RAPPORT DE PROJET TUTEURE 2012

Réalisation du projet AssiServi

Auteurs :

Dorian CHEIGNON - Clémentine GAUGY –

Sébastien GAUTHERON – Guillaume LAFOUGE

Enseignant responsable :

Marie Lefevre

IUT Lyon 1

l'excellence technologique

Remerciements

Nous souhaitons remercier Mme Lefèvre, notre enseignante responsable, pour nous avoir suivis et guidés durant toute la durée de notre projet. De part son expérience, elle a su nous apporter son aide et nous conseiller dans nos prises de décisions, ainsi que pour nous fixer des objectifs en terme de délais.

Nous tenons également à remercier le département Informatique de l'IUT pour nous avoir permis de réaliser ce projet ,et avoir mis à notre disposition un espace de travail, ainsi que le matériel informatique.

Sommaire

Introduction

Dans le cadre de notre 2ème année à l'IUT de Lyon 1, au département informatique, nous avons dû réaliser un projet ayant pour but de mettre en application les connaissances, les notions et les méthodologies qui nous ont été enseignées. Ce travail nous a permis de mettre en pratique nos connaissances et d'améliorer notre esprit collaboratif au sein d'une équipe de 4 étudiants.

Pour la partie théorique, nous avons tiré profit des cours de modélisation et de gestion de projet. La partie technique repose sur les concepts abordés en Java, en Web et en SQL, ainsi que sur la découverte et l'implémentation de nouvelles technologies et de nouveaux outils.

Ce projet s'est déroulé en trois phases principales : premièrement, l'analyse des besoins des utilisateurs, deuxièmement, la conception des structures du site et des applications et enfin, le développement de ces derniers.

En fin ce projet a eu pour but de développer notre autonomie de travail, nos méthodes de recherche et d'organisation. L'amélioration de notre esprit de collaboration au sein d'une équipe a été très enrichissante.

I. Présentation du projet

1. Choix du sujet

Nous avons choisi de créer un projet qui réponde à une demande actuelle, qui est le gain de temps, que ce soit pour le client ou pour le fournisseur, dans le domaine de la restauration.

En effet, les restaurateurs peuvent apprécier le fait que les clients puissent commander avant d'arriver car ils en tirent plusieurs avantages, comme une meilleure organisation et d'éviter de se retrouver submerger par les commandes.

Du point de vue du client, les bénéfices concernent essentiellement la diminution du temps d'attente car les personnes qui travaillent, par exemple, ont peu de temps pour manger le midi.

Nous avons décidé d'appeler notre projet « AssiServi » pour la simple raison qu'en utilisant nos applications, le client n'a plus qu'à aller au restaurant et comme il a déjà commandé, son plat est prêt à son arrivée. Il n'a donc plus qu'à s'assoir et il sera servi dans de courts délais.

2. Objectifs

L'objectif de notre projet tuteuré a donc été de développer un site Internet portail dédié qui est appuyé par deux applications Smartphones (une application Android et une application Apple) qui viennent en appui.

Il s'agit de proposer aux restaurateurs et aux clients un site portail qui soit, à la fois, un moyen de valoriser les restaurants et leurs menus, et un outil qui facilite l'accès à des informations et leurs modifications. Les restaurateurs peuvent donc augmenter leur rendement grâce à un service plus rapide et plus efficace.

Le site portail permet le téléchargement des applications mobiles par le client, et la gestion du restaurant et des commandes par le restaurateur.

Les applications mobiles permettent la consultation des restaurants répertoriés, ainsi que la prise de commandes par les clients.

II. Analyse technique

1. Outil pour la gestion de versions

Pour gérer au mieux les différentes versions, nous avons choisi d'utiliser le serveur de gestion de versions XP-DEV qui nous alloue de la mémoire, et afin de déposer et de récupérer des documents sur le serveur, nous avons installé le logiciel client Tortoise.

2. Outil utilisé pour la modélisation

Afin de bien cerner nos objectifs et de pouvoir les modéliser, nous avons commencé par créer des modèles UML (Unified Modeling Language) en utilisant le logiciel ArgoUml. C'est le logiciel que nous avons utilisé lors de nos précédents projets à l'Iut, il ne nous a donc pas fallu de temps pour apprendre à l'utiliser et il était simple pour nous d'y avoir accès.

3. Outils utilisés pour la base de données

La base de donnée de type MySQL car ce système présente l'avantage d'être gratuit et portable. De plus, la base de données dont nous disposons à l'IUT est facilement administrable grâce à PhpMyAdmin.

4. Outils utilisés pour la conception du site Web

Nous avons tout d'abord utilisé le langage XHTML, un langage de balisage qui succède au langage HTML et qui répond aux nouvelles normes des pages Web en raison de la qualité d'affichage médiocre des nouveaux appareils numériques portables.

Nous avons également utilisé le langage CSS pour la présentation de nos pages Web car il répond aux standards définis par le World Wide Web Consortium (W3C).

Nous avons aussi utilisé le langage PHP (Hypertext Preprocessor) qui reste la base du site. Il nous permet la gestion des formulaires d'inscription, la connexion… Nous avons appris le langage PDO (PHP Data Objects) qui est une extension du PHP qui permet la gestion de la base de données (requêtes de modification, d'insertion…).

Pour créer des animations et des effets, nous avons utilisé les langages JavaScript et JQuery.

De plus, on utilise le langage AJAX pour que nos tables de la base de données soient mises à jour en temps réel.

Pour le développement, nous nous sommes principalement servis de l'IDE (Environnement de Développement Intégré) NetBeans, il s'agit de l'IDE que nous maîtrisons le mieux car nous nous sommes exercés durant notre formation. De plus, il nous permet de programmer en PHP, en HTML et en CSS. Il nous est aussi arrivés de programmer sur DreamWeaver car il propose deux modes : un mode de création qui permet de faire la mise en page directement à l'aide d'outils et un autre mode qui permet d'afficher et de modifier directement le code (HTML ou autre).

5. Outils utilisés pour la conception des applications mobiles

a. L'application Android

Les langages utilisés sont les langages Java et XML. Java est utilisé pour la création des classes et des méthodes et leurs implémentations. Quant à XML, il est utilisé pour répertorier toutes les propriétés des objets graphiques.

L'IDE utilisé est Eclipse car les tutoriels de programmation d'applications Android le conseillent et qu'il est l'un des seuls environnements de développement qui permet le téléchargement rapide de plug-in et du SDK (Software Development Kit). Ces derniers contiennent des bibliothèques de classes prédéfinies, ainsi qu'un émulateur Android, ils sont mis à disposition par Google.

b. L'application IPhone

Le langage utilisé est l'Objective-C, il s'agit d'un langage de programmation orienté objet particulièrement exploité par Apple. Il a été nécessaire de télécharger la bibliothèque JSON pour déchiffrer ce format, et d'autres frameworks pour pouvoir modifier les interfaces et les boutons.

Pour l'IDE utilisé, nous n'avons pas eu le choix, il a fallu programmer avec X-CODE car il est préinstallé sur les machines Mac et il n'en existe pas d'autre. Il contient un simulateur IOS qui permet de visualiser le résultat.

III. Analyse fonctionnelle

1. Les besoins des utilisateurs

a. Le restaurateur

Pour commencer, à l'arrivée sur le site, le restaurateur devra basculer en mode « Restaurant ».

S'il n'est pas encore inscrit, il devra cliquer sur le bouton « Créer un compte ». Il sera alors redirigé sur la page d'inscription. Lors de la première étape, il devra renseigner ses informations personnelles comprenant, entre autre, la saisie de son adresse mail et la création de son mot de passe. En deuxième étape, il renseignera ses informations professionnelles, c'est-à-dire le nom du restaurant, son adresse, le type du restaurant… La troisième étape sera la validation des saisies et aura un lien permettant de revenir à l'accueil. Pour finaliser l'inscription, le restaurateur recevra un mail de confirmation et il sera rediriger à l'accueil.

S'il est inscrit, il n'aura qu'à se connecter avec son adresse mail et son mot de passe.

Une fois connecté, il accédera à un tableau de bord qui lui permettra de recevoir en temps réel les commandes passées par les clients. Il pourra donc les accepter ou les refuser. Il pourra ajouter des plats à la base de données, ainsi que consulter les informations concernant son profil et son restaurant.

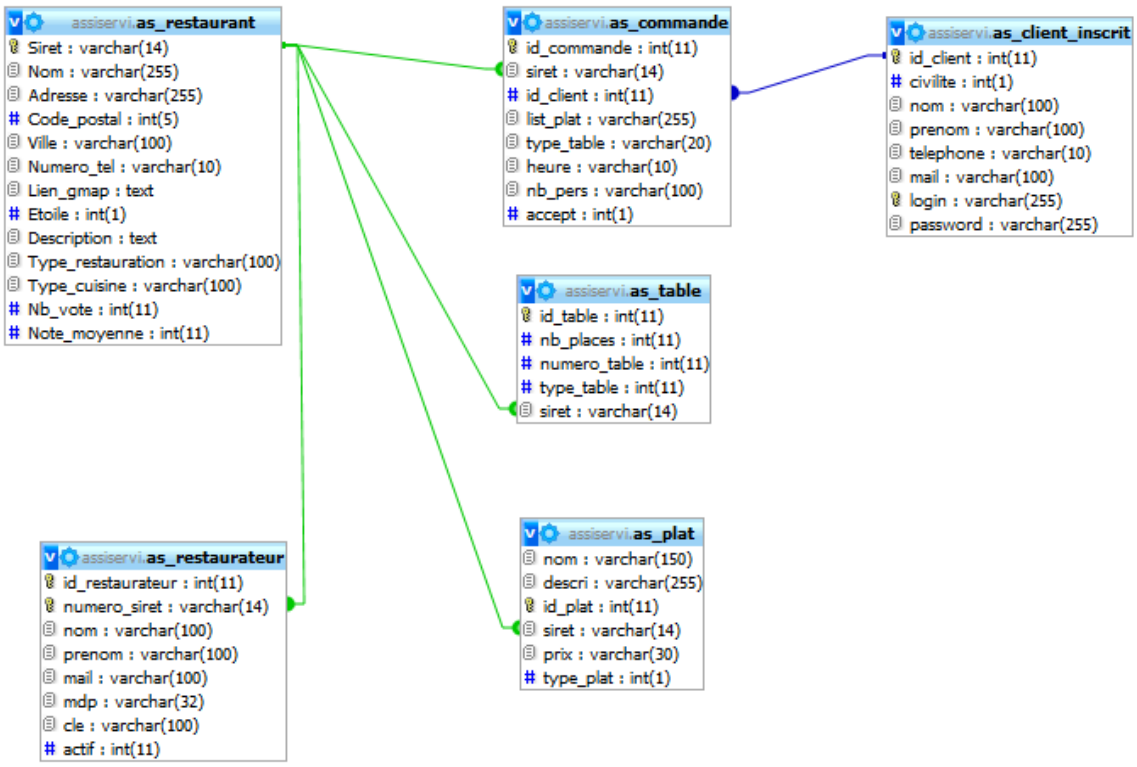
b. Le client

En arrivant sur le site, le client aura la possibilité de télécharger soit l'application Android, soit l'application Android, selon le système d'exploitation de son Smartphone. Il sera alors redirigé soit sur Google Market, soit sur Apple Store, afin de lancer le téléchargement.

Une fois l'application installée, il devra voir la liste des restaurants disponibles et lorsqu'il en sélectionnera un, il aura accès aux informations le concernant et à la liste des plats. Il pourra ensuite vérifier la commande avec un récapitulatif des plats qu'il aura sélectionné.

Pour valider la commande, il faudra alors qu'il crée un compte ou s'il l'a déjà créé, qu'il s'identifie par son adresse mail et son mot de passe.

2. Diagramme de classe généré par PhpMyAdmin



3. Planning prévisionnel et réel

1. Diagramme de Gantt

L'une de nos premières tâches a été d'élaborer un calendrier prévisionnel du déroulement du projet, et de nous répartir le travail à effectuer. Ne connaissant pas vraiment la durée de chacune des phases du projet, nous avons estimé des durées, qui n'ont pas été exactement respectées, soit parce que nous les avions mal estimées, soit parce que nous avons été confrontés à des problèmes qui nous ont ralenti.

Semaines		44	45	46	47	48	49	50	51	52	1	2	3	4	5	6	7	8	9	10	11	12	13
Début		31/10	7/11	14/11	21/11	28/11	5/12	12/12	19/12	26/12	2/1	9/1	16/1	23/1	30/1	6/2	13/2	20/2	27/2	5/3	12/3	19/3	26/3
Fin		6/11	13/11	20/11	27/11	4/12	11/12	18/12	25/12	1/1	8/1	15/1	22/1	29/1	5/2	12/2	19/2	26/2	4/3	11/3	18/3	25/3	1/4
Etude préalable																							
	Observation																						
	Conception/Organisation																						
	Appréciation																						
Etude détaillée																							
Réalisation																							
	Etude technique																						
	Programmation																						
	Jeux d'essais																						
	Tests et correction																						
									Vacances									Vacances				Evaluation	Stage

2. Répartition des tâches

Nous avons choisi de répartir les tâches comme ceci :

Guillaume Lafouge	Création de la page d'accueil du site et du formulaire d'inscription
Clémentine Gaugy	Création du tableau de bord et sa feuille de style CSS
Dorian Cheignon	Création de l'application IPhone
Sébastien Gautheron	Création de l'application Android

4. Le développement

1. La structure du site Web

En premier lieu, la partie du développement reposait sur la structure du site, notamment la mise en application directe de nos travaux dirigés de cours de Web.

Pour ce faire, il nous fallait mettre en place un site ergonomique tout en essayant de respecter aux mieux nos objectifs et notre plan de site.

Pour commencer, nous nous sommes intéressés à la page d'accueil. Elle contient une slide bar qui permet de passer du mode « Client » au mode « Restaurant ». En mode « Client », elle s'articule autour des images cliquables permettant le téléchargement des applications. Et bien sûr, Le passage en mode «Restaurant » permet ensuite d'accéder au formulaire d'inscription. Elle comprend également un bouton de connexion qui permet par la suite l'accès au tableau de bord.

La page d'accueil est plutôt simple car elle n'implémente que des fonctions HTML basiques, sauf pour la slide bar qu'il a fallu créer en JavaScript.

Pour, le formulaire, nous avons utilisé les langages HTML, PHP, JavaScript et JQuery. Le langage HTML nous permet de créer les champs de saisie grâce à la balise « Input ». Cette balise nous a permis de masquer les caractères lors de la saisie du mot de passe. Les listes déroulantes et les « Text Area » ont également été créés avec ce langage. Le langage PHP est utilisé pour récupérer les variables, qui ont été saisies dans les champs correspondants, afin de les stocker dans la base de données. Les langages JQuery et JavaScript ont été utilisés pour manipuler les feuilles de style, ils nous ont permis d'obtenir des effets visuels plus attrayants.

Pour la création du tableau de bord, les langages utilisés sont les suivants : HTML, MySql, PHP, CSS et JavaScript. Le langage PHP sert à récupérer les données dans la base pour afficher les commandes et le langage HTML sert à afficher la page concernée. Le CSS nous a permis de créer un design agréable. Les boutons et le menu ont été réalisés avec le logiciel Photoshop. Le langage JavaScript nous a permis de réaliser la rubrique pour modifier le profil du restaurateur. Le tableau de bord comprend donc un menu composé de quatre boutons.de plus, en affichant les commandes en cours, le restaurateur pourra les accepter ou les refuser. Un mail de confirmation ou d'annulation leur sera alors envoyé.

Tout notre site devant se polariser autour d'une base de données, la création des tables fut une étape primordiale.

La sécurité s'effectue à l'aide d'une variable de session, ainsi les visiteurs du site qui ne sont pas connectés ne peuvent pas accéder au tableau de bord.

Etant donné que nous utilisons la PDO, le site sera compatible avec la prochaine version de PHP (la version 6).

2. La structure des applications

a. L'application IPhone

L'application peut-être vue comme une structure générale qui englobe toutes ses fonctionnalités. Cela est possible grâce à la classe « AppsDelegat » qui peut être implémentée en Objective-C. Cette classe est en fait un gestionnaire d'application.

En lançant l'application, on voit une liste des restaurants répertoriés qui est crée en récupérant les données nécessaires dans la base de données par l'intermédiaire du format JSON. Ce format permet le transport des données. Pour obtenir cette vue, on affiche les variables souhaitées dans les cases d'un tableau par des requêtes à la table « UITableView ».

Et chaque ligne du tableau possède un gestionnaire d'évènement qui permet l'affichage des détails du restaurant sélectionné.

Depuis la vue de tous les restaurants, il est possible d'accéder à l'onglet carte qui affiche une carte Google Map avec la position de tous les restaurants. Si l'on clique sur un petit pin de localisation, on affiche le nom du restaurant à l'aide d'une fonction de type « MKPointAnnotation ». Il y a également un bouton dans la description d'un restaurant qui permet d'afficher la carte en vue satellite.

A partir de la vue des détails du restaurant, un autre gestionnaire permet d'afficher la carte avec le récapitulatif des plats dont les variables sont récupérées par le format JSON. La vue des plats comprend le nom de ces derniers ainsi que leurs descriptions et leurs prix.

Cette dernière vue comprend également un gestionnaire sur le bouton représenté par un panier. Lorsque l'on a fini de choisir les plats, on appuie sur ce bouton pour afficher le récapitulatif de la commande et un nouveau gestionnaire sous la forme d'un bouton « Confirmer » pour la valider.

L'utilisateur arrive sur une vue Horaires qui comprend elle aussi un gestionnaire d'évènement qui écoute et attend qu'il touche l'écran. On est alors redirigé vers une vue dans laquelle, on est invité à saisir son identifiant et son mot de passe. Ces variables sont récupérées par un script PHP qui les vérifie dans la base.

Au lancement de l'application, en dessous de la liste, on remarque un ensemble de quatre vues gérées par un contrôleur de vue (Tab View Controller): « Restaurants », « Carte », « Recherche » et « Plus ».

Par exemple, la vue « Plus » permet à un utilisateur de créer un compte par la saisie de ses informations personnelles. Ces informations sont envoyées à la base de données par l'intermédiaire d'une Url construite avec des variables.

b. L'application Android

Dans une application Android, les classes Java sont indépendantes des vues (Layout). On peut associer une vue à une classe Java grâce à la fonction :

setContentView(R.layout.[nom layout]);

Au lancement de l'application, l'utilisateur tape une fois sur l'écran pour la lancer, il arrive alors sur l'écran principal qui possède 3 onglets grâce à un objet « TabHost ».

Sur l'onglet « Restaurant », on affiche une « listview » personnalisée.Sur Android, les « listview » sont des éléments assez complexes. Pour les personnaliser, il faut créer une sorte de modèle que l'on applique à la « listview ».

L'onglet carte affiche une carte Google Map avec la position de tout les restaurants, si l'on clique sur un petit pin de localisation on affiche le nom du restaurant à l'aide d'une fonction de type « AlertDialog ». Il y a également une option en menu qui permet d'afficher la carte en vue satellite.

L'onglet inscription contient des champs pour que l'utilisateur entre ces données personnelles, toutes les erreurs sont gérées en cas d'oubli d'un champ ou d'inscription avec un pseudo déjà pris.

Si l'on clique sur l'onglet « Restaurant », on passe à la vue de description du restaurant, il y a objet de type « TabHost » a l'intérieur de cette vue, contrairement à la vue principale où le « TabHost » était codé en dur dans le code Java, ce « TabHost » ci doit être placé dans le « layout », puis il faut placer la totalité de la vue dans une « ScrollView »pour que les utilisateurs ayant une résolution différente puisse voir la totalité de l'écran.

Pour voir les plats, il faut cliquer sur le bouton "Afficher les plats", la vue qui s'affiche alors contient une « ListView » avec catégorie. Pour la réaliser, il faut créer une classe indépendante qui gère les catégories avec des méthodes spéciales. Ainsi qu'un autre modèle de « ListView » pour afficher les informations des plats comme le nom, le prix et la description qui défile de gauche à droite lors du focus. Un simple clic sur un plat ouvre une « AlertDialog » qui permet d'ajouter ce plat à la classe commande qui gère dans des « Arraylist » tout les plats qui sont ajoutés. A la fin de la sélection, un clic sur le petit panier qui est une « ImageButton » permet de passer à la vue suivante, le récapitulatif de la commande.

Le récapitulatif de la commande est encore une « Listview », une réutilisation de la « listview » des restaurants. La vue est placée dans une « ScrollView » mais une partie de l'écran en bas est statique pour que le client puisse faire défiler la liste des plats commandés et que le prix total et le bouton pour valider ne bouge pas.

Un clic sur le bouton Valider permet de passer à la vue suivante où le client peut sélectionner l'heure à laquelle il souhaite arriver, il s'agit d'un objet « TimePicker », il peut également indiquer le nombre de convives et sa préférence de table grâce à deux objets « Spinner ».

Après validation la vue suivante demande une identification, les erreurs de champs vides et d'identification sont gérées, une fois l'identification correctement effectuée, une fenêtre

« AlertDialog » s'ouvre pour demander au client s'il veut vraiment confirmer sa commande, la commande est envoyée dans la base et le client est ramené à l'écran principal.

Tout au long de l'application, toutes les requêtes s'exécutent dans un « thread » différent de la vue, permettant d'afficher une fenêtre de chargement dans une « AlertDialog » avec un objet « ProgressBar » pour faire attendre l'utilisateur.

3. Relation entre les modules

5. Les difficultés rencontrées et les solutions apportées

a. Le site Web

Pour commencer, lors de la mise en forme de notre site, notre choix initial d'utiliser des images d'extension « .gif » s'est avéré être une mauvaise idée car nous nous sommes confrontés à un problème d'affichage. C'est pourquoi, nous avons décidé d'utiliser des images d'extension « .png » et qui permettent la superposition grâce à la gestion de la transparence.

Concernant le CSS, nous avons également dû réfléchir à la meilleure solution pour la résolution d'écran. Celle-ci étant d'utiliser le pixel comme unité de mesure.

Dans la page d'Accueil, le bouton permettant de passer d'un mode d'utilisateur à l'autre ne fonctionnait pas. Il a fallu que nous recherchions des solutions dans des forums adaptés et la solution a été d'installer un plugin JavaScript de la bibliothèque JQuery.

Ensuite, nous avons dû trouver une solution pour gérer la connexion. Lors de l'authentification, si le restaurateur se trompait d'adresse, il pouvait quand même se connecter car son adresse mail était stockée dans une variable « $POST », qui ne gère pas les variables cryptées. Il a donc fallu que nous stockions la variable de l'adresse mail dans un « $SESSION » lors de l'envoi de la requête à la base de données. Elle peut être cryptée et donc être comparée à la variable correspondante stockée dans la base et qui a déjà été cryptée.

Lors de l'ajout d'un plat, la requête d'insertion dans la base de données ne voulait pas s'exécuter. Malgré de nombreux tests, nous ne trouvions pas l'erreur. Puis en cherchant dans la base de données, nous nous sommes rendus comptes que l'attribut « Desc» (pour la description des plats) de la table « Plats » était appelé « Desc » qui est un mot clé du langage SQL signifiant « Décroissant ». Il a donc suffi de changer le nom de l'attribut « Desc » en « Description » pour résoudre cette difficulté.

Dans le formulaire d'inscription, nous avons remarqué que pour que la saisie du code postal soit correcte, il fallait qu'il soit de type « Varchar5 » et non de type « Int ». En effet, le type « Int », ne prend pas en compte un zéro placé en première position.

Toujours pour le formulaire d'inscription, nous avons souhaité que les différentes étapes soient réunies sur une même page pour rendre l'inscription fluide. Nous n'avons donc pas pu passer par la fonction « Submit » qui nous obligeait à créer une page par étape et nous ne pouvions pas non plus utiliser « $POST » qui lui permet de rester sur la même page mais ne prend pas en compte les redirections. La solution a été d'utilisée la fonction « header location » qui gère mieux les redirections.

En cours de développement, nous avons cru que la validation de l'inscription par l'envoi d'un mail ne marchait plus. En réalité, cette fonction marchait, c'était juste que si plusieurs inscriptions se font simultanément ou dans un cours intervalle de temps, le délai d'envoi est plus long.

De plus, l'apprentissage du PDO, nous a fait commettre quelques erreurs de débutants. Par exemple, nous avons mal paramétré les types dans certaines fonctions comme « bindValue » qui associe une

valeur à un paramètre. Cependant, nous pouvons affirmer que cet apprentissage de l'extension PDO nous sera très utile dans notre vie professionnelle car dans la prochaine version de PHP (la version 6), la gestion des bases de données se fera entièrement via cette extension.

Il ne faut pas oublier que l'on ne pouvait pas héberger le site sur le serveur de l'Iut car ce dernier ne gère pas les variables « session ». Nous avons donc hébergé le site sur un hébergeur distant et gratuit : WebHost. Ce dernier propose une gestion des bases de données supportant l'extension PDO. Néanmoins, les applications mobiles ne pouvaient pas se connecter sur la base de données WebHost. En outre, les développeurs Smartphones arrivaient à se connecter via la base de données hébergée à l'Iut, mais malheureusement cette dernière ne gère la PDO. Il a donc fallu modifier toutes les requêtes de connexion et gestion de la base de données en requêtes MySqlQuery.

Un de nos principaux problèmes a été la gestion des commandes. En effet, nous voulions que l'ajout d'une commande par un client génère une fenêtre pop-up dans le tableau de bord du restaurateur pour le prévenir. Nous avons eu beaucoup de mal à faire afficher en temps réel la dernière commande insérée. La solution a été de créer un rafraîchissement automatique en utilisant une métadonnée qui demande au serveur de réactualiser la liste des commandes.

Il a également fallu trouver une solution pour que si le restaurateur consulte la liste des commandes depuis le tableau de bord et qu'il ouvre le détail d'une commande, même si le rafraîchissement s'effectue en même temps, la page reste telle quelle. Nous avons donc créé une balise « » qui aura pour identifiant le numéro de la ligne correspondante à la commande. Le contenu et la taille de cette balise étant contrôlés par le CSS, nous avons indiqué que lorsque l'identifiant de la « » est ciblé grâce au mot clé du CSS « target » que la taille change et que le contenu soit lisible. La balise « <a href…..> » nous a permis de cibler la balise « » en renvoyant le numéro de la ligne correspondante.

Enfin comme dans tous projets informatiques, nous avons rencontrés quelques bugs, notamment par rapport à la gestion des erreurs. Les scripts JavaScript ne la gérant pas, nous n'avons pas encore pu corriger tous les bugs.

b. Les applications mobiles

Tout d'abord, nous avons dû comprendre le format JSon que nous avons choisi pour faire transiter les informations entre la base et les applications. Nous avons choisi ce format car il est plus léger en terme taille de données (nous aurions pu choisir le format XML mais celui ci est beaucoup plus lourd). Le SDK d'Apple ne prenant pas en compte ce format, nous avons dû télécharger un Framework et l'ajouter à notre projet afin de déchiffrer les données. Dans le SDK Android, les fonctions JSON sont déjà incluses.

Nous avons eu également des problèmes avec la récupération des données depuis les scripts PHP, les accents ne passaient pas, cela affichait des caractères non reconnu.

Ensuite nous avons dû comprendre comment fonctionne le modèle MVC (Model View Controller) qui est le modèle utilisé pour les applications.

Nous avons dû comprendre le fonctionnement d'une API Google (Google Map) afin de transformer des adresses administratives (adresse - ville - code postal) en coordonnés GPS. Puisque les applications utilisent des coordonnés pour positionner des points sur une carte. Le SDK d'Apple propose un Framework qui interagis avec l'API de Google, mais celui ci ne fait que la reverse géo localisation (coordonnées -> position). Nous avons dû transmettre via le service en ligne de Google les adresses administratives afin de pouvoir récupérer leurs coordonnées et pour pouvoir par la suite utiliser le reverse géo localisation proposé pas le Framework d'Apple. Du coté de l'application Android, pour pouvoir utiliser l'API Google, il faut créer un projet de type « Google API ». Nous ne le savions pas à la création du projet, il a donc fallu en refaire un nouveau.

La gestion de la mémoire a été une de nos principales préoccupations. Une mauvaise allocation ou une mauvaise libération de la mémoire faisait planter l'application en nous signalant un mauvais accès à la mémoire. Il nous a donc été impossible de passer outre.

Le typage des objets a été un problème durant notre développement. Nous avions pris pour définir le numéro de Siret, le type « Int ». Nous avons remarqué que ce type n'accepte pas de nombres à plus de 10 chiffres. Et pour la clé primaire des restaurants (leur numéro Siret) le nombre de chiffres est de 14 chiffres. Nous avons donc dû également changer son type. En Objective-C, il existe une classe qui prend en compte tous les nombres : la classe « NSNumber ». Donc il nous a simplement fallu changer tous les types « Float, Int... » par « NSNumber ». La manipulation des données entre la base et les applications a demandé beaucoup de « cast ».

Sur Android, il y a une énorme contrainte de résolution d'écran, on ne peut pas se référer à son Smartphone pour développer, car les utilisateurs ont des Androphones différents qui n'ont pas la même taille d'écran. Il faut donc toujours penser par rapport à la plus petite résolution et construire les vues en conséquence.

6. Limites et évolutions possibles du projet

Premièrement, il nous reste certaines fonctions à implémenter dont celle permettant la géo localisation. Il s'agissait d'une partie importante pour nous, mais étant donné que nous nous sommes heurtés à de nombreux problèmes, nous avons dû revoir nos ambitions à la baisse.

Nous avons pensé, que par la suite, il serait intéressant d'implémenter une interface administrateur qui permette de consulter puis de valider les restaurants ajoutés depuis la dernière visite. En effet, nous ne gérons pas la vérification de la validité du numéro de Siret saisi par le restaurateur.

Lors de notre phase d'analyse, nous avions pensé créer un module de notation des restaurants ainsi que la possibilité pour les clients de laisser des commentaires. Le module existe mais nous ne l'avons pas implémenté pour la raison suivante : nous avons privilégié le développement des outils dynamiques du site au détriment de fonctions similaires à celle-ci.

Par manque de temps, certaines de nos fonctionnalités ne sont pas abouties. Par exemple, nous prévoyons de prévenir le client que sa commande a été validée par l'envoi d'un télémessage (sms). Nous avions prévu de permettre aux utilisateurs non inscrits de pouvoir commander mais cela nous a posé des problèmes à cause des requêtes à la base de données.

Nous n'avons également pas eu le temps de créer une rubrique de gestion des tables dans le tableau de bord. Nous aurions voulu permettre au restaurateur de visualiser les tables qui sont encore disponibles en fonction du nombre de place mais nous l'avons abandonné car elle nous a finalement paru inutile. Il nous paraît plus judicieux de prévoir une fonction qui refuse automatiquement les demandes de commandes si le restaurant n'a plus de places disponibles.

Pour finir, si notre site venait à être mis en ligne, certaines rubriques devraient être alimentées souvent. En particulier, il faudrait répertorier beaucoup plus de restaurants et que ceux-ci mettent souvent à jour les plats proposés, afin que le public soit toujours intéressé par notre projet que le site ne tombe pas dans l'oubli.

Conclusion

Après quelques mois de travail intense, nous sommes extrêmement fiers de vous proposer un projet fonctionnel et abouti.

Etant donné l'importance du projet en termes de quantité de travail, nous ne pensions pas pouvoir terminer notre projet dans les temps. En effet, nous avons fait le choix de créer un concept et il nous a donc fallu réfléchir à comment nous allions nous organiser. De plus, notre concept de projet contient la création d'un site Web, ainsi que la création de deux applications mobiles qui ont nécessité l'apprentissage de nouveaux langages de programmation.

Cependant, ce projet nous a vraiment séduit et satisfait car il nous a permis de nous prouver à tous les quatre que nous sommes capables de réaliser un projet de A à Z par nous-mêmes.

Notre esprit d'équipe et notre sens de l'organisation s'en trouvent améliorés, et malgré que nos esprits d'analyse et de recherche nous aient parfois menés à des obstacles importants, nous avons réussi à les dépasser pour terminer ce projet grâce à notre motivation.