



RAPPORT DE STAGE



Marc
GIORDANI

Projet FideCoin

Développement d'une plateforme de fidélisation

**CONCEPTEUR DEVELOPPEUR INFORMATIQUE
DU 06 MARS 2017 AU 05 MAI 2017**

— “ We can only see a short distance ahead, but we can see plenty

there that needs to be done. ”

Alan Turing « Computing machinery and intelligence » 1950

— “ Chi va pianu va sanu E chi va sanu va luntanu. ”

(Celui qui va lentement va sûrement, et celui qui va sûrement va loin.)

— “ Erscht wenn de baum g'fâlle esch, siejt m'r wir gross er esch ”

(Ce n'est que lorsque l'arbre est tombé qu'on se rend compte de sa grande taille)

— “ На то и ум, чтобы достичь того, чего хочешь. ”

(L'intelligence sert précisément à réaliser ce que vous voulez.)

F.Dostoïevski (1821-1881)

Table des matières

CHAPITRE 1 – PRÉSENTATION GÉNÉRALE.....	4
1. Avant-Propos	5
2. Abstract.....	5
3. Remerciements	6
4. Introduction	7
A. Mon profil et mon parcours	7
B. Tuteur de stage.....	7
5. La démarche générale.....	8
A. Le cadre de développement.....	8
B. Les interlocuteurs et acteurs du projet	8
C. Gestion du projet en Agile	9
D. Démarche d'analyse en UML	11
E. Les outils de développement.....	12
CHAPITRE 2 – CAHIER DES CHARGES.....	17
1. Présentation préliminaire du projet.....	18
2. Définition des besoins fonctionnels.....	18
3. Tableau des fonctionnalités.....	19
CHAPITRE 3 – LE PLANNING	20
4. Prémisses.....	21
5. Diagramme de GANTT	21
CHAPITRE 4 – ANALYSE.....	23
1. Le Diagramme de Cas d'Utilisation	24
2. Fiches de description textuelle de Cas d'Utilisation	25
A. Authentification	25
B. Créer compte / s'inscrire	26
C. Visualiser / Afficher les restaurateurs	27
D. Visualiser les bonnes affaires e-commerce.....	28
3. Les Maquettes.....	29
4. Diagramme de Navigation	34
5. Les Diagrammes de Séquences Systèmes	35
A. Authentification	35
B. Afficher restaurateurs.....	36
6. Les Diagrammes d'Activité Authentification et Inscription.....	37

CHAPITRE 5 – CONCEPTION DE LA BASE DE DONNEES.....	38
1. Introduction	39
2. Démarche utilisée et règles de gestion.....	40
3. La démarche ascendante.....	41
A. Création du dictionnaire de données	42
B. Graphes des dépendances fonctionnelles	44
C. Diagramme de classes.....	45
A. Le Modèle Physique de Données	48
4. Le code SQL (LLD) de création de la base de données	52
CHAPITRE 6 – CONCEPTION DE L’APPLICATION.....	56
CHAPITRE 7 – DEVELOPPEMENT	57
1. Le code de la page APP	58
2. Le code de la page d’Authentification	58
A. Authentification.html	58
B. Authentification.scss	59
C. Authentification.ts.....	60
3. Le code de la page d’Inscription	62
A. Registration.html	62
B. Registration.scss	62
C. Registration.ts	62
CHAPITRE 8 – DEPLOIEMENT	63
1. Le déploiement	64
2. Le Diagramme de déploiement.....	65
CHAPITRE 9 – CONCLUSION	66
CHAPITRE 10 – ANNEXES	67
1. Correspondances Projet/Reac.....	68
2. Code SQL complet.....	69
3. Bibliographie et Webographie	78
D. Bibliographie.....	78
E. Webographie	78
4. Glossaire / Lexique	79
5. Table des illustrations.....	80

CHAPITRE 1 – PRESENTATION GENERALE

1. AVANT-PROPOS

A travers ce rapport, je vais vous présenter de manière synthétique ma première expérience de travail en équipe pour la création d'une application et d'un site pour lancer une startup. Celle-ci s'articulera autour de ce nouvel outil mis à disposition des utilisateurs pour acheter leurs repas et faire en même temps de bonnes affaires. Lors de cette expérience, j'ai pu mettre en pratique, les compétences acquises lors de ma formation chez M2i.

Ce rapport retrace la réflexion que j'ai dû mener pour comprendre et analyser les besoins du projet qui m'a été confié par Kamel Khenissi le chef du projet, et présente la démarche, les méthodologies et les outils que j'ai utilisés pour y répondre. Pour finalement aboutir à la conception et au codage de l'application.

Le développement en équipe pour créer une application et un site dans le but de lancer une startup est un projet très existant. En effet la liberté que l'on a par rapport à un projet d'entreprise très cadré est grisant mais à la fois effrayant : nous devons nous-même trouver les cadres de conceptions et de développement ainsi que trouver et conserver une rigueur qui permet de mener à bien le projet.

2. ABSTRACT

Through this report, I will present to you in a synthetic way my first experiment of team work for the creation of an application and a site to launch a startup. This one will articulate around this new tool placed at the disposal them users to buy their meals and to make good deals at the same time. During this experiment, I could put into practice, the competences acquired during my formation at M2i.

This report recalls the reflection which I had to carry out to understand and analyze the needs for the project which was entrusted to me by Kamel Khenissi the chief of the project, and presents the approach, methodologies and the tools which I used to answer it. For finally leading to the design and the coding of the application.

The development equips some to create an application and a site with an aim of launching a startup is a very existing project. Indeed the freedom which one has compared to a very tallied project of company is graying but at the same time alarming: we must ourself find the executives of designs and development like finding and preserving a rigour which makes it possible to conclude the project.

3. REMERCIEMENTS

Je tiens à remercier toutes les personnes qui ont contribué au succès de mon stage et qui m'ont aidé lors de la rédaction de ce rapport.

Tout d'abord, j'adresse mes plus profonds remerciements à mes professeurs, Mr Pascal BUGUET et Sébastien MALORON qui, grâce à leur écoute et leurs conseils, m'ont énormément aidé durant ce stage. Il fut d'une aide précieuse dans les moments les plus délicats.

Je tiens à remercier mon maître de stage, Mr Alain OTHMAN, de URBATIL, pour son soutien et son suivi, le temps passé ensemble et le partage de son expertise. Grâce aussi à sa confiance, j'ai pu m'accomplir dans cette mission.

Un grand merci à toute l'équipe pédagogique de M2i et les intervenants professionnels responsables de la formation Concepteur Développeur Informatique, pour avoir assuré la partie théorique de celle-ci.

Enfin, je tiens à remercier ma femme, pour son soutien et sa patience. Elle m'a permis de réaliser ce projet et accepté que je m'absente régulièrement pendant 6 mois pour suivre cette formation.

Et je tiens aussi féliciter mes coéquipiers dans notre équipe : Alexandre COLONNA, Farodj DJERDI et Kamel KHENISSI, pour m'avoir supporté pendant 2 mois !

Before	Now	After
	Présentation, introduction	Cahier des Charges

4. INTRODUCTION

A. Mon profil et mon parcours

Après avoir travaillé dans l'imprimerie depuis mes 25ans, j'ai décidé de me réorienter vers l'informatique. J'ai bénéficié d'un financement en congé individuel de formation pour me consacrer à l'obtention du titre de Concepteur-Développeur Informatique au sein du centre de formation M2I FORMATION.

Alors que l'informatique est une passion depuis le collège, mon parcours ne m'a pas réellement donné l'opportunité d'effectuer une formation en informatique. Dans mon contexte familial, je devais être imprimeur comme mon grand-père, alors qu'à l'époque j'aimais tout particulièrement la programmation et l'automatisme. La passion que j'avais de travailler dans l'informatique n'a pas pu être concrétisée.

Actuellement, je suis en poste dans l'entreprise Newworks du groupe Electrogeloz en qualité d'Opérateur Numérique. Ce poste n'est pas en rapport avec mon niveau de Licence Professionnelle qui aurait dû m'amener à un poste à responsabilité. Après réflexion, j'ai décidé de me reconvertis dans les métiers de l'informatique.

Actuellement, et depuis plusieurs années, la demande de concepteur-développeur dans le web et les applications mobile est très forte.

Puis j'ai beaucoup échangé avec un ami, ingénieur de développement logiciel, qui m'a encouragée à franchir le pas et à me former au métier de concepteur dévelopeur informatique, persuadé que j'avais les aptitudes pour réussir.

Suite à ce stage je retournerai dans mon entreprise mais je continuerai à travailler sur ce projet qui est loin d'avoir été mené à son terme. Et en parallèle je continuerai à chercher des postes dans le Front-End surtout dans les langages Java pour le mobile en natif, en hybride avec Ionic 2 et AngularJS 2 et autres. Je continuerai également à me former pour parfaire mes connaissances : pourquoi pas avec des formations proposées par le CNAM pour obtenir un niveau Ingénieur.

B. Tuteur de stage

Monsieur Alain OTHMAN a joué le rôle de directeur de stage pendant la période du stage qui a été effectué dans les locaux de M2i Paris Picpus.

Ses compétences en gestion de projet Agile en informatique lui ont permis de superviser ce projet.

5. LA DEMARCHE GENERALE

A. Le cadre de développement

B. Les interlocuteurs et acteurs du projet

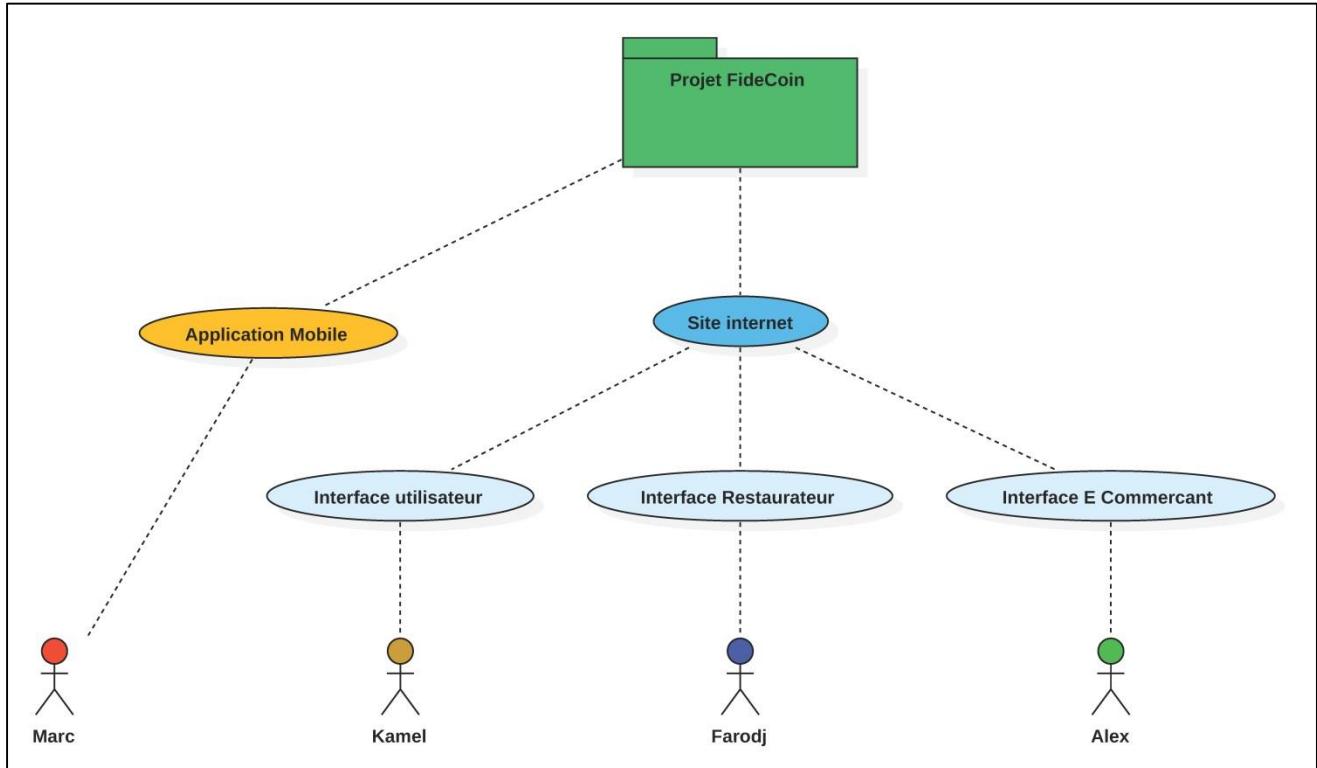


FIGURE 1 : SCHEMA DE REPARTITION DES DOMAINES

C. Gestion du projet en Agile

a. Une manière très différente d'envisager la création d'une application mobile

Les méthodes agiles sont apparues au début des années 1990 pour répondre aux besoins de réactivité d'organisations en croissance rapide et de plus en plus complexes. L'Agile Manifesto, rédigé en 2001, a posé les fondements théoriques de l'Agile, en définissant notamment 4 valeurs communes à toutes les méthodes dites agiles :

- **Les individus et leurs interactions** plus que les processus et les outils
- **Des logiciels opérationnels** plus qu'une documentation exhaustive
- **La collaboration avec les clients** plus que la négociation contractuelle
- **L'adaptation au changement** plus que le suivi d'un plan

Concrètement, dans le cadre d'une méthode agile, le développement a lieu dans une logique itérative, l'objectif est d'améliorer et d'adapter le produit, cycle après cycle, plutôt que de chercher à réaliser un produit parfait dès le départ. Les méthodes agiles, auparavant réservées au développement de logiciels, connaissent un succès croissant dans le monde de la création de site internet et donc dans notre cas pour la création d'application mobile qui se trouve être le mélange des deux mondes. La méthode Scrum notamment est de plus en plus utilisée, et ce même pour des projets de taille assez modeste.

Il faut néanmoins garder à l'esprit que chaque méthode agile dispose de sa propre logique et de conventions bien précises. Ce sont des méthodes de travail pas toujours naturelles, qui prennent beaucoup de temps avant d'être acceptées par tous, et donc efficaces. Naturellement, les agences qui travaillent en agile, notamment en Scrum, appliquent rarement la méthode sur le bout des doigts. Chacun l'adapte en fonction des contraintes internes et du type de projet / client. Je vais ici vous décrire rapidement les grandes lignes d'une méthode de travail agile simplifiée, d'inspiration Scrum, telle qu'elle pourrait être déployée dans une agence de taille moyenne.

b. Mise en place

L'esprit de travail inhérent aux méthodes agiles n'est pas du tout naturel, que ce soit en tant que client ou en tant que développeur. Le manque de visibilité peut parfois déranger. La mise en place d'un projet en agile est donc avant tout un travail pédagogique. Il faut prendre le temps d'expliquer et de faire adhérer à la méthode déployée l'ensemble des parties prenantes au projet de création de site internet.

Concrètement, il faut également attribuer des rôles. Dans notre exemple, nous allons considérer que le client est représenté par un homme, Kamel, qu'on appellera **Product Owner** (vocabulaire Scrum, de plus en plus utilisé). Du côté de l'agence, il n'y a pas de chef de projet, mais un **Scrum Master**, Marc (c'est-à-dire moi-même), dont le rôle est de s'assurer que l'organisation est bien comprise. (Théoriquement), il ne s'agit ni d'un chef de projet déguisé, ni un intermédiaire entre le client et les développeurs. Les développeurs sont auto-organisés et peuvent échanger directement avec Kamel, le Product Owner, dès qu'ils en ont besoin.

Au tout début du projet, Kamel liste, sans entrer dans le détail, l'ensemble des fonctionnalités que doit comporter le site web. Ce document est appelé *Product Backlog*. A partir de maintenant, la première itération, ou *sprint*, peut commencer.

c. Déroulé d'un sprint type

On suppose ici que Kamel et l'équipe de l'agence web ont décidé de travailler sur la base de sprint de 14 jours.

- **Jour 0** – Kamel et le reste de l'équipe fixent l'objectif général du sprint à partir du Product Backlog.
- **Jour 1** – L'équipe définit la liste des tâches à réaliser (*user stories*) au cours du sprint, le Sprint Backlog. Chaque tâche est évaluée en temps (ou en points de complexité) afin d'ajuster en fonction de la capacité de développement disponible sur le sprint.
- **Jour i, avec *i* variant de 2 à 13** – Daily scrum. L'équipe se réunit pendant 10 mn maximum, généralement au petit matin. Chaque membre de l'équipe a une minute pour dire ce sur quoi il a travaillé et les difficultés rencontrées. Les échanges doivent se prolonger en petit comité en dehors du Daily Scrum.
- **Jour 13** – Kamel et le reste de l'équipe (ou souvent juste le Scrum master) retravaillent un peu le Product Backlog et définissent l'objectif principal du prochain sprint.
- **Jour 14 – Démo** – L'équipe présente à Kamel les résultats du travail réalisé au cours du dernier sprint. La démo se fait généralement sur un environnement de pré-production (et non en local...).
- **Jour 14 –** L'équipe définit la liste des tâches du prochain sprint (Sprint Backlog).

d. Fin du projet

Pas de recette interminable contrairement à un développement dans le cadre d'une méthodologie plus traditionnelle. Le principe d'une démo à la fin de chaque sprint fait que les principaux bugs et problèmes fonctionnels sont identifiés, puis corrigés, tout au long du développement, et pas uniquement à la fin. Cela ne signifie pas pour autant qu'il ne faut consacrer au moins un sprint aux « finitions », que ce soit au niveau du graphisme, des textes voire des fonctionnalités. Toute la partie Optimisation SEO (méta-infos, réécriture d'urls, maillage interne, etc.), Analytics (installation script de tracking, paramétrage Google Analytics, etc.) et emailing transactionnel peut également faire l'objet d'un sprint dédié.

D. Démarche d'analyse en UML

Comme n'importe quel type de projet, un projet informatique nécessite une phase d'analyse, suivie d'une étape de conception.

Dans la phase d'analyse, on cherche d'abord à bien comprendre et à décrire de façon précise les besoins des utilisateurs ou des clients. Que souhaitent-ils faire avec l'application ? Quelles fonctionnalités veulent-ils ? Pour quel usage ? Comment l'action devrait-elle fonctionner ? C'est ce qu'on appelle « l'analyse des besoins ». Après validation de notre compréhension du besoin, nous imaginons la solution. C'est la partie analyse de la solution.

Dans la phase de conception, on apporte plus de détails à la solution et on cherche à clarifier des aspects techniques, tels que l'installation des différentes parties logicielles à implémenter.

Pour réaliser ces deux phases dans un projet informatique, nous utilisons des méthodes, des conventions et des notations. UML fait partie des notations les plus utilisées aujourd'hui.

UML signifie « Unified Modeling Language » ou Langage de modélisation unifié en français. C'est un langage de modélisation qui permet de représenter graphiquement les besoins des utilisateurs. On utilise pour cela des diagrammes.

C'est une démarche qui se base sur une approche objet. Cette approche s'appuie sur 4 principes fondamentaux :

- Décrire un problème, un état (Analyse),
- Décrire une solution (Conception),
- Proposer une solution (Réalisation),
- Modéliser la maintenance et l'évolution du système (Test et Maintenance).

L'approche objet nécessite une démarche itérative et incrémentale, c'est-à-dire que le concepteur doit faire des allers-retours entre les diagrammes initiaux et, les besoins du client et des utilisateurs perçus au fur et à mesure de la conception du logiciel afin de le modifier si nécessaire. Elle est guidée par les besoins du client et est centrée sur le diagramme de classes qui décrit aussi bien des actions que des informations dans une même entité. Les autres diagrammes nous aident à voir clair dans les besoins et dans la solution qui est à développer. Ils permettent de compléter le diagramme de classes.

E. Les outils de développement

Le développement mobile peut se faire de 3 façons différentes :

- En Natif : Les interactions avec le système sont avancées, les performances optimums. Mais on doit passer par les App Stores. Et on doit écrire un code par système (Android, iOS, Windows Mobile...)
- En HTML5 : Ne demande que les connaissances de développement Web. Les mises à jours sont instantanées et la distribution de l'application n'est pas restreinte.
- En Hybride : Prend les qualités du développement Natif et du développement Web.

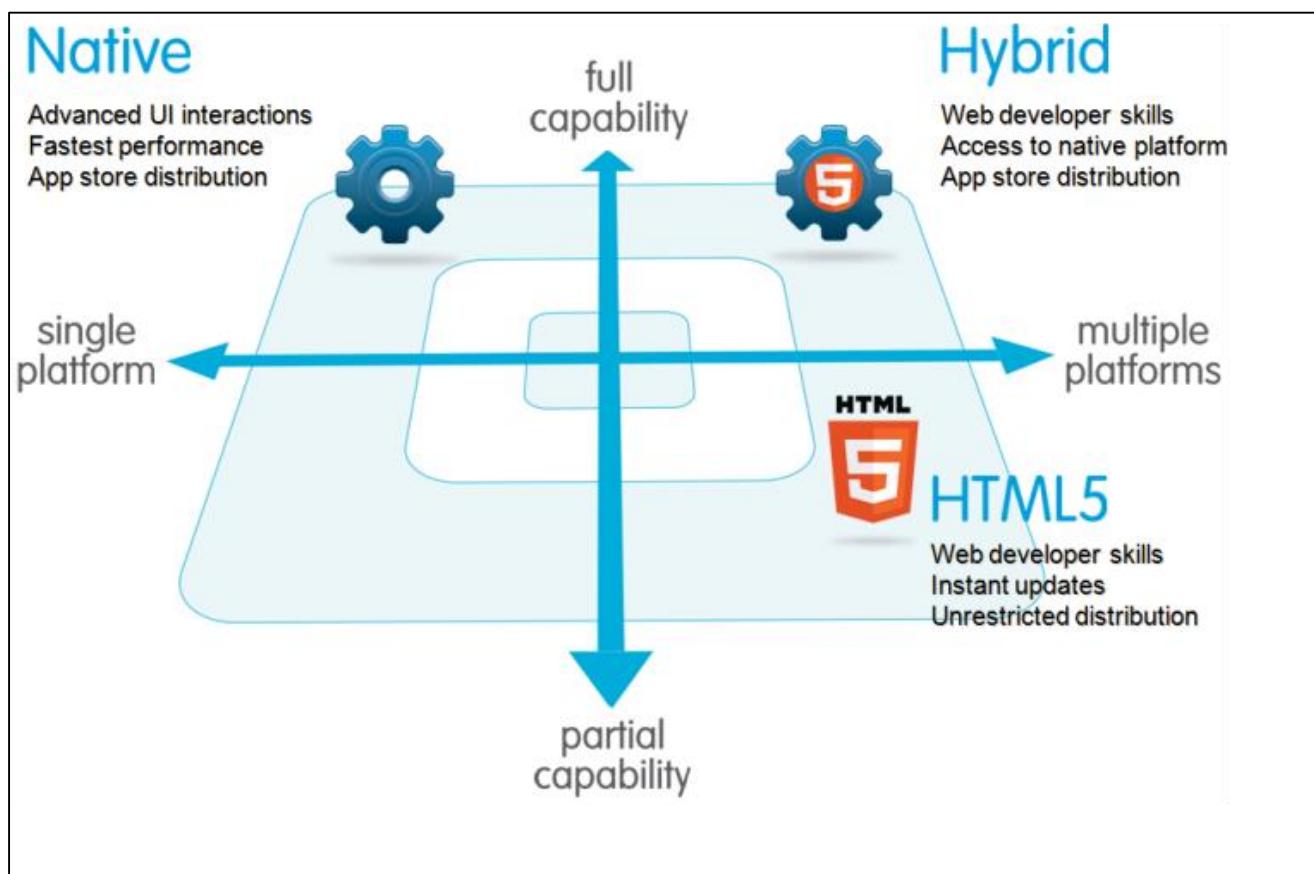


FIGURE 2 : DIFFÉRENCES ENTRE DEVELOPPEMENT NATIF, HYBRIDE ET WEB



C'est pour cela que j'ai choisi de développer mon application en Hybride. Et pour cela je vais utiliser ionic 2 qui est un système à la mode et qui a été déjà éprouvé.

Je vais vous présenter brièvement le développement web mobile avec ionic2 :

a. Le Web et mobile : deux mondes à part

Le web est basé depuis la fin des années 90 sur un mode orienté serveur :

- Un serveur HTTP qui centralise tous les traitements
- Des clients HTML/CSS qui affichent passivement l'information
- De multiples tentatives existent pour 'dynamiser' le côté client, Javascript en tête
- Un handicap : ça rappelle furieusement le modèle mainframe/ client texte des années 60/70,
- Un souhait : préserver un standard ouvert 2.

Le mobile s'appuie sur des outils de développements plus sophistiqués, mais dédiés à des plateformes incompatibles entre elles :

- Android : Langage Java - Environnement propriétaire Google, sur base Linux - Développement Android Studio - Distribution : Google Play
- iOS : Langage Swift - Environnement propriétaire Apple, sur base NetBSD - Développement XCode - Distribution : App Store
- FirefoxOS... Windows Phone... Blackberry... Ubuntu... Tizen...

b. Mais la mixité subsiste :

D'un côté, il est possible de faire tourner des choses assez évoluées dans le navigateur de son téléphone. De l'autre, il est aujourd'hui possible d'accéder à des fonctions 'natives' depuis son navigateur.

c. Historiquement : les "sites mobiles"

A l'époque des premiers smartphones, il était nécessaire de faire une version spécifique des sites Internet pour visualisation sur téléphone mobile

- ✓ Avantages : version légère et bien adaptée aux smartphones peu performants, facilité de développement
- ✓ Inconvénients ↗ nécessité de gérer deux versions en parallèle ↗ aucune exploitation des possibilités natives des téléphones

d. Vers une version unique des sites web : le responsive design

Les possibilités d'élasticité (grille fluide) et d'adaptation aux devices (media queries) des feuilles de style CSS permettent de maintenir une version unique des sites web tout en s'adaptant au format d'affichage

- ✓ Avantages ↗ adaptation relativement simple (que des hacks CSS) ↗ une seule version des sites web à maintenir
- ✓ Inconvénients ↗ aucune exploitation des possibilités natives des téléphones

e. Un pas vers les fonctions natives : HTML5

La version HTML5 propose, de manière plus ou moins aboutie, la possibilité d'accéder à certaines fonctions natives depuis son navigateur web

- ✓ Avantages ▷ possibilité de cumuler avec les capacités de responsive design ▷ fonctions accessibles en quelques balises
- ✓ Inconvénients ▷ toutes les fonctions ne sont pas présentes ▷ compatibilité aléatoire suivant les navigateurs

f. L'enjeu des fonctions natives

Stockage offline - Géolocalisation - Gestuelles tactiles - Bluetooth - Accéléromètre, boussole... - Notifications - Caméra...

g. La barrière des App Stores

- Google et Apple ont fait le choix du point d'entrée unique vers l'application : l'app store (ou playstore)
- Pour diverses raisons de contrôle qualité, de sécurité... et parfois de censure
- Cette barrière impose des critères techniques nombreux et peu simples à franchir.

h. 1ères solutions : Apache Cordova/Adobe PhoneGap

- Idée : proposer des passerelles entre les fonctions natives, et des pages conçues en HTML/CSS
- Naissance du concept d'Application Hybride
- Aujourd'hui intégralement en opensource et géré par l'Apache Foundation

i. Limitations de Cordova

- Ne propose pas de solution pour l'interface utilisateur (gestuelles tactiles, etc...)
- La publication vers les stores Google/Apple reste complexe
- A voir plus comme une série de librairies que comme un Framework à part

j. Surcouche et structuration: la solution d'un Framework

Idée : compléter la philosophie (et les briques techniques) de Cordova par un véritable framework capable de :

- Faciliter le développement
- Industrialiser la publication sur les stores
- Donner la possibilité de créer des modules optionnels
- 'Casser' la logique de page au sens HTTP du terme

k. Côté Facebook : React

- Framework développé en interne
- Destiné au développement des applications Facebook avant tout
- Essentiellement côté « front »
- Basé sur Javascript
- Fait appel à des composants natifs autant que possible

l. Autres Framework

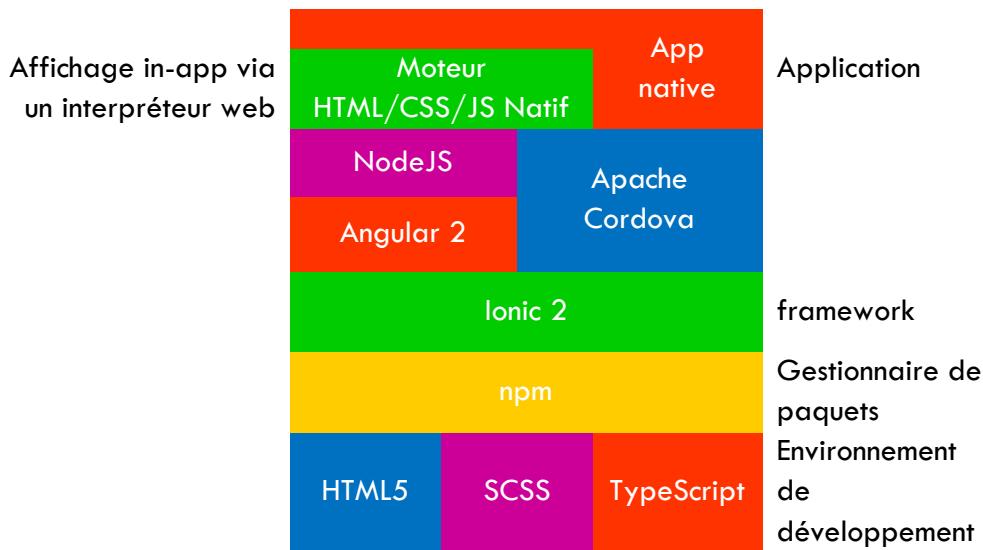
Onsen UI - Intel XDK - Sencha Touch - Kendo UI - Framework 7 - JQuery Mobile - Mobile Angular UI - Famo.us - Monaca - Trigger.IO - Electron (pour applis desktop).

m. Ionic

Créé en 2013. Repose entièrement sur Cordova pour la partie native et sur AngularJS pour la cinématique des pages.

La version 2 sortie est sortie fin 2016, TypeScript remplace Javascript. Angular2 remplace AngularJS

n. Architecture d'Ionic2



ANGULAR 2 – PRÉSENTATION



Tour d'horizon

Au travers d'Angular 2, Google a fait table rase du passé, en remettant à plat de nombreux concepts présents dans Angular 1.

FIGURE 3 : LOGO ANGULARJS 2

Cette stratégie a été motivée par 4 principes fondateurs :

- Augmenter les performances : l'un des plus grands reproches qui ait été fait à Angular sont ses lacunes en termes de performance. Pour pallier ce problème, les développeurs d'Angular ont décidé de se reposer davantage sur les briques natives du navigateur, comme par exemple, les prometteurs WebComponents ;
- Améliorer la productivité : en affichant une syntaxe expressive basée sur la syntaxe de ES2015/TypeScript (annotations, import, types, ...), plutôt que sur des surcouches ;
- S'adapter au mobile : la conception modulaire du framework permet de réduire considérablement son empreinte mémoire sur les terminaux mobiles ;
- Embrasser les nouveaux standards du Web... en se basant sur des technologies telles que le ShadowDom, Observables et autres nouveautés apportées par ES2015.

Rupture avec l'existant

Des « directives » aux « scopes », en passant par le « two-way data binding », Angular 2 n'en garde que les stigmates. Google nous offre ici un tout nouveau framework qui ne tient de son prédecesseur que le nom.

Voici une liste non exhaustive des concepts qui disparaissent :

- « Two-way data binding » : la création de cycle dans le graphe de détection des changements impliquait de nombreux problèmes de performance et de compréhension ;
- « Controllers » : désormais, les contrôleurs font partie intégrante du contexte this des composants ;
- « Scope / RootScope » : l'ensemble des « scopes » sont maintenant liés au contexte this des composants ;
- « Watch / Observe / Apply / Digest » : la détection des changements est effectuée par Zone.js ;
- « .module / .directive / .factory » : les dépendances sont injectées à partir d'annotations « Spring-like » et les imports sont effectués selon la syntaxe ES2015.



FIGURE 4 : LOGO
TYPESCRIPT

TypeScript

Les développeurs d'Angular avaient dans un premier temps parié sur AtScript, surcouche à ES6 écrite par leurs soins.

Avec l'intégration des décorateurs à Traceur, transpileur de TypeScript, c'est finalement vers ce dernier que l'équipe s'est tournée. TypeScript n'est pas imposé, mais son usage est encouragé car il apporte de nombreuses facilités de développement, comme par exemple une auto-complétion très poussée, pour peu que l'on utilise un IDE digne de ce nom. Par ailleurs, en imposant une syntaxe plus stricte et un typage statique, TypeScript rend le code plus lisible et maintenable.

Néanmoins, on peut utiliser ES2015. Il est même possible de coder ses propres composants Angular 2 avec du EcmaScript 5.

Navigateurs supportés / Compatibilité

Tous les navigateurs modernes sont supportés. Angular 2 peut même tourner sur IE9, avec l'utilisation des polyfills. En revanche, il faut tirer un trait définitif sur les versions antérieures d'Internet Explorer.

Performances

Angular 2 affiche des performances de rendu jusqu'à dix fois supérieures à son prédecesseur. Il serait également supérieur en performance à l'ensemble de ses concurrents directs (React, ...).



FIGURE 5 : LOGO JASMINE ET KARMA

Tests

Les développeurs de chez Google préconisent un framework éprouvé, Jasmine. Plus l'intégration

avec Karma. En ce qui concerne les tests bout-en-bout, la documentation prévoit des tests au niveau composant sans les interactions avec le DOM (ce qui n'était pas possible avec Angular 1), et des tests sur les composants directement dans le DOM.

CHAPITRE 2 – CAHIER DES CHARGES

Before	Now	After
Présentation	Cahier des Charges	Planning, Diagramme des Cas d'Utilisation

1. PRESENTATION PRELIMINAIRE DU PROJET

2. DEFINITION DES BESOINS FONCTIONNELS

Dans notre cas je vais vous décrire les éléments nécessaires et demandés par le Product Owner pour l'application mobile. Ils ont été déterminés lors des premières réunions. C'est le Product Backlog en méthode Agile.

L'utilisateur client de l'application mobile doit pouvoir :

- ✓ S'inscrire,
- ✓ Se connecter (s'Authentifier),
- ✓ Demander son mot de passe si est oublié,
- ✓ Afficher un lien vers la recherche de restaurateur pour commander et un lien pour accéder aux bonnes affaires,
- ✓ Visualiser son compte,
- ✓ Visualiser ses Clés et Coins,
- ✓ Se déconnecter.

Dans la partie commande gastronomique, l'utilisateur de l'application mobile doit pouvoir :

- ✓ Afficher / Rechercher les restaurateurs en fonction de sa position ou d'une adresse,
- ✓ Choisir le Restaurateur,
- ✓ Afficher les détails du Restaurateur,
- ✓ Afficher la carte du Restaurateur (Menus, produits... etc),
- ✓ Choisir ses produits ou menus,
- ✓ Créer un panier,
- ✓ Valider sa commande,
- ✓ Utiliser ses points de fidélités et choisir les options de livraison et de paiement,
- ✓ Faire une réclamation sur une commande,
- ✓ Noter et commenter une commande et donc un commerçant.

Dans la partie bonnes affaires, l'utilisateur de l'application mobile doit pouvoir :

- ✓ Visualiser les bonnes affaires,
- ✓ Visualiser le détail d'un produit,
- ✓ Visualiser le prix (dépense d'une clé),
- ✓ Acheter le produit,
- ✓ Payer,
- ✓ Noter et commenter une commande et donc un e-commerçant.

Détail important : Le Product Owner souhaite que l'utilisateur est l'obligation de se connecter pour pouvoir utiliser l'application.

3. TABLEAU DES FONCTIONNALITES

Fonctionnalité	Description	Type
S'inscrire	CRUD	Fonctionnel
Afficher Restaurateurs	R	Fonctionnel
Afficher détails Restaurateurs	R	Fonctionnel
Afficher Carte	R	Fonctionnel
Afficher Produits	R	Fonctionnel
Gérer Panier	CRUD	Fonctionnel
Evaluer commande	CRUD	Fonctionnel
Réclamation Restaurateur	CRUD	Fonctionnel
Visualiser Bonnes affaires	R	Fonctionnel
Visualisé détails du produit	R	Fonctionnel
Visualisé Prix	CRU	Fonctionnel
Acheter produit	CRU	Fonctionnel
Evaluer commande	CRUD	Fonctionnel

FIGURE 6 : TABLEAU DES FONCTIONNALITES

CHAPITRE 3 – LE PLANNING

Before	Now	After
Cahier des Charges	Planning	Diagramme des Cas d'Utilisation

4. PREMISSES

J'ai réalisé le diagramme de Gantt au début du projet après avoir déterminé les tâches à effectuer. Et il planifie et synthétise la succession des tâches du projet. Et il permet de surveiller l'avancement des différentes tâches. Celles-ci peuvent dépendre d'un ou plusieurs tâches précédentes.

5. DIAGRAMME DE GANTT

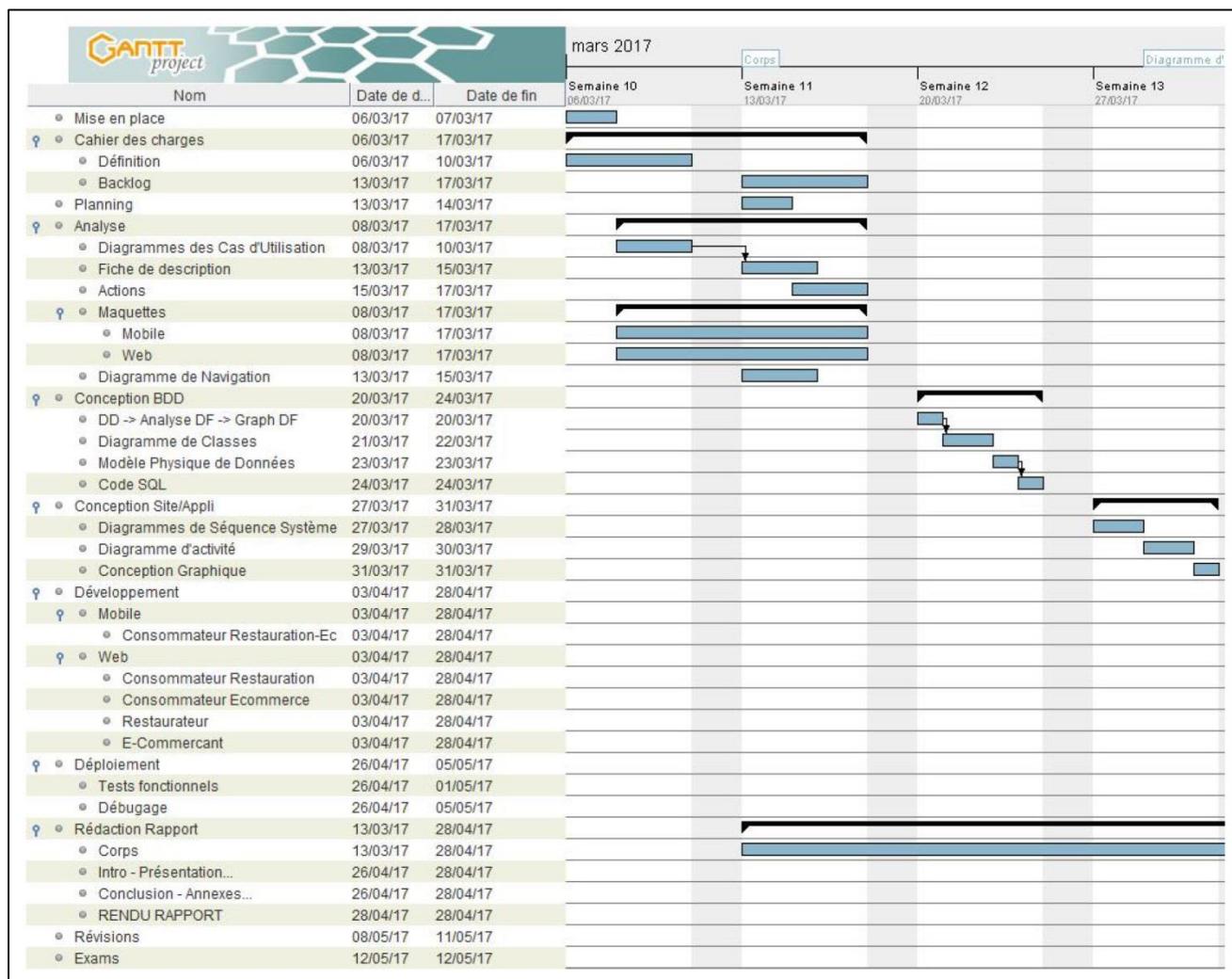


FIGURE 7 : DIAGRAMME DE GANTT 1/2

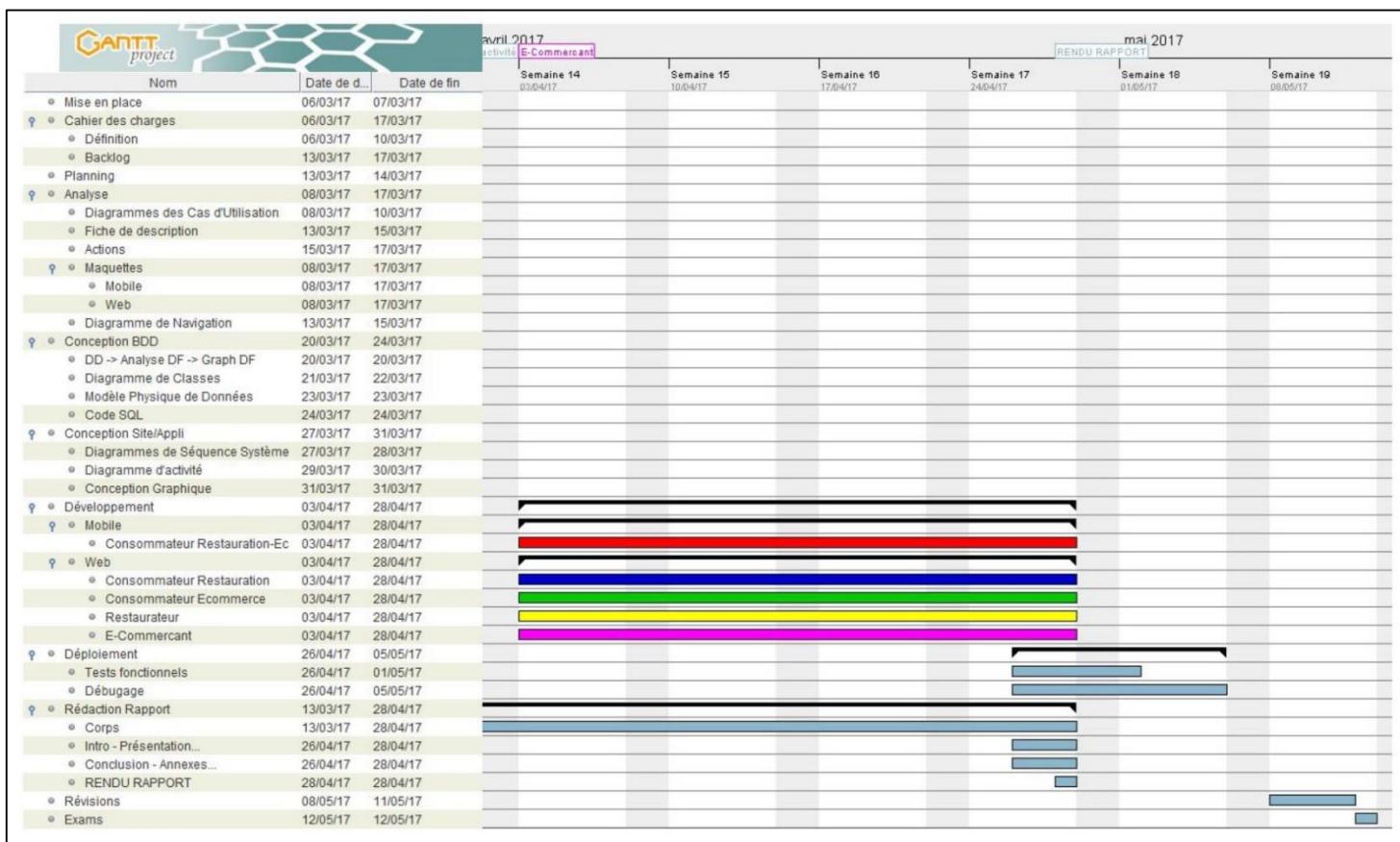


FIGURE 8 : DIAGRAMME DE GANTT 2/2

CHAPITRE 4 – ANALYSE

Before	Now	After
Planning	Diagramme des Cas d'Utilisation	Maquettes

1. LE DIAGRAMME DE CAS D'UTILISATION

Les diagrammes de cas d'utilisation permettent d'illustrer les tâches fondamentales attendues du système d'un point de vue de leurs utilisateurs. Un diagramme de cas d'utilisation est un ensemble de cas d'utilisation, il permet de définir un processus métier.

Un cas d'utilisation permet de modéliser un besoin utilisateur. Il permet de formaliser les interactions des acteurs avec le système.

Grâce au cahier des charges nous avons pu déterminer des acteurs principaux, secondaires et les cas d'utilisation. C'est grâce à un travail d'équipe lors de plusieurs réunions que nous avons pu mettre au point ces diagrammes. Sur ce projet nous avons un diagramme de cas d'utilisation pour le commerçant, pour l'e-commerçant et l'utilisateur. Je vais vous présenter le diagramme d'utilisation d'un utilisateur mobile.

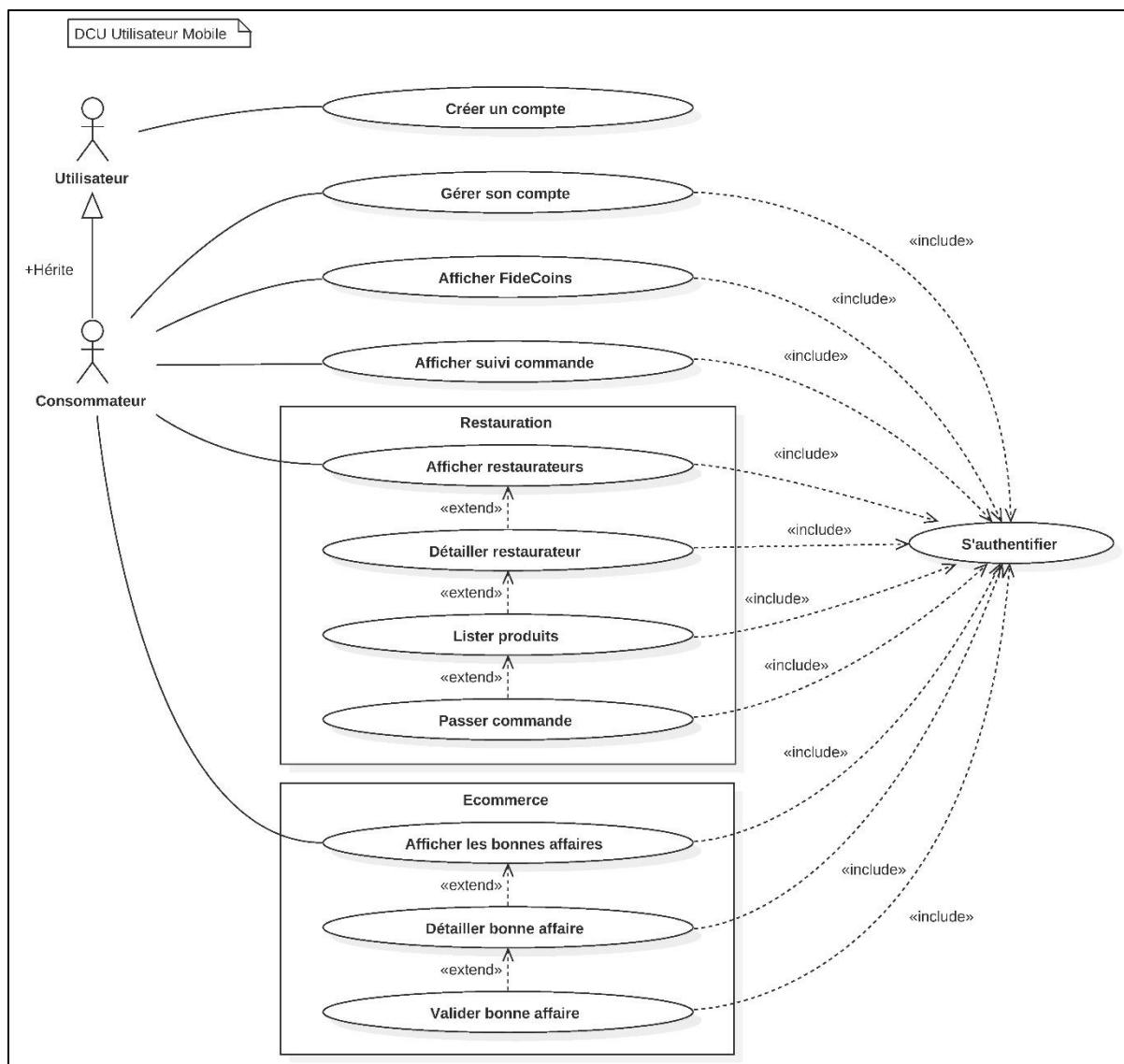


FIGURE 9 : DIAGRAMME DE CAS D'UTILISATION - UTILISATEUR MOBILE

2. FICHES DE DESCRIPTION TEXTUELLE DE CAS D'UTILISATION

A. Authentification

Etapes	Description
Identification du CU	Titre : Authentification Résumé : Gestion de l'authentification pour accéder aux fonctionnalités Acteur : Utilisateur Date de création : 07/04/2017 Date de dernière modification : 07/04/2017 Version : 1.0 Auteur : Marc GIORDANI
Pré-conditions	Écran d'authentification disponible Connexion à la BD disponible
Scenario nominal	Saisies de l'e-mail et du mot de passe Contrôle de surface Requête vers le serveur Contrôle des saisies côté serveur Contrôle d'existence dans la BD
Scenarios alternatifs	Les zones de saisies ne sont pas remplies ou remplies avec des formats incorrects (e-mail non conforme, mot de passe non conforme). L'authentification échoue pour non-correspondance de valeurs dans la BD une première puis une deuxième fois.
Scenarios d'erreurs des cas alternatifs	L'authentification échoue pour non-correspondance de valeurs dans la BD une troisième fois. La procédure d'authentification est verrouillée.
Post-conditions	Authentification réussie et affichage de la page d'Accueil
Exigences non fonctionnelles	Temps de réponse maxi : 2 secondes
Besoins d'IHM	Page d'authentification

FIGURE 10 : FICHES DE DESCRIPTION TEXTUELLE DES CAS D'UTILISATION

B. Créer compte / s'inscrire

Etapes	Description
Identification du CU	Titre : Créer compte Résumé : Crédation d'un compte pour devenir un consommateur sur la plateforme Acteur : Utilisateur Date de création : 07/04/2017 Date de dernière modification : 07/04/2017 Version : 1.0 Auteur : Marc GIORDANI
Pré-conditions	Ecran de création de compte disponible Connexion à la BD disponible
Scenario nominal	Saisie des informations Contrôle de surface Requête vers le serveur Contrôle des saisies côté serveur
Scenarios alternatifs	Les zones de saisies ne sont pas remplies ou remplies avec des formats incorrects (e-mail non conforme, mot de passe non conforme).
Scenarios d'erreurs des cas alternatifs	
Post-conditions	Création de compte réussie L'utilisateur reçoit un email de confirmation et doit le valider L'utilisateur peut s'authentifier
Besoins d'IHM	Page de création de compte

C. Visualiser / Afficher les restaurateurs

Etapes	Description
Identification du CU	<p>Titre : Visualiser les restaurateurs</p> <p>Résumé : Afficher la liste des restaurateurs partenaires de FideCoin</p> <p>Acteur : Consommateur</p> <p>Date de création : 07/04/2017</p> <p>Date de dernière modification : 07/04/2017</p> <p>Version : 1.0</p> <p>Auteur : Marc GIORDANI</p>
Pré-conditions	<p>Ecran avec liste des restaurateurs disponible proche de la position du restaurateur ou proche de l'adresse entrée par l'utilisateur</p> <p>Connexion à la BD disponible</p> <p>Utilisateur authentifié</p>
Scenario nominal	<p>Demande la page de la liste des restaurateurs en fonction de la position du consommateur ou de l'adresse entrée par le consommateur</p> <p>>> requête vers le serveur</p> <p>Le consommateur peut voir la liste des restaurateurs</p>
Scenarios alternatifs	L'utilisateur n'est pas authentifié
Scenarios d'erreurs des cas alternatifs	
Post-conditions	Le consommateur peut sélectionner un restaurateur pour voir son menu
Besoins d'IHM	Page de la liste des restaurateurs

D. Visualiser les bonnes affaires e-commerce

Etapes	Description
Identification du CU	<p>Titre : Visualiser les bonnes affaires e-commerce</p> <p>Résumé : Le consommateur peut voir les bonnes affaires en vente</p> <p>Acteur : Consommateur</p> <p>Date de création : 07/04/2017</p> <p>Date de dernière modification : 07/04/2017</p> <p>Version : 1.0</p> <p>Auteur : Marc GIORDANI</p>
Pré-conditions	Ecran des bonnes affaires en vente de l'e-commerçant disponible Connexion à la BD disponible Utilisateur authentifié
Scenario nominal	Le consommateur voit les bonnes affaires
Scenarios alternatifs	L'utilisateur n'est pas authentifié
Scenarios d'erreurs des cas alternatifs	
Post-conditions	Le consommateur peut voir le détail d'une bonne affaire
Besoins d'IHM	Page des bonnes affaires e-commerce

Before	Now	After
Diagramme des Cas d'Utilisation	Maquettes	Diagramme de Navigation

3. LES MAQUETTES

Des maquettes pour la partie mobile existaient déjà mais n'étaient plus adaptés. En effet en retravaillant sur le projet en équipes nous avons revu certaines choses. Et comme j'ai décidé de développer l'application mobile en IONIC2, j'ai décidé de refaire les maquettes. Pour cela j'ai utilisé IONIC CREATOR. C'est un site internet qui permet de créer une maquette de projet hybride facilement. Et le résultat est plus proche du résultat final.

De plus avec cet outil j'ai la possibilité d'exporter le projet pour pouvoir commencer à coder l'application. Par contre comme c'est un outil en IONIC1, je devrais retravailler les éléments et recoder en utilisant les spécificités d'IONIC2 : angular2, typescript, des noms de balises différentes...

Je me suis concentré sur la partie commerçants restaurateurs qui est la plus importante. Au lancement de l'application, la partie E-commerce sera uniquement web.

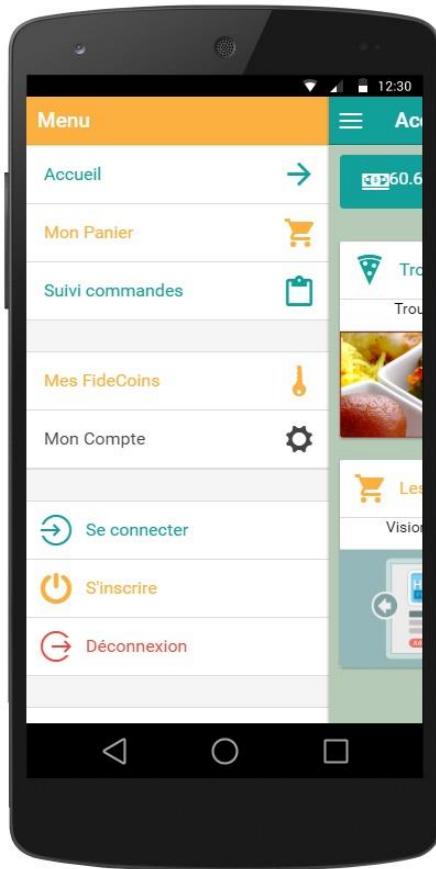
A. S'Authentifier

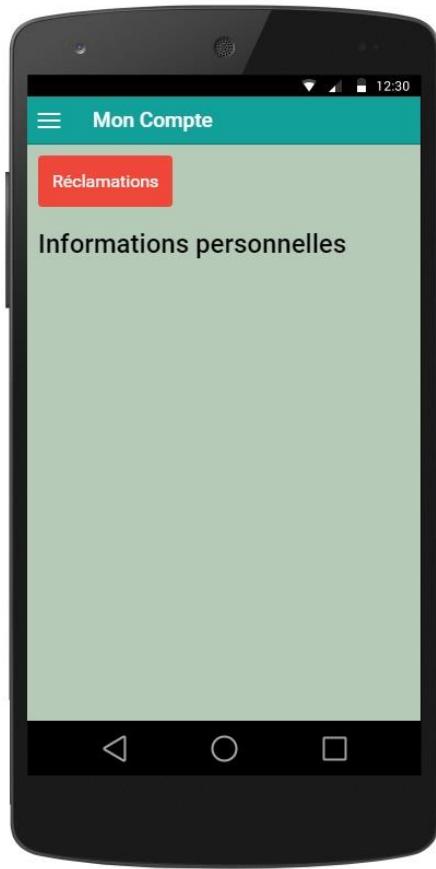
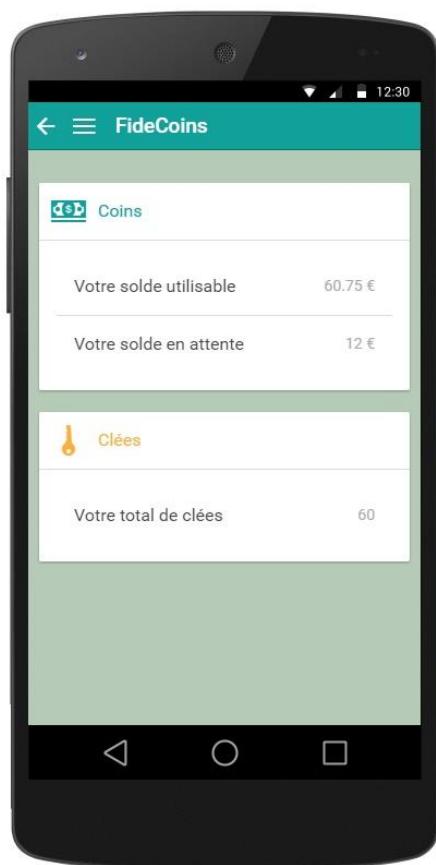
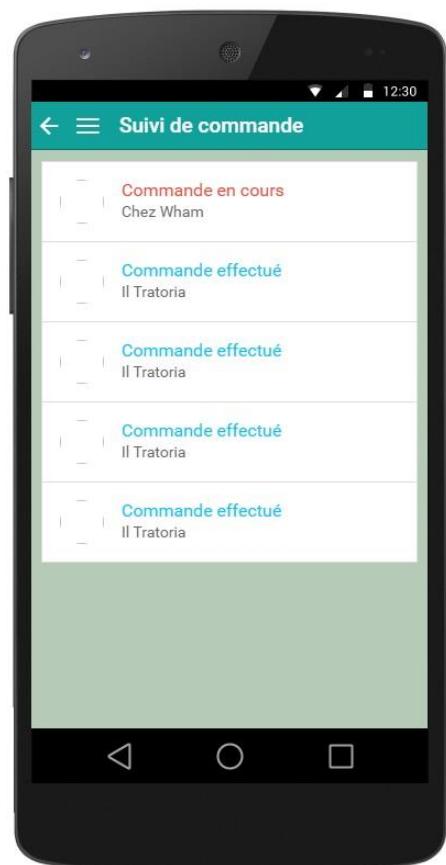


B. S'enregistrer

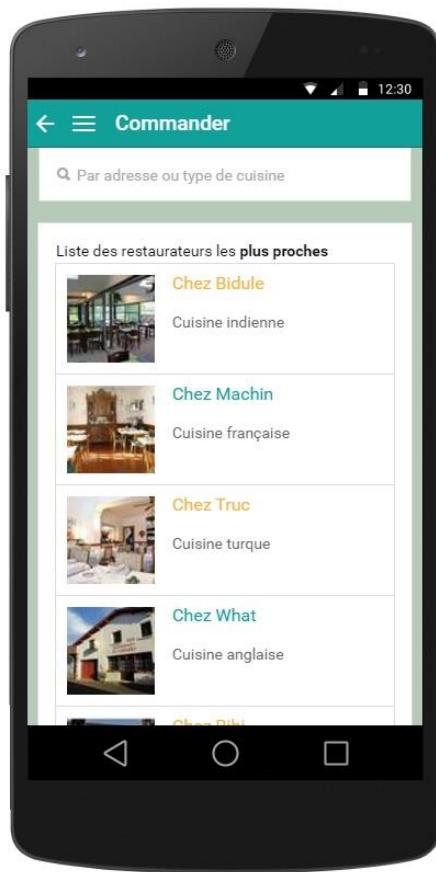


FIGURE 11 : MAQUETTES

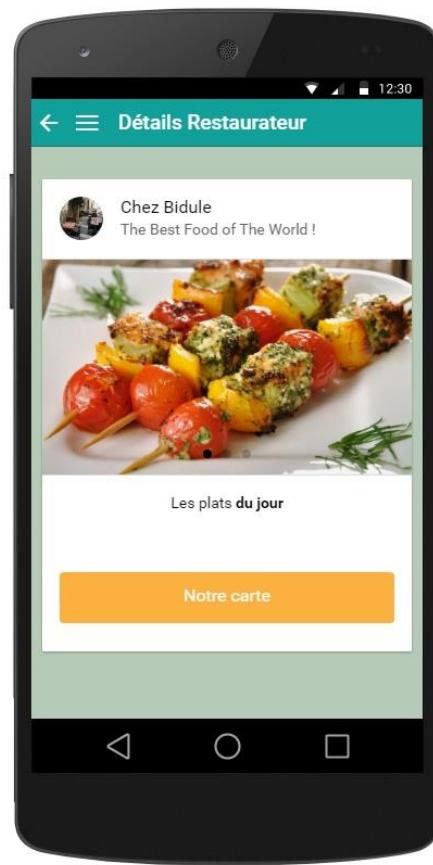
C. Menu de navigation**D. Accueil****E. Mot de passe oublié 1****F. Mot de passe oublié 2**

G. Mon Compte**H. Réclamations****I. FideCoins****J. Suivi Commande**

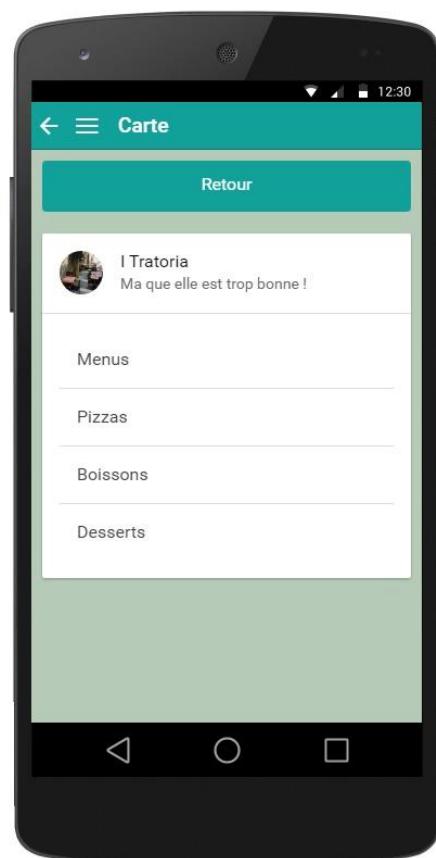
K. Liste Restaurateurs



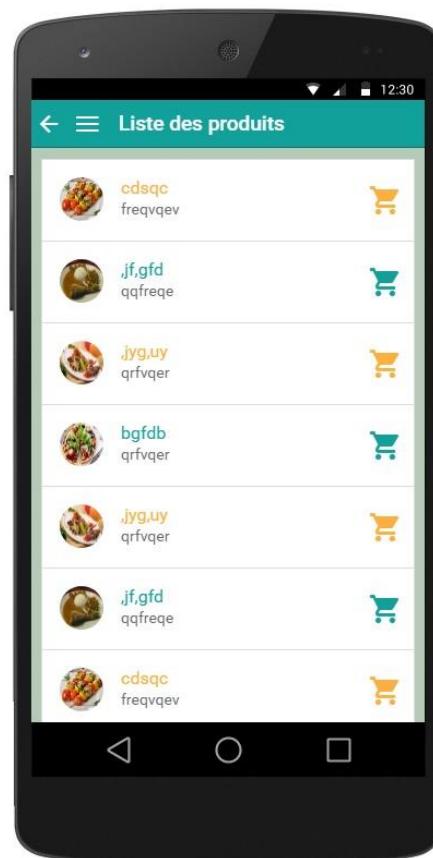
L. Détails Restaurants

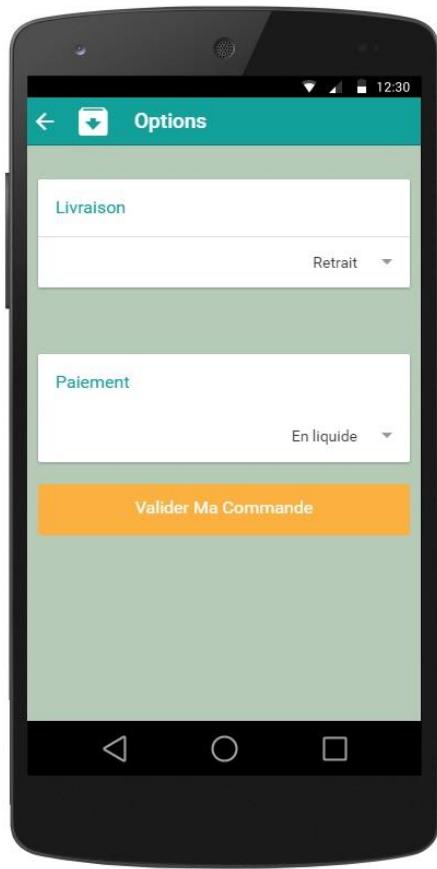
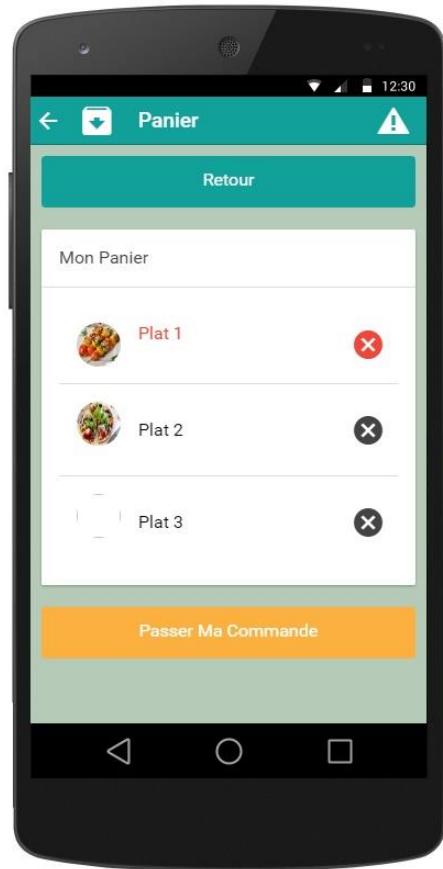


M. Carte Restaurant



N. Liste Produits



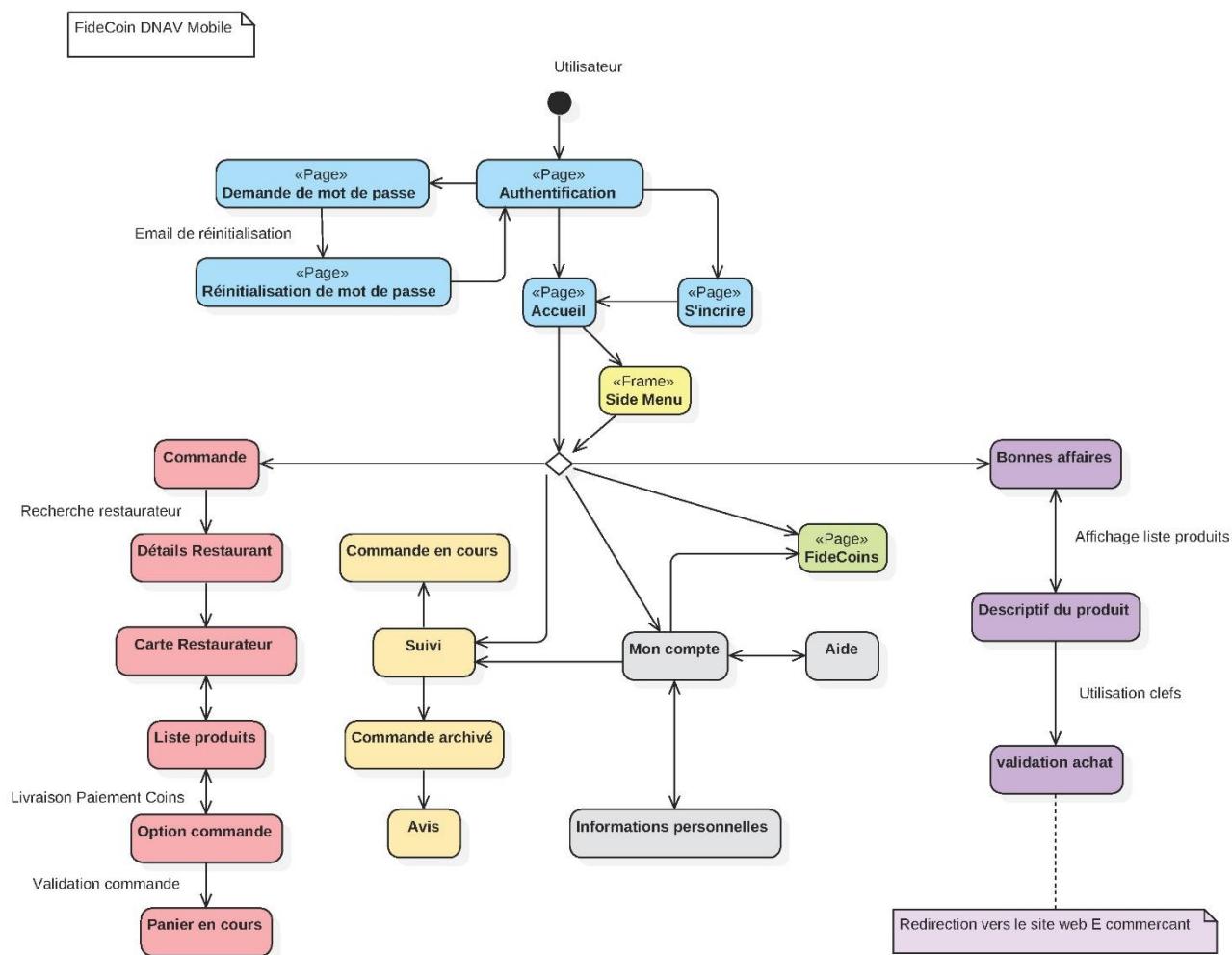
O. Options commande**P. Panier****Q. Avis**

Before	Now	After
Maquettes	Diagramme de Navigation	Diagramme de Séquence Système

4. DIAGRAMME DE NAVIGATION

Le diagramme de navigation, permet, en s'aidant des maquettes, de représenter la cinématique de l'application, la navigation entre les différentes pages ou menu de celle-ci. Il est d'une grande aide pour nous aider à définir nos futurs controllers et views.

J'ai représenté le Diagramme de Navigation avec plusieurs couleurs différentes pour bien séparer les activités possibles sur l'application.



<u>Légende :</u>	La partie authentification		La partie compte	
	La partie commande		La partie suivi de commande	
	La partie bonnes affaires		La partie fidélité	
	La partie Menu de navigation			

FIGURE 12 : DIAGRAMME DE NAVIGATION

Before	Now	After
Diagramme de Navigation	Diagrammes de Séquences Systèmes	Diagramme d'Activité

5. LES DIAGRAMMES DE SEQUENCES SYSTEMES

A. Authentification

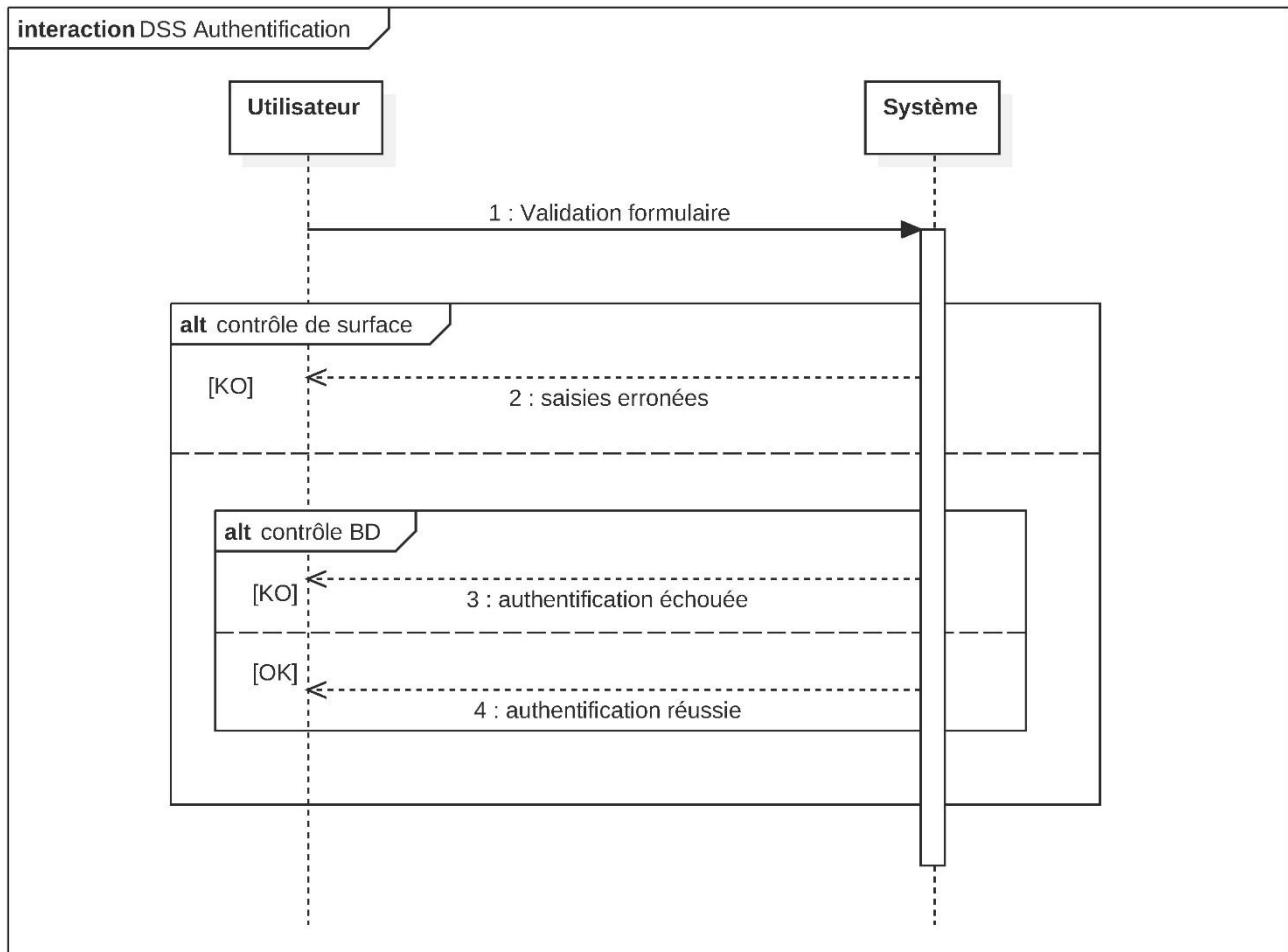


FIGURE 13 : DIAGRAMME DE SEQUENCE SYSTEME AUTHENTIFICATION

B. Afficher restaurateurs

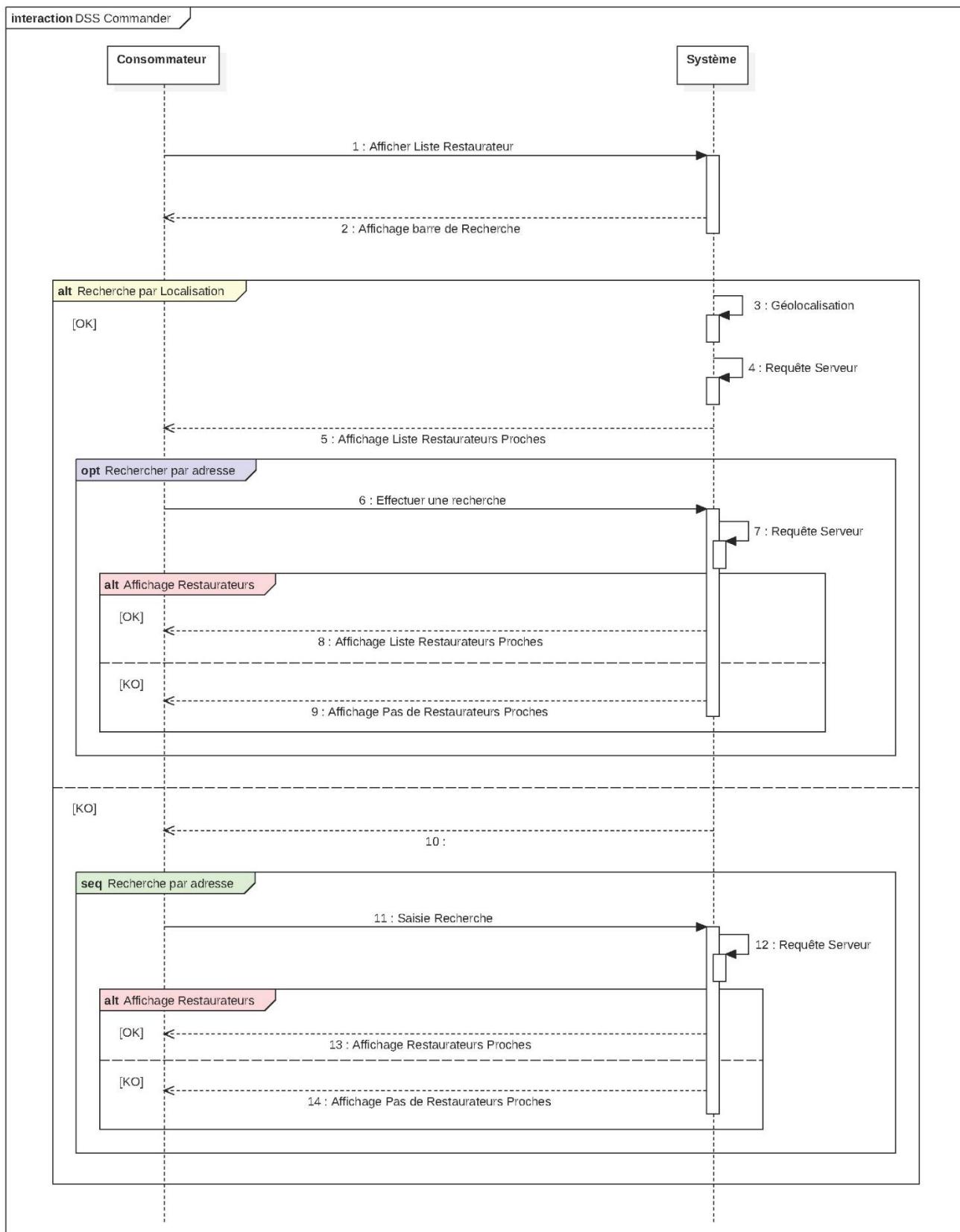


FIGURE 14 : DIAGRAMME DE SEQUENCE SYSTEME - AFFICHER RESTAURATEURS

Before	Now	After
Diagrammes de Séquences Systèmes	Diagramme d'Activité	

6. LES DIAGRAMMES D'ACTIVITE AUTHENTIFICATION ET INSCRIPTION

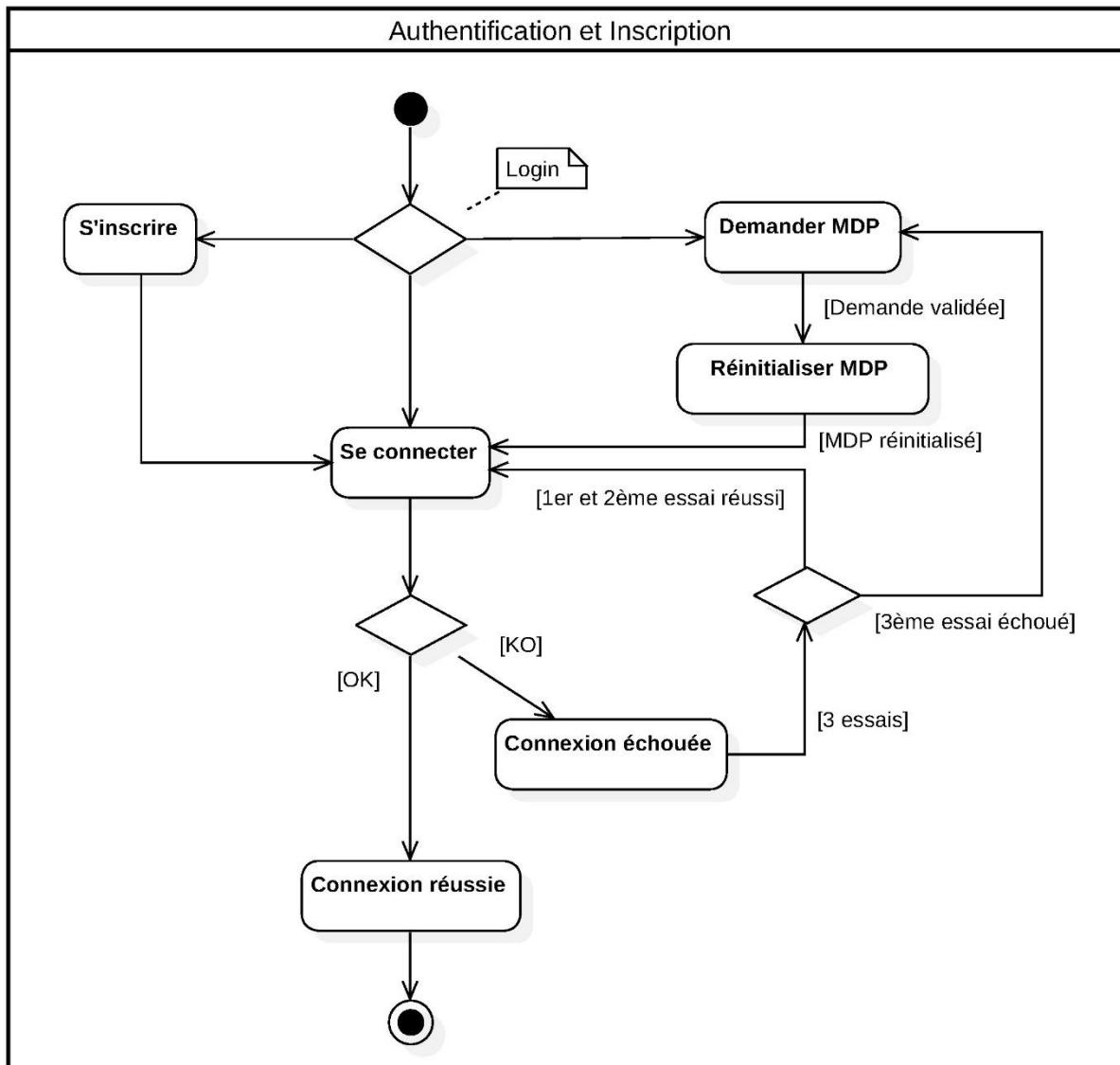


FIGURE 15 : DIAGRAMME D'ACTIVITE ET AUTHENTIFICATION ET INSCRIPTION

CHAPITRE 5 – CONCEPTION DE LA BASE DE DONNEES

Before	Now	After
Diagramme d'Activité	Création Base de Données	Dictionnaire de Données

1. INTRODUCTION

Pour créer la Base de Données je vais tout d'abord créer le Diagramme de classes du domaine. Les démarches présentées ici ne font pas parti ni d'UML ni de Up et dérivées (RUP, TUP, 2TUP).

Il y 3 principales démarches :

- La démarche par l'analyse du discours (ou règles de gestion),
- La démarche ascendante,
- La démarche par les formes.

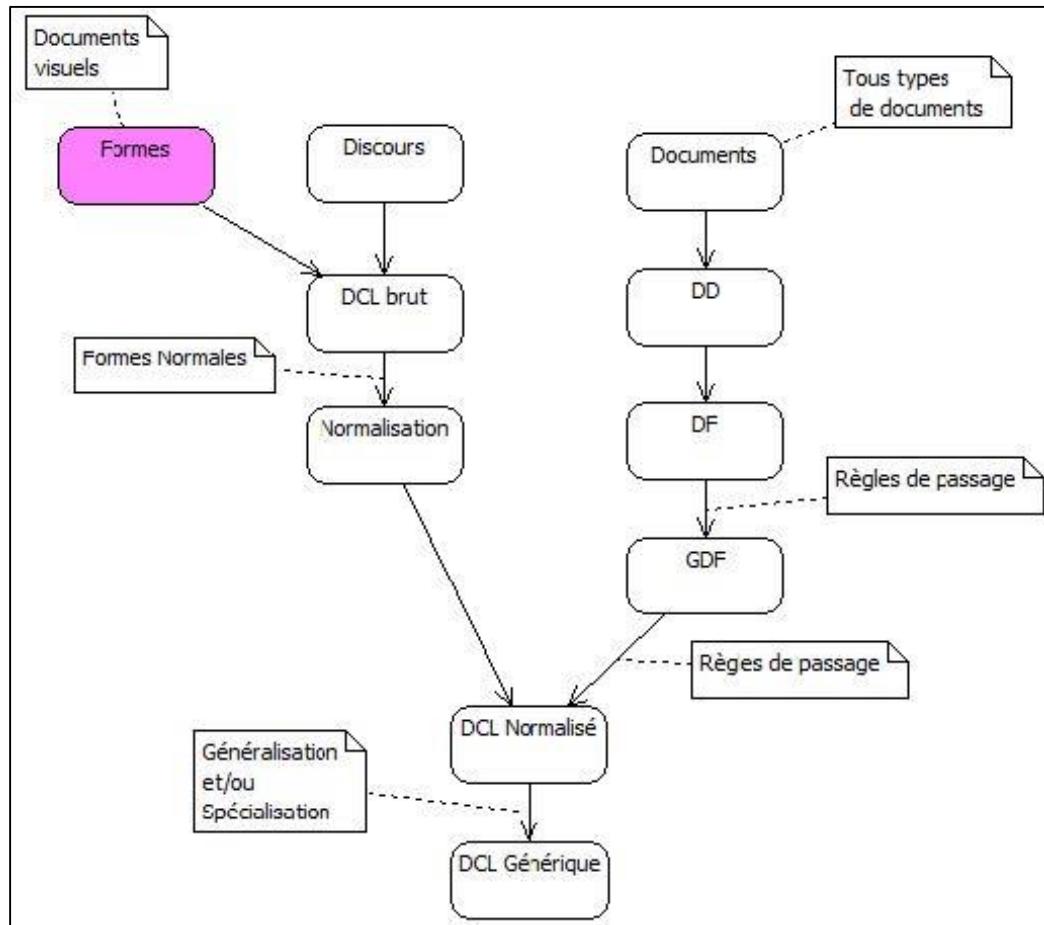


FIGURE 16 : LES 3 DEMARCHEES DE CREATION DU DCL

Grâce au cahier des charges fonctionnel, où le demandeur (notre Product Owner Kamel) exprime ses besoins en termes de fonctions de service et contraintes, nous pouvons créer le Diagramme de classes.

A partir des fonctions de service et de contraintes sont définies des critères d'appréciation ainsi que leurs niveaux qui eux-mêmes sont assortis d'un certain degré de flexibilité.

Nous avons élaboré ces éléments précédemment grâce aux DCU et au backlog du SCRUM.

2. DEMARCHE UTILISEE ET REGLES DE GESTION

Les classes et les associations sont repérées directement à partir du discours.

Un nom devient une classe et un verbe une association.

La démarche complète est la suivante :

- Repérer les classes,
- Repérer les associations (éventuellement les nommer),
- Poser les multiplicités,
- Lister les attributs et les placer,
- Lister les méthodes et les placer.

Mais pour notre projet nous avons décidé de ne pas seulement s'appuyer sur le discours.

Je me suis aidé de maquettes réalisées précédemment pour en tirer les données sous la forme d'un dictionnaire de données.

3. LA DEMARCHE ASCENDANTE

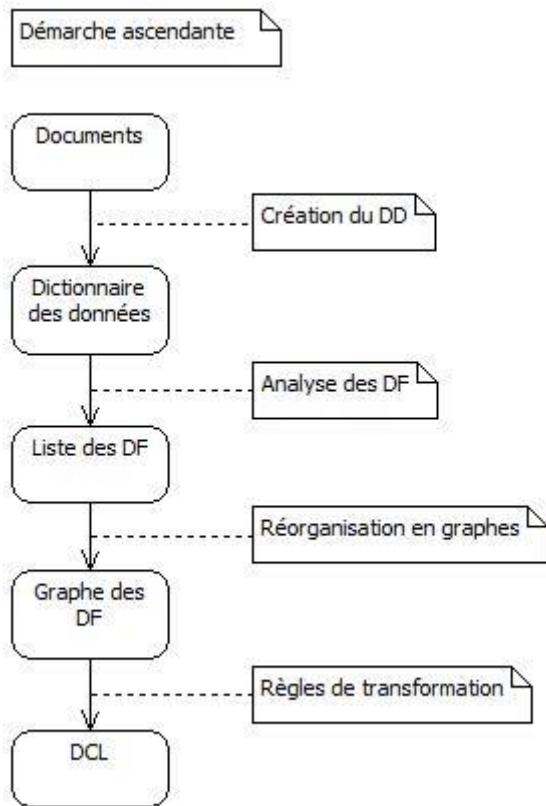


FIGURE 17 : DEMARCHE ASCENDANTE

Pour créer notre diagramme de classes j'ai utilisé la méthode ascendante qui permet de lever les ambiguïtés suite à une analyse du discours (ou règles de gestion).

Before	Now	After
Création Base de Données	Dictionnaire de Données	Graph des Dépendances Fonctionnelles

A. Création du dictionnaire de données

	Libellé	Code	Type	Longueur
1	id utilisateur	idUser	int	11
2	pseudo utilisateur	aliasUser	string	20
3	prénom utilisateur	firstNameUser	string	50
4	nom utilisateur	lastNameUser	string	50
5	email utilisateur	emailUser	string	50
6	mdp utilisateur	passwordUser	string	20
7	date inscription utilisateur	dateRegistrationUser	date	20
8	avatar utilisateur	avatarUser	string	20
9	id adresse	idUserAddress	int	11
10	Adresse	userAddress	string	50
11	complément d'adresse	additionalAddress	string	50
12	id type adresse	idAddressType	int	11
13	name adresse	nameAddressType	string	20
14	id Ville	idCity	int	11
15	nom ville	nameCity	string	50
16	cp	postalCode	string	5
17	id téléphone	idPhone	int	11
18	téléphone fixe	numberPhone	int	10
19	id type téléphone	idPhoneType	int	11
20	type téléphone	phoneType	string	20
21	id evaluation utilisateur	idEvaluation	int	11
22	date evaluation utilisateur	dateEvaluation	date	20
23	note evaluation utilisateur	noteEvaluation	int	1
24	commentaire evaluation utilisateur	textEvaluation	string	140
25	id fidecoin	idFideCoin	int	11
26	clé total	totalKeys	double	8
27	coins total	totalCoins	double	8
28	nombre clée en attente utilisateur	waitingKeys	double	8
29	nombre coins en attente utilisateur	waitingCoins	double	8
30	id réclamation	idClaim	int	11
31	date réclamation	dateClaim	date	20
32	texte réclamation	textClaim	string	140
33	id role utilisateur	idRole	int	11
34	type du rôle	typeRole	string	20
35	id commerçant	idMerchant	int	11
36	nom commerçant	nameMerchant	string	50
37	catégorie commerçant	categoryMerchant	string	50
38	cout livraison commerçant	deliveryCost	double	8
39	montant commande min commerçant	minOrderAmount	double	8
40	description congés	vacationDescription	string	140
41	Raison sociale resto	companyName	string	50
42	statut juridique commerçant	legalStatus	string	20
43	numéro SIREN	sirenNumber	int	10
44	id type commerçant	idMerchantType	int	11
45	type de commerçant	merchantType	string	20
46	id commande	idPurchaseOrder	int	11
47	date de commande	datePurchaseOrder	date	20
48	total commande	totalPurchaseOrder	double	8

FIGURE 18 : DICTIONNAIRE DE DONNEES

49	id produit	idProduct	int	11
50	nom produit resto	nameProduct	string	50
51	description produit resto	descriptionProduct	string	140
52	prix produit resto	priceProduct	double	8
53	id photo	idPhoto	int	11
54	nom Photo	namePhoto	string	50
55	description photo	descriptionPhoto	string	50
56	id TVA	idTva	int	11
57	valeur tva	valueTva	double	8
58	id produit restaurateurs	IdMerchantOrder	int	11
59	clées gagnées	winKeys	double	8
60	coins gagnés	winCoins	double	8
61	coins dépensés	spentCoins	int	8
62	status de livraison	deliveryStatus	string	20
63	id ligne de commande	idOrderLine	int	11
64	quantité commandé	quantityOrderLine	int	11
65	montant ligne commandé	sumOrderLine	double	8
66	nom du produit	nameProduct	string	20
67	prix du produit	priceProduct	double	8
68	description produit	descriptionProduct	string	140
69	valeur tva produit	valueTva	double	8
70	id catégorie resto	idCategory	int	11
71	nom categorie resto	nameCategory	string	50
72	id horaire	idSchedule	int	11
73	description horaire	descriptionSchedule	string	140
74	id jour	idDay	int	11
75	nom jour	nameDay	string	11
76	id tranche horaire	idTimeSlot	date	11
77	début de la tranche	startTimeSlot	date	11
78	fin de la tranche	endTimeSlot	date	11
79	id statut ouverture	idOpeningState	int	11
80	nom statut ouverture	nameOpeningState	string	20
81	id produit ecom	idProductItem	int	11
82	quantité article	quantityItem	int	11
83	id vente produit ecom	idSale	int	11
84	date mise en vente ecom	offeringDateSale	date	20
85	statut vente ecom	statusSale	int	11
86	prix produit actuel ecom	currentPrice	double	8
87	date de vente finale ecom	dateSale	date	20
88	total clées vente	totalKeysSale	int	8
89	vente terminé	endedSale	boolean	2
90	id commande ecom	idTraderOrder	int	11
91	nombre clef dépensé user	spentKeyUser	int	8
92	total clefs commande ecom	totalKeys	int	8

Légende :

Les données User



Les données Restaurateur



Les données E-Commerçant



Les données Communes Restaurateur et E-Commerçant



Before	Now	After
Dictionnaire de Données	Graph des Dépendances Fonctionnelles	Diagramme de classes

B. Graphes des dépendances fonctionnelles

Je n'ai représenté sur ce graphique que les sommets des arbres qui représentent les attributs identifiant de classe pour une meilleure lisibilité.

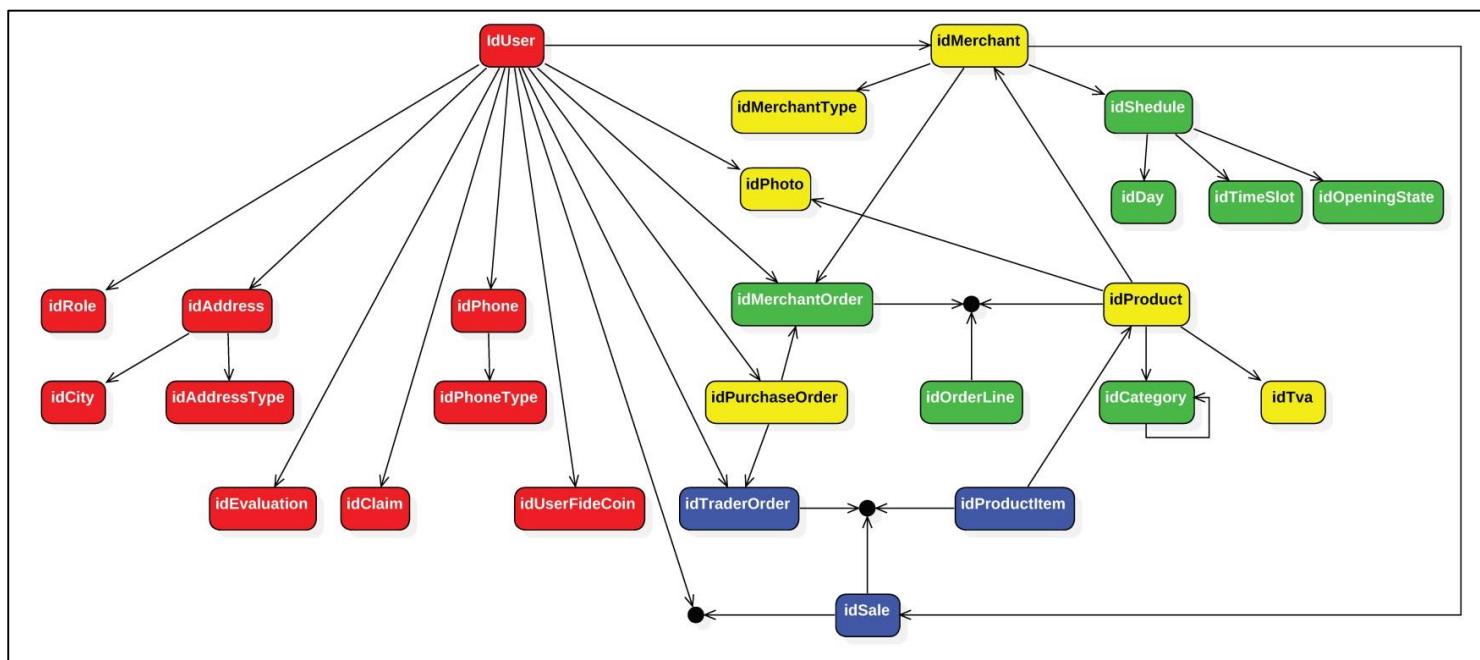


FIGURE 19 : GRAPHIQUE DES DEPENDANCES FONCTIONNELLES FIDEKOIN

Before

Graph des
Dépendances Fonctionnelles

Now

Diagramme de classes

After

Modèle Physique de Données

C. Diagramme de classes

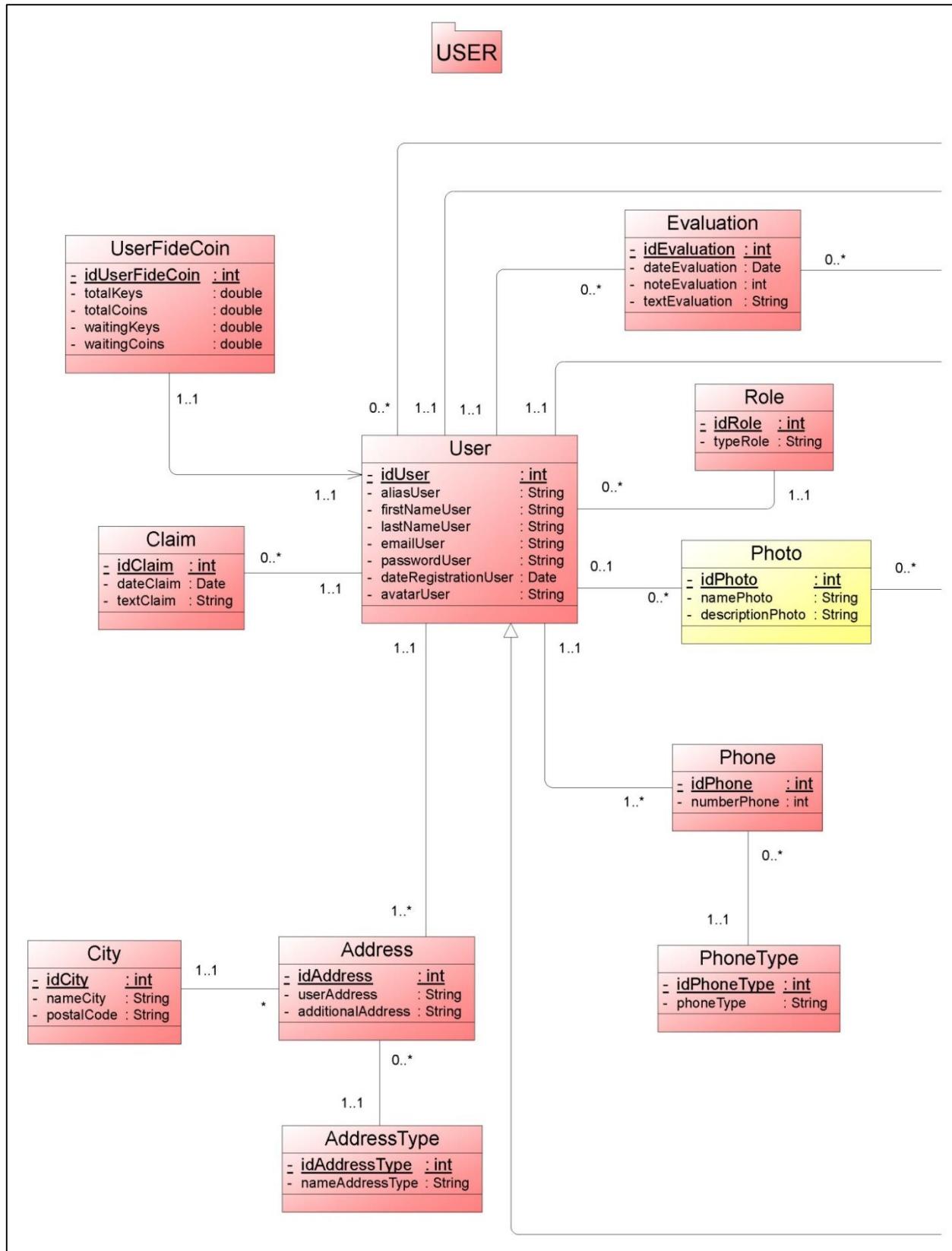


FIGURE 20 : DIAGRAMME DE CLASSES USER

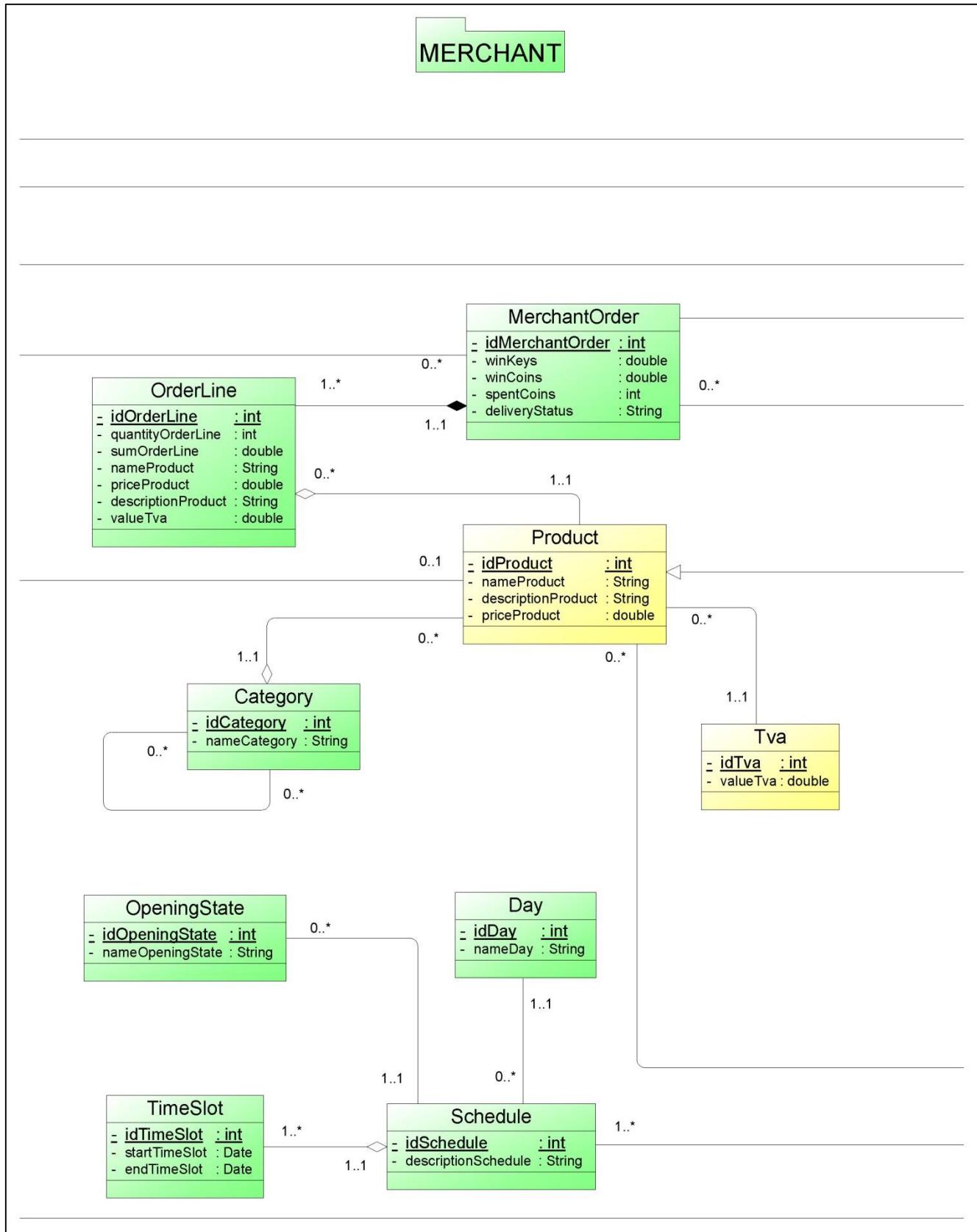


FIGURE 21 : DIAGRAMME DE CLASSES MERCHANT

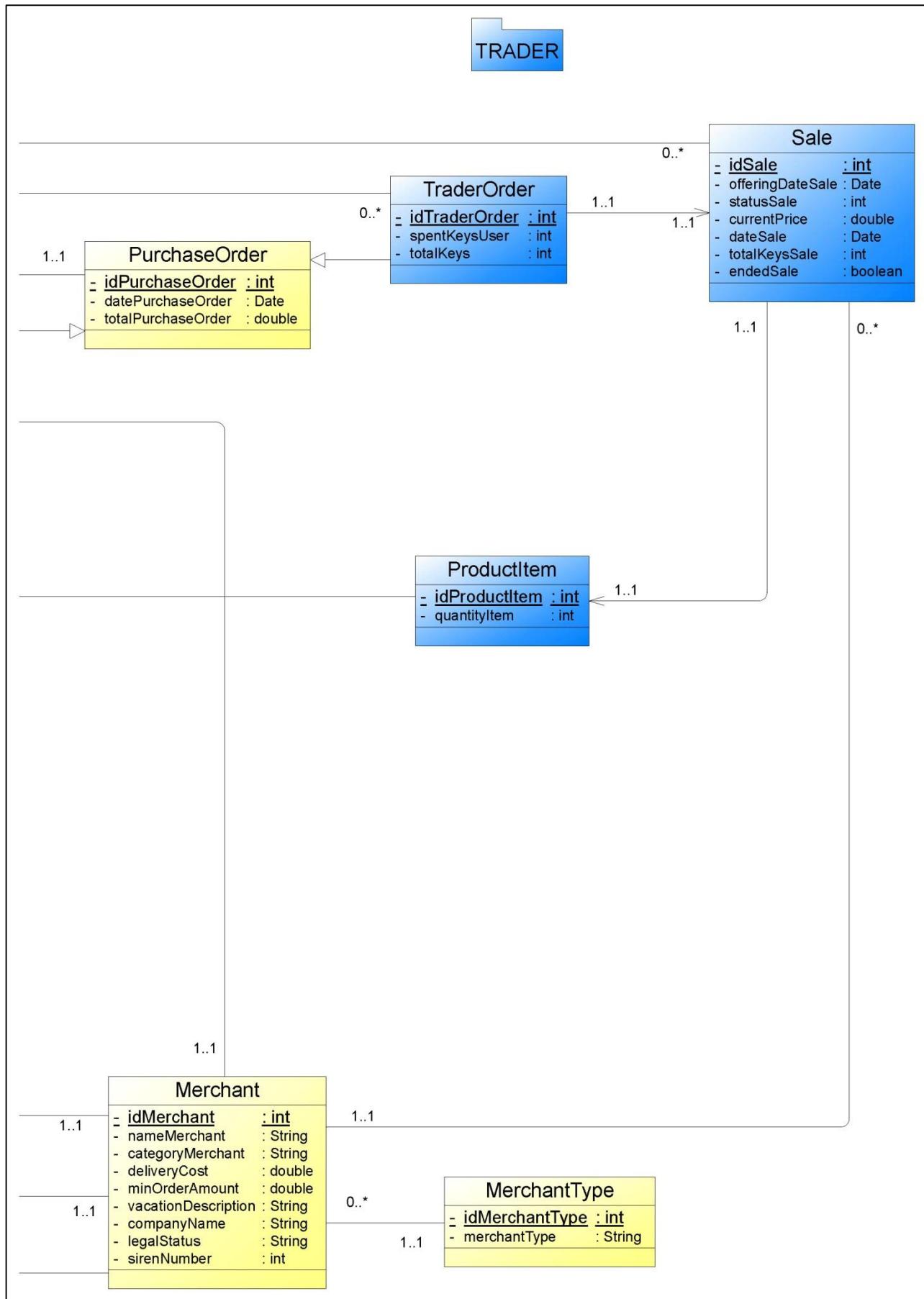


FIGURE 22 : DIAGRAMME DE CLASSES TRADER

Before	Now	After
Diagramme de classes	Modèle Physique de Données	Code SQL (LDD)

A. Le Modèle Physique de Données

a. Règles de passage du modèle Objet au modèle relationnel

Elément UML	Elément relationnel	Commentaire
Classe	Table	
Attribut <i>identifiant</i>	Clé primaire	
Attribut	Colonne	
Association de multiplicité 1-1..*	Clé étrangère	
Association de multiplicité 1-1	Clés étrangères « réciproques »	
Association de multiplicité *-*	Table de jointure avec une clé primaire composée de 2 clés étrangères	
Classe association	Table de jointure avec une clé primaire composée de 2 clés étrangères et autant de colonnes que d'attributs de la classe-association	
Agrégation	Tables, clé étrangère	Suppression avec affectation de NULL
Composition	Tables, clé étrangère	Suppression en cascade
Réflexive	Comme une association binaire	
Méthode	Fonction ou procédure stockée	
Contrainte	Fonction ou procédure stockée ou trigger	

b. Les schémas

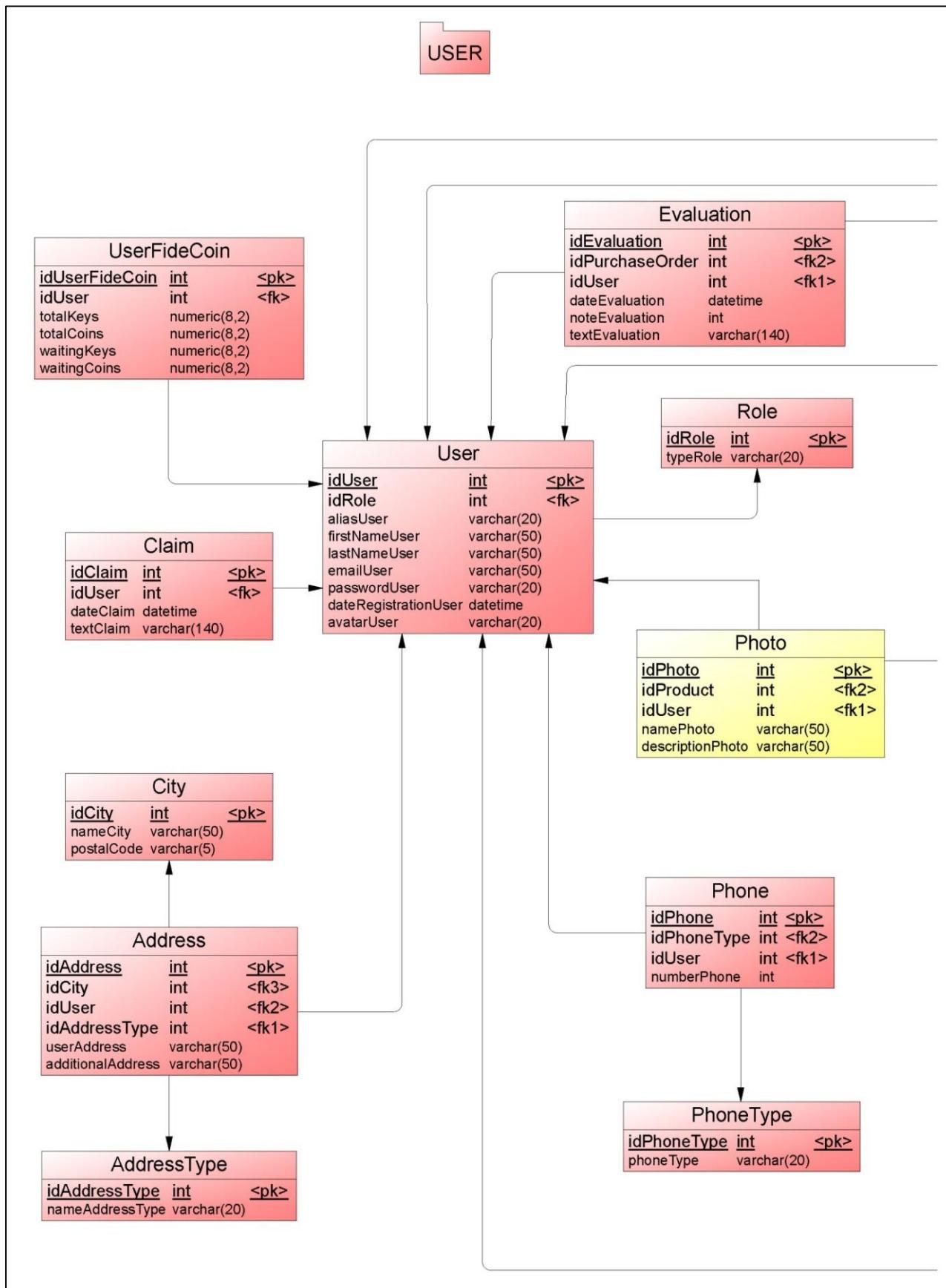


FIGURE 23 : MODELE PHYSIQUE DE DONNEES USER

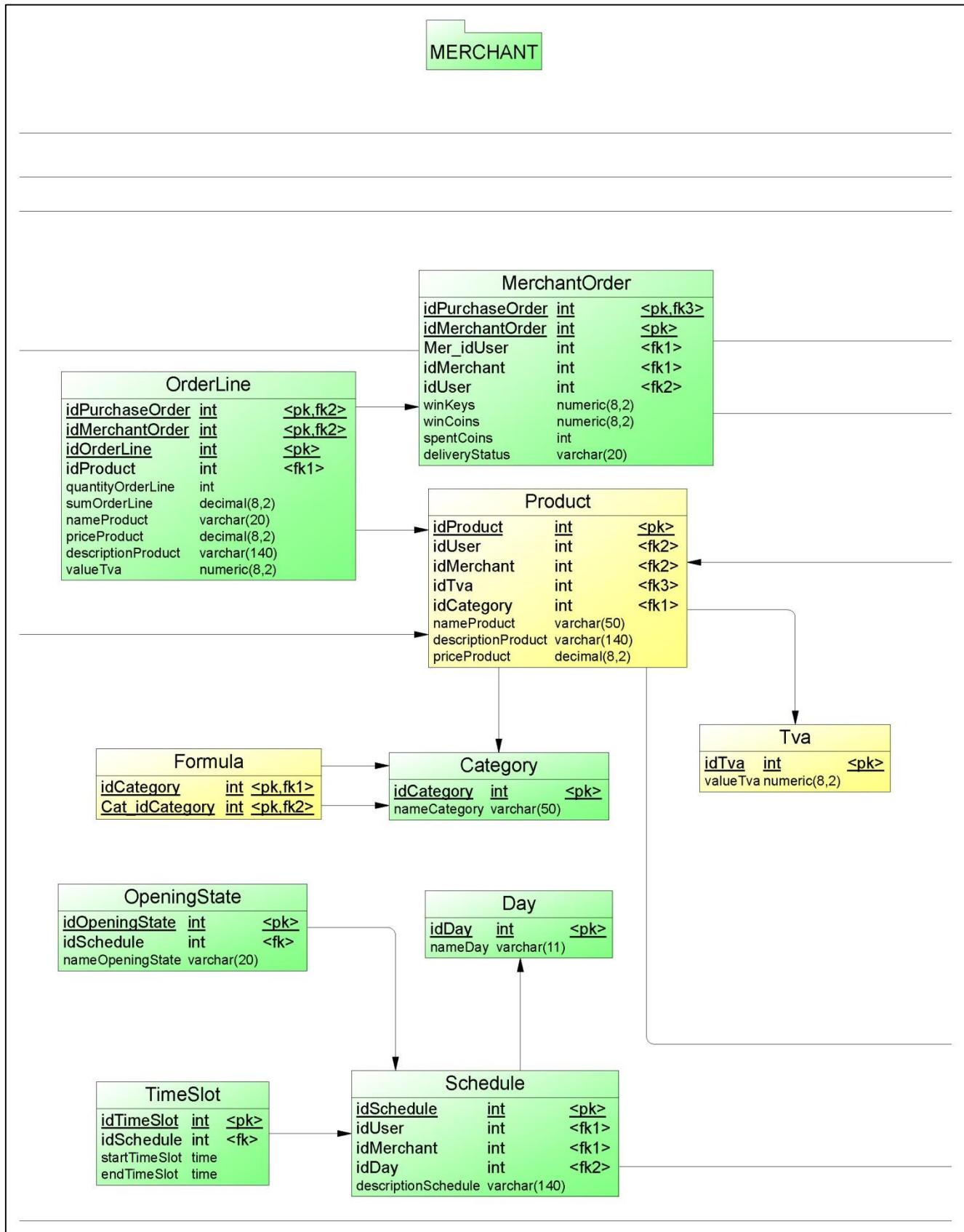


FIGURE 24 : MODELE PHYSIQUE DE DONNEES MERCHANT

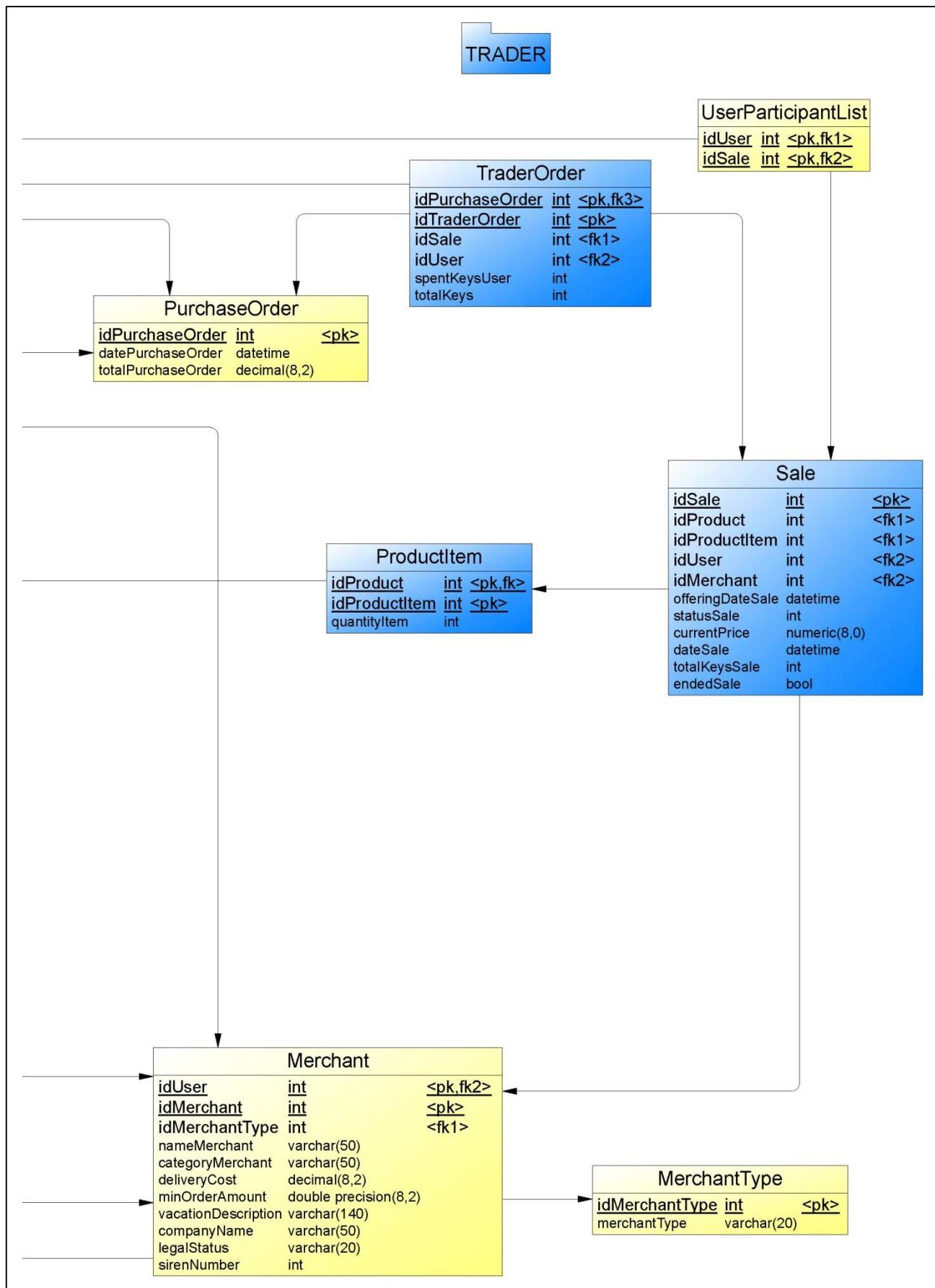


FIGURE 25 : MODELE PHYSIQUE DE DONNEES TRADER

Before	Now	After
Modèle Physique de Données	Code SQL (LDD)	

4. LE CODE SQL (LDD) DE CREATION DE LA BASE DE DONNEES

Le code complet se trouve en Annexe. Vous trouverez ici les tables liées à l'utilisateur consommateur.

```
/*
=====
/* Nom de SGBD : MySQL 5.0 */
/* Date de création : 12/04/2017 13:14:36 */
=====*/
/*
=====
/* Database : fidecoin */
/* ===== */
drop database if exists fidecoin;
create database fidecoin CHARACTER SET=utf8 COLLATE=utf8_unicode_ci;

use fidecoin;

/*
=====
/* Table : User */
/* ===== */
create table User
(
    idUser          int not null auto_increment,
    idRole          int not null,
    aliasUser       varchar(20),
    firstNameUser   varchar(50),
    lastNameUser    varchar(50),
    emailUser       varchar(50),
    passwordUser    varchar(20),
    dateRegistrationUser datetime,
    avatarUser      varchar(20),
    primary key (idUser)
);

/*
=====
/* Table : Role */
/* ===== */
create table Role
(
    idRole          int not null auto_increment,
    typeRole        varchar(20),
    primary key (idRole)
);
```

FIGURE 26 : CODE SQL (LDD) DE CREATION DE LA BD

```
/*=====
/* Table : Address
/*=====*/
create table Address
(
    idAddress      int not null auto_increment,
    idCity         int not null,
   idUser          int not null,
    idAddressType   int not null,
    userAddress     varchar(50),
    additionalAddress  varchar(50),
    primary key (idAddress)
);

/*=====
/* Table : Address_Type
/*=====*/
create table Address_Type
(
    idAddressType   int not null auto_increment,
    nameAddressType  varchar(20),
    primary key (idAddressType)
);

/*=====
/* Table : City
/*=====*/
create table City
(
    idCity          int not null auto_increment,
    nameCity        varchar(50) not null,
    postalCode      varchar(5),
    primary key (idCity)
);

/*=====
/* Table : Phone
/*=====*/
create table Phone
(
    idPhone         int not null auto_increment,
    idPhoneType     int not null,
    idUser          int not null,
    numberPhone     int,
    primary key (idPhone)
);

/*=====
/* Table : Phone_Type
/*=====*/
create table Phone_Type
(
    idPhoneType     int not null auto_increment,
    phoneType       varchar(20),
    primary key (idPhoneType)
);
```

```
/*=====
/* Table : User_FideCoin                               */
/*=====*/
create table User_FideCoin
(
    idUserFideCoin      int not null auto_increment,
   idUser              int not null,
    totalKeys          numeric(8,2),
    totalCoins          numeric(8,2),
    waitingKeys         numeric(8,2),
    waitingCoins        numeric(8,2),
    primary key (idUserFideCoin)
);

/*=====
/* Table : Claim                                     */
/*=====*/
create table Claim
(
    idClaim            int not null auto_increment,
    idUser              int not null,
    dateClaim           datetime,
    textClaim           varchar(140),
    primary key (idClaim)
);

/*=====
/* Table : Evaluation                                */
/*=====*/
create table Evaluation
(
    idEvaluation        int not null auto_increment,
    idPurchaseOrder     int not null,
    idUser               int not null,
    dateEvaluation       datetime,
    noteEvaluation       int,
    textEvaluation        varchar(140),
    primary key (idEvaluation)
);

/*=====
/* Table : Purchase_Order                            */
/*=====*/
create table Purchase_Order
(
    idPurchaseOrder      int not null auto_increment,
    datePurchaseOrder    datetime,
    totalPurchaseOrder   decimal(8,2),
    primary key (idPurchaseOrder)
);
```

```

/*=====
/* Table : Merchant_Order
/*=====*/
create table Merchant_Order
(
    idPurchaseOrder      int not null,
    idMerchantOrder      int not null,
    Mer_idUser           int not null,
    idMerchant            int not null,
   idUser                int not null,
    winKeys              numeric(8,2),
    winCoins              numeric(8,2),
    spentCoins           int,
    deliveryStatus        varchar(20),
    primary key (idPurchaseOrder, idMerchantOrder)
);

/*=====
/* Table : Trader_Order
/*=====*/
create table Trader_Order
(
    idPurchaseOrder      int not null,
    idTraderOrder         int not null,
    idSale                int not null,
    idUser                int not null,
    spentKeysUser         int,
    totalKeys             int,
    primary key (idPurchaseOrder, idTraderOrder)
);

/*=====
/* Table : User_Participant_List
/*=====*/
create table User_Participant_List
(
    idUser                int not null,
    idSale                int not null,
    primary key (idUser, idSale)
);

/*=====
/* Table : Sale
/*=====*/
create table Sale
(
    idSale                int not null auto_increment,
    idProduct             int not null,
    idProductItem          int not null,
    idUser                int not null,
    idMerchant             int not null,
    offeringDateSale       datetime,
    statusSale              int,
    currentPrice           numeric(8,0),
    dateSale                datetime,
    totalKeysSale           int,
    endedSale               bool,
    primary key (idSale)
);

```

CHAPITRE 6 – CONCEPTION DE L'APPLICATION

CHAPITRE 7 – DEVELOPPEMENT

1. LE CODE DE LA PAGE APP

2. LE CODE DE LA PAGE D'AUTHENTIFICATION

A. Authentication.html

```
<ion-content padding class="backgrd">

<ion-img src="assets/icon/FideCoinLogoSeul.png" id="logo"></ion-img>

<ion-list no-lines>

<form [formGroup]="authForm">

    <div class="spacer"></div>

    <!-- INPUT EMAIL-->
    <ion-item class="form">
        <ion-input placeholder="Votre email" [(ngModel)]="email" formControlName="email"
        type="email" [class.invalid]="!authForm.controls.email.valid &&
        (authForm.controls.email.dirty || submitAttempt)"></ion-input>
        </ion-item>
        <div *ngIf="!authForm.controls.email.valid && (authForm.controls.email.dirty || submitAttempt)" class="error">Veuillez entrer un email valide</div>
        <div class="spacer"></div>

    <!-- INPUT PASSWORD-->
    <ion-item class="form">
        <ion-input placeholder="Votre mot de passe" [(ngModel)]="password"
        formControlName="password" type="password"
        [class.invalid]="!authForm.controls.password.valid && (authForm.controls.password.dirty
        || submitAttempt)"></ion-input>
        </ion-item>
        <div *ngIf="!authForm.controls.password.valid &&
        (authForm.controls.password.dirty || submitAttempt)" class="error">Veuillez entrer un mot de passe valide</div>
        <div class="spacer"></div>

        <button *ngIf="!authForm.controls.password.valid &&
        (authForm.controls.password.dirty || submitAttempt)" id="forgot" ion-button
        small color="greeny" (tap)="showForgotPasswordPage()">Mot de passe oublié
    ?</button>
        <div class="spacer"></div>

        <button ion-button block color="orangy" (click)="validateAuth()">
        Connexion</button>
        <div class="spacer"></div>

        <button ion-button block color="greeny" (click)="signup()">Ou créez votre Compte</button>
    </form>

</ion-list>

</ion-content>
```

B. Authentification.scss

```
page-authentification {  
    #logo {  
        background: url("../assets/icon/FideCoinLogoSeul.png");  
        display: block;  
        margin-left: auto;  
        margin-right: auto;  
        height: 135px  
    }  
    #forgot {  
        display: block;  
        margin: 15px auto;  
    }  
    .backgrd {  
        background-color: #B5CBB7;  
        //background: url('../assets/img/drawable-port-mdpi-screen.png') no-repeat  
        center;  
        //background-size: cover;  
    }  
    .form {  
        background-color: rgba(0, 0, 0, 0);  
    }  
    .error {  
        color: red;  
        font-size: 0.8em;  
        text-align: center  
    }  
    ion-input {  
        background-color: #f2f2f2;  
    }  
    .invalid {  
        border: 3px solid red;  
    }  
    .spacer{  
        margin: 10px;  
    }  
}
```

C. Authentification.ts

```

import {FormBuilder, FormGroup, Validators} from '@angular/forms';
import {Component} from '@angular/core';
import {NavController, NavParams} from 'ionic-angular';

import {ServiceProvider} from '../../providers/service-provider';

//Pages
import {WelcomePage} from '../welcome/welcome';
import {ForgotPassword1Page} from "../forgot-password1/forgot-password1";
import {RegistrationPage} from "../registration/registration";
import {ShoppingCartPage} from '../shopping-cart/shopping-cart';

@Component({selector: 'page-authentification', templateUrl: 'authentification.html'})

export class AuthentificationPage {

  users : any[];

  authForm : FormGroup;
  submitAttempt : boolean = false;
  email : string;
  password : string;

  constructor(public navCtrl : NavController, public navParams : NavParams, public
  FormBuilder : FormBuilder, public service : ServiceProvider) {

    this.authForm = FormBuilder.group({
      email: [
        '', Validators.compose([
          Validators.minLength(11),
          Validators.maxLength(30),
          Validators.pattern('^[a-zA-Z0-9.!#$%&*+=?^`{|~}-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$'),
          Validators.required
        ])
      ],
      password: [
        '', Validators.compose([
          Validators.maxLength(30),
          // Regex Password with UpperCase, LowerCase, Number or SpecialChar and min 8
          // Chars
          Validators.pattern('(?=^.{8,$})(?=.*\d)(?=.*[W+])(?![\n])(?=.*[A-Z])(?=.*[a-z]).*$'),
          Validators.required
        ])
      ]
    });

    this.email = navParams.get('email');
    this.password = navParams.get('password');
  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad AuthentificationPage');
  }
}

```

```
validateAuth() {
    this.submitAttempt = true;

    if (!this.authForm.valid) {
        this.submitAttempt = true;
    } else {
        console.log("success!")
        console.log(this.authForm.value);
        this.submitAttempt = false;
        this
            .navCtrl
            .push(WelcomePage, {
                email: this.email,
                password: this.password
            })
    }
}

showForgotPasswordPage() : void {
    this
        .navCtrl
        .push(ForgotPassword1Page);
}

signup() : void {
    this
        .navCtrl
        .push(RegistrationPage);
}

basketToggle() : void {
    this
        .navCtrl
        .push(ShoppingCartPage);
}

getData() {
    this
        .service
        .getData()
        .subscribe(data => this.users = data, err => console.log(err));
}
```

3. LE CODE DE LA PAGE D'INSCRIPTION

A. Registration.html

B. Registration.scss

C. Registration.ts

CHAPITRE 8 – DEPLOIEMENT

Before	Now	After
	Déploiement	

1. LE DEPLOIEMENT

Le diagramme de déploiement permet de représenter la structure physique du système, sur laquelle sont répartis les différents composants ainsi que leurs relations.

On retrouve principalement, dans un diagramme de déploiement, 4 types d'éléments :

- Les nœuds (nodes), qui sont représentés par des cubes, sont des composants physiques du système (Serveur, Ordinateur...).
- Les composants, qui sont représentés par un rectangle, représentent les différentes briques logicielles du système. Les composants colorés(en bleu dans notre cas), représentent des composants que nous n'avons pas conçu, ce sont le plus souvent des solutions logicielles tierces déjà en place sur le système.
- Les associations, sont de simples lignes représentant les voies de communications entre les nœuds du système, ainsi que leurs composants.

Pour Le projet FideCoin, le déploiement correspond donc à :

- Des installateurs qui permettent le déploiement de l'application sur :
 - Android : un fichier *.apk,
 - iOS : un fichier *.ipa,
 - Windows Mobile : *.cab.
- Un serveur apache2 sur lequel nous déployeront nos différents scripts HTML, PHP, Node.js, Javascript. Il permettra la consultation à distance.
- Un serveur de base de données relationnelle MySQL, sur lequel sera exécuté notre LDD lors du déploiement, ce qui aura pour effet de créer le schéma.

2. LE DIAGRAMME DE DEPLOIEMENT

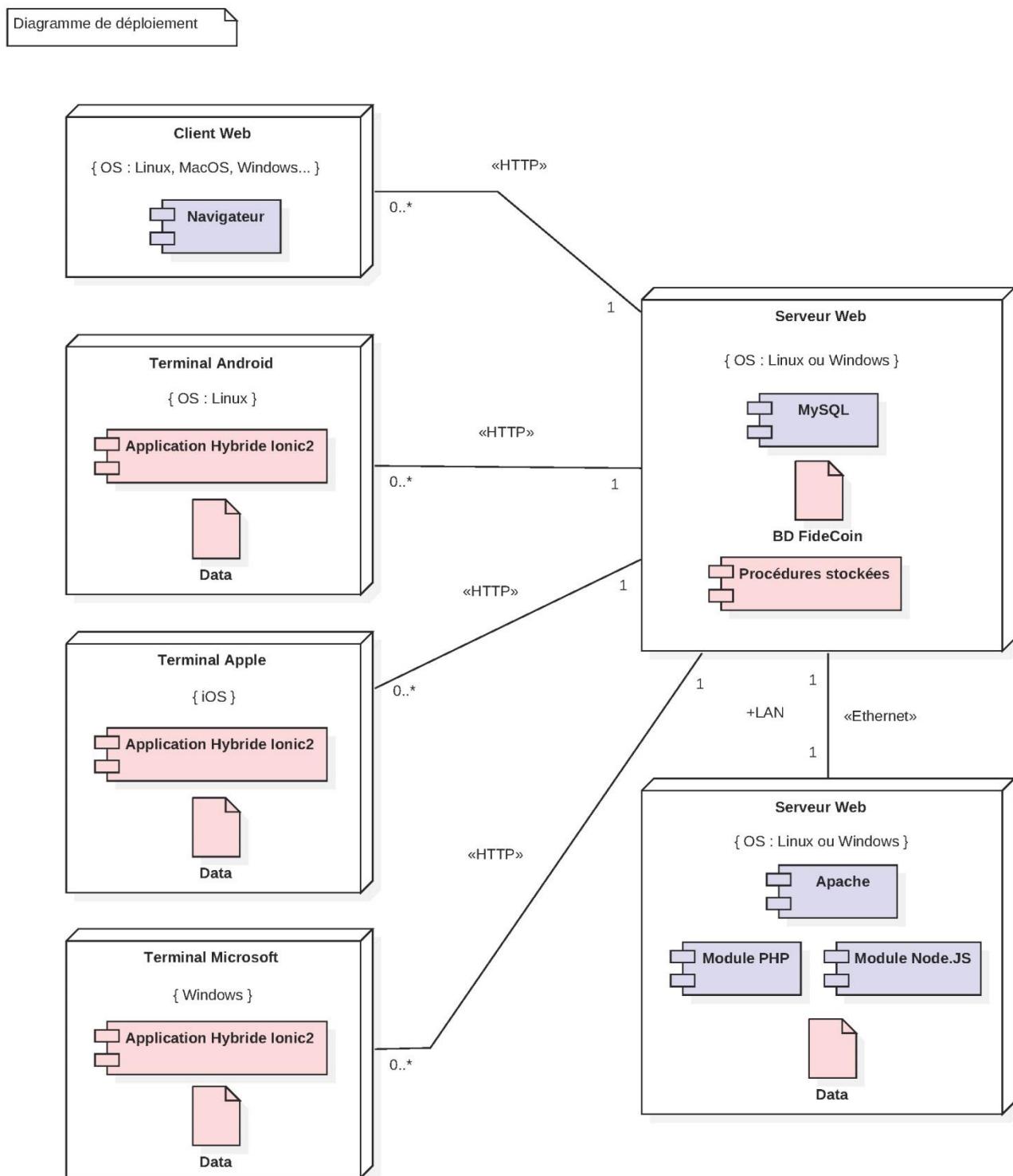


FIGURE 27 : DIAGRAMME DE DEPLOIEMENT

CHAPITRE 9 – CONCLUSION

CHAPITRE 10 – ANNEXES

1. CORRESPONDANCES PROJET/REAC

COMPETENCES DU REAC	CORRESPONDANCES DANS LE PROJET
Développer des composants d'interface	
Maquetter une application	UML (Diagramme de Cas d'Utilisation, Maquettes, Diagramme de Séquences Système, Diagramme de Navigation, Diagramme d'Activité, Diagramme de Séquence Détaillé) avec StarUml Maquettes de l'IHM avec IONIC CREATOR
Développer une interface utilisateur	Ionic2, HTML5, CSS3-SCSS, AngularJS2
Développer des composants d'accès aux données	Langage serveur (PHP, ...) avec du SQL et du JSON.
Développer des pages web en lien avec une base de données	HTML5, CSS3-SCSS, AngularJS2, AJAX, langage serveur (PHP, ..)
Développer la persistance des données	
Concevoir une base de données	Le Diagramme de classes et le Modèle Physique des Données (MPD) avec PowerAMC
Mettre en place une base de données	SQL (LLD) : Création de la BD, Création des tables et mise en place des contraintes (FK) sur les tables avec PowerAMC (CREATE, ALTER, DROP, GRANT et REVOKE)
Développer des composants dans le langage d'une base de données	Le langage procédural de MySQL
Utiliser l'anglais dans son activité professionnelle en informatique	Diagrammes UML, maquettes, développement, commentaires du code
Développer une application n-tiers.	
Concevoir une application	UML (Diagramme de Séquence Détaillé)
Collaborer à la gestion d'un projet informatique	Gestion de projet en Agile avec la méthode SCRUM
Développer des composants métier	Modèles en Typescript.
Construire une application organisée en couches	MVC (Diagrammes de séquence détaillés) Modèle : *.ts – Vue : *.html – Contrôleur : Ionic/Angular2
Développer une application de mobilité numérique	Géolocalisation, Ionic2 application hybride
Préparer et exécuter les plans de tests d'une application	Karma et Jasmine pour les tests unitaires
Préparer et exécuter le déploiement d'une application	UML (Diagramme de Déploiement) avec StarUml – Crédit d'un exécutable *.apk pour Android – Déploiement - Installation d'un serveur de BD, de la BD.

FIGURE 28 : TABLEAU DE CORRESPONDANCES PROJET/REAC

2. CODE SQL COMPLET

```

/*=====
/* Nom de SGBD : MySQL 5.0
/* Date de création : 12/04/2017 13:14:36
/*=====

/*=====
/* Database : fidecoin
/*=====

drop database if exists fidecoin;
create database fidecoin CHARACTER SET=utf8 COLLATE=utf8_unicode_ci;

use fidecoin;

/*=====
/* Table : Address
/*=====

create table Address
(
    idAddress      int not null auto_increment,
    idCity         int not null,
   idUser          int not null,
    idAddressType   int not null,
    userAddress     varchar(50),
    additionalAddress  varchar(50),
    primary key (idAddress)
);

/*=====
/* Table : Address_Type
/*=====

create table Address_Type
(
    idAddressType   int not null auto_increment,
    nameAddressType  varchar(20),
    primary key (idAddressType)
);

/*=====
/* Table : Category
/*=====

create table Category
(
    idCategory      int not null auto_increment,
    nameCategory     varchar(50),
    primary key (idCategory)
);

/*=====
/* Table : City
/*=====

create table City
(
    idCity          int not null auto_increment,
    nameCity        varchar(50) not null,
    postalCode      varchar(5),
    primary key (idCity)
);

```

```

/*=====
/* Table : Claim
/*=====*/
create table Claim
(
    idClaim          int not null auto_increment,
   idUser           int not null,
    dateClaim        datetime,
    textClaim        varchar(140),
    primary key (idClaim)
);

/*=====
/* Table : Day
/*=====*/
create table Day
(
    idDay            int not null auto_increment,
    nameDay          varchar(11),
    primary key (idDay)
);

/*=====
/* Table : Evaluation
/*=====*/
create table Evaluation
(
    idEvaluation     int not null auto_increment,
    idPurchaseOrder  int not null,
    idUser           int not null,
    dateEvaluation   datetime,
    noteEvaluation   int,
    textEvaluation   varchar(140),
    primary key (idEvaluation)
);

/*=====
/* Table : Formula
/*=====*/
create table Formula
(
    idCategory       int not null,
    Cat_idCategory   int not null,
    primary key (idCategory, Cat_idCategory)
);

/*=====
/* Table : Merchant
/*=====*/
create table Merchant
(
    idUser           int not null,
    idMerchant       int not null,
    idMerchantType   int not null,
    nameMerchant     varchar(50),
    categoryMerchant varchar(50),
    deliveryCost     decimal(8,2),
    minOrderAmount   double precision(8,2),
    vacationDescription varchar(140),
    companyName      varchar(50),
);

```

```

legalStatus          varchar(20),
sirenNumber         int,
primary key (idUser, idMerchant)
);

/*=====
/* Table : Merchant_Order
*/
=====*/
create table Merchant_Order
(
    idPurchaseOrder      int not null,
    idMerchantOrder     int not null,
    Mer_idUser          int not null,
    idMerchant          int not null,
    idUser              int not null,
    winKeys             numeric(8,2),
    winCoins            numeric(8,2),
    spentCoins          int,
    deliveryStatus      varchar(20),
    primary key (idPurchaseOrder, idMerchantOrder)
);

/*=====
/* Table : Merchant_Type
*/
=====*/
create table Merchant_Type
(
    idMerchantType       int not null auto_increment,
    merchantType         varchar(20),
    primary key (idMerchantType)
);

/*=====
/* Table : Opening_State
*/
=====*/
create table Opening_State
(
    idOpeningState        int not null auto_increment,
    idSchedule            int not null,
    nameOpeningState      varchar(20),
    primary key (idOpeningState)
);

/*=====
/* Table : Order_Line
*/
=====*/
create table Order_Line
(
    idPurchaseOrder      int not null,
    idMerchantOrder     int not null,
    idOrderLine          int not null,
    idProduct            int not null,
    quantityOrderLine    int,
    sumOrderLine         decimal(8,2),
    nameProduct          varchar(20),
    priceProduct         decimal(8,2),
    descriptionProduct   varchar(140),
    valueTva             numeric(8,2),
    primary key (idPurchaseOrder, idMerchantOrder, idOrderLine)
);

```

```

/*=====
/* Table : Phone
/*=====
create table Phone
(
    idPhone          int not null auto_increment,
    idPhoneType      int not null,
   idUser           int not null,
    numberPhone      int,
    primary key (idPhone)
);

/*=====
/* Table : Phone_Type
/*=====
create table Phone_Type
(
    idPhoneType      int not null auto_increment,
    phoneType        varchar(20),
    primary key (idPhoneType)
);

/*=====
/* Table : Photo
/*=====
create table Photo
(
    idPhoto          int not null auto_increment,
    idProduct        int,
    idUser           int,
    namePhoto        varchar(50),
    descriptionPhoto varchar(50),
    primary key (idPhoto)
);

/*=====
/* Table : Product
/*=====
create table Product
(
    idProduct        int not null auto_increment,
    idUser           int not null,
    idMerchant       int not null,
    idTva            int not null,
    idCategory       int not null,
    nameProduct      varchar(50),
    descriptionProduct varchar(140),
    priceProduct     decimal(8,2),
    primary key (idProduct)
);

/*=====
/* Table : Product_Item
/*=====
create table Product_Item
(
    idProduct        int not null,
    idProductItem    int not null,
    quantityItem     int,
    primary key (idProduct, idProductItem)
);

```

```
/*=====
/* Table : Purchase_Order
/*=====*/
create table Purchase_Order
(
    idPurchaseOrder      int not null auto_increment,
    datePurchaseOrder    datetime,
    totalPurchaseOrder   decimal(8,2),
    primary key (idPurchaseOrder)
);

/*=====
/* Table : Role
/*=====*/
create table Role
(
    idRole              int not null auto_increment,
    typeRole            varchar(20),
    primary key (idRole)
);

/*=====
/* Table : Sale
/*=====*/
create table Sale
(
    idSale               int not null auto_increment,
    idProduct            int not null,
    idProductItem        int not null,
   idUser                int not null,
    idMerchant           int not null,
    offeringDateSale     datetime,
    statusSale            int,
    currentPrice          numeric(8,0),
    dateSale              datetime,
    totalKeysSale         int,
    endedSale             bool,
    primary key (idSale)
);

/*=====
/* Table : Schedule
/*=====*/
create table Schedule
(
    idSchedule           int not null auto_increment,
    idUser                int not null,
    idMerchant           int not null,
    idDay                 int not null,
    descriptionSchedule  varchar(140),
    primary key (idSchedule)
);
```

```
/*=====
/* Table : Time_Slot
/*=====*/
create table Time_Slot
(
    idTimeSlot      int not null auto_increment,
    idSchedule      int not null,
    startTimeSlot   time,
    endTimeSlot     time,
    primary key (idTimeSlot)
);

/*=====
/* Table : Trader_Order
/*=====*/
create table Trader_Order
(
    idPurchaseOrder  int not null,
    idTraderOrder    int not null,
    idSale           int not null,
   idUser            int not null,
    spentKeysUser    int,
    totalKeys        int,
    primary key (idPurchaseOrder, idTraderOrder)
);

/*=====
/* Table : Tva
/*=====*/
create table Tva
(
    idTva          int not null auto_increment,
    valueTva       numeric(8,2),
    primary key (idTva)
);

/*=====
/* Table : User
/*=====*/
create table User
(
    idUser         int not null auto_increment,
    idRole         int not null,
    aliasUser      varchar(20),
    firstNameUser  varchar(50),
    lastNameUser   varchar(50),
    emailUser      varchar(50),
    passwordUser   varchar(20),
    dateRegistrationUser datetime,
    avatarUser     varchar(20),
    primary key (idUser)
);
```

```

/*=====
/* Table : User_FideCoin
/*=====*/
create table User_FideCoin
(
    idUserFideCoin      int not null auto_increment,
   idUser              int not null,
    totalKeys          numeric(8,2),
    totalCoins          numeric(8,2),
    waitingKeys         numeric(8,2),
    waitingCoins        numeric(8,2),
    primary key (idUserFideCoin)
);

/*=====
/* Table : User_Participant_List
/*=====*/
create table User_Participant_List
(
    idUser            int not null,
    idSale             int not null,
    primary key (idUser, idSale)
);

alter table Address add constraint FK_Association_13 foreign key (idAddressType)
    references Address_Type (idAddressType) on delete restrict on update restrict;

alter table Address add constraint FK_Association_14 foreign key (idUser)
    references User (idUser) on delete restrict on update restrict;

alter table Address add constraint FK_Association_15 foreign key (idCity)
    references City (idCity) on delete restrict on update restrict;

alter table Claim add constraint FK_Association_5 foreign key (idUser)
    references User (idUser) on delete restrict on update restrict;

alter table Evaluation add constraint FK_Association_1 foreign key (idUser)
    references User (idUser) on delete restrict on update restrict;

alter table Evaluation add constraint FK_Association_29 foreign key (idPurchaseOrder)
    references Purchase_Order (idPurchaseOrder) on delete restrict on update restrict;

alter table Formula add constraint FK_Formula foreign key (idCategory)
    references Category (idCategory) on delete restrict on update restrict;

alter table Formula add constraint FK_Formula2 foreign key (Cat_idCategory)
    references Category (idCategory) on delete restrict on update restrict;

alter table Merchant add constraint FK_Association_39 foreign key (idMerchantType)
    references Merchant_Type (idMerchantType) on delete restrict on update restrict;

alter table Merchant add constraint FK_Generalisation_2 foreign key (idUser)
    references User (idUser) on delete restrict on update restrict;

alter table Merchant_Order add constraint FK_Association_22 foreign key (Mer_idUser,
idUser)
    references Merchant (idUser, idMerchant) on delete restrict on update restrict;

alter table Merchant_Order add constraint FK_Association_24 foreign key (idUser)
    references User (idUser) on delete restrict on update restrict;

```

```

alter table Merchant_Order add constraint FK_Generalisation_5 foreign key
(idPurchaseOrder)
    references Purchase_Order (idPurchaseOrder) on delete restrict on update restrict;

alter table Opening_State add constraint FK_Association_46 foreign key (idSchedule)
references Schedule (idSchedule) on delete restrict on update restrict;

alter table Order_Line add constraint FK_Association_32 foreign key (idPurchaseOrder,
idMerchantOrder)
    references Merchant_Order (idPurchaseOrder, idMerchantOrder) on delete restrict on
update restrict;

alter table Order_Line add constraint FK_Association_33 foreign key (idProduct)
references Product (idProduct) on delete restrict on update restrict;

alter table Phone add constraint FK_Association_3 foreign key (idUser)
references User (idUser) on delete restrict on update restrict;

alter table Phone add constraint FK_Association_36 foreign key (idPhoneType)
references Phone_Type (idPhoneType) on delete restrict on update restrict;

alter table Photo add constraint FK_Association_12 foreign key (idUser)
references User (idUser) on delete restrict on update restrict;

alter table Photo add constraint FK_Association_26 foreign key (idProduct)
references Product (idProduct) on delete restrict on update restrict;

alter table Product add constraint FK_Association_16 foreign key (idCategory)
references Category (idCategory) on delete restrict on update restrict;

alter table Product add constraint FK_Association_35 foreign key (idUser, idMerchant)
references Merchant (idUser, idMerchant) on delete restrict on update restrict;

alter table Product add constraint FK_Association_40 foreign key (idTva)
references Tva (idTva) on delete restrict on update restrict;

alter table Product_Item add constraint FK_Generalisation_3 foreign key (idProduct)
references Product (idProduct) on delete restrict on update restrict;

alter table Sale add constraint FK_Association_30 foreign key (idUser, idMerchant)
references Merchant (idUser, idMerchant) on delete restrict on update restrict;

alter table Sale add constraint FK_Association_44 foreign key (idProduct, idProductItem)
references Product_Item (idProduct, idProductItem) on delete restrict on update
restrict;

alter table Schedule add constraint FK_Association_23 foreign key (idUser, idMerchant)
references Merchant (idUser, idMerchant) on delete restrict on update restrict;

alter table Schedule add constraint FK_Association_34 foreign key (idDay)
references Day (idDay) on delete restrict on update restrict;

alter table Time_Slot add constraint FK_Association_45 foreign key (idSchedule)
references Schedule (idSchedule) on delete restrict on update restrict;

alter table Trader_Order add constraint FK_Association_31 foreign key (idSale)
references Sale (idSale) on delete restrict on update restrict;

alter table Trader_Order add constraint FK_Association_37 foreign key (idUser)
references User (idUser) on delete restrict on update restrict;

```

```
alter table Trader_Order add constraint FK_Generalisation_4 foreign key (idPurchaseOrder)
references Purchase_Order (idPurchaseOrder) on delete restrict on update restrict;

alter table User add constraint FK_Association_42 foreign key (idUser)
references User (idUser) on delete restrict on update restrict;

alter table User_FideCoin add constraint FK_Association_4 foreign key (idUser)
references User (idUser) on delete restrict on update restrict;

alter table User_Participant_List add constraint FK_UserParticipantList2 foreign key
(idSale)
references Sale (idSale) on delete restrict on update restrict;

alter table User_Participant_List add constraint FK_UserParticipantList1 foreign key
(idUser)
references User (idUser) on delete restrict on update restrict;
```

3. BIBLIOGRAPHIE ET WEBOGRAPHIE

D. Bibliographie

- ✓ Pascal Roques, « UML 2, Modéliser une application web », Eyrolles, 2008.
- ✓ Christian SOUTOU et Frédéric BROUARD, "UML 2 pour les Bases de Données", Eyrolles, 2012.

E. Webographie

- ✓ Agile : <https://www.lafabriquedunet.fr>
- ✓ Ionic2 : <https://ionicframework.com/docs/>
- ✓ Git : <https://git-scm.com/>
- ✓ HTML : <http://www.w3.org/>
- ✓ PHP : <http://www.php.net/>
- ✓ SQL : <http://sqlpro.developpez.com/>
- ✓ MySQL : <http://www.mysql.fr/>
- ✓ JavaScript : <https://developer.mozilla.org/fr/docs/Web/JavaScript>
- ✓ Angular2 : <https://angular.io/docs/ts/latest/> - <http://blog.xebia.fr>
- ✓ Sass CSS : <http://sass-lang.com/guide>

4. GLOSSAIRE / LEXIQUE

Mots clés	Description
UML	Unified Modeling Language
ORM	Object Relational Mapping
API	Application Programming Interface (Bibliothèques)
IDE	Integrated Development Environnement
HTML	Hyper Text Markup Language (Langage de balisage hypertexte)
CSS	Cascading Style Sheet (Feuilles de style en cascade)
SCSS	Syntactically Awesome Stylesheets
HTTP5	Hyper Text Transfert Protocol: Protocole de communication entre 2 logiciels (un client et un serveur)
JS	JavaScript
AngularJS2	Extension du langage HTML par de nouvelles balises (tags) , par opposition à l'utilisation systématique de l'élément div et à la définition des éléments de présentation en JavaScript.
AJAX	Asynchronous JavaScript and Xml⁷
URL	Uniform Resource Locator
CRUD	Concerne le LMD (Langage de manipulation de données). C : create - R : read - U : update - D : delete
IHM	Interface Home Machine
DCL	Diagramme de classe
MPD	Modèle physique de données
SGBDR	Système de Gestion de Bases de Données Relationnelles
SQL	Structured Query Language (Langage de Requête Structurée)
LDD	Data Definition Language (Langage de Définition de Données)
LMD	Data Manipulation Language (Langage de manipulation de données)
JDBC	Java DataBase Connectivity
JDK	Java Development Kit
JRE	Java Runtime Environment
DAO	Data Access Object
IDE	Integrated Development Environment (environment de développement)
MVC	Modèle – Vue - Controleur
AGL	Atelier de génie logiciel

5. TABLE DES ILLUSTRATIONS

Figure 1 : Schéma de répartition des Domaines.....	8
Figure 2 : Différences entre Développement Natif, Hybride et Web.....	12
Figure 3 : Logo AngularJS 2	15
Figure 4 : Logo TypeScript.....	16
Figure 5 : Logo Jasmine et Karma	16
Figure 6 : Tableau des fonctionnalités.....	19
Figure 7 : Diagramme de GANTT 1/2.....	21
Figure 8 : Diagramme de Gantt 2/2.....	22
Figure 9 : Diagramme de Cas d'Utilisation - utilisateur Mobile.....	24
Figure 10 : Fiches de Description Textuelle des Cas d'Utilisation.....	25
Figure 11 : Maquettes.....	29
Figure 12 : Diagramme de Navigation.....	34
Figure 13 : Diagramme de Séquence Système Authentification.....	35
Figure 14 : Diagramme de Séquence Système - Afficher restaurateurs.....	36
Figure 15 : Diagramme d'Activité et Authentification et Inscription	37
Figure 16 : Les 3 démarches de création du DCL.....	39
Figure 17 : démarche ascendante.....	41
Figure 18 : Dictionnaire de données.....	42
Figure 19 : Graphique des Dépendances Fonctionnelles FideCoin.....	44
Figure 20 : Diagramme de classes User.....	45
Figure 21 : Diagramme de classes Merchant	46
Figure 22 : Diagramme de classes Trader.....	47
Figure 23 : Modèle Physique de Données User	49
Figure 24 : Modèle Physique de Données Merchant	50
Figure 25 : Modèle Physique de Données Trader.....	51
Figure 26 : Code SQL (LLD) de creation de la BD	52
Figure 27 : Diagramme de déploiement.....	65
Figure 28 : Tableau de Correspondances Projet/REAC	68