit pouvoir couvrir les besoins des cas d'utilisation de la figure 14.
— Nombre d'utilisateurs illimité.
— Compatible avec le PSP PostFinance.
— Possibilité de création de plusieurs groupes. C'est un point très important que le logiciel de réservation doit respecter. BerraRégion étant un groupe de plusieurs petites entreprises. Il est important qu'elles puissent jouir d'une zone qui leur est dédiée.
— Attribution des cours aux moniteurs en fonction de leur niveau.

2.4.2 Yoplanning

Yoplanning est codé en python pour la partie Backend, et utilise PostgreSQL pour l'administration des bases de données. La génération de requêtes entre le Frontend est codée en angular et le backend est réalisé à l'aide du protocole AMS.

C'est un logiciel de planification spécialement dédié à la gestion et la vente d'activités. Les fournisseurs proposent plusieurs offres à différents prix. Chaque abonnement implique une charge fixe par mois ainsi qu'un prélèvement à chaque transaction.

Comme cité plus haut 2.1.2, actuellement, le logiciel de réservation utilisé pas l'ESS la Berra est Yoplanning. Toutes les fonctions liées aux réservations sont donc disponibles. De plus le logiciel met à disposition une API REST[4] qui permet de couvrir tous les besoins liés en cas d'utilisation. Yoplanning est aussi compatible avec le PSP PostFinance. Le tableau ci-dessous donne quelques caractéristique du produit.

Caractéristiques	**Valeur**
Prix	27.99 euros/mois
Interopérabilité	Via API
Gestion client	Non géré
API	Type RESTful
Format de données de l'API	JSON
Personnalisation des produits	Oui
Plateforme multi-services	Oui
Vue statistique	Oui (mais ne correspond aux attentes du projet)
Limitation de membres	Non
Limitation de réservations	Non
Gestion des groupes de participants	Oui
Moyen de paiement compatible	Stripe (PostFinance sur demande)

Table 2 – Caractéristiques de YoPlanning

2.4.3 SimplyBook.me

SimplyBook.me est une autre solution très similaire à YoPlanning, celle-ci est aussi équipée d'une API[13] de type RESTful qui fournit toutes les informations nécessaires. L'avantage de ce logiciel est que celui-ci prend en considération les comptes clients, ce qui veut dire qu'un suivi des commandes de chaque client est faisable grâce à leur ID respectif sans devoir être traité par le ERP.

Le désavantage du logiciel SimplyBook.me (la solution la plus chère 59.90 dollars) est sa limite de nombre de réservations possible par mois. Cette limite peut être augmentée à un certain coût.

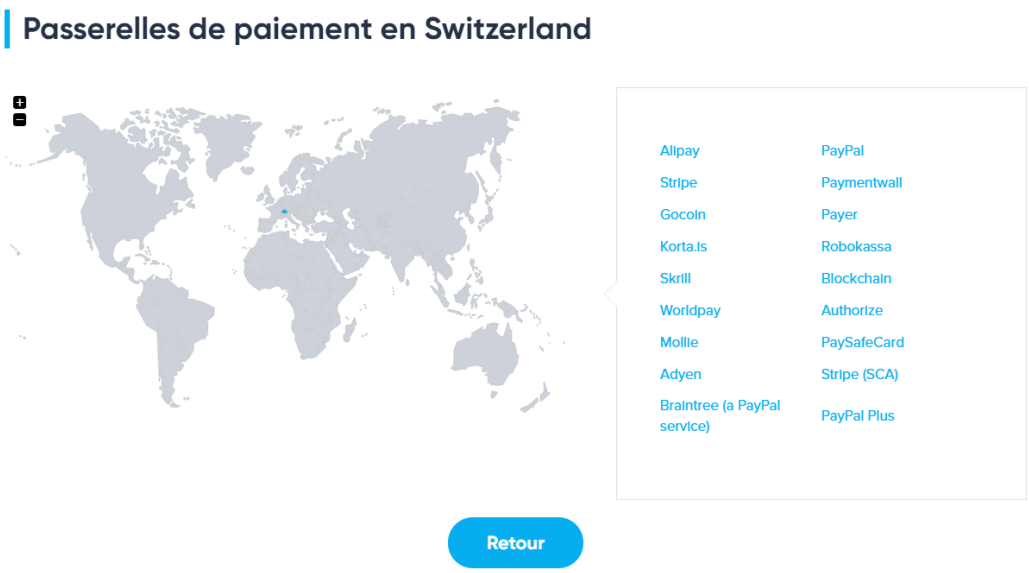


FIGURE 13 – Passerelles de paiements disponible sur simplybook.me[14]

Comme nous pouvons le voir sur la figure ci-dessus, simplybook.me est compatible avec un grand nombre de passerelle de paiement. Comme cité au préalable, la plupart des logiciels de réservations s'efforce d'être compatible avec des passerelles disponibles partout dans le monde ou en Europe.
L'analyse du logiciel simplybook.me s'arrête donc ici.

2.4.4 Bookeo

Bookeo représente un choix très intéressant, avec un abonnement à prix fixe. Cependant, le pack le plus cher, soit 119.95 dollars se limite à 60 employés compris dans l'abonnement. Un forfait spécial peut être demandé pour avoir plus d'employés. Cependant, le prix de l'abonnement va lui aussi augmenter

Le logiciel quant à lui permet une bonne gestion des réservations et des ressources. L'API[15] est très bien faite avec une grande variété de requêtes disponibles via

celle-ci.

Il est également important de noter que le logiciel permet l'intégration de plusieurs moyens de paiement, à l'exemple de PayPal ou Stripe. Le choix du moyen de paiement utilisé pour ce projet aura donc un fort impact sur le choix du logiciel de réservation et vice-versa.

Le logiciel ne respectant pas plusieurs contraintes du projet, il ne sera donc pas présent dans la matrice de décision.

2.4.5 Prise de décision du logiciel de réservation

Durant la phase d'analyse, différents logiciels de réservation ont été analysés. Malheureusement aucun autre logiciel ne respecte toutes les contraintes demandées pour le projet. C'est pourquoi aucune matrice ne sera faite puisque la seule solution correspondante au cahier des charges est celle déjà utilisée par l'école de ski de la Berra c'est à dire Yoplanning.

En utilisant ce logiciel, nous avons les ressources nécessaires à notre projet, mise à part la possibilité de faire un remboursement avec le moyen de paiement PostFinance. Cependant, après vérification par téléphone, cette fonction est en train d'être mise en place par les développeur de Yoplanning et devrait prochainement être disponible pour leurs clients.

2.5 ERP

Pour cette partie, il est important de se poser la question sur l'utilité d'un ERP[3] dans le projet. En effet, il serait possible de choisir de ne pas prendre de ERP, car celui-ci a de très fortes chances de coûter de l'argent tant par sa licence que par sa maintenance. Cependant, il est important de prendre en considération que s'abstenir d'un tel outil nécessite beaucoup d'heures de codage et de préparation afin de parvenir à un résultat plus ou moins similaire à celui qu'un ERP standard pourrait fournir, et ce, sans compter la maintenance régulière d'une telle solution.

Comme la durée du projet est plutôt restreinte et que nous ne sommes pas des développeurs à proprement parler, l'utilisation d'un ERP sera essentielle. Étant donné que la solution finale récupérera et traitera des données de divers systèmes, il est plus facile de tout centraliser afin de permettre une meilleure gestion des ressources. C'est pour cette raison que sur le diagramme de composant de la solution idéale un ERP a été intégré.

Cette partie portera sur l'analyse des différents critères importants qui définiront quel progiciel de gestion intégré représente la meilleure solution. Ensuite, 2 ERPs pouvant correspondre aux attentes définies ci-dessous seront analysés. Une matrice

de décision se basant sur les critères définis au préalable sera effectuée afin de définir le choix à faire.

2.5.1 Critères

Les critères qui serviront à définir quel ERP représente la meilleure solution seront listés dans cette section. L'idéal représente un ERP qui puisse si possible couvrir immédiatement tous les besoins sans intervention. Cependant, si ce n'est pas possible de trouver "l'outil" parfait, il est important que le choix se porte sur un ERP permettant d'ajouter des fonctions au travers de modules ou d'APIs.

Les contraintes liées à l'utilisation d'un ERP dans notre système se baseront sur la figure ci-dessous.

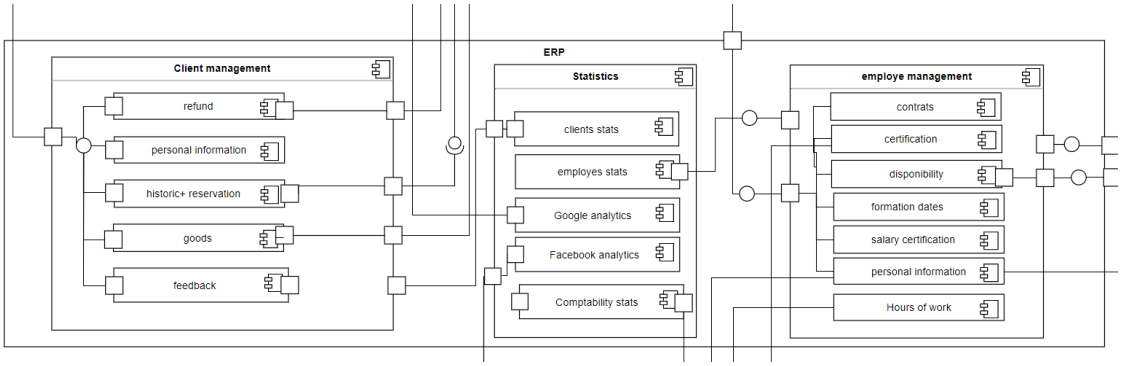


FIGURE 14 – ERP de la solution idéal

Contraintes

- Le prix du logiciel : Le mandant veut une solution ne coûtant pas plus de 1000.- par année entre le coût de l'abonnement ou de l'acquisition et de la maintenance.
- L'interopérabilité du produit.
- L'évolutivité de la solution dans le temps.
- Possibilité de gérer des comptes clients.
- Visualisation statistiques des divers métriques.
- Gestion du personnel (salaires, fiches salaire, etc.).

2.5.2 Odoo

Odoo[16] est un ERP/CRM disponible dans une version payante, mais aussi dans une version gratuite open source. La version gratuite est la version dite communautaire. Le prix de la version payante dépassant de loin la contrainte de prix, avec plus de 100.- par mois en activant tous les modules souhaités, l'analyse portera uniquement sur la version gratuite.

Odoo initialement appelé TinyERP est sorti en juillet 2014 et n'a cessé d'évoluer, jusqu'en novembre 2015 où la première publication du logiciel Odoo sous licence LGPLV3[17] a été publiée. C'est cette même version qui sera utilisée si les résultats d'analyse valident l'utilisation de cet ERP.

La version open source a quelques inconvénients, les mises à jour des modules et du logiciel lui-même ne sont pas automatiques et nécessitent donc un certain travail de maintenance. Cependant, ce progiciel reste une très bonne alternative. Codé à l'aide du langage python, il utilise une base de données de type PostgreSQL. L'avantage de ce logiciel est dans un premier temps son prix, mais aussi et surtout la facilité de création de modules afin de le faire réagir de la manière souhaitée. En effet, avec un autre logiciel, créer un module répondant aux besoins risque d'être très compliqué voire impossible pour certains.

Cet ERP peut être une solution pour le projet, en effet il fournit une API[18] très complète, il est aussi alimenté par une grosse communauté très active. Par conséquent un grand nombre de modules pouvant plus ou moins correspondre à nos attentes est disponible.

Caractéristiques	**Valeur**
Prix	Gratuit
Interopérabilité	Via API
Gestion du personnel	Oui
Gestion client	Oui
API	Type RESTful
Format de données de l'API	JSON
Vue statistique	Oui

Table 3 – Caractéristiques de Odoo

2.5.3 Dolibarr

Dolibarr[19] est aussi un très bon concurrent, il est aussi open source et a été confectionné spécialement pour les petites et moyennes entreprises. Il est aussi doté d'une architecture modulaire. Écrit en PHP et fonctionnant avec des bases de données de type MySQL ou PostgreSQL, il fonctionne avec toutes les configurations PHP.

Cet ERP est un peu moins connu qu' Odoo, il a été créé en avril 2002, il est disponible dans une version open source avec licence GPLV3[20]. Plus facile à installer qu'un ERP comme Odoo, il est disponible dans un format d'application web afin d'y avoir accès partout.

Dolibarr n'est pas non plus 100% gratuit, car suivant les modules utilisés, ceux-ci peuvent être payants. Cependant, cet ERP reste une solution très intéressante de par sa devise des 3S : simple d'installation, simple d'utilisation et simple de développement.

Caractéristiques	**Valeur**
Prix	Gratuit (suivant les modules)
Interopérabilité	Via API
Gestion du personnel	Oui
Gestion client	Oui
API	Type RESTful
Format de données de l'API	JSON
Vue statistique	Oui (mais ne correspond aux attentes du projet)

Table 4 – Caractéristiques de Dolibarr

2.5.4 Tryton

Tryton[21] constitue aussi une bonne solution. Cet ERP est à l'origine un fork en 2008 de Odoo. Celui-ci est codé en python et utilise aussi PostgreSQL comme moteur de base de données. Il est disponible sous licence GPLV3 et est donc construit comme Odoo. Le développement de nouveaux modules est aussi prévu. Cependant, la documentation que ce soit au niveau utilisateur, administrateur, ou développement est plutôt maigre et difficile à apprivoiser. De plus, cet ERP ne dispose pas de gestion RH. Certes pour ce , ceci n'a pas un gros impact, mais ce travail est étroitement lié avec un autre projet en développement actuellement et celui-ci nécessite d'avoir une gestion RH intégrée.

Les deux projets devront être intégrés sur un seul et unique ERP, c'est pourquoi l'analyse de cette solution s'arrêtera ici.

2.5.5 Matrice de décision de L'ERP

La recherche de la solution idéale représente un vrai challenge, un grand nombre d'ERP avec de grosses promesses sont disponibles sur le marché. Les contraintes définies en début de section ont permis de faire un gros tri sur ceux-ci. Notamment la contrainte de prix. En effet la plupart des ERPs coûtent une somme très importante. La seule manière de respecter cette contrainte a été d'analyser des solutions open source. Elles-mêmes n'étant pas toujours vraiment gratuites.

Critères	**Coeficient**	**Odoo**	**Dolibarr**
Prix	**2**	4	4
Fonctions implémentées	**1**	4	3
Gestion du personnel	**2**	5	5
Gestion client	**1**	5	5
Contenu de l'API	**2**	4	4
Vue statistique	**1**	5	3
Evolutivité	**1**	4	2
Facilité d'utilisation	**1**	3	5
Total		**47**	**44**

Table 5 – Matrice de décision de l'ERP

L'ERP qui correspond le plus aux besoins du projet est Odoo. C'est cet ERP qui sera utilisé pour ce travail.

2.6 Logiciel de comptabilité

En Suisse, il existe plusieurs logiciels de comptabilité spécifiquement conçus pour la législation comptable en place. Il est donc important de ne citer que ce type de logiciel adapté à notre pays.
Ces logiciels proposent des logiciels métiers offrant des solutions pour les difficultés liées aux métiers du chiffre (facturations, paiements, comptabilité, TVA, salaires, LPP, AVS, etc).
Dans le cadre de ce projet, le point important est la comptabilité et les paiements. Les autres fonctionnalités restent un grand plus pour la gestion d'une entreprise.

Contraintes Le logiciel doit posséder une API afin de pouvoir lier le logiciel avec les autres composants du système final.

2.6.1 Les critères

- Une API est à disposition et fournit les requêtes nécessaires afin de pouvoir développer pour recevoir et envoyer des informations.
- La comptabilité ainsi que les paiements et les salaires sont intégrés à la solution.
- Les coûts mensuels ou annuels.
- Le fournisseur du logiciel facilite le développement de solutions.
- La mise en place est facile et documentée.
- Gestion de moyenne entreprise possible (plus de 120 employés).
- Qualité de l'interface utilisateur.
- Adapté à la législation suisse.

L'objectif est d'arriver au résultat comme présenté ci-dessous. L'objectif est donc de lier le logiciel de comptabilité avec l'ERP. C'est le projet de Mr Guibert.

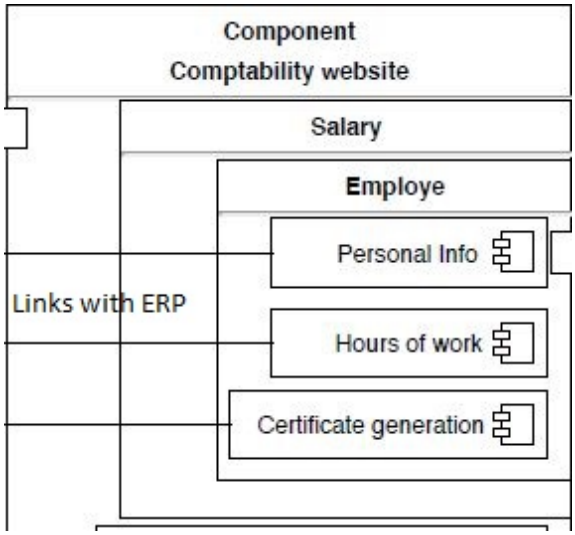


FIGURE 15 – Partie comptabilité du diagramme de composants

2.6.2 Les logiciels

Une liste de solutions a été contactée et en fonction de la mise à disposition d'une API ou non, elle est intégrée à notre matrice de décision. La liste de logiciels de comptabilité contactés est :

- Crésus [22]
- Winbiz [23]
- Banana [24]
- Comptabase [25]
- Bexio [26]

Il est a noté que Crésus, Comptabase ainsi que Banana n'offrent pas d'API et sont donc directement écartés de l'analyse. L'analyse porte donc sur les deux logiciels suivants : Winbiz et Bexio.

Winbiz WinBiz est un logiciel d'entreprise qui centralise la comptabilité, la facturation, la gestion des salaires et la gestion des heures de toute entreprise. Il est conçu en coopération permanente avec les autorités cantonales et fédérales, les caisses sociales ainsi que les instituts financiers suisses. Ses fonctionnalités sont étroitement liées aux besoins administratifs des PME suisses ainsi qu'aux nouveautés technologiques et législatives.
Les extensions caisse et commerce sont importantes pour permettre d'avoir les fonctionnalités liées à la vente online ainsi que la vente au point of sale. Pour avoir toutes ces extensions, il faut débourser 89 CHF par mois.

Bexio Bexio est un autre logiciel de comptabilité très complet offrant des fonctionnalités importantes pour le projet telle que la comptabilité en ligne, la gestion des salaires online. Pour avoir la gestion des salaires inclus, il faut prendre le pack Pro+ qui est au prix de 99 CHF par mois.

Conclusion Bexio n'a pas de partie caisse ni commerce. C'est pourquoi, elle est également exclue de la comparaison.
Il faut également mettre en évidence qu'une grande partie du marché est client chez winbiz. Typiquement les écoles de ski de la région romande utilise pour la plus grande partie d'entre elles Winbiz. Il est important de prendre cela en compte afin de mettre sur pied un projet pouvant potentiellement être utilisé par d'autres écoles de ski qui ont déjà manifesté leur intérêt. Le choix se porte donc sur Winbiz comme composant logiciel de comptabilité

2.7 Problèmes rencontrés

A la suite de l'analyse des solutions choisies, il a été constaté le problème suivant : le paiement manager ne prend actuellement pas en charge les remboursements avec le PSP PostFinance. En effet, le lien entre le paiement manager et le PSP est une partie gérée exclusivement par le logiciel de réservation choisi(YoPlanning). Il est donc actuellement impossible de réaliser la gestion des remboursements. Il est prévu que YoPlanning mette à disposition la gestion des remboursements avec le PSP PostFinance mais pas avant janvier 2020. Nous allons donc prendre en compte et prévoir dans le projet mais cette partie ne sera pas implémentée.

2.8 Conclusion analyse

L'analyse a permis de définir les différentes solutions qui seront utilisés afin de réaliser le projet. Il sera donc réaliser avec le logiciel de réservation YoPlanning, l'ERP open-source Odoo, le logiciel de comptabilité Winbiz ainsi que le PSP PostFinance. Sumup a quant à lui été retenu comme solution de mPOS mais son intégration directe ne fait pas partie des limites du projet. Les choix réalisés ont été pris par rapport à plusieurs critères qui ont été définis dans chaque partie spécifique à chaque solution à analyser.

2.9 Priorisation pour la suite du projet

Pour la suite du projet il a été décidé de diviser le projet global en différentes parties et de prioriser les parties les plus importantes du projet. Pour définir l'ordre de priorisation, les contraintes d'implémentations par rapport au diagramme de composants ont été analysées.

1. La partie client de l'ERP est le point central sur lequel est basé le reste du projet. C'est donc le premier objectif à réaliser.
2. La partie authentification sur le site internet est un aspect important du projet afin de proposer une expérience client plus agréable.
3. La partie concernant les remboursements directement depuis l'ERP doit être prise en compte dans le développement du projet mais ne pourra pas être mis en place. L'interface n'étant pas mise en place du côté du logiciel de réservation
4. La partie statistique a un caractère important pour l'analyse des données. Cependant, il n'apporte rien de plus à l'expérience client. Cette partie peut être approfondie dans un autre projet d'analyse et d'exploration des données issues des différentes sources à disposition (ERP, Facebook, Google Analytics) afin de mettre cela en place dans un CRM.

Avec cette priorisation, il est possible d'effectuer chaque partie les unes après les autres en ayant ainsi des étapes intermédiaires fonctionnelles. Cette séparation permet également d'avoir un cadre de projet et une vision plus claire sur les objectifs à atteindre.

3 Conception

Cette partie du rapport permettra de structurer la phase de developpement, en définissant plus précisément les objectis à atteindre et en précisant les tâches qui devront être faites durant celle-ci.

3.1 Workpackages

Pour mieux se rendre compte de la charge de travail et des divers tâches à réaliser durant la phase de développement, les activités vont être découpés en de multiples tâches d'une demi-journée maximum.

3.1.1 Workpackage 1 - Mise en place de l'environnement

Résumé : Les tâches regroupées dans ce chapitre représentent les divers activités qui seront nécessaires pour mettre en place l'environnement de test.
Date de création : 27.11.2019

Préconditions : disposer d'un serveur et d'une copie du site de l'ESS la Berra.

Objectifs :
L'environnement devra contenir le site internet, un accès à la webApp Yoplanning, ainsi que le server Odoo et sa base de données.

Tâche	Auteur	durée [h]
Jalon 1 - Odoo fonctionnel		**4**
Récupération des sources pour Odoo	Raphaël	1
Vérification des dépendances Pour Odoo	Raphaël	1
Mise en place de Odoo	Raphaël	2
Jalon 2 - Base de données fonctionnelle		**8**
Analyse des modules disponibles	Raphaël	2
Analyse de l'architecture de la base de données fournis par Odoo	Raphaël	2
Schématisation de l'architecture de la base de données	Raphaël	1
Modification de la base de données en fonction des besoins	Raphaël	3
Jalon 3 - site internet fonctionnel		**5**
Mise en place d'un serveur web pour le site internet	Célestin	2
Importation d'une copie du site internet de l'ESS la Berra	Célestin	1
Modification du site afin de ne garder que l'essentiel	Célestin	2

Table 6 – Estimation du temps pour le workpackage 1

3.1.2 Workpackage 2 - Authentification

Résumé : Les tâches suivantes regroupent les différentes activités nécessaires pour permettre au client de s'authentifier sur le site internet. Ce qui implique l'intégration d'une gestion des utilisateurs, la sécurisation de le méthode d'authentification mais aussi la conception et la mise en place du frontend liée à ce cas d'utilisation. **Date de création :** 28.11.2019

Préconditions : le site internet mise en place dans le package "Mise en place de l'environnement" doit être fonctionnel.

Objectifs :
Le clients doit pouvoir créer un compte ou se connecter de manière sécurisée sur le site internet.

Tâche	**Auteur**	**durée [h]**
Jalon 4 - Authentification		**6**
Concevoir la méthode d'authentification	Raphaël	2
Mettre en place l'environnement	Raphaël	1
Implémentation du backend	Raphaël	3
Jalon 5 - Frontend authentification		**6**
Conception des mockups	Raphaël	2
Implémentation du frontend	Raphaël	4

Table 7 – Estimation du temps pour le workpackage 2

3.1.3 Workpackage 3 - Traitement des données de YoPlanning à Odoo

Résumé : Ce sont les tâches nécessaires afin de pouvoir réaliser la liaison entre le logiciel métier Yoplanning et son API avec l'ERP utilisé dans ce projet (Odoo).
Date de création : 28.11.2019

Préconditions : le server Odoo et sa base de données et l'accès à l'API de YoPlanning sont disponibles

Objectifs :
L'objectif est de remplir les informations des clients directement dans l'ERP et gérer la synchronisation entre les deux composants.

Tâche	Auteur	durée [h]
Jalon 6 - Diagrammes et contraintes pour le traitement des données		**7**
l'API fourni par le logiciel de réservation est testée pour voir la forme des données pouvant être récupérée	Célestin	3
Le diagramme de séquence est réalisé entre les deux composants afin de pouvoir schématiser le problème de la synchronisation	Célestin	1
Définition des conditions et des contraintes de synchronisation des données clients entre le logiciel de réservation et l'ERP	Célestin	2
Jalon 7 - Implémentation		**10**
Le choix de la technologie utilisé pour réaliser les requêtes est fait et justifié	Célestin	2
Implémenter la première importation des données présentes dans le logiciel de réservation	Célestin	3
Gérer les webhook pour gérer la synchronisation entre le logiciel de réservation et l'ERP	Célestin	5
Jalon 8 - Test		**2**
Tester l'infrastructure mise en place	Célestin	2

Table 8 – Estimation du temps pour le workpackage 3

3.1.4 Workpackage 4 - l'affichage des données dans le compte client

Résumé : Les tâches regroupées dans ce workpackage regroupent les tâches nécessaires à réaliser l'affichage des informations du client dans son compte personnel. Il faudra pour cela lier le compte personnel d'Odoo avec le login. L'utilisation de l'API d'Odoo est pour cela obligatoire. Il est cependant nécessaire de déterminer le language qui sera utilisé pour réaliser les requêtes nécessaires.
Date de création : 28.11.2019

Préconditions : L'environnement devra contenir le site internet ainsi que le server Odoo et sa base de données.

Objectifs :
L'affichage du compte client est fonctionnel. Le client peut accéder à son compte et ses informations. Le design n'est pas nécessaire.

Tâche	Auteur	durée [h]
Jalon 9 - Analyse technologies utilisées		**4**
Définition des technologies utilisées frontend pour l'affichage chez le client	Célestin	2
Définition des technologies utilisées pour la récupération des données depuis l'ERP vers le site internet	Célestin	2
Jalon 10 - Conception affichage		**4**
Réalisation diagramme de séquence et d'activité de l'affichage	Raphaël	2
Realisation d'une mockup sommaire pour l'affichage des informations du client	Raphaël	2
Jalon 11 - Réalisation affichage		**8**
réalisation des requêtes pour récupérer les informations du client depuis l'ERP	Raphaël	4
Organiser les informations du client au bon endroit par rapport à la mockup	Célestin	4

TABLE 9 – Estimation du temps pour le workpackage 4

3.2 Planification

wp	Jalons	Auteur	date
1	Jalon 1 - Odoo fonctionnel	Raphaël	11.12.2019
1	Jalon 3 - site internet fonctionnel	Célestin	11.12.2019
1	Jalon 2 - Base de donnée fonctionnel	Raphaël	18.12.2019
3	Jalon 6 - Diagrammes et contraintes pour le traitement des données	Célestin	18.12.2019
2	Jalon 4 - Authentification	Raphaël	08.01.2020
2	Jalon 5 - Frontend authentification	Raphaël	08.01.2020
3	Jalon 7 - Implémentation	Célestin	08.01.2020
3	Jalon 8 - Test	Célestin	08.01.2020
4	Jalon 9 - Analyse technologies utilisées	Célestin	22.01.2020
4	Jalon 10 - Conception affichage	Raphaël	22.01.2020
4	Jalon 11 - Réalisation affichage	Raphaël	22.01.2020
4	Jalon 11 - Réalisation affichage	Célestin	22.01.2020
	Rendu du rapport		31.01.2020

TABLE 10 – Planification des dates de rendu des jalons

4 Développement

Ce chapitre décrira toute les étapes qui ont été nécessaires pour atteindre les différents jalons tout en respectant les contraintes. Le chapitre sera divisé par workpackage afin de conserver la même structure.

Le premier chapitre consiste en un tableau représentant le temps réel effectué pour mettre en place les différents jalons et les temps estimés afin de pouvoir comparer si ceux-ci étaient bons.

Jalons	**Temps estimé**	**Temps réel**
Jalon 1 - Odoo fonctionnel	4H	4H
Jalon 3 - Site internet fonctionnel	5H	6H
Jalon 2 - Base de données fonctionnelle	8H	4H
Jalon 6 - Diagrammes et contraintes pour le traitement des données	7H	10H
Jalon 4 - Authentification	6H	12H
Jalon 5 - Frontend authentification	6H	4H
Jalon 7 - Implémentation	10H	16H
Jalon 8 - Test	2H	2H
Jalon 9 - Analyse technologies utilisées	4H	2H
Jalon 10 - Conception affichage	4H	2H
Jalon 11 - Réalisation affichage	8H	10H

Table 11 – Comparaison du temps effectué par rapport au temps estimé pour chaque jalons

4.1 Workpackage 1 - Mise en place de l'environnement

Pour effectuer ce workpackage l'utilisation de la documentation mise à disposition par Odoo[27][28] est indispensable.

4.1.1 Equipement

Pour mettre en place l'environnement de test du projet, une machine virtuelle a été créée dans la Demilitarized zone (Zone démilitarisée) de l'école d'ingénieur et d'architecture de Fribourg. Le port 22 (port SSH) est ouvert afin de permettre le contrôle de celle-ci à distance.

OS	Ubuntu
Version	18.04.3
CPU	2
RAM	2[GB]
Stockage	30[GB]
Connexion port	22(SSH)
	80(HTTP)
	443(HTTPS)
	8069(Odoo)
	5432(PostgreSQL)
	30306(PhpMyAdmin)
utilisateur	compte AAI

TABLE 12 – Caractéristiques de la machine virtuelle

4.1.2 Jalon 1 - Installation de la base de données

Le serveur Odoo a besoin d'une base de données pour fonctionner. Odoo utilise le système de gestion de base de données PostgreSQL[29].
Pour l'installer, il faut effectuer la commande suivant en tant qu'administrateur (sudo) :

apt-get install postgresql -y

Avant de passer à l'installation même du serveur Odoo, il faut encore ajouter un utilisateur sur la base de données. Les commandes suivantes sont effectuées dans le terminal :

sudo -u postgres createuser -s root

Le SGBD PostgreSQL créer par default un accès pour l'administration des base de données sur le port 5432. La commande suivante permet de vérifier si le port 5432 de la machine virtuelle est actif. netstat -na | grep "5432"

Le SGDB est maintenant fonctionnel. Les prochains chapitres définiront comment ajouter une base de données Odoo sur celui-ci.

4.1.3 Jalon 1 - Installation du server Odoo

Les instructions d'installation d'Odoo se trouvent aussi dans la documentation d'installation du progiciel cité plus haut dans ce chapitre. Des informations supplémentaires seront ajoutées afin d'éviter d'éventuelles erreurs qui pourraient intervenir

plus tard dans le processus d'installation.
Dans un premier temps, il faudra réaliser les commandes suivantes en mode administrateur (sudo su) :

```
wget -O - https://nightly.odoo.com/odoo.key | apt-key add -
echo "deb http://nightly.odoo.com/13.0/nightly/deb/ ./" > ...
... /etc/apt/sources.list.d/odoo.list
apt-get update && apt-get install odoo
```

Après avoir effectué ces commandes, le serveur peut être mis en place avec la commande :

sudo service odoo start

À l'aide de la commande "sudo tail -f /var/log/odoo/odoo-server.log" il faut vérifier qu'il n'y ait pas de WARNING de type :

`odoo.addons.base.models.res_currency:` The num2words python library is not installed, amount-to-text features won't be fully available.

Si c'est le cas, il faudra effectuer la commande suivante pour installer pip3 qui permettra d'ajouter les librairies manquantes :

```
sudo apt-get install python3-pip
sudo pip3 install num2words
sudo service odoo restart
```

la commande "sudo tail -f /var/log/odoo/odoo-server.log" montre maintenant que le serveur est activé et en fonction.

En se rendant sur la page "adresseIpServeur :8069" la page suivante s'affiche. Ceci signifie que le serveur est fonctionnel.

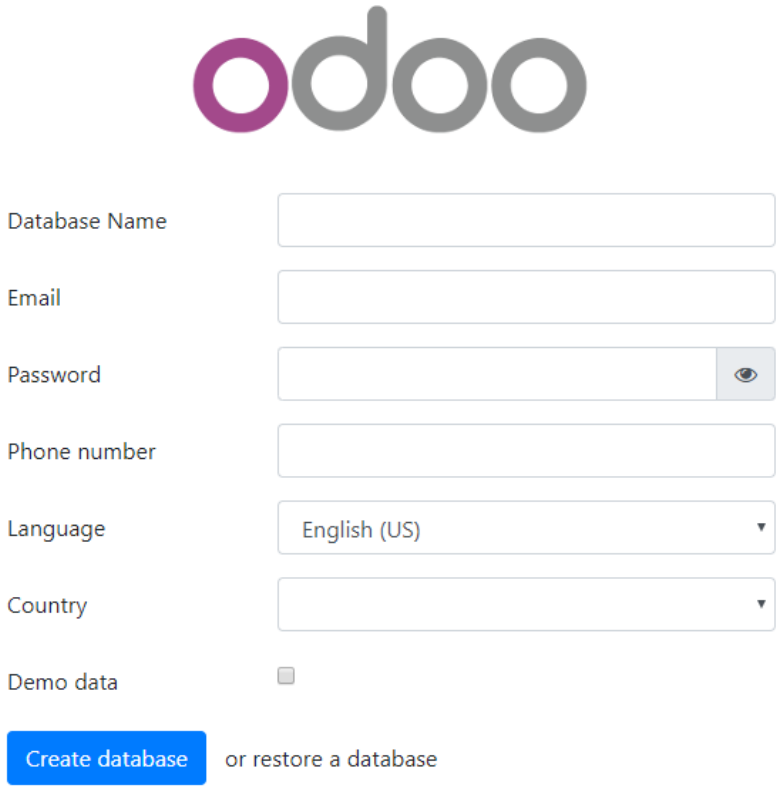


FIGURE 16 – Première page fournis par Odoo

En cas de problème, il faut vérifier si le port 8069 est bien ouvert.

4.1.4 Jalon 1 - Configuration d'Odoo

Ce chapitre est dépendant des deux chapitres précédents. A cette étape de la mise en place de l'environnement, le serveur odoo doit être fonctionnel et atteignable depuis une autre machine via la combinaison de son adresse IP et du port 8069. La base de données doit aussi être fonctionnelle et avoir déjà été initialisée en ajoutant l'utilisateur root.
La figure 16 est la première étape afin de configurer le serveur. Pour ce projet, le nom de la base de données sera :"berra-region-odoo".
Après avoir rempli tous les champs avec les informations correspondantes, le système propose de remplir la base de données avec des informations de démonstrations. Afin de faciliter l'analyse de la base de données nécessaire pour le jalons suivants, cette option sera activée.

Odoo est maintenant en place et la page actuelle est une boutique d'applications et de modules. Dans un premier temps en se rendant dans les paramètres odoo ("settings" sur l'interface web), il est possible de modifier les utilisateurs de la plateforme. il est possible à cette étape d'ajouter des utilisateurs et de définir leur droits au sein du

progiciel.

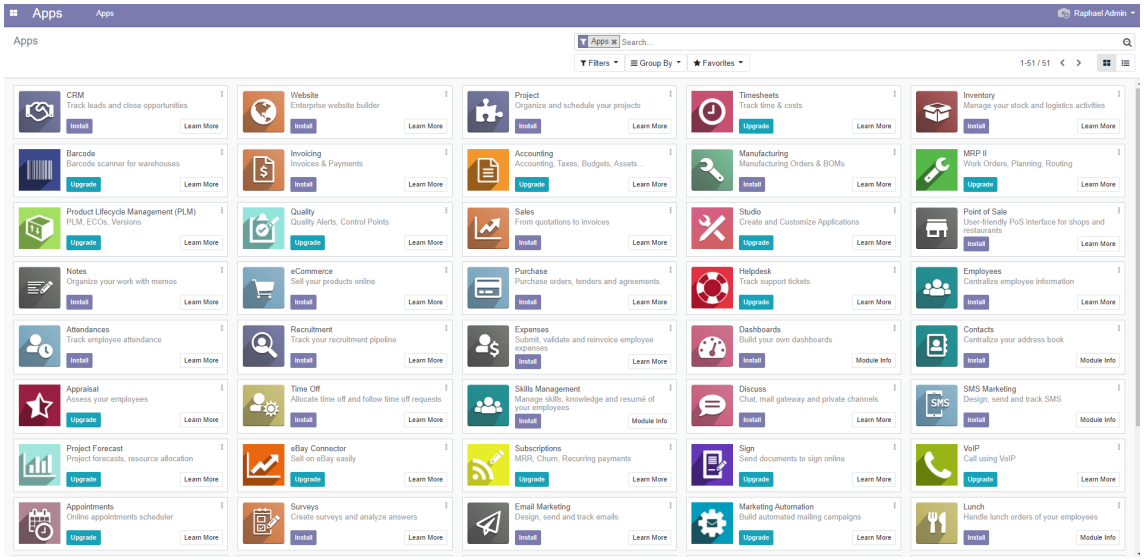


FIGURE 17 – Première page après la configuration de base Odoo

Afin de faciliter le travail de développement, le mode développeur a été activé sur la plateforme. Pour cela il faut se connecter à la plateforme,et aller tout en bas de la page settings et cliquer sur le lien "activate the developper mode". Cette fonction permet d'afficher le nom des variables ainsi que d'autres informations la concernant simplement en survolant la variable.

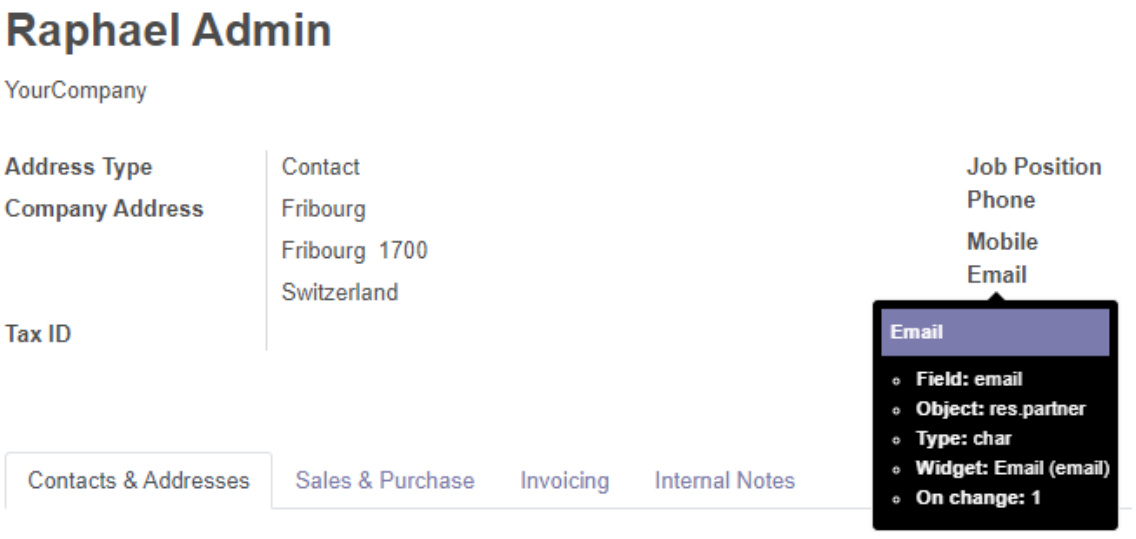


FIGURE 18 – Visualisation des variables.

4.1.5 Jalon 1 - Remarque

La machine virtuelle mise à disposition par l'établissement n'avait que les ports 22 (SSH), 80(HTTP), 443(HTTPS) ouverts. Lorsque l'installation d'Odoo a été faite,

il était impossible de se connecter à l'interface Odoo depuis une autre machine que le serveur lui-même. Pour comprendre la provenance du problème beaucoup de recherches ont été faites et ont permis de trouver la résolution à ce problème et en révéler un autre pouvant générer la même erreur. Le firewall de la machine virtuelle en est la source. Pour le résoudre, il suffit d'entrer la commande suivante :

```
sudo ufw allow 8069
sudo ufw allow 80
sudo ufw allow 443
sudo ufw allow 5432
```

Ces commandes indiquent au firewall de la machine virtuelle d'autoriser les demandes de connexion via les ports spécifiés.

4.1.6 Jalon 2 - Analyse des modules disponibles

La version communautaire de Odoo a énormément de modules qui ont été implémentés par la communauté elle-même. Il est donc important de prendre le temps d'analyser ceux-ci afin de voir si certains de ceux-ci correspondent aux besoin du projet.

FIGURE 19 – Première page lors de la configuration d'essai fournis par Odoo

La figure ci-dessus est la configuration de base fournie par Odoo lorsqu'une demande d'essai du logiciel est faite.

Message
Le module message s'active automatiquement lorsque le module employés est installé. Il permet de pouvoir gérer les communications internes en pouvant effectuer des groupes de discussion ainsi que des communications privées avec tous les membres de l'entreprise.

Employés
Ce module permet de gérer le personnel de l'entreprise, en ayant toutes les informations utiles pour chaque employé. Il est aussi possible d'ajouter des départements ainsi que des type de jobs.

Sur la figure 19, le module de gestion du personnel est présent. Cependant il y a par conséquent pas de gestion des clients et des partenaires.

Contacts
Le module contact présent sur la droite de la figure 17, lui, permet de regrouper les partenaires et les clients sur un seul module. Et c'est par la même occasion le seul module de base permettant de gérer des clients. De plus, ce module implémente automatiquement un historique des paiements fait pour chaque client, cette fonction sera très utile lors de la réalisation des workpackage 3 et 4. **Le module "contacts" sera donc installé**. Il suffit de se rendre sur l'onglet "apps" et de cliquer sur le bouton install, odoo se chargera du reste.

Invoicing
Le module Invoicing est un module qui permet la gestion des factures et des paiements en attente. C'est aussi grâce à ce module qu'il est possible d'avoir un historique des paiements pour chaque client **Il sera donc aussi installé.**

4.1.7 Jalon 2 - Analyse de la structure des données d'Odoo

Odoo a une architecture à 3 niveaux. Le niveau le plus bas correspond à la couche de stockage. Juste au-dessus la couche logique, c'est cette couche uniquement qui communique avec la base de données, et elle est gérée par le serveur lui-même. Pour finir, la dernière couche est la couche de présentation et d'interaction avec les utilisateurs[30].

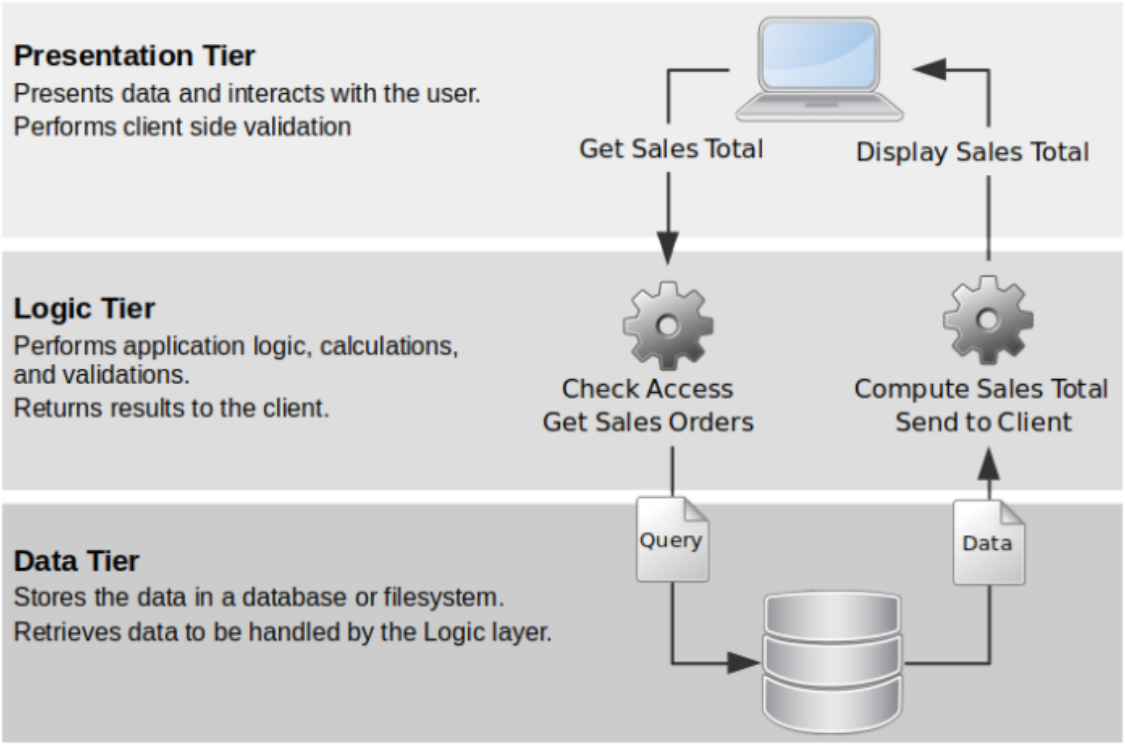


FIGURE 20 – Architecture d'Odoo

Couche de Stockage
La couche de stockage est gérée par un serveur PostgreSQL, C'est le seul serveur de base de données pris en charge par Odoo. Les fichiers binaires, comme les images ou les documents sont quant à eux stockés dans un répertoire appelé "filestore". Ce qui implique que pour faire une backup complète il faut copier la base de données mais aussi le répertoire "filestore"

Couche logique
C'est uniquement cette couche qui communique avec la base de données et est gérée par les serveur Odoo. Cette manière de faire permet de garantir une sécurité des contrôles d'accès et une cohérence. Dans cette couche se trouve un **ORM** qui fournit un API qui est utilisé par les différent modules. L'ORM représente les divers entités de donnée par des modèles. Ces modèles sont des classe python ayant plusieurs méthodes permettant la gestion de ceux-ci, comme par exemple : create(), search(), ou read().

Couche de présentation
Responsable des données et de l'interaction avec l'utilisateur, c'est sur cette couche que la partie authentification devra être réalisée. Cette couche communique avec le serveur Odoo via le protocole RPC. Ce protocole permet de communiquer avec un serveur distant, et est très souvent utilisé pour la conception de micro-noyaux.

4.1.8 Jalon 3 - Mise en place d'un serveur web

Afin de pouvoir héberger un site internet sur le serveur à disposition, il est impératif d'y installer un serveur web. Toute l'installation et le transfert des fichiers se feront par SSH. Kitty est donc utilisé pour cela. C'est un client SSH qui permet d'établir la connexion avec le serveur. Cette machine n'a que l'OS installé de base. Afin de pouvoir y héberger un site internet, il est obligatoire d'installer un serveur web sur le serveur. Afin d'avoir tous les composants nécessaires pour la gestion d'un site internet, la plateforme Linux, Apache, Mysql, PHP a été choisie. Afin de réaliser cela, une marche à suivre [31] a été suivie. Il est important de vérifier les ports ouverts et d'autoriser les connections entrantes. Pour cela, la commande : sudo ufw allow in "Apache Full" est utilisée. Il est important de vérifier le bon fonctionnement des différents composants avant de continuer. PHP est testé en ajoutant ce code : $<$?php phpinfo() ; ?$>$ dans la page info.php présente à la racine. La page est appelée et affiche les informations présente dans cette figure21

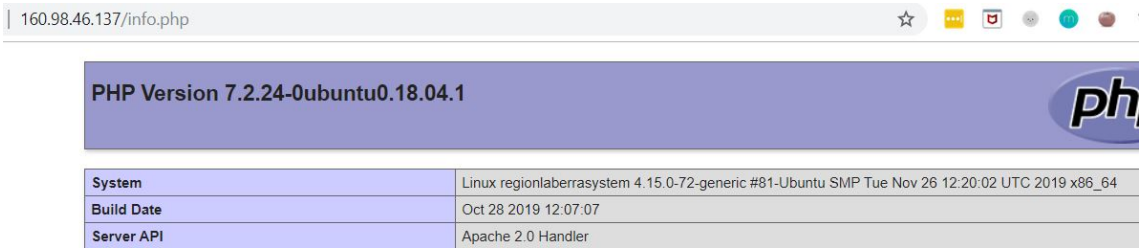


FIGURE 21 – Informations fournies par info.php

Il est également important de tester la connexion de MySQL. Pour cela, il est important d'essayer de s'y connecter avec la commande. sudo mysql -u username -p. C'est un succès.

Il est également nécessaire d'installer phpmyadmin afin de permettre la gestion des bases de données. Ceci a été réalisé avec une autre marche à suivre disponible sur internet [32]. Afin de pouvoir se connecter à PhpMyAdmin, il faut également accepter les connections entrantes vers le port utilisé par PhpMyAdmin qui est le 3306 [22]. Pour cela, une demande a été réalisée afin d'ouvrir ce port sur la machine virtuelle.

```
celestin.rumo@regionlaberrasystem:/$ sudo ufw allow 3306
Rule added
Rule added (v6)
celestin.rumo@regionlaberrasystem:/$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
22                         ALLOW IN    Anywhere
8069                       ALLOW IN    Anywhere
80                         ALLOW IN    Anywhere
443                        ALLOW IN    Anywhere
80,443/tcp (Apache Full)   ALLOW IN    Anywhere
3306                       ALLOW IN    Anywhere
22 (v6)                    ALLOW IN    Anywhere (v6)
8069 (v6)                  ALLOW IN    Anywhere (v6)
80 (v6)                    ALLOW IN    Anywhere (v6)
443 (v6)                   ALLOW IN    Anywhere (v6)
80,443/tcp (Apache Full (v6)) ALLOW IN    Anywhere (v6)
3306 (v6)                  ALLOW IN    Anywhere (v6)
```

FIGURE 22 – Autorisation du port 3306

Il est ensuite possible de se rendre sur la page de connexion de PhpMyAdmin. 23

FIGURE 23 – Autorisation du port 3306

4.1.9 Jalon 3 - Installation de Wordpress

lorsque le serveur est mis en place, il est possible de passer à l'installation de Wordpress. Le site est actuellement réalisé avec Wordpress. Une autre marche à suivre [33] est disponible afin de réaliser cette étape. La création d'une base de données relative à WordPress est obligatoire avec la commande : CREATE DATABASE wordpress DEFAULT CHARACTER SET utf8 COLLATE utf8_unicode_ci Il est nécessaire de créer un utilisateur avec tous les accès afin de pouvoir gérer la base de données : GRANT ALL ON wordpress.* TO 'wordpressuser'@'localhost' IDENTIFIED BY 'password' Afin d'autoriser Wordpress d'utiliser le fichier .htaccess, il est nécessaire de modifier le fichier apache2.conf afin d'autoriser de passer outre la règle de base avec la configuration suivante : <Directory /var/www/html/> AllowOverride All </Directory> Il est ensuite nécessaire de continuer l'installation selon la marche à suivre précédente.

4.1.10 Jalon 3 - Importation d'une copie du site internet de l'ESS la Berra

La récupération de la base de données ainsi que des fichiers Wordpress depuis l'hébergeur se sont fait depuis le cPanel de l'hébergeur pour la base de données et avec une connexion en ftp afin de récupérer les fichiers présents dans le html.
Un problème a été rencontré ici car le logiciel de gestion de base de données est MariaDB sur l'hébergeur du site de l'ESS la Berra. Afin d'éviter tout problème dû à cela, le gestionnaire a été changé sur le serveur mis en place pour le projet. La démarche[34] est simple et rapide. La base de données ainsi récupérée peut être importé dans la base de données existante sur le serveur avec PhpMyAdmin. Pour importer tous les documents relatifs à Wordpress, un client ftp est utilisé, Filezilla dans le cadre de ce projet. Afin d'utiliser un port ouvert, sftp est utilisé. Les documents sont importés dans le répertoire /var/www/html/
Le répertoire s'appelant public_html dans le serveur source, c'est important d'utiliser un utilitaire afin de modifier tous les chemins. La cli de wordpress[35] est installée pour mettre à jour tous ces chemins avec la fonction search-replace.
Le document wp-conf.php24 doit être modifié afin de contenir les informations correctes de la base de données du serveur du projet.

```
define( 'DB_NAME', 'wordpress' );

/** MySQL database username */
define( 'DB_USER', 'root' );

/** MySQL database password */
define( 'DB_PASSWORD', 'berraRegion1234' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The Database Collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );
```

FIGURE 24 – Fichier de configuration wp-conf.php

Pour mettre à jour le bon thème, il est possible de lister les thèmes disponibles avec la commande : wp theme list et d'activer un de ces thème avec la commande : wp theme activate "theme".

Problème rencontré
Wordpress fonctionne avec des tables pour les différentes parties du site internet. Malheureusement, en utilisant toutes les tables présentes dans le site original de l'ESS la Berra, une erreur apparaît et bloque le site internet. Afin de contourner cela et de n'utiliser que le strict nécessaire, les tables indispensables ont été renommées de telle manière à ce qu'elles soient utilisées par Wordpress pour la mise en place du site internet. Les deux tables sont :

- La table Post pour les pages
- La table terms pour le menu

L'index d'origine des tables du site original de l'ESS la Berra présent à chaque début de nom de table est : "wp_1". L'index de l'installation sur le nouveau serveur web avec Wordpress installé dessus est "w_". Les deux tables précédemment nommées ont donc été renommées avec l'index "wp_". Cela a permis d'avoir une installation minimale du site internet de l'ESS la Berra.

4.2 Workpackage 2 - Authentification

Pour effectuer ce workpackage, le workpackage 1 (mise en place de l'environnement) doit être terminé. A la fin de ce workpackage, il sera possible de s'authentifier et de créer un compte sur le site internet.

Tous les codes de ce workpackage sont disponibles en annexe dans le dossier annexes/code/login. Dans ce rapport, se trouveront uniquement les parties de code jugées intéressantes.

4.2.1 Jalon 4 - Mise en place

Il est possible de créer de nouveau clients via l'API d'Odoo[36] et de leur donner accès uniquement aux ressources qui seront utiles pour ce projet. l'API fourni par Odoo fonctionne avec le langage python, Ruby, PHP et Java. Le site internet est fait à l'aide du CMS Wordpress qui lui utilise le langage PHP. Afin de simplifier le travail d'intégration, les nouvelles pages crées dans ce workpackage seront en PHP, par conséquent les requêtes vers l'API Odoo aussi.

Comme dit précédemment, le site est construit à l'aide du CMS WordPress. Ce qui rend la tâche plus compliquée que prévue. WordPress est un très bon CMS fait pour aider à construire un site web plus ou moins rapidement à l'aide d'une interface graphique. Cependant, lorsque des fonctions spéciales ou des pages personnalisées veulent y être intégrées, il faut au préalable créer des plugins ou des templates en fonction des besoins.

Afin de simplifier le développement de pages internet de ce workpackage, un serveur wamp64 a été installé. De cette manière, les différentes requêtes peuvent être testées sur l'ordinateur de travail et seront ensuite ajoutées au serveur web dès que les tests seront validés.

4.2.2 Jalon 4 - Installation de l'environnement local

Les ordinateurs utilisés pour travailler sur ce projet utilisent un système d'exploitation Windows (version 10). C'est pourquoi une installation wamp64 sera installée plutôt que xampp. Wamp64 est spécialement conçu pour les systèmes d'exploitation Windows.

Pour lancer l'installation, il suffit de se rendre sur le site `http://www.wampserver.com/` et d'installer la version correspondante au système sur laquelle elle va être installée.

C'est dans le dossier "www" disponible sous : "disque C –> wamp64 –> www" que le dossier de travail va être déposé.

4.2.3 Jalon 4 - Diagrammes de séquence

Dans ce chapitre, les deux diagrammes correspondant au processus de connexion et de création de compte seront présentés. Ceux-ci représentent les différents acteurs impliqués dans le processus ainsi que les différents échanges qu'ils ont entre eux. Dans ces diagrammes, la page "historic" est renvoyée au clients sans interaction avec le

serveur Odoo, ceci est pour simplifier le schéma. En réalité, il y a une interaction avec le serveur, mais ce processus est décrit dans un workpackage dédié à la création de cette page (chapitre 4.4.2).

Diagramme de Connexion

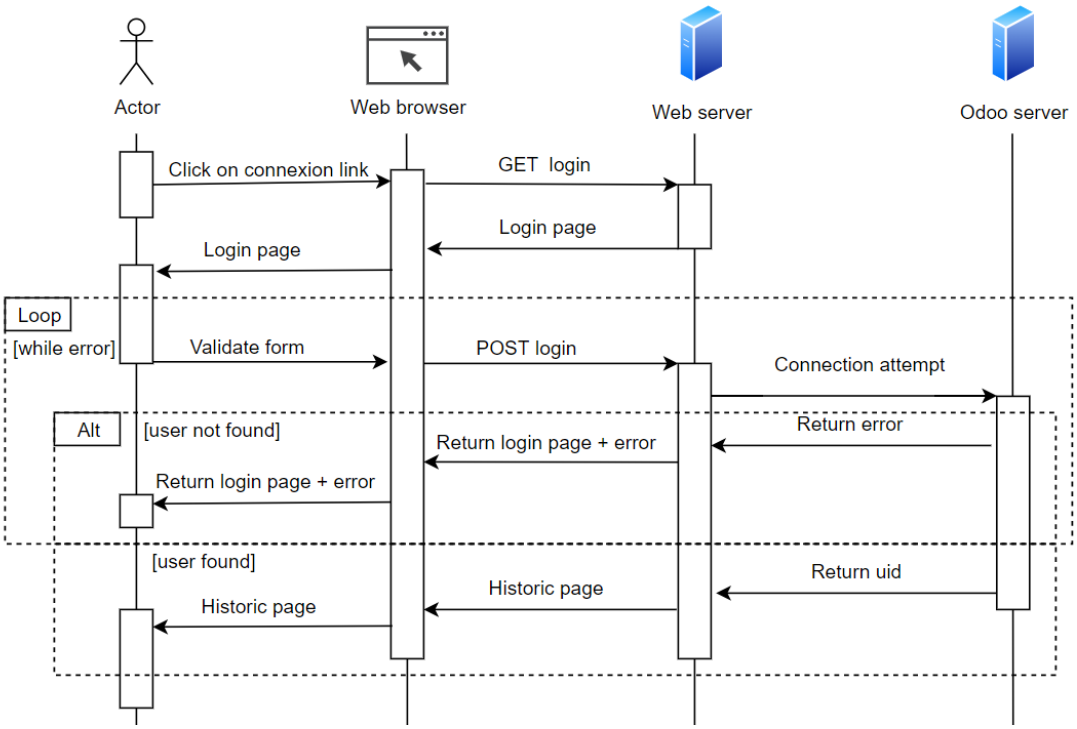


FIGURE 25 – Diagramme séquence : connexion

Diagramme de Création de compte

Dans le diagramme ci-dessous, on remarque que la page de création de compte n'est pas directement accessible. L'utilisateur doit d'abord se rendre sur la page de connexion sur laquelle il trouvera un lien qui le redirigera vers la page lui permettant de se créer un compte.

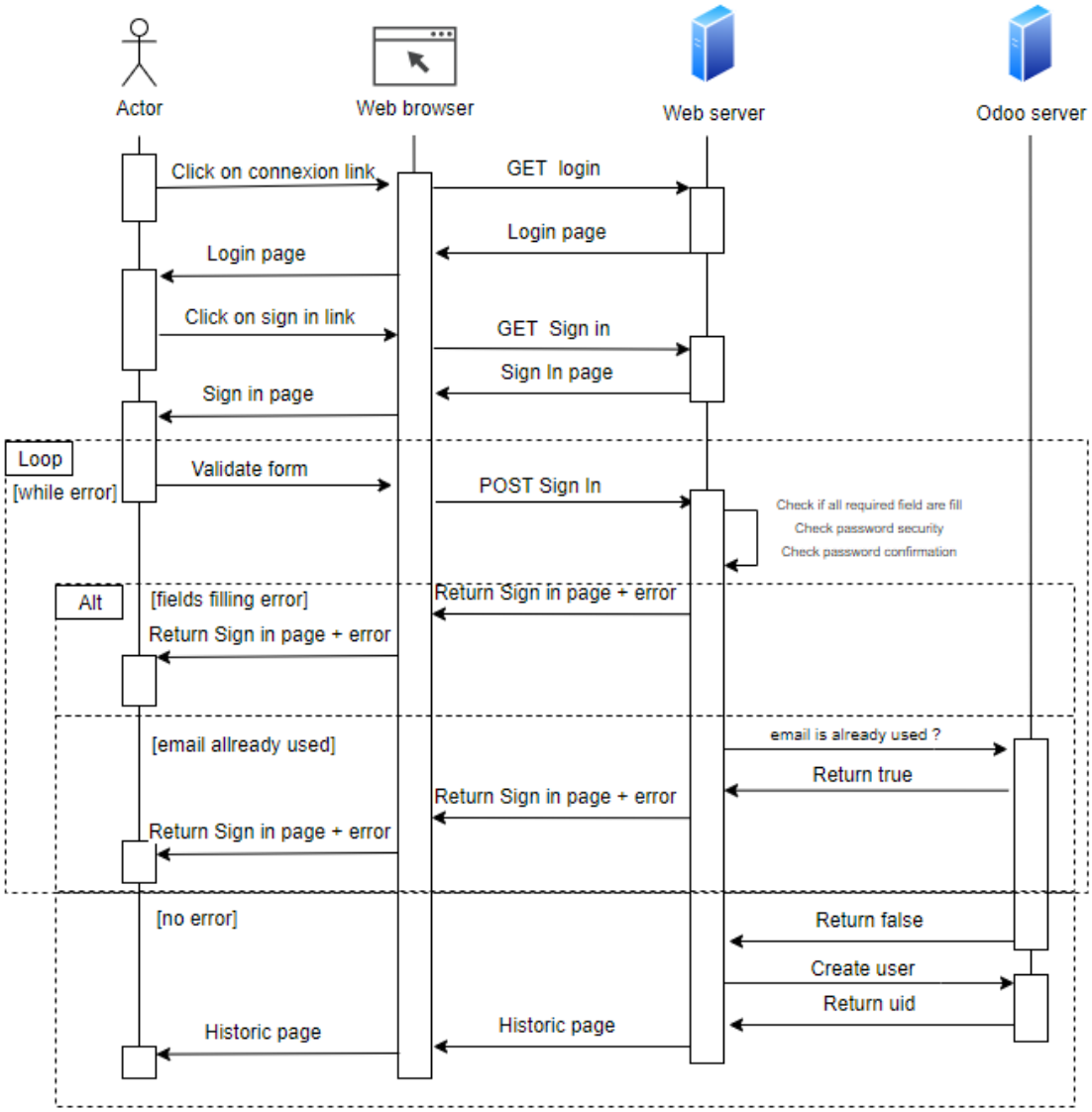


FIGURE 26 – Diagramme séquence : création d'un compte

Pour alléger le travail du serveur web au moment du traitement du formulaire, dans un premier temps celui-ci vérifiera toutes les données dont le processus de traitement est local au serveur et ne nécessite aucune interaction avec d'autre acteur du diagramme. Il traitera dans un premier temps si tous les champs requis sont remplis, ensuite la qualité du mot de passe en termes de sécurité, et le mot de passe de confirmation. Ces variables sont vérifiables en interne grâce à des fonctions implémentées en PHP et intégrées à la page web.

De cette manière, le serveur peut retourner au client qu'il a fait une erreur lors de la complétion du formulaire d'inscription sans avoir besoin d'effectuer une requête sur le serveur Odoo. C'est uniquement lorsque toutes les variables qui peuvent être vérifiées en interne seront validées que le serveur effectuera une requête vers le serveur Odoo pour obtenir les informations qui permettront de terminer le processus de validation.

4.2.4 Jalon 4 - Connection via l'API

Comme expliqué précédemment, le langage PHP sera utilisé. Cependant pour que PHP puisse interpréter le protocole XML-RPC[37] il est obligatoire d'installer la librairie "Ricord" disponible sur le site "`https://github.com/poef/ripcord`" de plus il faut aussi que PHP supporte XML-RPC.

Pour ce faire il faut d'abord savoir quelle version de PHP est utilisée sur la machine. La commande "php -v" permet d'avoir la réponse. Dans ce projet, la version PHP utilisée est la 7.2, celle-ci n'est plus active depuis novembre 2019. Cependant, des mises à jour sont encore faites jusqu'au 30 novembre 2020, uniquement pour des problèmes lié à la sécurité. Il est donc préférable de mettre à jour le site internet de la Berra dans les prochains mois. Cette tâche aurait pu être réalisée dans ce projet, malheureusement le temps de développement est compté. Pour avoir un résultat minimal, il faudra négliger cette étape qui n'est pour l'instant pas une contrainte pour se concentrer sur d'autre tâches. Dès que la version est connue, il suffit d'entrer la commande suivante avec la version PHP correspondante à celle présente sur le serveur.

sudo apt-get install php7.2-xmlrpc

XML-RPC
XML-RPC est un protocole RPC qui permet à des processus s'exécutant dans des environnements différents d'interagir avec le serveur. Celui-ci permet d'appeler une fonction avec n'importe quel langage de programmation. Cela permet de fournir un service web utilisable par tout le monde sans restriction de système ou de langage.

Pour s'authentifier via l'API en PHP il faut inclure la libraire Ripcord(ligne 13).

```php
8   if(isset($_POST['submit']) ) {
9       $username = $_POST['email'];
10      $password = $_POST['pwd'];
11
12      //Connexion initialisation
13      require_once('./ripcord/ripcord.php');
14      $common  =  ripcord :: client ( $url.'/xmlrpc/2/common');
15      $uid  =  $common -> authenticate ( $db ,  $username ,  $password ,  array ());
16      // fetch object structure
17      $models = ripcord::client("$url/xmlrpc/2/object");
18      // check if user exist
19      if ($uid == false) {
20        echo "Wrong Password or email";
21      }else{
22
23        echo ("Welcome");
24        $_SESSION['uid'] = $uid;
25        $_SESSION['password'] = $password;
26      }
27
28  }
```

FIGURE 27 – Code pour l'authentification

Dans un premier temps, il faut différencier les 2 moyens d'arriver sur la page de login.

1. Lors de la première requête de la page (lorsque l'utilisateur souhaite entrer ses données pour se connecter).
2. lors que le client a validé son formulaire de connexion (avec une requête de type POST).

La vérification faite à la ligne 8 correspond à cette réflexion. En récupérant les données du formulaire une tentative de connexion est effectuée avec celles-ci. La fonction "authenticate" présente à la ligne 15 retourne l'id de l'utilisateur en cas de connexion réussie et 0 en cas d'échec. C'est pour cette raison qu'un test est fait à la ligne 19 afin d'informer l'utilisateur que la connexion a échoué.

La variable superglobale `"$_SESSION"` est utilisée afin de transmettre les informations de connexion aux autres pages qui en auront besoin.

A cette étape du projet, lorsque la connexion est effectuée un message Bienvenue est affiché sur la page, en attendant d'avoir créé les autres pages.

4.2.5 Jalon 4 - Mise en place du formulaire d'inscription

La procédure pour la création de compte est plus ou moins similaire à la connexion à un compte à quelques détails près. Un formulaire et une vérification de la requête POST seront aussi faits.

Afin de garantir une meilleure sécurité, il est important de gérer tous les type d'erreur que pourrait faire un nouvel utilisateur lors de la création de son compte, comme

par exemple, utiliser une adresse email déjà enregistrée sur la plateforme, ne pas confirmer correctement son mot de passe, ou choisir comme pays de résidence un pays inexistant.

Vérification du mot de passe
Afin de garantir une meilleure sécurité aux nouveaux utilisateurs, une vérification de la robustesse du mot de passe est effectuée lors de la création d'un compte. Pour réaliser cette fonction le code présent sur la page "`https://www.codexworld.com/how-to/validate-password-strength-in-php/`" a été utilis, les figures ci-dessous en sont un aperçu.

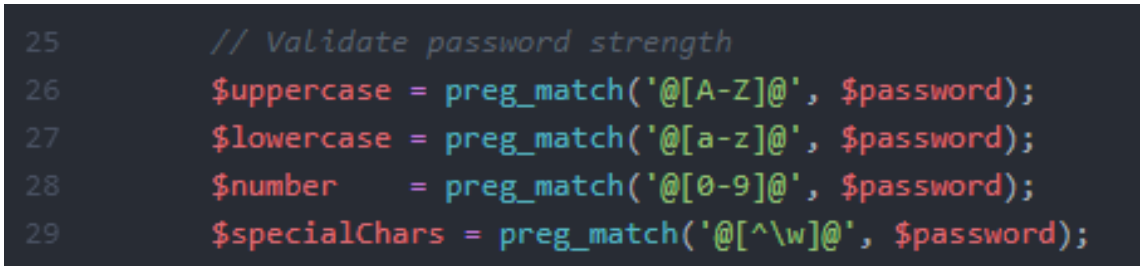


FIGURE 28 – Critère de complexité du mot de passe

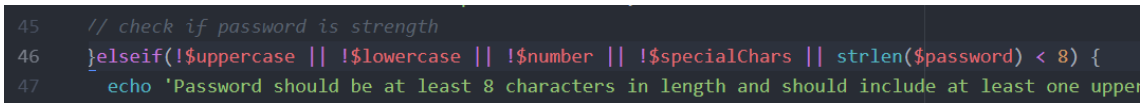


FIGURE 29 – Vérification du mot de passe

4.2.6 Jalon 4 - Création des nouveaux utilisateurs via l'API

Avant de pouvoir créer un compte via l'api fourni par Odoo, il faut comprendre comment ceux-ci sont construits. Il faut aussi comprendre les fonctions fondamentales qui permettent d'interagir avec les données d'Odoo.

Structure des modèles utilisateurs
Odoo différencie plusieurs types de contact, il y a les compagnies, les utilisateurs, et les partenaires. Une personne qui ne représente que quelqu'un ayant eu contact avec l'entreprise (paiements, facturation, message) sera considérée dans Odoo comme un partenaire. Dès lors que celui-ci se crée un compte avec un mot de passe, celui-ci devient un utilisateur. Une compagnie représente une entité qui lie plusieurs partenaires.

Il faudra donc créer un utilisateur lorsqu'une personne remplit le formulaire de création de compte. Un utilisateur est caractérisé par un model dans Odoo appelé : "res.users" et contient 193 attributs. Certains d'entre eux sont des attributs auxquels une valeur est automatiquement attribuée lors de la création d'un nouvel utilisateur, d'autres sont disponibles uniquement à la lecture. Tous ces attributs sont

disponibles sur le portail Odoo dans "Settings–>Technical–>Models–>res.users". Pour ce projet, ce sont uniquement les champs suivants qui seront utilisés.

- name = String
- login = String
- email = String
- password = String
- street = String
- city = String
- phone = String
- `country_id` = référence sur le pays par son id
- `groups_id` = tableau contenant les référence de groupe par leur id.

Fonction utiles
L'API fourni par Odoo permet la mise en place d'un système dit "SCRUD" (search, create, read, update, delete).Ces fonctions sont respectivement :

- 'search'
- 'create'
- 'read'
- 'write'
- 'unlink'

Requête de vérification d'email
Cette requête doit être faite à l'aide des informations de connexion de l'administrateur. C'est uniquement lui, qui peut voir tous les utilisateurs de la plateforme et ainsi vérifier si l'émail proposé par le futur client est disponible ou non.

```
30      // Check email request
31      $check_email = $models->execute_kw($db, $uid, $password,
32      'res.users', 'search',array(array( array('email', '=', $email)))); // display criteria
```

FIGURE 30 – Vérification de l'email

La fonction 'search' retourne un tableau vide si aucune adresse email telle que celle qu'on essaie d'enregistrer existe. C'est uniquement lorsque ce test sera validé que l'utilisateur sera créé à l'aide du code suivant.

```php
$user_id = $models->execute_kw($db, $uid, $password,
    'res.users',
    'create', // Function name
    array( // Values-array
        array( // First record
            'name'=>$name,
            'login'=> $email,
            'email'=>$email,
            'password' => $pwd,
            'street'=>$street,
            'city'=>$city,
            'phone'=>$phone,
            'country_id' => $country_id[0]['id'],
            'groups_id' => array(8,22)
        )
    )
);
```

FIGURE 31 – Création d'un client via l'api

Sur la figure ci-dessus on remarque le champ **'login'** celui-ci représente le nom d'utilisateur pour se connecter à son compte. Pour ce projet, l'email sera utilisé comme tel. Toutes les informations qui seront récupérées de Yoplanning par le biais de l'api sont basées sur la fait que Yoplanning utilise l'email des clients comme ID. C'est pourquoi, le même système sera utilisé pour le projet. De cette manière le processus de vérification est amélioré et surtout allégé.

Groupe de sécurité
Odoo gère les accès au portail et aux ressources via des group d'accès. Sur la figure ci-dessus (figure31), on remarque que l'utilisateur, aura les accès au groupe 8. Celui-ci représente la possibilité de se connecter au portail Odoo, et de voir les paiements qui lui sont dédiés.

4.2.7 Jalon 5 - Conception des formulaires

Ci-dessous sont présentés les deux mockups qui représenteront ce à quoi devra ressembler la page web une fois intégrée au site WordPress. A noter que ces représentations ne sont pas les versions définitives. Les pages vont être intégrées sur un site disposant déjà d'un thème. Le but sera donc d'y intégrer les pages en respectant le style du thème actuel. Le but de ces mockup est d'avoir une ligne directrice pour la structure des pages.

Mockup connexion

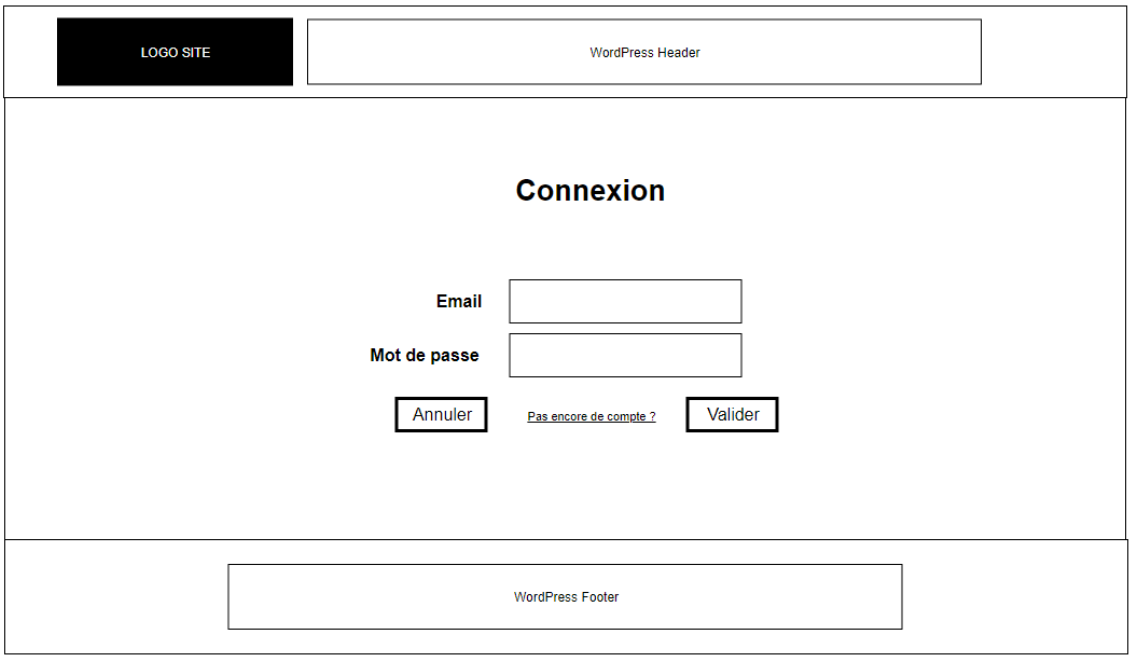


FIGURE 32 – Mockup formulaire de connexion

Mockup Créer un compte

FIGURE 33 – Mockup formulaire d'inscription

4.2.8 Jalon 5 - Mise en place du frontend

Pour la partie Frontend le code pour le formulaire de connexion ou de création de compte a été récupéré sur le site de W3School (`https://www.w3schools.com/howto/howto_css_login_form.asp`). Pour la partie modification de style, elle interviendra plus tard. La page sera intégrée sur un site web disposant déjà d'un thème, c'est pourquoi les pages seront dans un premier temps intégrées au site pour voir les effets du thème actuel sur celle-ci et des modifications seront ajoutées en conséquence.

```
<h1>Login</h1>
<form action="index.php" method="post">
  <div class="imgcontainer">
    <img src="img/profile.png" alt="Avatar" class="avatar">
  </div>

  <div class="container">
    <div class="email field">
      <label for="email"><b>Email</b></label>
      <input type="text" placeholder="Enter Email" name="email" required>
    </div>
    <div class="Password field">
      <label for="psw"><b>Password</b></label>
      <input type="password" placeholder="Enter Password" name="pwd" required>
    </div>
    <div class="login">
      <button type="button" class="cancelbtn">Cancel</button>
      <button type="submit" name="submit">Login</button>
    </div>
    <div class="signin">
      <a href="add.php">pas encore de compte ?</a>
    </div>
  </div>
</form>
```

FIGURE 34 – Code du formulaire de login

Grâce à la classe field qui a été ajouté, il est possible de rapidement effectuer des modifications de style en CSS sur la plupart des éléments en même temps. Le code CSS est disponible dans "annexes/code/login/css".

4.2.9 Jalons 4 et 5 - Ajout d'une page personnalisée sur WordPress

Pour ajouter du contenu personnalisé sur un site WordPress il faut d'abord définir quel type sera la personnalisation. En effet, en général, on utilise un plugin pour modifier des fonctionnalités du site, et des templates pour en modifier l'apparence. Mais faire l'inverse est aussi possible. Le risque en ajoutant des modifications de fonctionnement au travers d'une template, est que lors d'un changement de thème

les templates n'étant plus disponibles, les modifications qu'elle apportait ne le seront par conséquent plus non plus.

Étant donné que les pages connexion et inscription modifient le fonctionnement du site il faudrait réaliser un plugin. Cependant il faut bien plus de temps pour créer un plugin qu'une template. Le temps de développement restant est restreint, c'est pourquoi pour avoir un minimum de résultat d'ici la fin, les pages seront créées sous forme de template pour gagner du temps.

structure du répertoire

Les template créées dans ce projet seront placées dans le dossier correspondant au thème actuellement actif sur le site de la Berra (wpzoom-balance). Dans le cas de ce projet le chemin d'accès à ce répertoire est `/var/www/html/wp-content/themes/wpzoom-balance/`.

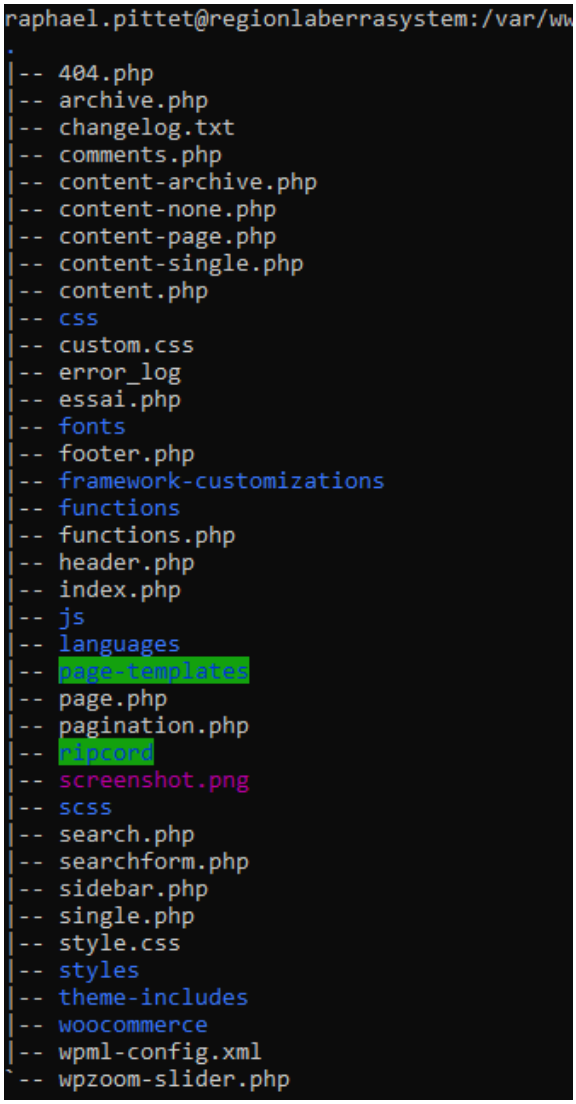


FIGURE 35 – Structure du thème actif

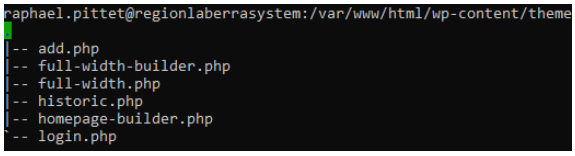


FIGURE 36 – Structure du dossier "page-template"

La figure 35 représente l'état du dossier actuellement, En temps normal, le dossier ripcord n'est pas présent, un paragraphe est disponible dans ce chapitre qui explique comment ajouter ce dossier. C'est le dossier "page-template" qui sera exploité pour ajouter des pages personnalisées. C'est pour cette raison que sur la figure 36, les fichiers "login.php", "add.php", "historic.php" sont présent.

Mode debugging WordPress
Avant de commencer toute manipulation dans les fichiers WordPress il est préférable d'activer le mode debugger [38]. Celui-ci permet d'afficher les logs d'erreur sur la page directement concernée ou sur un fichier dédié. Pour cela il faut éditer le fichier "wp-config.php" présent dans le dossier racine du site. A la ligne où se trouve l'instruction "define('WP_DEBUG', false);" il faut remplacer le paramètre "false" par "true". La figure suivante est un aperçu du fichier wp-cong.php après l'activation de mode débuguer.

```
/**
 * For developers: WordPress debugging mode.
 *
 * Change this to true to enable the display of notices during development.
 * It is strongly recommended that plugin and theme developers use WP_DEBUG
 * in their development environments.
 *
 * For information on other constants that can be used for debugging,
 * visit the Codex.
 *
 * @link https://codex.wordpress.org/Debugging_in_WordPress
 */
define( 'WP_DEBUG', true );
```

FIGURE 37 – Template login

Créer une template
La création d'une template sur WordPress est très rapide en effet il suffit d'ajouter la ligne suivante sur la page qu'on souhaite définir comme template.

< ?php /* Template Name : NAME */ ?>

Celle-ci doit bien évidemment être un fichier php. Ainsi en ajoutant cette page dans le répertoire "page-templates" celle-ci sera déjà considéré comme une template utilisable lors de l'Edition d'une page sur la page d'administration du site internet.

Cependant il est possible d'ajouter d'autre champs qui informeront par exemple sur l'auteur de la page. L'image ci-dessous montre de quelle manière ceci a été intégré sur la page login.

```php
<?php
/*
Template Name: LOGIN
 * Custom template used for custom php code display
 * @author    Pittet Raphael
 */
?>
```

FIGURE 38 – Template login

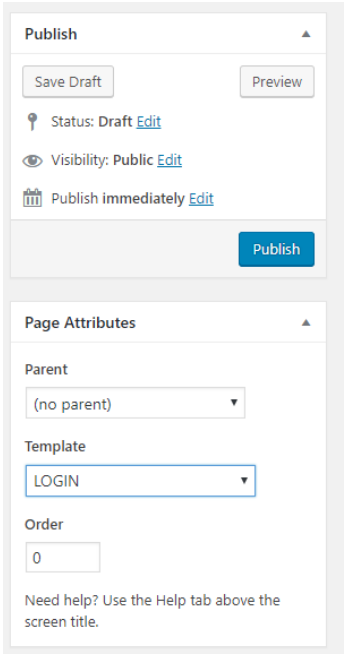


FIGURE 39 – Utilisation de la page dans l'admin

Attention ! La page doit contenir uniquement le contenu qui devra être affiché dans le body de la page et ne doit pas être construite avec les balises html, head, body. Si des fichier css, ou des scripts php tel que la librairie ripcord doivent être intégrés, il faut les intégrer à la page via des fonctions fournis par WordPress. Les paragraphes suivants expliqueent plus en détails comment le faire. La template LOGIN ressemble donc à la figure ci-dessous avec du code php à la place du commentaire du même nom.

```php
<?php
/*
Template Name: LOGIN
 * Custom template used for custom php code display
 * @author    Pittet Raphael
 */

 // PHP CODE

?>
  <h1>Connexion</h1>
  <form action="./login" method="post">
    <div class="container">
      <div class="email field">
        <label for="email"><b>Email</b></label>
        <input type="text" placeholder="Email" name="contact-email" required>
      </div>
      <div class="Password field">
        <label for="psw"><b>Mot de Passe</b></label>
        <input type="password" placeholder="Mot de passe" name="pwd" required>
      </div>
      <div class="login">
        <button type="button" class="cancelbtn">Annuler</button>
        <button type="submit" name="submit">Connexion</button>
      </div>
      <div class="signin">
        <a href="../enregistrement/">pas encore de compte ?</a>
      </div>
    </div>
  </form>
```

FIGURE 40 – Rendu de la template : à l'étape 1

ajout des scripts PHP

Pour ajouter un script js ou php sur une template, WordPress fournit une fonction appelé "`get_template_part()`"[39]. Lorsque cette fonction est appelée par une "page-template" WordPress va rechercher dans le répertoire principal du thème un fichier correspondant à celui entré comme paramètre dans la fonction.

Sur toutes les pages qui seront créées dans ce projet, la librairie "ripcord" sera nécessaire. Comme présenté dans le chapitre 4.2.4, cette libraire était importée à l'aide de la fonction php require(). Désormais, celle-ci devra être appelée de la manière suivante :

```
get_template_part('./ripcord/ripcord');
```

En incluant le scripts php de cette manière, on évite d'éventuelles erreurs, mais surtout on conserve une structure en accord avec le fonctionnement du CMS. Le chemin d'accès fournit dans l'exemple ci-dessus correspond au chemin d'accès à utiliser si le dossier ripcord est déposé au même niveau que sur la figure35.

WordPress fournit aussi deux autres fonction similaires à la précédente Cependant, celles-ci sont exprès faites pour ajouter le footer et le header. En se référant à la mockup de la section 4.2.7 (Jalon 5 - Conception des formulaires), il faut ajouter le header et le footer déjà présents sur le site de la berra afin de rester dans le même univers. La figure suivante représente le résultat à cette étape du projet.

```php
<?php
/*
Template Name: LOGIN
 * Custom template used for custom php code display
 * @author    Pittet Raphael
 */

 // PHP CODE
get_header();

?>
  <h1>Connexion</h1>
  <form action="./login" method="post">
    <div class="container">
      <div class="email field">
        <label for="email"><b>Email</b></label>
        <input type="text" placeholder="Email" name="contact-email" required>
      </div>
      <div class="Password field">
        <label for="psw"><b>Mot de Passe</b></label>
        <input type="password" placeholder="Mot de passe" name="pwd" required>
      </div>
      <div class="login">
        <button type="button" class="cancelbtn">Annuler</button>
        <button type="submit" name="submit">Connexion</button>
      </div>
      <div class="signin">
        <a href="../enregistrement/">pas encore de compte ?</a>
      </div>
    </div>
  </form>
<?php get_footer(); ?>
```

FIGURE 41 – Rendu de la template : à l'étape 2

ajout du CSS

Pour ajouter un fichier de style tel que CSS dans une template il y a 2 manières. La première consiste à utiliser les fonctions fournies par WordPress de la même manière que pour les scripts mais avec la fonction wp_enqueue_style(). Le lien suivant fournit plus de détails concernant cette fonction.`https://developer.wordpress.org/themes/basics/including-css-javascript/`.

La deuxième consiste à ne pas ajouter de style via le fichier lui-même mais depuis l'interface administrateur. C'est cette méthode qui sera utilisée pour ce projet. En effet, le site de la Berra a déjà un thème qui définit plus un moins le style de tout le type d'éléments intégrable dans une page web.

Cette manière de procéder permet d'intégrer dans un premier temps la page sur le site afin d'avoir un aperçu des résultats avec les modifications faites par le thème en général, et de pouvoir adapter en fonction de cela l'apparence finale de la page. Pour réaliser ces modifications il faut se connecter à l'outils d'administration et aller dans –> appearence –> customize —> additionnal CSS. Tous les fichiers css ajouté sur le site WordPress sont disponibles en annexe dans annexes/code/login/css

autre modifications à apporter

Les deux pages créées dans ce projet contiennent des formulaires ; ce qui implique que lorsque celui-ci est validé (posté), une action doit être faite. Dans le cas de

ces formulaires l'action est une simple redirection sur la même page en précisant le chemin relatif à la page actuelle (ex : ./nom-fichier.php). A parti du moment où ces page ne sont plus utilisées comme telles mais comme template, ajouter un chemin relatif n'est plus possible car la page en elle-même ne sera pas générée dans le dossier page-template. Cependant WordPress fournit la encore une fonction qui permet de récupérer l'url d'une page dynamiquement au moment de sa génération. Cette méthode est appelée "_get_page_link()" est a donc été utilisée de la manière suivante :

```
33    <form action="<?php _get_page_link();?>" method="post">
```

FIGURE 42 – Utilisation de la méthode _get_page_link

transfert des fichiers sur le serveur
Pour transférer les fichiers sur la machine virtuelle, le protocole scp est utilisé. Les fichiers sont envoyés depuis un dossier local, sur un serveur distant. Comme cité dans le paragraphe "structure du répertoire" les pages php créées doivent être insérées dans le dossier "page-templates" (figure35) disponible sous `/var/www/html/wp-content/themes/wpzoom-balance/`.

Pour envoyer un fichier avec le protocole scp depuis un dossier local vers un serveur distant, la commande se fait ainsi :

```
scp C:\wamp64\www\test\login.php raphael.pittet@160.98.46.137:/var/www/html/wp-content/themes/wpzoom-balance/page-templates/
```

FIGURE 43 – commande scp

Le chemin `C:/wamp64/www/test/login.php` représente le répertoire local.
"raphael.pittet@160.98.46.137" est le nom d'utilisateur ainsi que l'adresse IP du serveur distant. On ajoute ensuite après les " :" le chemin représentant le dossier de destination sur la machine virtuelle. Durant cette phase un problème est intervenu concernant les droits d'accès, le paragraphe problème rencontré traite du sujet et de la solution à utiliser.

Pour transférer un répertoire complet via scp, c'est le cas de la librairie ripcord il faut ajouter -r qui signifie qu'il va transferer de manière récursive jusqu'à avoir envoyé tout le répertoire.

Pour savoir dans quel dossier transférer les templates ainsi que la librairie, il faut se référer aux exemples donnés plus haut dans ce chapitre3536.

Création des pages sur WordPress
A cette étape du projet, les fichiers ont été édités de manière à fonctionner en tant que template sur WordPress et ont été placés dans les bons répertoires. Il faut se rendre sur la panneau d'administration du site et ajouter une page. Il ne faut pas éditer du contenu il faut juste donner un titre à cette page (pour la reconnaitre plus simplement) et sélectionner les template qui viennent d'être ajoutées. La figure 39 montre où cette sélection peut être faite.

Problèmes rencontrés
Lors du transfert des fichiers en scp sur la machine virtuelle, une erreur est survenue car les accès au dossier cible était trop restreints. Dans ce genre de situation, il suffit d'accorder les droit d'accès au dossier de réception. avec la commande "sudo chmod 777 [nom-fichier|dossier], qui permet à tous les utilisateurs de modifier le fichier/dossier, attention, cependant, à restaurer les droits après avoir transféré les fichiers.

Une autre erreur qui a généré un peu de retard sur le projet concerne une erreur de redirection. En effet, lorsqu'un formulaire est envoyé à un site WordPress (ce qui est le cas de ce projet), si la requête POST contient des variables telles que : name,email,subject,content [40], le formulaire est détourné par WordPress qui essaie d'en faire un commentaire. En effet, lorsque le formulaire était validé et envoyé, la redirection était faite sur la bonne URL, mais le message d'erreur " error 404" signifiant que la page n'est pas trouvée, était affiché à la place du contenu souhaité. Pour régler ce problème, il a fallu renommer ces variables autrement dans le cadre de ce projet, la transformation suivante a été sélectionné "name" en "contact-name".

4.2.10 Jalons 4 et 5 - Tests

Afin de valider les deux pages effectuées dans ce workpackage, une procédure de test est réalisée. L'objectif est que le client puisse créer un compte sur la plateforme internet et aussi se connecter s'il est déjà membre.

Rendu final des pages

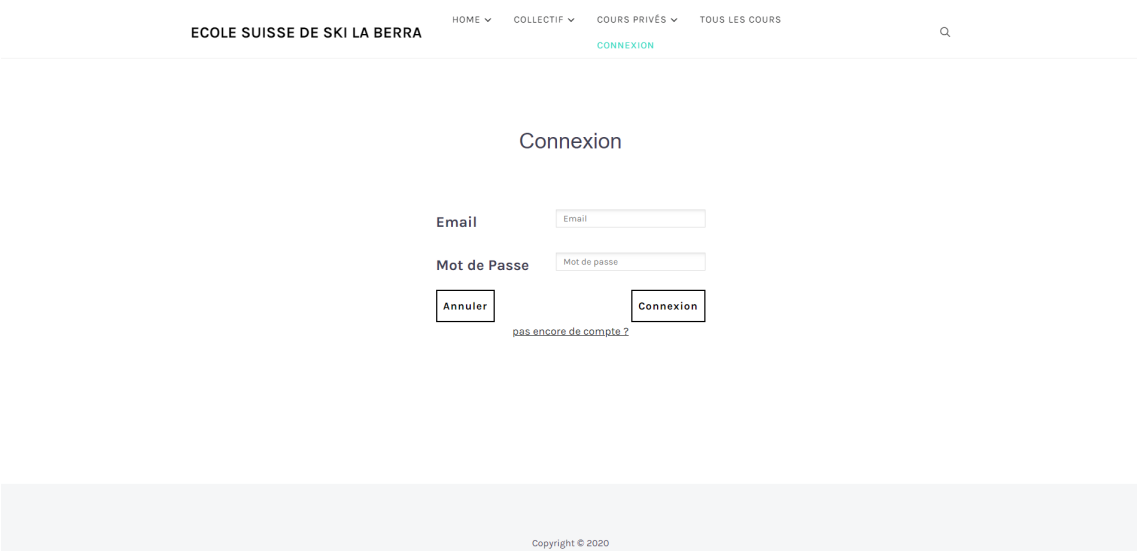


FIGURE 44 – Rendu page de connexion

FIGURE 45 – Rendu page de connexion

Test page login

Num.	test	Attendu	Resultat
1	Mauvaise combinaison	Erreur(mdp ou email)	Erreur(mdp ou email)
2	Bonne combinaison	connexion	connexion
3	Aucun compte	Erreur(mdp ou email)	Erreur(mdp ou email)

TABLE 13 – Tests page LOGIN

Test page login

Num.	test	Attendu	Resultat
1	Tout les champs ne sont pas remplis	Erreur	Erreur
2	Mot de passe pas assez sécurisé	Erreur(MDP trop léger)	Erreur(MDP trop léger)
3	Mot de passe de confirmation faux	Erreur(MDP confirmation)	Erreur(MDP confirmation)
4	Faux pays	Erreur(faux pays)	Erreur(faux pays)
5	email déjà utilisé	Erreur(email déjà pris)	Erreur(email déjà pris)

TABLE 14 – Tests page ADD

Les deux pages qui ont été testées sont fonctionnelles et couvrent les objectifs des tests. Les deux pages sont disponibles dans en annexe dans : annexes/code/login.

4.3 Workpackage 3 - Traitement des données de YoPlanning à Odoo

4.3.1 Préparation de l'environnement de travail

Le language de programmation python sera utilisé pour réaliser la liaison entre YoPlanning et Odoo. Ce language a beaucoup de librairies utiles pour travailler avec des données. Afin de pouvoir travailler sur windows, il est nécessaire d'installer Python. La dernière version ayant des problèmes de compatibilité avec certaines libraires qui ne sont pas encore mises à jour, il est nécessaire d'installer la version 3.7 de Python ou plus ancienne. Le fait d'avoir installer la version 3.8 au début a créé une perte de temps dû au temps nécessaire pour réaliser cette incompatibilité des versions de Python avec certaines de ces librairies. Dans le cadre du projet, cela s'est produit avec la librairie de Jupyter. Il permet d'utiliser l'application web Jupyter directement dans Pycharm
Une fois Python installé, il est nécessaire d'installer PIP qui est un outil très connu pour installer des paquets Python. Pour cela il faut télécharger un code python et le lancer à l'aide de la commande : Python get-pip.py.[41]
Après avoir installé PIP, il est possible d'installer toutes les librairies nécessaires : requests, jsonpath, json, ipython, xml-rpc. Pour installer un package/ librairie, il est nécessaire d'ouvrir une invite de commande et de tapper la commande : pip install "nom du paquet désiré". Les librairies requests, jsonpath et json sont les librairies qui vont être utilisées durant ce projet. Elles offrent chacune des outils nécessaires afin de récupérer et de traiter le type de données qui sera utilisé durant le projet.

- requests : La librarie requests offre une manière élégante et plus compréhensible pour réaliser des requêtes HTTP. Elle est très pratique à utiliser pour réaliser des requêtes sur une API API par exemple. C'est pour cette raison que la librairie sera utilisée.
- JSONPath : JSONPath est une librairie utilisée pour accéder à divers éléments de données au format JSON. Elle est très utile pour simplifier la manière d'accéder à des éléments dans une donnée au format JSON.
- json : la librairie json permet de convertir un format json en fichier texte ou l'inverse, cela peut s'avérer utile, notamment pour faire des tests.

PyCharm
PyCharm est un environnement de développement intégré utilisé pour programmer en Python. Il permet l'analyse de code et contient un débogueur graphique. Il permet d'intégrer beaucoup de librairies. La version professionnelle peut être obtenue avec une licence étudiante. La version professionnelle est nécessaire afin de pouvoir utiliser le package Jupyter. Jupyter.

Jupyter
Jupyter est une application web qui permet de réaliser des calepins ou notebooks, Cela est très pratique pour réaliser des tests et s'assurer des données que nous voulons traiter. Il peut être directement intégré à PyCharm afin de réaliser les tests directement depuis PyCharm. Les tests sont réalisés afin de s'assurer du format

de donnée reçu afin de savoir comment procéder au traitement par la suite. Il est également utiliser afin d'afficher les réponses des requêtes qui sont créées. Différents notebooks sont disponibles sur le répertoire afin de voir tous les tests effectués. [42]

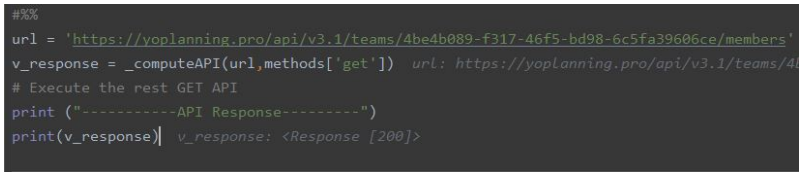


FIGURE 46 – Exemple de code python dans le l'envrionnement de développement Pycharm, utilisant un un notebook Jupyter

La figure 46 permet de voir qu'il s'agit d'un code python et que Jupyter permet d'afficher directement les données qui sont dans les variables. Cela crée un environnement favorable de travail afin de pouvoir s'assurer de l'exactitude des données reçues.

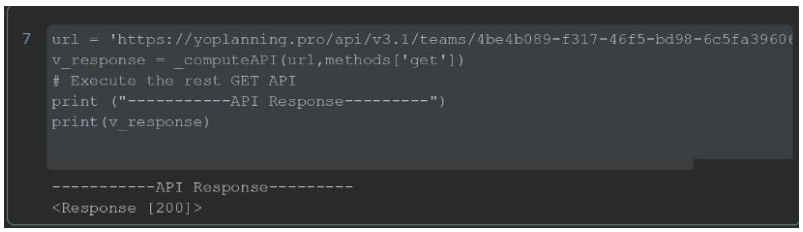


FIGURE 47 – Affichage du résultat du code exécuté directement dans Pycharm en utilisant un fichier Jupyter

La figure 47 permet de voir le résultat. La ligne "#%%" permet de séparer le fichier en plusieurs parties afin de pouvoir séparer le fichier et de lancer uniquement une partie du code afin de cibler les tests.

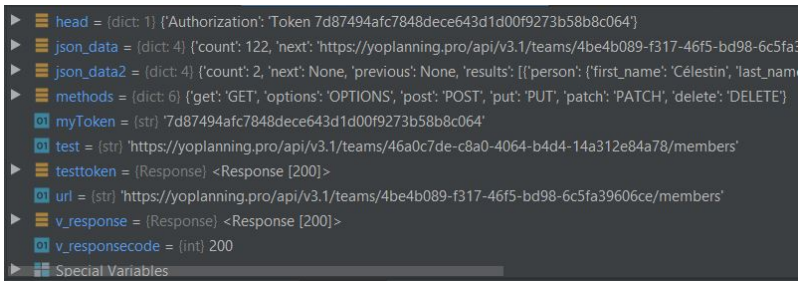


FIGURE 48 – Affichage des variables

La figure 48 permet d'avoir un aperçu de toutes les valeurs contenues dans les variables afin de débugger en cas de problème.

4.3.2 Jalon 6 - l'API de YoPlanning [43] est testée pour voir les données reçues

Afin de pouvoir utiliser l'API de YoPlanning, un token d'identification est nécessaire. Celui-ci est fourni par la société elle-même. Malheureusement, le token fourni n'était pas le bon et il ne permettait pas d'accéder à toutes les ressources nécessaires dans le cadre du projet. Il était impossible de réaliser des requêtes sur des ressources de l'école de ski. Les tests ont donc été faits sur une deuxième "Team" qui est Kayak aventure.

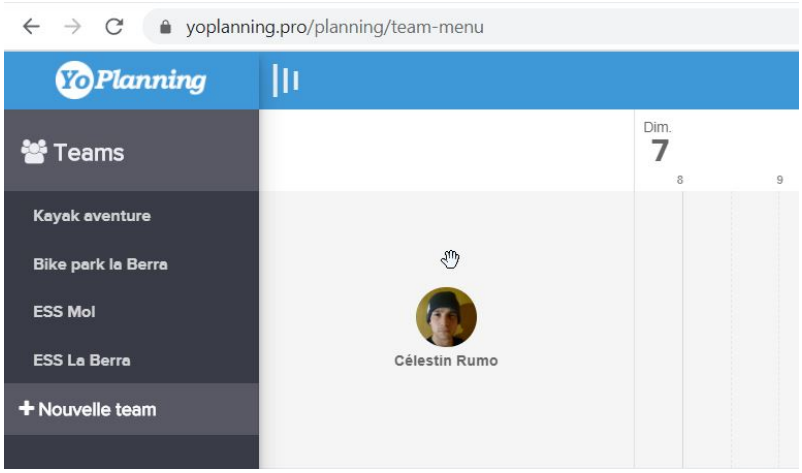


FIGURE 49 – Affichage des teams dans l'application web de YoPlanning

La figure 49 est l'affichage des teams sur lesquels un certain utilisateur a les droits. Cette affichage est l'affichage de l'application web mis à disposition par YoPlanning.

Le projet est cependant basé sur l'API de YoPlanning. Il est donc primordial de prendre connaissance de la documentation de l'API fourni par YoPlanning [43]. La première étape afin de pouvoir réaliser des tests est de comprendre le fonctionnement de l'API avec l'aide de la documentation. Cela permet de mettre en avant quelques points importants :

1. l'API est basé sur le style architectural REST (representational state transfer) définissant un ensemble de contraintes à utiliser pour créer des services web. [44]
2. l'authentification se fait avec un token qui permet d'envoyer des requêtes à l'API.
3. les différentes ressources sont identifiées avec l'aide d'un UUID généré avec la version 4 de UUID qui permet de générer un chiffre totalement aléatoire. [45]
4. les webhooks ou web callback est un système de notification afin d'être averti lorsqu'un évènement type survient sur une team spécifique. Lorsque l'évènement est enclenché une requête POST est envoyé à l'url défini par la personne ayant demandé à s'abonner.

5. L'API référence est le chapitre de la documentation permettant de prendre connaissance de type de requêtes à effectuer afin de récupérer les informations souhaitées.

.

Il est possible de commencer les tests de l'API après avoir pris connaissance de la documentation. La première étape est de récupérer les teams sur lesquelles nous avons des droits avec le token reçu. La référence API est : /api/v3.1/teams/. Une méthode générique de test a été réalisée par Monsieur Guibert et sera reprise afin de réaliser les tests. [46]

(a) Le code python permettant de récupérer les teams sur lesquelles le token octroie des droits

(b) la réponse émise directement dans Pycharm permettant de récupérer l'UUID

FIGURE 50 – Figures montrant l'utilisation de l'API

Les différents fichiers pouvant être récupérés de l'API sont disponibles dans le répertoire prévu à cet effet. [47]. Le temps de test afin de pouvoir récupérer toutes les informations nécessaires a malheureusement été très lent dû au délai de réponse très long de la part de YoPlanning. Le token fourni par leur soins n'avait malheureusement pas les droits suffisants pour réaliser les requêtes nécessaires. Les accès nécessaires ont été délivrés uniquement deux semaines avant la fin du projet. (15 Février). Le temps nécessaire afin de réaliser les tests a été rallongé dû aux problèmes causés par le manque de droit du token donné par YoPlanning. Du temps afin d'essayer de trouver d'où venait le problème a en effet été perdu car le bloquage dû au mauvais token a pris du temps pour être découvert. Le message d'erreur étant indiqué dans la documentation laissait à penser à un mauvais format de requête. Ce qui n'était pas le cas.

4.3.3 Jalon 6 - le problème de la synchronisation entre les logiciels est expliqué

Toutes les informations de réservation doivent être transmises depuis YoPlanning à Odoo. Il y a différentes situations possibles :

— le client n'a pas de compte et c'est sa première commande.
— le client n'as pas de compte mais a déjà réalisé une commande auparavant.
— le client a un compte qu'il a créé avant sa première réservation
— le client a un compte qu'il a créé après avoir déjà réalisé une ou plusieurs réservation(s). Il ne faut donc afficher que les commandes réalisées après la création du compte client.

Ces 4 situations différentes ne changent aucunement le diagramme de séquence ci-dessous. En effet, tous les nouveaux clients ainsi que toutes les commandes doivent être transmises à l'ERP Odoo afin de pouvoir réaliser des statistiques. Il est nécessaire de récupérer toutes les informations concernants les clients et les commandes afin d'avoir une vue d'ensemble. Chaque commande nécéssite un client déjà créé au préalable. Dans le cadre du projet, les différentes situations sont gérées.

— un nouveau client est créée.
— une nouvelle commande est créée.
— une commande est modifiée.
— une commande est supprimée.
— un paiement est créé.

La clé primaire d'un client est définie par son adresse email. Il y a deux raisons à ce choix.
— Le client qui veut créer un compte sera identifié avec son adresse email et son mot de passe. Il n'est pas possible d'avoir deux comptes liés à la même adresse email.
— Un utilisateur n'ayant pas encore créé de compte pourra réserver plusieurs cours pour des personnes différentes. la clé primaire d'un utilisateur n'est pas lié à son adresse email pour YoPlanning mais à son nom, prénom et adresse email. Chaque clé primaire se retrouve donc avec un UUID différent alors que les membres font partie de la même famille. Les familles peuvent être retrouvées avec l'adresse email utilisée.

4.3.4 Jalon 7 - Le choix de la technologie utilisée est décrit

L'environnement de travail a déjà été décrit dans la section du Jalon 6. Le choix quant au language de programmation va être approfondi ainsi que le framework utilisé. Les deux sont liés et il est important de définir les raisons de ces choix.

Python
Un des objectifs du projet est de créer une liaison entre le logiciel métier (YoPlanning) et l'ERP (Odoo). Afin de réaliser cela, une application web doit être mise en place. Il y a plusieurs languages qui sont très utilisés pour le développement Backend de ce type. Les deux plus utilisés actuellement sont Python et Javascript. La manipulation des données JSON est un élément à prendre en compte pour le choix du language utilisé.
l'ERP Odoo est un programme open source travaillant exclusivement en python en backend. L'affinité du développeur a également un poids dans le choix du language

de développement de l'application web.
Les points suivants ont donc été retenus afin de retenir Python comme language pour le développement de cette application :

- La manipulation des données JSON. Python offre en effet plusieurs librairies facilitant la manipulation des données étant au format JSON. Ces différentes librairies sont décrites dans le chapitre 4.3.1.
- Odoo est un ERP écrit en python. Il est donc judicieux d'utiliser le même language afin de prévoir la suite du projet et un développement durable en utilisant un seul et unique language de programmation pour le Backend du projet.
- L'affinité du programmeur a également une importance dans le choix du language. La prise en main des données JSON a été faite en utilisant Python. C'est donc un argument en plus en faveur de ce language de programmation.

Afin de pouvoir communiquer efficacement avec une API RESTfull, il est nécessaire d'installer un framework python qui permet de faire cela. Plusieurs framework sont disponibles et permettent de récupérer des requêtes REST. Le plus léger et modulaire est le Micro-framework Flask [48]. Flask permet de dispatcher les requêtes reçues en fonction de la méthode et de l'url sur lequel la requête est reçue. Cela est une caractéstiques clé de ce micro-framework.

Flask
Flask est un framework permettant de réaliser des applications web. Il est utilisé dans le cadre du projet pour un de ses outils qui est le dispatcher de requêtes RESTfull. Cet outil permet d'avoir un API endpoint. Cet endpoint est requis afin de recevoir les requêtes POST envoyées depuis l'API de YoPlanning lorsqu'il y a un nouvel évènement qui est survenu. Le micro-framework Flask est donc le point d'entrée de notre système et permet de récupérer les informations nécessaires afin de les transmettre plus loin pour traiter correctement l'information reçue.

```
@app.route('/api', methods=['POST'])
def api():
    clientup_data = request.get_json()
    response_data = {
        "sucess": True,
        "status_code": 200,
    }
    ypm.check_data_inside(clientup_data)
    return response_data
```

FIGURE 51 – Le point d'entrée réalisé avec le micro-framework Flask

La figure 51 montre à quel point il est facile de mettre en place un service web permettant de récupérer une donnée émise par un serveur distant sur l'url défini

par notre méthode. L'utilisation finale de ce micro-framework sera expliquée plus en détail dans le chapitre 4.3.6

4.3.5 Jalon 7 - Première importation des données présentes dans YoPlanning

Il avait été initialement prévue de récupérer toutes les informations déjà présentes dans les bases de données YoPlanning. Cette étape a été abandonnée car trop lourde. La quantité des données étant très grande, il faut réaliser énormément de test en amont afin de garantir le transfert efficace et correct des données d'un système à l'autre. Cette étape à elle seule pourrait faire l'objet d'un travail de semestre tant sa mise en place est complexe car elle demande une fiabilité totale. En outre, toutes les possibilités doivent être prises en compte et traitées afin de ne pas avoir d'erreur lors de la manipulation. Or, dans le cadre du projet, comme mentionné dans le capitre 4.3.3 nous ne gérons que 5 situations différentes parmi toutes les situations possibles.

4.3.6 Jalon 7 - Gestion de la synchronisation entre les deux systèmes

Un Callback ou webhook est un système de notification sur certains type d'évènements qui peut être mis en place. A chaque fois que l'évènement est déclenché, une requête POST est envoyé à un URL spécifié par l'utilisateur avec les informations nécessaires au format JSON. Une requête GET est ensuite effectuée afin de récupérer l'information et la traiter correctement en fonction de l'évènement. Ce système de notification est mise à disposition par YoPlanning comme expliqué au chapitre 4.3.2

les requêtes POST envoyé depuis l'API de YoPlanning arrivent toujours sur le même url. La requête contient un élément JSON comme dans la figure 52 présente ci-dessous.


```
{
    "resource":
        { "id": "ec39fef2-b6ed-471f-82a9-d84790d36731",
          "type": "client"
        },
    "link": "https://yoplanning.pro/api/v3.1/teams/46a0c7de-c8a0-4064-b4d4-14a312e84a78/clients/ec39fef2-b6ed-471f-82a9-d84790d36731",
    "event": "created"
}
```


FIGURE 52 – Le format JSON reçu lors d'une requête POST de YoPlanning

Une suite de manipulation est réalisée à la suite de la réception du POST sur l'url de notre application web. La vue d'ensemble de l'application est démontrée dans la figure 53 présente ci-dessous. Chaque élément faisant partie de ce diagramme de séquence est développé plus bas afin d'en comprendre la logique et l'utilité. La figure 55 est un diagramme plus complet expliquant plus précisémment chaque étape pour la création d'un client.

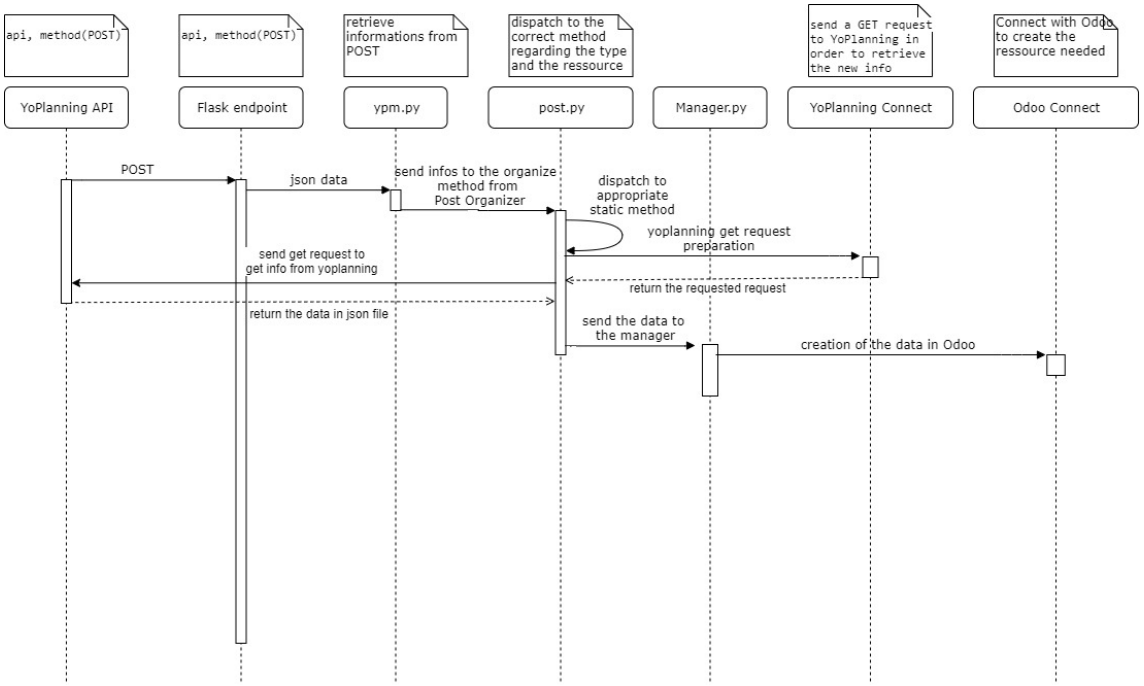


FIGURE 53 – Le diagramme de séquence de l'application web

Flask endpoint
La figure 51 permet de voir comment le endpoint est géré et la manipulation qu'il exécute afin de transmettre les données reçues plus loin. Il récupère le fichier json transmis dans le Body du post et le stock dans une variable. Cette variable est ensuite envoyée plus loin afin de traiter les données.

ypm.py
La donnée JSON reçue précédemment doit ensuite être séparée afin de récupérer les différents éléments composant ce POST. L'object transmis par YoPlanning 52 contient 3 éléments : .

1. une ressource qui est composé d'un objet contenant deux paires
   (a) id. L'id de la la resource qui a été modifié, c'est cet élément qui permet d'identifier de manière unique quel élément a été modifié dans le logiciel YoPlanning.
   (b) type. Le type de la resource modifié. Il permet de savoir quel est le type de l'élément modifié. Cela permet de connaître quelle référence dans la documentation de l'API de YoPlanning il est nécessaire d'utiliser.
2. link. Le lien qui permet d'envoyer une requête GET afin de récupérer les données ajoutées ou modifiées est directement envoyé par YoPlanning. Il est ainsi plus facile de générer la requête nécessaire à la récupération des données.
3. event. L'event permet de savoir quel type d'évènement a été levé. Il peut s'agit d'une création, suppression ou modification.

Ces différents éléments permettent de définir la manière dont devra être traitée l'information. Un objet est créé avec ces éléments et l'objet et ensuite envoyé plus

loin afin de définir comment gérer l'évènement qui est survenu.
La méthode organize est donc appelée avec les attributs post (l'objet), l'event et le type de la donnée modifiée.

post.py
En fonction du type de donnée et de l'event, une méthode est appelé.54. Elle permet d'appeler une méthode permettant de réaliser un "GET" sur l'API de YoPlanning afin de récupérer l'information sur laquelle il y a eu une alerte. Le formatage de la requête est réalisé par un autre fichier "yoplanningConnect.py" afin de pouvoir l'appeler depuis d'autres endroit en cas de besoin.
les données JSON ensuite récupérées sont transmises plus loin afin de traiter les données correctement en fonction de l'évènement et du type de la resource.

```
@staticmethod
def _serialize_new_client(post):
    infoYp = ypc.requestCreator(post.getLink())
    # new method in client manager to be able to create object + set all the attribut necessary
    cm.manageNewClient(infoYp)
    return
```

FIGURE 54 – Méthode statique appelé

Manager.py
Le fichier Manager est séparé en fonction du type de la donnée. Il y a donc 3 fichiers différents :

1. clientManager
2. orderManager
3. paymentManager

Chacun de ces fichiers contient la logique pour gérer la logique de gestion propre à chaque type et chaque évènement.
Les fichiers orderManager et paymentManager ne contiennent que la méthode de création de l'objet en question. Il a en effet été impossible de gérer les commandes et les paiements comme expliqué dans le chapitre suivant 4.3.8 Le clientManager est fonctionnel et contient plusieurs fonctions.

1. la méthode manageNewClient est appelé depuis la méthode statique 54
   (a) Un objet client est créé.
   (b) une méthode est appelé afin de vérifier si l'émail du client existe déjà dans l'ERP.
   (c) Si le client n'existe pas encore, alors le client est créé. et sinon rien ne se passe.

Afin de réaliser la connexion nécessaire avec l'ERP, un fichier a également été créé afin de pouvoir l'appeler depuis plusieurs endroits si nécessaire. Ce fichier est décrit ci-dessous.

OdooConnect.py
Ce fichier permet de créer un objet pour la connexion avec l'API externe de Odoo. un autre standard est utilisé. il s'agit du standard xmlrpc qui est totalement différent du RESTfull.

Diagramme pour le clientManager
Le diagramme suivant permet d'avoir une vue d'ensemble de l'intéraction des différents fichiers plus en détail et résumant le chapitre ci-dessus. Ce diagramme permet de visualiser les différentes étapes et est une aide pour la lecture du chapitre ci-dessus afin d'en améliorer la compréhension.

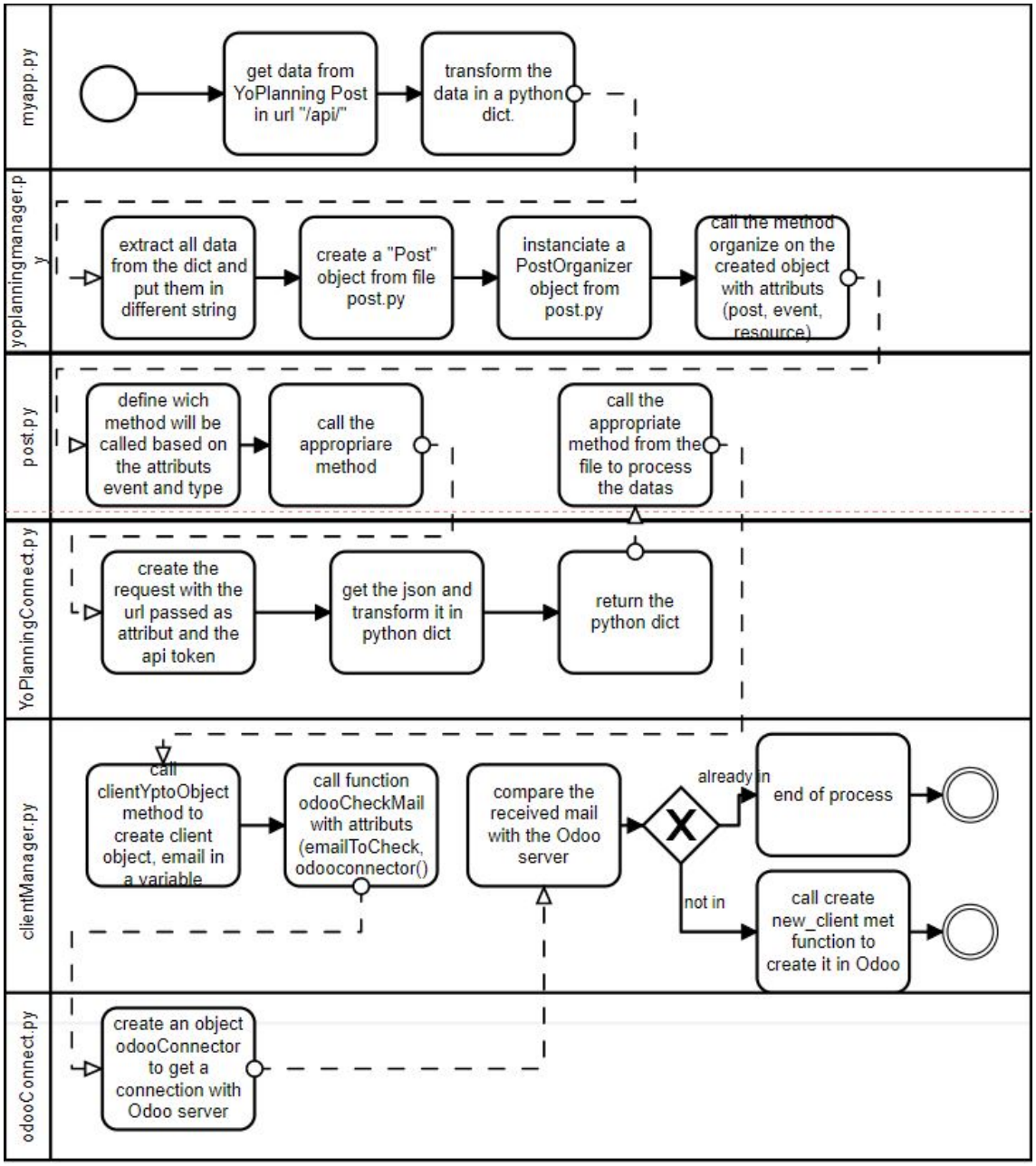


FIGURE 55 – Diagramme représentant les intéractions pour la création d'un client

4.3.7 Jalon 8 - Test

Afin de tester l'application web en local, une application permettant de générer des requêtes a été utilisée. Cette application porte le nom de Postman. [49]

Cette application permet de définir quel type de méthode utilisé (GET, POST, etc...), l'url. Le format de donnée dans le body peut être spécifié (JSON) dans notre cas et la réponse est directement perçue avec l'affichage en dessous.

FIGURE 56 – Information concernant l'url et la méthode à envoyer

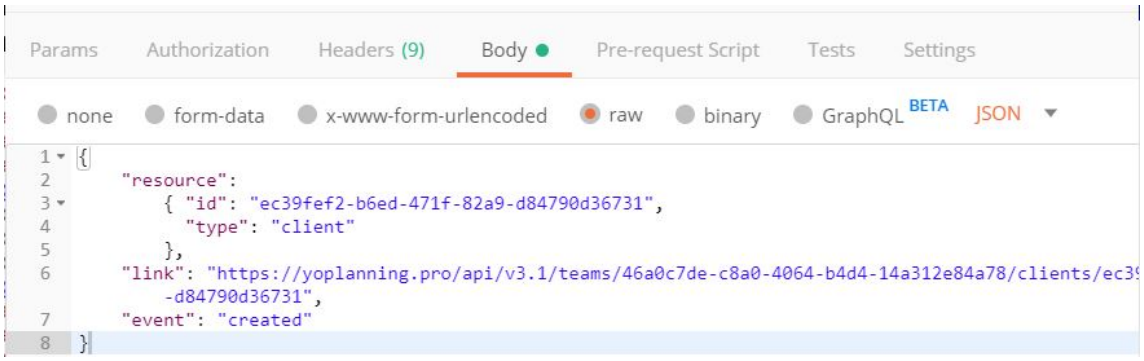


FIGURE 57 – Information concernant le body inclus dans la requête

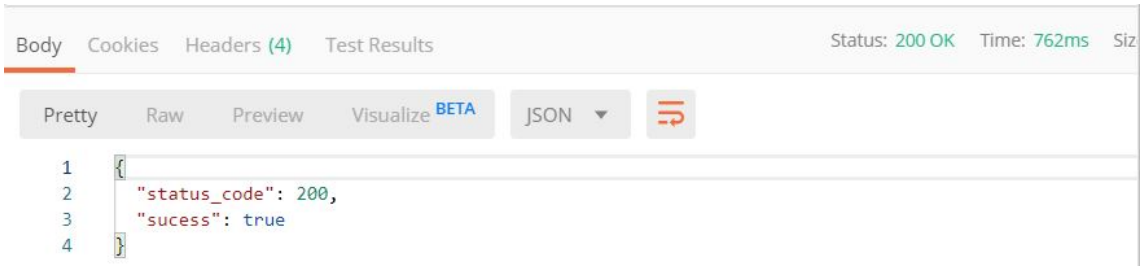


FIGURE 58 – Réponse de la requête

Afin de tester l'existence d'un client, il a été nécessaire de créer un client manuellement dans YoPlanning et de récupérer son ID afin de pouvoir "simuler" une notification arrivant sur notre application web. La figure 59 permet de voir que le test en fonction de l'adresse email est réalisé avec succès.

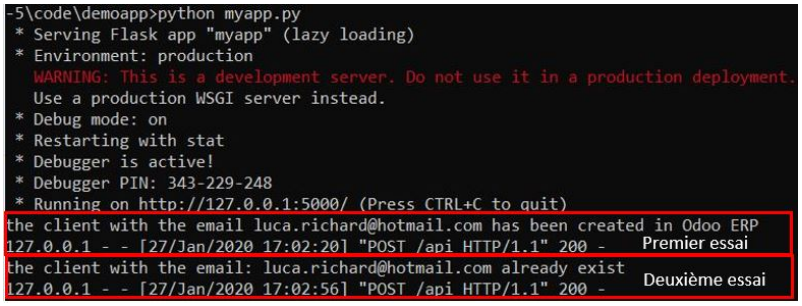


FIGURE 59 – Log dans le terminal de l'application web

La figure 60 ci-dessous montre le client nouvellement créé à partir de la notification YoPlanning envoyé vers l'application web.

FIGURE 60 – Fiche client nouvellement créé dans l'ERP

Ce test prend en compte les deux situations possibles ; soit le client a déjà un compte client, soit il n'en a pas encore un. Les deux situations sont gérées correctement. Aucuns tests unitaires n'ont été réalisé sur l'application web. La gestion des exceptions peut également être réalisée afin d'augmenter la fiabilité de l'application créée. Cela n'a pas été réalisé durant le projet par manque de temps. Les commentaires ont été insérés mais de manière générale le code est lisible et compréhensible. Le code est à disposition sur le git dans le répertoire prévu à cet effet. [50]

4.3.8 Workpackage 3 - problèmes

Les problèmes durant cet étape ont été de difficultés différentes. Toutes les sources des problèmes ont été trouvées mais uniquement certaines ont pu être traîtées dans le temps à disposition de ce projet.

problèmes résolus

1. Les problèmes liés à la prise en main, la compréhension et l'utilisation des données au format JSON ont été résolus, en partie avec l'aide d'un tutoriel disponible sur Udemy [51]
2. La prise en main de python et l'implémentation de certaines méthodes ont été facilitées et résolues grâce au tutoriel présent sur cette page. [52]
3. les autres problèmes de "Data types" ont été résolus en utilisant la console et en réalisant des "print" pour définir et cibler les problèmes afin de les résoudre.

4.3.9 Workpackage 3 - améliorations du code

Une discussion a eu lieu avec Mr Supcik afin de déterminer les améliorations possibles du code et les conventions de programmation du language Python. Les différents points ont été mis en avant :

1. Il n'y a pas de propriété privée pour les objets python. Les getters et setters n'ont donc aucune utilité dans ce language. Le concept de properties est utilisé à la place comme indiqué à cette page. [53] Ces modifications ont été apportées au code final et les getters et setters ont été enlevés car inutiles.
2. Le module Dataclass aurait pu être utilisé pour créer des objets à la place des classes normales utilisées actuellement dans l'application web. L'avantage des Data Classes est que les classes créées avec ce module contiennent certaines propriétés et fonctions de base afin de traiter plus facilement une donnée et sa représentation. Ce module est disponible depuis la version python 3.7. [54] Cette amélioration du code n'a pas été effectuée sur la version actuelle.

problème non-résolu

La création d'invoice, respectivement de commande n'a pas pu être réalisée dans le cadre du projet. Comme expliqué plus tard dans ce chapitre 4.4.4, une commande est un modèle composé d'autres modèles. Il y a donc beaucoup de dépendances comme schématisé dans la figure ci-dessous.

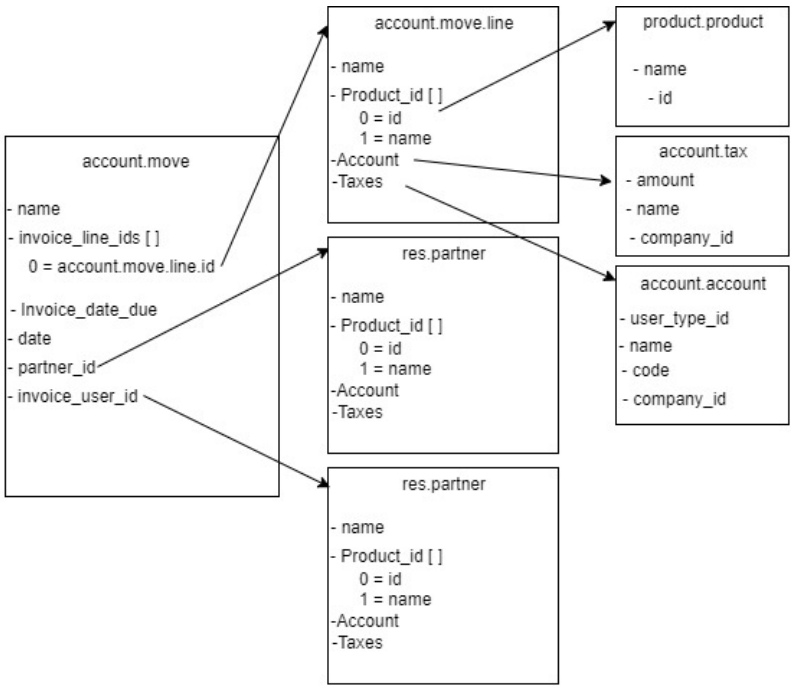


FIGURE 61 – Relation entre les modèles lors de la création d'une commande

Chaque modèle a beaucoup de champs à disposition. Il y a certains champs qui ont des caractéristiques spécifiques. Un champ peut être : required et/ou Readonly. Un tableau des modèles comportant des champs required ET Readonly est à disposition ci-dessous.

Account.move

#	Field Name	Field Label	Field Type	Required	Readonly	Type
1	currency_id	Currency	Many2one	YES	YES	Base Field
2	date	Date	date	YES	YES	Base Field
3	journal_id	Journal	Many2one	YES	YES	Base Field
4	name	Number	char	YES	YES	Base Field
5	state	Status	selection	YES	YES	Base Field
6	type	Type	selection	YES	YES	Base Field

Account.move.line

#	Field Name	Field Label	Field Type	Required	Readonly	Type
1	move_id	Journal Entry	Many2one	YES	YES	Base Field

FIGURE 62 – Champ étant required ET readonly

La source du problème se trouve ici. Nous essayons de créer un invoice en ajoutant des champs "required" et qui sont également "readonly". Or écrire un champ qui est en readonly n'est pas autorisé. Un autre inconvénient est qu'il y a énormément de liens entre les différents modèles comme vu dans la figure 61 qui ne sont pas nécessaires et complique inutilement la mise en place du système.

La solution idéale pour palier à ce problème est donc de créer un module dans Odoo spécifique à nos besoins avec les modèles jugés nécessaires à la mise en place du système pour qu'il soit opérationnel. La réalisation de ce module demande un temps conséquent et c'est la raison pour laquelle le module n'a pas été réalisé dans le cadre de ce projet. Il demande en effet une nouvelle analyse approfondie des besoins et une meilleure compréhension des modules disponibles dans Odoo.

Le problème ayant été découvert que très tard dans le projet, il a malheureusement été impossible d'y consacrer plus de temps afin de palier au problème avec succès.

4.4 Workpackage 4 - Affichage des données dans le compte client

Pour effectuer ce workpackage, le workpackage 1 (mise en place de l'environnement) doit être terminé ainsi que le numéro 2 (Authentification). Ce Workpackage est la suite du workpackage numéro 2. En effet dans ce chapitre, il va aussi falloir confectionner un template WordPress qu'il va falloir ajouter à la plateforme. Celle-ci sera accessible uniquement après s'être authentifiée sur la plateforme.

La page sera réalisée de la même manière que les 2 précédentes. Les mêmes technologies ainsi que procédure seront utilisées c'est pourquoi tout ne sera pas réexpliqué.

Le but de se workpackage est de créer une page qui permettra au client, une fois

connecté, de voir l'historique des paiements qu'il a effectué sur la plateforme. Cette page représente une partie des objectifs finaux du mandant. Le but serait de permettre au client de jouir d'un espace personnel, avec des contenus personnalisés dont un historique de ses paiements.

4.4.1 Jalon 9 - Analyse des modèles

Comme expliqué dans le Workpackage 2, Odoo stock toutes ses données sous forme de modèle. Ainsi, en se rendant sur le panneau d'administration d'Odoo dans "settings –>model" il est possible de visualiser tous les modèles stocker dans Odoo. Dans un premier temps, il faut trouver le model représentant les paiements. Celui-ci s'appelle "account.move". Pour déterminer son nom le mode développeur a été utilisé pour survoler une variable présente dans le module invoice et ainsi déterminer le nom du model dans lequel elle se trouve.

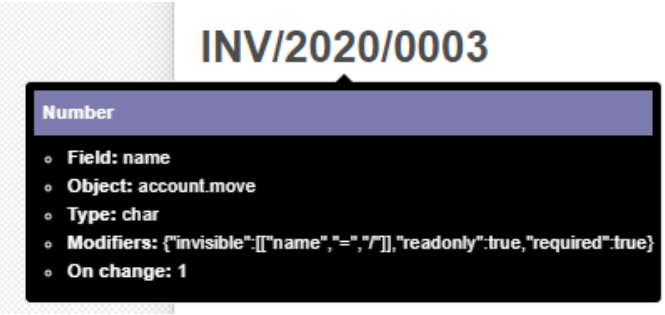


FIGURE 63 – Vérification de l'email

En se penchant plus sur le model, il est possible de visualiser tous les champs qui constituent le modèle. Ainsi on trouve 107 champs. Pour les besoins de ce workpackage ce sont uniquement les champs "name", "date", "invoice_due_date", "amount_total", "invoice_line_ids" qui seront utilisés.

Le model "account.move" est constitué de plusieurs "invoice_line_ids" ces identificateurs sont des références à d'autre modèles appelés "account.move.line" qui eux aussi références des produits par leur id, via la variable "product_id". Le seul moyen de récupérer le nom des produits pour lesquels le paiement est délivré est donc d'effectuer deux requêtes au minimum. Une première pour récupérer tout les paiements destinés au client et une autre pour récupérer le nom des produits référencés par chaque "account.move.line" de chaque "account.move".

4.4.2 Jalon 10 - Diagrammes de séquence

Ce chapitre permet de représenter tous les acteurs impliqués dans le processus de création de la page historique et les interactions qu'ils ont entre eux. Le diagramme présenté ci-dessous ne décrit pas en détails comment accéder à la page « historic ». Il décrit les différentes étapes nécessaires à sa génération. Pour savoir comment accéder à la page d'historique il faut se référer au workpackage 2.

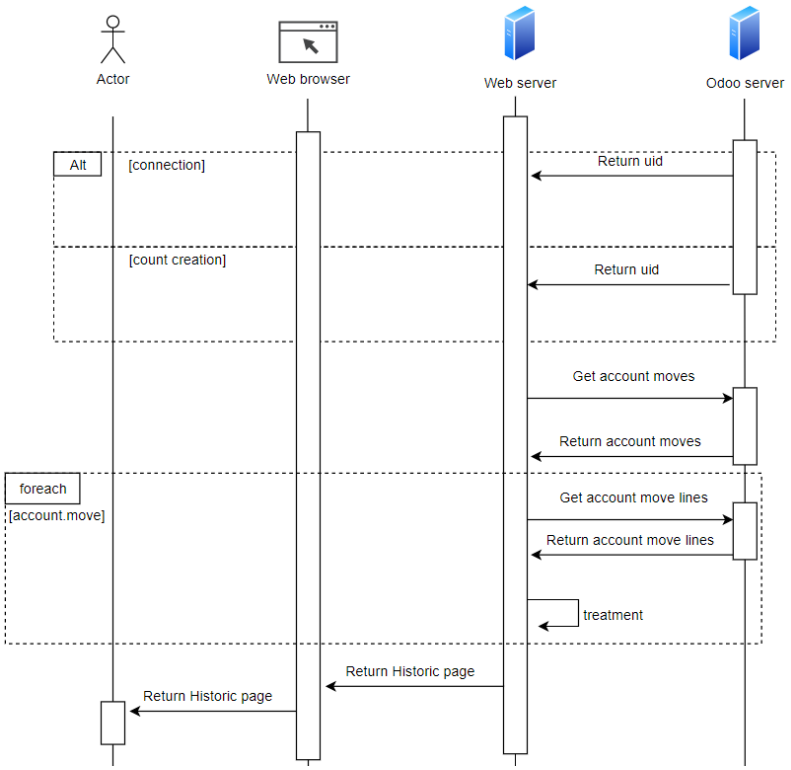


FIGURE 64 – Vérification de l'email

4.4.3 Jalon 10 - Mockup de l'affichage des paiements

La page d'historique devra elle aussi respecter la structure du thème actuel, en conservant le même footer et le même header. La mockup ci-dessous permet de donner des lignes directrices pour la réalisation du frontend.

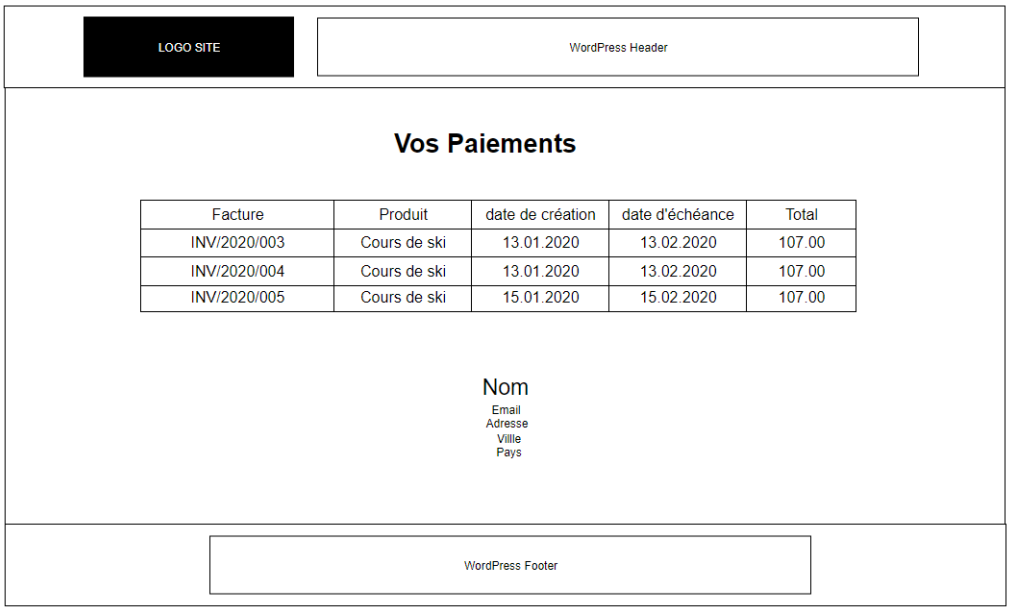


FIGURE 65 – Vérification de l'email

4.4.4 Jalon 11 - Réalisation des requêtes en Backend

Avant de commencer n'importe quel traitement ou requête, il a fallu récupérer les identifiants de connexion qui ont été transmis via les variables superglobales de PHP (`$_SESSION['uid']` et `$_SESSION['password']`). c'est uniquement après cette étape qu'une interaction est possible avec le serveur Odoo.

Récupération des données utilisateur
En se référant à la mockup de la page historique il faut récupérer les données du client afin de lui les présenter. Pour se faire il faut envoyer une requête à la plateforme avec les variables précédemment récupérées. Afin d'être sûr que le client reçoive ses informations et uniquement les siennes, la requête devra respecter le critère qui dit d'afficher uniquement les utilisateurs ayant l'id correspondant à celui du client. La figure ci-dessous représente la requête qui permet de récupérer ces informations.

```
20    //user informations
21    $perso = $models->execute_kw($db, $uid, $password,
22    'res.users', 'search_read',array(array(array('id', '=' , $uid))),
23    array('fields'=>array('name','email', 'street','city')));
24    $name = $perso[0]['name'];
25    $email = $perso[0]['email'];
26    $street = $perso[0]['street'];
27    $city = $perso[0]['city'];
```

FIGURE 66 – Récupération des données client

Récupération de l'historique des paiements
Dans un premier temps, il a fallu analyser exactement comment étaient construits les paiements sur la plateforme. Ceux-ci sont représentés par des modèles "account.move". Comme expliqué dans le chapitre précèdent, pour récupérer le nom du produit pour lequel le client à un paiement à effectuer, il est obligatoire d'effectuer deux requêtes. Le schéma ci-dessous permet de mieux comprendre pourquoi et de quelle variable les données seront exploitées.

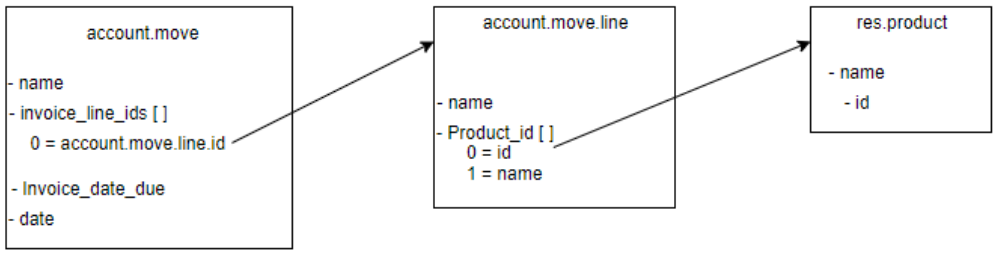


FIGURE 67 – Vérification de l'email

L'avantage est que dans les "account.move.line" la référence sur le produit n'est

pas uniquement représentée par son id mais aussi par son nom. Cela nous permet, d'éviter de faire 3 requêtes pour chaque ligne d'historique. Ainsi la requête à cette étape du projet ressemble à ceci.

```php
<?php
// search all user's invoice
$ids = $models->execute_kw($db, $uid, $password,
'account.move', 'search_read',array(array()),
  array(
    'fields' => array(
      'name','date', 'invoice_date_due', 'amount_total', 'invoice_line_ids'
      )
    )
  );
 // for each invoice
for ($i=0; $i <count($ids) ; $i++) {
  // for each invoiceline in invoice
  for ($j=0; $j <count($ids[$i]['invoice_line_ids']) ; $j++) {

  $product_id = $models->execute_kw($db, $uid, $password,
    'account.move.line', 'search_read',array(array(array('id','=',$ids[$i]['invoice_line_ids'][$j]))),
      array(
      'fields' => array(
        'product_id'
        )
      )
  );
  ?>
  <tr>
    <td><?=$ids[$i]['name']?></td>
    <td><?=$product_id[0]['product_id'][1]?></td>
    <td><?=$ids[$i]['date']?></td>
    <td><?=$ids[$i]['invoice_date_due']?></td>
    <td><?=$ids[$i]['amount_total']?></td>
  </tr>
  <?php
  }

}
 ?>
```

FIGURE 68 – Requête d'historique

4.4.5 Jalon 11 - Réalisation du frontend

La partie frontend doit permettre de s'approcher le plus possible de la mockup fournie au chapitre "Jalon 10 - Mockup de l'affichage des paiements". Il y a plusieurs manières de le faire, mais pour ce projet, une tableau a été créé. La figure ci-dessous représente uniquement le code html ajouté a cette page pour formater l'affichage sous forme de tableau. On remarquera que le code PHP a été commenté pour mieux visualiser la structure html.

```
<h1>Vos paiements</h1>

<div class="content">
  <div class="table">
  <table >
  <tr>
    <th>facture</th>
    <th>Produit</th>
    <th>Date de création</th>
    <th>Pour d'échéance</th>
    <th>total</th>
  </tr>
  <?php

  //PHP CODE{
    ?>

    <tr>
      <td><?=$ids[$i]['name']?></td>
      <td><?=$product_id[0]['product_id'][1]?></td>
      <td><?=$ids[$i]['date']?></td>
      <td><?=$ids[$i]['invoice_date_due']?></td>
      <td><?=$ids[$i]['amount_total']?></td>
    </tr>

    <?php
  // PHP END OF CODE
   ?>
</table>
</div>
<div class="side-bar">
  <div class="profil">
    <h2><?=$name?></h2>
    <ul>
      <li><?=$email?></li>
      <li><?=$street?></li>
      <li><?=$city?></li>
    </ul>
  </div>
</div>
</div>
```

FIGURE 69 – Requête d'historique

la manière d'utiliser les variable PHP présente sur la figure ci-dessus, est décrite dans le chapitre précédant.

4.4.6 Jalons 10 et 11 - Intégration de la page à WordPress

L'intégration de la page sur le site WordPress se fera de la même manière que les pages de connexion et de création de compte. Le développement total ne sera donc pas réexpliqué. Dans un premier temps, le commentaire PHP qui permet à WordPress de déterminer si la page représente une template a été ajouté (< ?php /* Template Name : NAME */ ?>). Ensuite les fonctions "get_header()" et "get_footer()" ont été ajoutées respectivement en haut et en bas du contenu de la page. La librairie ripcord a été importée à l'aide de la méthode "get_template_part()".

La page sera ensuite envoyée dans le répertoire "page-template" du site de la Berra

(cf. figure 35), pour pouvoir créer une nouvelle page depuis le panneau d'administration en ligne, en utilisant comme thème la page précédemment transférée.

Le nom donné à cette nouvelle page créée à partir de la template est "historique".

modifications des pages ADD et LOGIN
Afin de rediriger les clients sur leur page d'historique après une connexion sur la plateforme il faut ajouter une redirection sur ces deux pages. WordPress fournit là encore une fonction. Plutôt que d'utiliser la fonction PHP "header()", la fonction "wp_redirect()" sera utilisée de manière à travailler en harmonie avec la technologie utilisée. Ainsi dans le fichier login.php et add.php l'instruction "wp_redirect('../Historique/')" est visible. Cette manière de faire est fonctionnelle mais n'est pas la meilleure. En effet, si une personne change le nom de la page "Historique" depuis le panneau administratif, l'instruction sera obsolète et génèrera une erreur de redirection pouvant mener à une erreur 404, car l'URL défini ne sera plus accessible.

4.4.7 Jalon 10 et 11 - Tests

Rendu de la page
La figure ci-dessous représente le resultat obtenu en suivant la procédure détaillé dans ce chaptire.

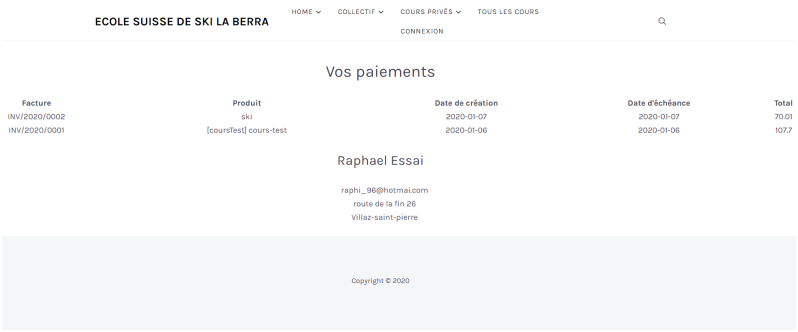


FIGURE 70 – Rendu historique

Tests

Num.	test	Attendu	Résultat
1	Affichage sans connexion	Non	Non*
2	Accès via processus "connexion"	Oui	Oui
3	Accès via processus "création de compte"	Oui	Oui
4	Le client ne voit que ses paiements	Oui	Oui
5	Le client ne voit que ses informations personnel	Oui	Oui

TABLE 15 – Tests page HISTORIC

*Lorsqu'on accède à la page en saisissant l'url dans le navigateur, celle-ci s'affiche. Elle a bien entendu aucune donnée client affiché, cependant une redirection sur la page de connexion , ou simplement une page avec un message d'erreur expliquant qu'il est nécessaire de se connecter au préalable pourrait faire partie des améliorations possibles de la page.

5 Résultats et améliorations possibles

Les résultat et améliorations importantes du projet sont des éléments à prendre en compte pour la suite du projet. Ces aspects n'ont pas été pris en compte dans l'analyse. L'analyse a en effet été réalisée pour le projet entier et certains problèmes étaient très difficiles à prévoir étant donné que les logiciels en question étaient utilisé dans le cadre d'un développement pour la première fois.

5.1 Résultat

la figure ci-dessous représente la situation actuelle du projet. Les différents points sont développés dans ce chapitre.

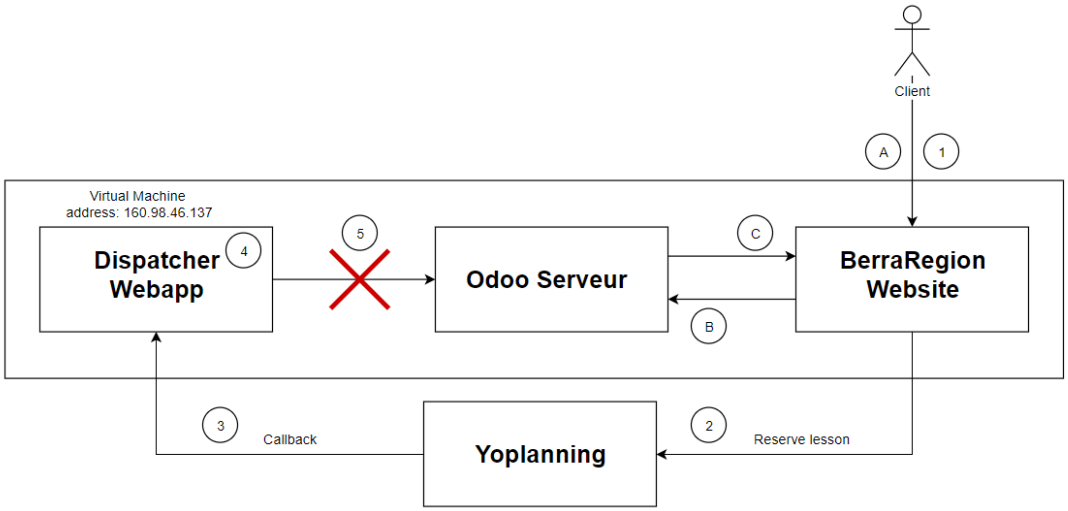


FIGURE 71 – Situation à la fin du projet

1. Le client peut, grâce au plugin fournit par Yoplanning, se rendre sur le site internet et réserver une leçon de ski . Ceci en passant par l'étape connexion ou non.
2. lorsque la commande est validée le plugin transmet ces informations à Yoplanning.
3. Lorsqu'une nouvelle réservation est faite un callback à destination du "Dispatcher Webapp" est envoyé.
4. le "Dispatcher Webapp" traite le callback, afin de déterminer quel traitement devra être effectué en fonction de divers conditions.
5. La "Dispatcher Webapp" envoie les données traitées au serveur Odoo. Cette étape du processus n'est pas disponible. Les raisons sont expliquées au point 4.3.9

— A : Le client se connecte sur la plateforme.
— B : Le client se rend sur la page d'historique. La plateforme envoie une requête pour récupérer les paiements destinés au client.
— C : les données sont transmises au client.

5.2 Création de module Odoo

Comme expliqué dans les problèmes non résolus du workpackage 3 4.3.9, la création d'un module sur Odoo utilisant des modèles de données propres aux besoins du projets est nécessaire afin de pouvoir atteindre l'objectif. La gestion des commandes des clients ne sera réalisable qu'avec la réalisation d'un module. Odoo bride extrêmement les capacités à faire intéragir ses modèles de bases avec des logiciels externes. Le positionnement d'Odoo sur le marché des logiciels open-sources gratuit n'est pas très lisible avec beaucoup de liens inter-modèles menant à des extensions payantes.

La suite du projet logique serait la réalisation de ce module et de ces modèles afin de pouvoir concrétiser la première partie mise en place lors de ce projet. Le travail réalisé n'est pas perdu et a permis de prendre connaissance de l'environnement de développement et les contraintes de développement qui sont apparues en prenant connaissance plus en profondeur des logiciels utilisés.

5.3 Plugin WordPress

Dans ce projet les différentes pages qui ont été créées ont été intégrées au site internet par le biais de template. Cette méthode est fonctionnelle, mais ne représente pas une solution idéale.

En effet, si le groupe BerraRegion souhaite modifier le thème qu'il utilise les pages créées ne seront plus disponibles et utilisables. Il faudra alors réeffectuer la procédure d'intégration présentée dans ce projet sur le nouveau site.

La meilleure solution serait de faire un plugin WordPress. Celui-ci est un peu plus long et légèrement plus complexe à réaliser. cependant, il permet à l'administrateur du site web de modifier le thème à volonté sans avoir peur de perdre les pages de connexion. Les plugin ne dépendent pas du thème utilisé. De plus, intégrer les pages avec un plugin permet même d'ajouter la possibilité de se connecter à Odoo sur un autre site WordPress simplement en ajoutant le dossier représentant le plugin sur le nouveau site concerné.

5.4 Mise en place d'un site internet

Il pourrait également être plus judicieux de réaliser une analyse plus approfondie de la mise en place d'un site internet. Wordpress étant très adéquat pour des sites internet simples. L'implémentation de plusieurs fonctionalités pouvant être souhaitée par le groupe BerraRegion, celle-ci pourrait rendre l'exploitation d'un site wordpress très compliquée. Cela peut également faire l'office d'un autre projet à réaliser.

5.5 Meilleure collaboration avec les partenaires

La collaboration avec les partenaires n'est pas un élément à négliger. En effet, l'implication des différentes parties concernées est très importante. Il est nécessaire et important que chaque partie ait conscience de ses responsabilités et devoirs lors de l'acceptation d'un projet. Cet aspect du travail a amené quelques soucis ainsi que

des pertes de temps. La gestion de cette situation était compliquée étant donné la position de Mr Rumo en tant que mandant / étudiant mais elle a permis de mettre à ce jour l'importance primordiale de cet aspect.

6 Conclusion

Dans un premier temps, nous pensons que réaliser un premier projet durant un semestre est une très bonne expérience qui nous a permis de vraiment nous rendre compte des difficultés que comporte un projet de cette envergure.

L'estimation du temps nécessaire pour effectuer les différentes tâches a été très difficile à cause de l'apprentissage nécessaire des technologies utilisése. Le fait de travailler avec plusieurs logiciels ayant chacun leur contrainte, nous a forcé à prendre énormément de temps pour comprendre ces logiciels et être capable de développer avec ces solutions. Ceci explique pourquoi nous n'avons pas rempli tous les objectifs du cahier des charge.

A travers ce projet, nous avons pu nous rendre compte des difficultés que peuvent générer des variables externes au projet. Comme expliqué dans le chapitre 5 précédent. La collaboration avec un ou des partenaires externes n'étant pas assez impliqués est difficile à gérer. L'implication n'étant pas forcément au rendez-vous et le délai de réponse étant long. Ce projet nous a fait prendre conscience que ce facteur n'est pas à négliger dans le cadre d'un projet car il peut vite rendre les délais internes d'un projet compliqué à respecter.

Les points réalisés dans le cadre de ce projet nous satisfont. l'ERP est en place et est capable de communiquer avec le site Wordpress actuel. Il est également possible pour un client de se créer un compte en utilisant la page de connexion/ d'enregistrement disponible sur le site internet. A travers ce compte, le client a accès à ses commandes.

La récupération des données depuis YoPlanning est fonctionnelle et le le dispatching est réalisé correctement en fonction de l'évènement et du type de resource. Il est cependant impossible de créer une commande et également un paiement dû au problème évoqué au chapitre 4.3.9.

En se référant à la priorisation des objectifs énoncé au point 2.9, nous pouvons cibler précisémment la raison pour laquelle l'objectif numéro 1 n'est pas atteint. Le deuxième objectif est réalisé. Les objectifs 3 et 4 n'étaient pas réalisables dans le cadre de ce projet.

Une ligne directrice est fournie pour les personnes qui seraient en charge de la suite de ce projet. Il y a également un chapitre dédié aux futurs modifications ou améliorations à apporter au projet.

Nous avons particulièrement apprécié travailler sur ce projet, car celui-ci nous a permis de prendre en main beaucoup de technologies. La gestion d'une API, le formatage des données au format JSON, le développement sur Wordpress, l'amélioration des compétences en PHP, la prise en main de l'ERP opensource Odoo, le système de composition latex, le logiciel de gestion de versions décentralisé GIT. Réaliser ce projet nous a permis de prendre en main tous ces éléments et nous sommes contents d'avoir pu acquérir ces capacités.

Déclarations d'honneur

Je, soussigné, Raphaël Pittet, déclare sur l'honneur que le travail rendu est le fruit d'un travail personnel. Je certifie ne pas avoir eu recours au plagiat ou à toutes autres formes de fraudes. Toutes les sources d'information utilisées et les citations d'auteur ont été clairement mentionnées.

Date _____ Signature _____

Je, soussigné, Célestin Rumo, déclare sur l'honneur que le travail rendu est le fruit d'un travail personnel. Je certifie ne pas avoir eu recours au plagiat ou à toutes autres formes de fraudes. Toutes les sources d'information utilisées et les citations d'auteur ont été clairement mentionnées.

Date _____ Signature _____

7 Versions

Machine virtuelle

OS	Ubuntu
Version	18.04.3
CPU	2
RAM	2[GB]
Stockage	30[GB]
Connexion port	22(SSH)
	80(HTTP)
	443(HTTPS)
	8069(Odoo)
	5432(PostgreSQL)
	30306(PhpMyAdmin)
utilisateur	compte AAI

TABLE 16 – Caractéristiques de la machine virtuelle

Logiciels et librairies

Type	Nom	version
Logiciel	Wamp64	3.1.9
Logiciel	PHP	7.2
Logiciel	Python	3.7.2
Logiciel	Atom	1.43.0
Logiciel	Apache	2.4.29
Progiciel	Odoo	13.0-20200103
Système d'exploitation	Windows 10 (Famille)	10.0.18362
Librairie	Flask	1.1.1
Librairie	Ipython	7.1.1
Librairie	JSON5	0.8.5
Librairie	JSONpath	0.82
Librairie	Jupyter	1.0.0
Librairie	Jupyter-client	5.3.4
Librairie	Notebook	6.0.2
Librairie	Pypi XMLRPC	2019.4.13

TABLE 17 – Version des logiciels

Table des figures

1 Situation actuelle du groupe BerraRégion 6
2 Plugin Yoplanning 7
3 Plugin Yoplanning 8
4 Plugin Yoplanning 10
5 Graphique représentant la répartition des transactions par moyen et lieu de paiement 12
6 Schéma des interconnections pour un paiement en ligne 14
7 Tableau des rentrées d'argent du mandant durant l'hiver 15
8 Coûts pour la Berra par Paiement Service Provider 15
9 Graphique 3D représentant les coûts(axe vertical bleu) des différentes solutions disponibles en fonction du nombre de transaction(axe horizontal vert) et du volume d'argent pour une durée d'un mois(axe horizontal rouge) 16
10 Intégration de PostFinance dans le plugin YoPlanning 18
11 Frais lié à Sumup 19
12 Logiciel de réservation de la situation idéal 21
13 Passerelles de paiements disponible sur simplybook.me[14] 23
14 ERP de la solution idéal 25
15 Partie comptabilité du diagramme de composants 29
16 Première page fournis par Odoo 40
17 Première page après la configuration de base Odoo 41
18 Visualisation des variables. 41
19 Première page lors de la configuration d'essai fournis par Odoo 42
20 Architecture d'Odoo 44
21 Informations fournies par info.php 45
22 Autorisation du port 3306 46
23 Autorisation du port 3306 46
24 Fichier de configuration wp-conf.php 48
25 Diagramme séquence : connexion 50
26 Diagramme séquence : création d'un compte 51
27 Code pour l'authentification 53
28 Critère de complexité du mot de passe 54
29 Vérification du mot de passe 54
30 Vérification de l'email 55
31 Création d'un client via l'api 56
32 Mockup formulaire de connexion 57
33 Mockup formulaire d'inscription 57
34 Code du formulaire de login 58
35 Structure du thème actif 59
36 Structure du dossier "page-template" 59
37 Template login 60
38 Template login 61
39 Utilisation de la page dans l'admin 61
40 Rendu de la template : à l'étape 1 62
41 Rendu de la template : à l'étape 2 63

42 Utilisation de la méthode _get_page_link . . . 64
43 commande scp . . . 64
44 Rendu page de connexion . . . 65
45 Rendu page de connexion . . . 66
46 Exemple de code python dans le l'envrionnement de développement Pycharm, utilisant un un notebook Jupyter . . . 68
47 Affichage du résultat du code exécuté directement dans Pycharm en utilisant un fichier Jupyter . . . 68
48 Affichage des variables . . . 68
49 Affichage des teams dans l'application web de YoPlanning . . . 69
50 Figures montrant l'utilisation de l'API . . . 70
51 Le point d'entrée réalisé avec le micro-framework Flask . . . 72
52 Le format JSON reçu lors d'une requête POST de YoPlanning . . . 73
53 Le diagramme de séquence de l'application web . . . 74
54 Méthode statique appelé . . . 75
55 Diagramme représentant les intéractions pour la création d'un client . 76
56 Information concernant l'url et la méthode à envoyer . . . 77
57 Information concernant le body inclus dans la requête . . . 77
58 Réponse de la requête . . . 77
59 Log dans le terminal de l'application web . . . 78
60 Fiche client nouvellement créé dans l'ERP . . . 78
61 Relation entre les modèles lors de la création d'une commande . . . 79
62 Champ étant required ET readonly . . . 80
63 Vérification de l'email . . . 81
64 Vérification de l'email . . . 82
65 Vérification de l'email . . . 82
66 Récupération des données client . . . 83
67 Vérification de l'email . . . 83
68 Requête d'historique . . . 84
69 Requête d'historique . . . 85
70 Rendu historique . . . 86
71 Situation à la fin du projet . . . 88

Liste des tableaux

1 Matrice de décision du moyen de paiement . . . 17
2 Caractéristiques de YoPlanning . . . 22
3 Caractéristiques de Odoo . . . 26
4 Caractéristiques de Dolibarr . . . 27
5 Matrice de décision de l'ERP . . . 28
6 Estimation du temps pour le workpackage 1 . . . 32
7 Estimation du temps pour le workpackage 2 . . . 33
8 Estimation du temps pour le workpackage 3 . . . 34
9 Estimation du temps pour le workpackage 4 . . . 35
10 Planification des dates de rendu des jalons . . . 36

11 Comparaison du temps effectué par rapport au temps estimé pour chaque jalons 37
12 Caractéristiques de la machine virtuelle 38
13 Tests page LOGIN 66
14 Tests page ADD 66
15 Tests page HISTORIC 87
16 Caractéristiques de la machine virtuelle 93
17 Version des logiciels 93

Références

[1] *Yoplanning*. Vakario. 2019. URL : https://yoplanning.com/ (visité le 02/10/2019).

[2] *projet marketing outcome*. HEIA-FR. 2019. URL : https://gitlab.forge.hefr.ch/celestin.rumo/projet-de-semestre-5/blob/doc/Annexes/projet_marketing/Projet_marketing_OutCome.pdf (visité le 02/10/2019).

[3] *Informations ERP*. wikipédia. 2019. URL : https://fr.wikipedia.org/wiki/Progiciel_de_gestion_int%C3%A9gr%C3%A9 (visité le 30/10/2019).

[4] *API YoPlanning*. YoPlanning. 2019. URL : https://yoplanning.pro/api/v3/docs/ (visité le 24/10/2019).

[5] *SAP*. SAP. 2019. URL : https://fr.wikipedia.org/wiki/SAP_(progiciel) (visité le 12/11/2019).

[6] *Diagramme de composant de la solution souhaitée*. BerraRegion. 2019. URL : https://gitlab.forge.hefr.ch/celestin.rumo/projet-de-semestre-5/blob/docC/Annexes/diagramme/composant_diagram_ideal_solution.pdf (visité le 25/11/2019).

[7] *Communiqué de presse BNS*. BNS. 2018. URL : https://www.snb.ch/fr/mmr/reference/pre_20180531/source/pre_20180531.fr.pdf (visité le 24/10/2019).

[8] *Rapport BNS*. BNS. 2018. URL : https://www.snb.ch/fr/mmr/reference/paytrans_survey_report_2017/source/paytrans_survey_report_2017.fr.pdf (visité le 24/10/2019).

[9] *Clearing*. bank. 2019. URL : https://fr.wikipedia.org/wiki/Compensation (visité le 25/11/2019).

[10] *Stripepaiement*. Stripe. 2019. URL : https://stripe.com/fr-ch/payments/payment-methods-guide#fiches-techniques-des-diff%C3%A9rents-moyens-de-paiement (visité le 25/11/2019).

[11] *PostFinanceacceptance*. PostFinance. 2019. URL : https://www.six-payment-services.com/fr/home/products/acceptance.html (visité le 25/11/2019).

[12] *myPos*. mypos. 2019. URL : https://www.mypos.eu/fr/our-story (visité le 13/09/2019).

[13] *API SimplyBook*. Simplybook.me. 2019. URL : https://simplybook.me/api/developer-api/tab/doc_api (visité le 24/10/2019).

[14] *Passerelles de paiements compatibles avec simplybook.me*. SimplyBook.me. 2019. URL : https://simplybook.me/fr/integrations-global/payments/CH (visité le 25/11/2019).

[15] *API Bookeo*. Bookeo. 2019. URL : https://www.bookeo.com/api/ (visité le 24/10/2019).

[16] *informations wikipédia Odoo*. wikipédia. 2019. URL : https://fr.wikipedia.org/wiki/Odoo (visité le 13/09/2019).

[17] *Informations LGPLV3*. wikipédia. 2019. URL : https://fr.wikipedia.org/wiki/Licence_publique_g%C3%A9n%C3%A9rale_limit%C3%A9e_GNU (visité le 29/04/2019).

[18] *API Odoo*. Odoo. 2019. URL : https://www.odoo.com/documentation/master/webservices/odoo.html (visité le 25/10/2019).

[19] *Informations wikipédia Dolibarr*. wikipédia. 2019. URL : https://fr.wikipedia.org/wiki/Dolibarr (visité le 22/10/2019).

[20] *Informations GPLV3*. wikipédia. 2019. URL : https://fr.wikipedia.org/wiki/Licence_publique_g%C3%A9n%C3%A9rale_GNU (visité le 29/09/2019).

[21] *Informations wikipédia Tryton*. wikipédia. 2019. URL : https://fr.wikipedia.org/wiki/Tryton (visité le 13/09/2019).

[22] *Cresus*. epsitec. 2019. URL : https://www.epsitec.ch (visité le 29/10/2019).

[23] *Winbiz*. Winbiz. 2019. URL : https://www.winbiz.ch/winbiz-cloud/ (visité le 29/10/2019).

[24] *Banana*. Banana. 2019. URL : https://www.banana.ch/fr/home (visité le 29/10/2019).

[25] *Comptabase*. Comptabase. 2019. URL : https://www.comptabase.ch/ (visité le 29/10/2019).

[26] *Bexio*. Bexio. 2019. URL : https://www.bexio.com/fr-CH/packages-et-prix (visité le 29/10/2019).

[27] *Documentation d'installation d'odoo*. Odoo. 2019. URL : https://www.odoo.com/documentation/13.0/setup/install.html (visité le 08/12/2019).

[28] *Documentation de deploiement d'odoo*. Odoo. 2019. URL : https://www.odoo.com/documentation/10.0/setup/deploy.html (visité le 08/12/2019).

[29] *Informations wikipedia PostgreSQL*. wikipedia. 2019. URL : https://fr.wikipedia.org/wiki/PostgreSQL (visité le 08/12/2019).

[30] *Odoo 12 Development Essentials - Fourth Edition*. O'Reilly. 2018. URL : https://learning.oreilly.com/library/view/odoo-12-development/9781789532470/cfcc20ef-e262-4827-b82f-a9a7e2ab38b1.xhtml.

[31] *Digitalocean LAMP*. digitalocean. 2019. URL : https://www.digitalocean.com/community/tutorials/how-to-install-linux-apache-mysql-php-lamp-stack-on-ubuntu-16-04 (visité le 10/12/2019).

[32] *Digitalocean PHPmyadmin*. digitalocean. 2019. URL : https://www.digitalocean.com/community/tutorials/how-to-install-and-secure-phpmyadmin-on-ubuntu-18-04 (visité le 10/12/2019).

[33] *Digitalocean Wordpress*. digitalocean. 2019. URL : https://www.digitalocean.com/community/tutorials/how-to-install-wordpress-with-lamp-on-ubuntu-16-04 (visité le 10/12/2019).

[34] *mysqlmaria*. geekflare. 2019. URL : https://geekflare.com/mysql-to-mariadb-migration/ (visité le 10/12/2019).

[35] *wp-cli*. wp-cli. 2019. URL : https://wp-cli.org/fr/ (visité le 10/12/2019).

[36] *Odoo 13 - External API*. Odoo. 2020. URL : https://www.odoo.com/documentation/13.0/webservices/odoo.html (visité le 03/01/2020).

[37] *Documentation XML-RPC*. Wikipedia. 2020. URL : https://fr.wikipedia.org/wiki/XML-RPC (visité le 04/01/2020).

[38] *Debugging in WordPress*. WordPress. 2020. URL : https://wordpress.org/support/article/debugging-in-wordpress/ (visité le 24/01/2020).

[39] *get-template-part référence*. WordPress. 2020. URL : https://developer.wordpress.org/reference/functions/get_template_part/ (visité le 25/01/2020).

[40] *forum WordPress*. WordPress. URL : https://wordpress.stackexchange.com/questions/77337/page-returns-404-with-post-variables-but-not-without.

[41] *Tutoriel install PIP*. Liquid web. 2018. URL : https://www.liquidweb.com/kb/install-pip-windows/ (visité le 07/01/2020).

[42] *Test Jupyter*. Gitlbab. 2020. URL : https://gitlab.forge.hefr.ch/celestin.rumo/projet-de-semestre-5/tree/master/Annexes/code/test_jupyter (visité le 28/01/2020).

[43] *API YoPlanning*. YoPlanning. 2020. URL : https://yoplanning.pro/api/v3.1/docs/ (visité le 26/01/2020).

[44] *Representational state transfer*. Wiki. 2020. URL : https://fr.wikipedia.org/wiki/Representational_state_transfer (visité le 26/01/2020).

[45] *Universal Unique Identifier*. Wiki. 2020. URL : https://fr.wikipedia.org/wiki/Universal_Unique_Identifier (visité le 26/01/2020).

[46] *API test method*. Loic Guibert. 2020. URL : https://gitlab.forge.hefr.ch/loic.guibert/ps5/blob/master/modules/yoplanning/textApi.py (visité le 26/01/2020).

[47] *Example JSON*. BerraRegion. 2020. URL : https://gitlab.forge.hefr.ch/celestin.rumo/projet-de-semestre-5/tree/master/Annexes/code/exemple_json (visité le 26/01/2020).

[48] *Flask*. Flask. 2020. URL : https://www.palletsprojects.com/p/flask/ (visité le 27/01/2020).

[49] *Postman*. Postman. 2020. URL : https://www.getpostman.com/ (visité le 27/01/2020).

[50] *Répertoire du code*. BerraRegion. 2020. URL : https://gitlab.forge.hefr.ch/celestin.rumo/projet-de-semestre-5/tree/master/Annexes/code/demoapp (visité le 30/01/2020).

[51] *rest api testing using python for beginners*. Udemy. 2020. URL : https://www.udemy.com/course/rest-api-rest-api-testing-using-python-for-beginners/ (visité le 28/01/2020).

[52] *factory method python*. realython. 2020. URL : https://realpython.com/factory-method-python/ (visité le 28/01/2020).

[53] *Getters et setters*. dauzon. 2020. URL : http://dauzon.com/lire-Python-Getters-et-setters-passez-votre-chemin-31 (visité le 28/01/2020).

[54] *Data Classes*. pyhon.org. 2020. URL : https://www.python.org/dev/peps/pep-0557/ (visité le 28/01/2020).

[55] *Diagramme de composant de la situation idéale*. HEIA-FR. 2019. URL : https://gitlab.forge.hefr.ch/celestin.rumo/projet-de-semestre-5/blob/doc/Annexes/diagramme/composant_diagram_ideal_solution.pdf (visité le 02/10/2019).

[56] *Odoo 12 Development Essentials - Fourth Edition*. O'Reilly. 2018. URL : https://learning.oreilly.com/library/view/odoo-12-development/9781789532470/.

[57] *Jupyter*. Jupyter Org. 2018. URL : https://jupyter.org/ (visité le 26/01/2020).

Glossaire

acquéreurs Dénomination d'une entreprise dans la chaîne de valorisation des cartes de crédit. Les acquéreurs délivrent les infrastructures nécessaires pour l'opération des transactions de cartes de crédit.. 13

API une interface de programmation d'application (désignée par le terme API pour application programming interface) est un ensemble normalisé de classes, de méthodes, de fonctions et de constantes qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels. Les principales méthodes sont les méthodes GET, POST, PUT et Delete qui permettent respectivement de recevoir, créer, mettre à jour et supprimer une ressource.. 67, 100

Demilitarized zone (Zone démilitarisée) Sous-réseau séparé du réseau local et d'internet qui contient les terminaux succeptibles, d'être accédés via internet.. 38

endpoint Un endpoint dans l'environnement API est le point de contact avec lequel une API peut intéragir avec un autre system. Un endpoint peut généralement être un url d'un server ou d'un service.. 72

facilitateur de paiement Un facilitateur de paiement assure l'encaissement des opérations auprès d'un acquéreur pour le compte de ses commerçants affiliés. Il a le rôle de fournisseur de service et est enregistré comme tel auprès de leur(s) acquéreur(s) pour faciliter les transactions de sous-marchands.. 13, 18

Flask Flask est un micro-framework Micro-framework qui vise à garder le code de base simple mais extensible. Flask ne prendra pas beaucoup de décisions pour la personne qui l'utilise, par exemple quelle base de données utiliser. Les décisions qu'il prend, telles que le moteur de templates à utiliser, sont faciles à modifier. Tout le reste est à faire par le développeur, de sorte que Flask puisse répondre à tous les besoins.. 72

Jupyter Jupyter est une application web qui permet de réaliser des calepins ou notebooks. Ces notebooks sont utilisés en science des données pour explorer et analyser des données. 67

Linux, Apache, Mysql, PHP C'est un ensemble de logiciels libres permettant de construire des serveurs de sites web. L'acronyme original se réfère aux logiciels suivants : « Linux », le système d'exploitation ; « Apache », le serveur Web ; « MySQL ou MariaDB », le serveur de base de données. 45

Micro-framework Un micro framework est un framework qui tente de fournir uniquement les composants absolument nécessaires à un développeur pour créer une application. Dans le cas des frameworks d'applications Web, un micro framework peut être spécifiquement conçu pour la construction d'API pour un autre service ou une autre application.. 72, 100

Payement Service Provider Service autorisé à effectuer des transactions avec des services de paiement en ligne ou des organisations de cartes de crédit. 12, 15

Point of sale lieu de vente (d'un produit) ; succursale (d'une chaîne commerciale).. 11

PyCharm PyCharm est un environnement de développement intégré utilisé pour programmer en Python qui permet l'analyse de code et contient un débogueur graphique.. 67

python Python est un langage puissant, à la fois facile à apprendre et riche en possibilités. Il dispose de nombreuses fonctionnalités intégrées au langage. Il est, en outre, très facile d'étendre les fonctionnalités existantes avec des bibliothèques qui aident le développeur à travailler sur des projets particuliers. Plusieurs bibliothèques peuvent ainsi être installées pour, par exemple, développer des interfaces graphiques en Python.. 67

YoPlaning YoPlanning est un logiciel métier de réservation du type Software as a service. 20

Acronymes

mPOS mobile Point Of Sale. 18, 19

PSP Payement Service Provider. 12