Table des matières

Table des figures

Introduction

ETANT donné la place prise par le e-commerce dans le monde 2.0 avec des ventes de plus de 1.4 Trillions USD pour 2014 et le nombre de sociétés qui se convertissent dans le e-commerce et dans la vente en ligne, le commerce en ligne est devenu une partie importante de la vie occidentale et une manière de faire de très bonnes affaires.

Dans ce cadre, plusieurs types de commerces sont apparus depuis deux ans :

Le m-commerce ou mobile commerce enrichissant et proposant des services de commerce sur les smart phones permettant aux utilisateurs de consulter les offres facilement, de recevoir des promotions live en fonction de leurs positions et de faciliter le payement en ligne en utilisant le téléphone.

Les promotions en ligne ou les coupons permettant de faire diminuer les prix pour vendre plus et pour attirer le consommateur.

Les sites d'achat groupé qui consiste à valider une offre souvent avec une réduction de -50.

Les plates-formes de social shopping permettant de vendre des produits en intégrant la notion sociale, de partage et de découverte des offres.

Le but de ses plateformes est d'attirer encore plus d'utilisateurs pour trouver la meilleur manière d'augmenter les ventes, de fidéliser et de faire découvrir un nouveau service. Sauf que dans un monde de plus en plus connecté où le client est plus que jamais roi, les consommateurs s'attendent à être traiter de manière personnelle, à découvrir une offre unique et à faire de très bonnes offres.

D'un autre côté, les sociétés cherchent à augmenter leurs ventes et la loyauté de leurs consommateurs en gardant leurs marges et sans diminution des prix. Pour cela, les solutions existantes présentent chacune des lacunes qui ne satisfassent pas la demande des sociétés, certains comme les coupons et les sites de ventes groupées ont montré qu'après 8 ans d'existence que les sociétés perdent de l'argent dans des offres sans un retour sur investissement et une loyauté à la marque quasi nulle et une qualité de service en détérioration. Il est donc impératif de travailler sur une nouvelle façon d'attirer les consommateurs se basant sur les préférences des consommateurs pour lui proposer une offre personnalisée au lieu de proposer des discounts qui font perdre de l'argent, diminuer les charges et les prix et ne fidélisent pas le client.

1 Présentation de l'entreprise d'accueil

1.1 Présentation de l'organisme d'accueil

La société Trick Consulting est une start-up dont le siège social est sis au 16, Rue Habib Thamer 1000 Tunis-Tunisie, est une société de consulting et d'ingénierie informatique en analyse avancée des bases des données et des Big Data travaillant avec les leaders de la télécommunication, du e-commerce et des services financiers en Europe et en Afrique.

1.2 Les services de Trick Consulting

Consulting et solution Big Data : Le BigData et Advanced Analytics est devenue une branche à part entière en informatique permettant une analyse d'une très grande quantité de données pour extraire des informations importantes a□ n de comprendre le comportement des clients pour augmenter les ventes et la fidélisation des acheteurs, proposer de nouveaux services et améliorer les offres existantes.

2 Objectifs visés (cahier des charges)

2.1 Description du travail demandé

Dans le cadre de son travail, l'entreprise est amenée à réaliser une plateforme de commerce personnalisée, c'est dans ce contexte que s'inclus notre projet intitulé « Développement du back end user pour une plateforme » qui consiste à la conception et la réalisation du module destiné à l'utilisateur (le cyberconsommateur) de cette plateforme avec un couplage faible basé sur la spécification REST (Representational state transfer)[1].

2.2 Spécification des besoins

Dans ce paragraphe nous allons citer les différents besoins fonctionnels et non fonctionnels que le système doit assurer.

2.2.1 Les besoins fonctionnels

Notre Solution doit permettre un ensemble de fonctionnalités répondant aux besoins de notre acteur unique pour ce module de l'application « le cyberconsommateur ».

[1] REST (REpresentational State Transfer) est un style d'architecture pour les systèmes hypermédia distribués. Ce style d'architecture est particulièrement bien adapté au World Wide Web mais n'en est pas dépendant

Une Timeline de suivis des activités de ses amis et de sa Community : L'utilisateur en se connectant va voir dans sa timeline :
Les offres dont le statut dans « Offer-Status-table » est Public, qui sont dans la même localisation que l'utilisateur et qui appartiennent aux même Trend suivis par l'utilisateur ou publié par la « Community » de l'utilisateur.

Importer les Like et les Commentaires sur avec les offres pour les montrer dans la Timeline
Un système de pagination pour importer les données par tranche. L'utilisateur peut alors cliquer sur Like, commenter ou adhérer à cette offre et AngularJS envoi les informations en temps réel vers la base de données pour mettre à jour les tables.

Les autres utilisateurs qu'il ne les suive pas (suiveurs ou commentateurs sur des publications dans son timeline) avec le lien d'accès à quelques informations de son profil ainsi que les options « follow » et « signaler ».

La gestion du profil de l'utilisateur et des trends suivis : L'utilisateur peut mettre à jour les informations sur son profil, chercher des amis grâce à leurs noms ou gérer les trends qu'il est en train de suivre.

Gestion des offres Privés : Le client a accès aux offres qui lui sont destinées. L'utilisateur peut alors rejeter l'offre qui ne lui correspond pas ou la publier dans son espace public.

Gestion des offres en cours : Visualisation des offres en cours dont le statut est Public avec le temps restant, le nombre de personne adhérent, le nombre de Like et de commentaire.

Historique des deals : Liste des anciens deals de cette personne avec le statut
« Closed », en cliquant sur un deal alors les informations sont importées pour consultation.

2.2.2 Les besoins non fonctionnels

- Ergonomie :

 La page d'accueil du portail présente une importance primordiale : elle doit donner à l'utilisateur et l'entreprise une vision de ce qu'il peut attendre du plateforme.
 Le module doit présenter une interface claire, simple et facile à utiliser.

- Rapidité :

Le système doit répondre à différentes requêtes dans un temps très réduit.

- Robustesse et sécurité:

Le système doit évoluer avec l'augmentation de nombre des requêtes et garantir en même temps la sécurité d'accès pour tous les clients.

2.3 Modélisation des besoins

La modélisation au niveau de la spécification est basée sur les diagrammes des cas d'utilisation. Ces diagrammes décrivent les utilisations requises d'un système, ou ce qu'un système est supposé faire.

2.3.1 Les diagrammes des cas d'utilisation

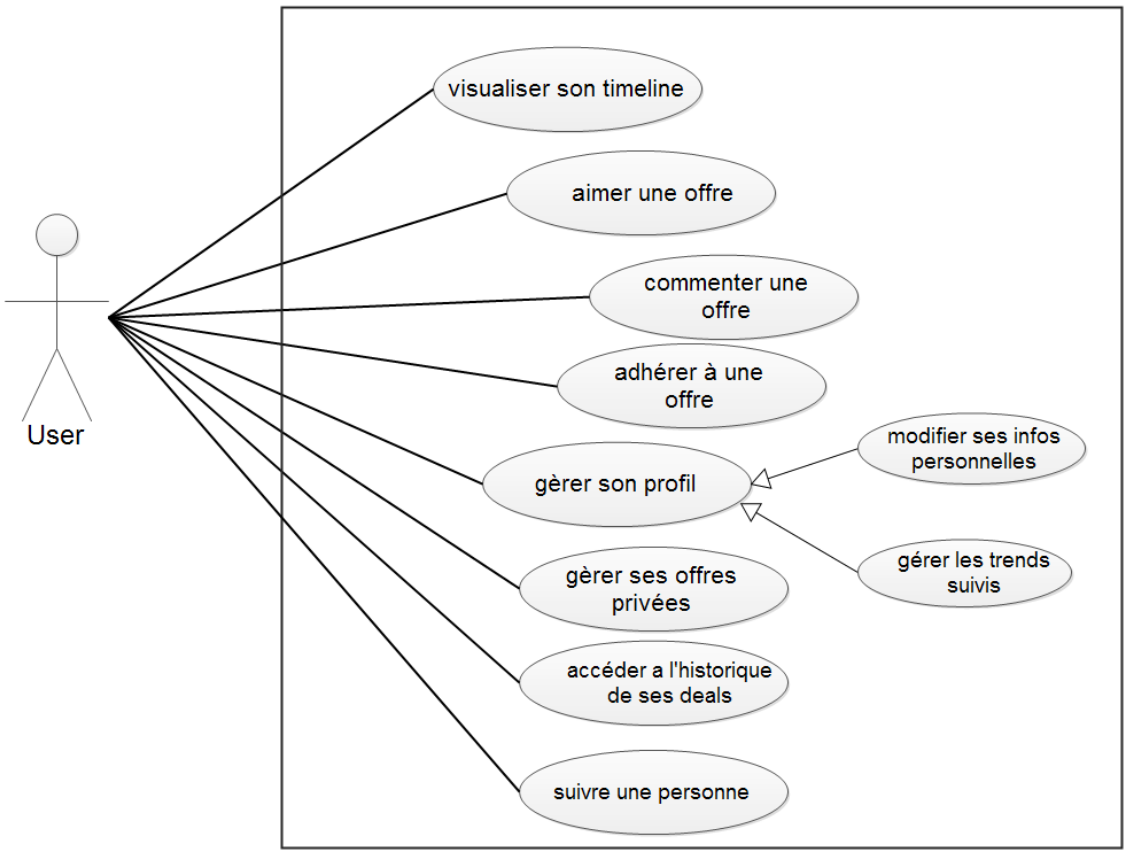

Figure 1 : Diagramme des cas d'utilisation

2.4 La conception globale

Dans cette section nous allons définir les architectures physique et logique que nous avons utilisées pour réaliser notre plateforme.

2.4.1 Architecture physique

L'architecture physique (également nommée architecture technique) décrit l'ensemble des composants matériels supportant l'application. Pour le nôtre, nous avons opté pour l'architecture 3-tiers qui vise à faciliter le déploiement, la maintenance ainsi une amélioration de la sécurité des données.

Ces trois composants sont :

- Un client, c'est-à-dire l'ordinateur demandeur de ressources, équipée d'une interface utilisateur (généralement un navigateur web) chargée de la présentation.
- Le serveur d'application (appelé également middleware). chargé de fournir la ressource mais faisant appel à un autre serveur.
- Le serveur des données, fournissant au serveur d'application les données dont il a besoin.

Le principe de l'architecture est illustré dans la figure suivante :

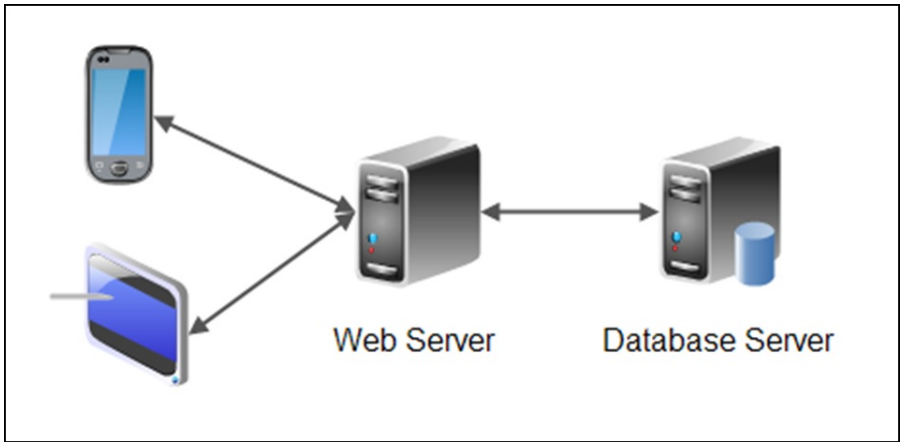

Figure 2 : Architecture physique de l'application

2.4.2 Architecture logique

L'architecture logique est la manière dont les composants logiques d'une solution sont organisés et intégrés. Nous avons opté pour une architecture en couches.

Le principe des couches applicatives repose sur le fait que chaque couche ne traite de manière autonome qu'une partie bien précise du système, un ensemble de fonctionnalités bien définies dans un cadre de responsabilités restreint. Cela contribue à diviser un problème global complexe en une suite de petits problèmes simples et permet une résolution plus facile, plus structurée et plus pérenne de l'ensemble. Ainsi la couche d'accès aux données sera en charge des opérations de

lecture/écriture depuis ou vers des sources de données externes diverses ; la couche de services (ou couche métier) fournira quant à elle la logique métier, etc.

Chacune des couches publie ensuite ses services spécialisés à destination d'une autre couche du système de niveau supérieur. Chaque couche se concentre donc sur ses préoccupations propres (accès aux données, logique métier, présentation, etc.) et fait appel à une ou plusieurs couches de niveau inférieur lorsqu'elle sort de sa sphère de responsabilités.

2.5 Conception détaillée

2.5.1 Schéma de la base de données

Ce diagramme sert à présenter d'une manière formelle les liaisons qui existent entre les différents nœuds de la base de données. La figure 3 montre les différentes nœuds et relations de la base de données graphique.

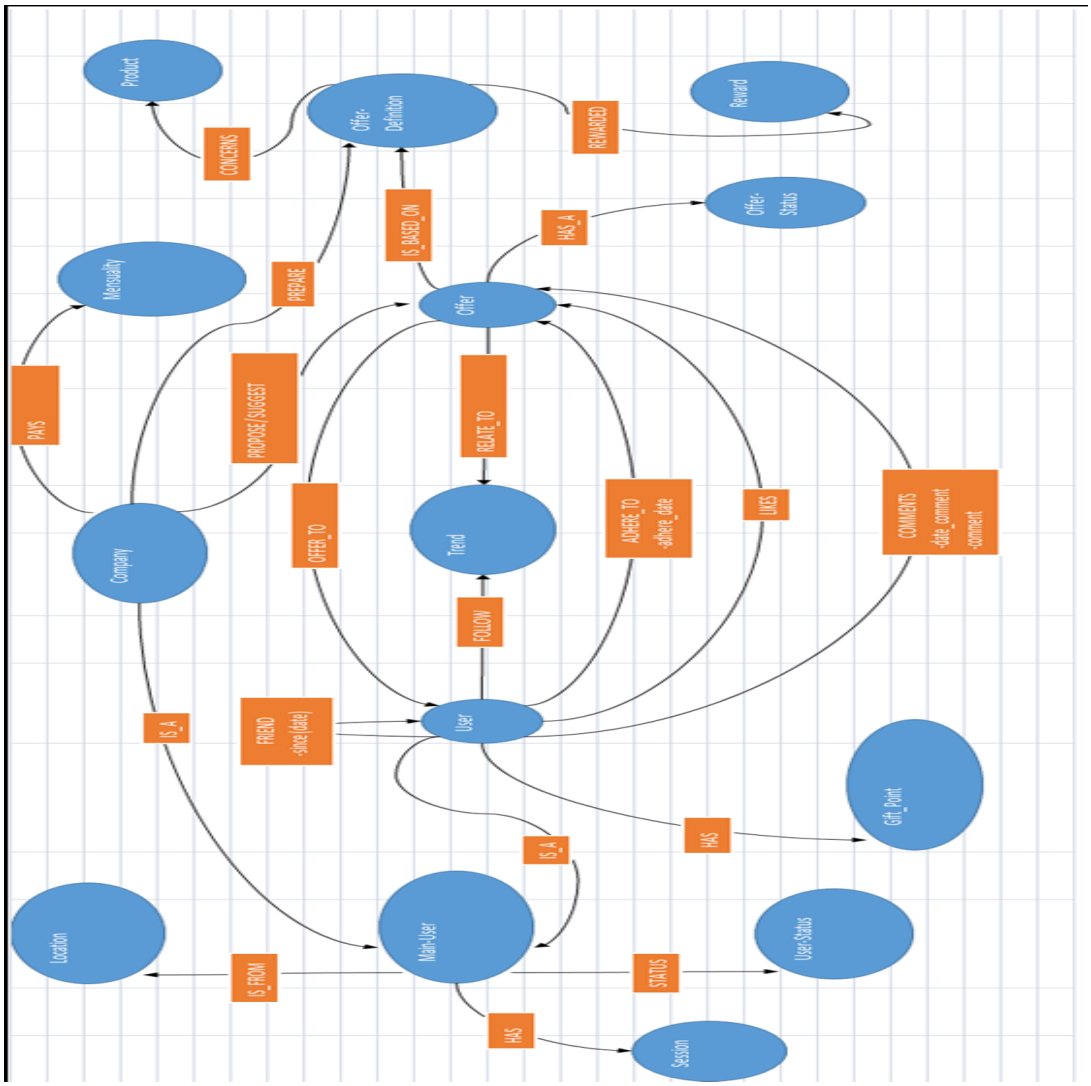

Figure 3 : Schéma de la base de données

2.5.2 Diagrammes de paquetages

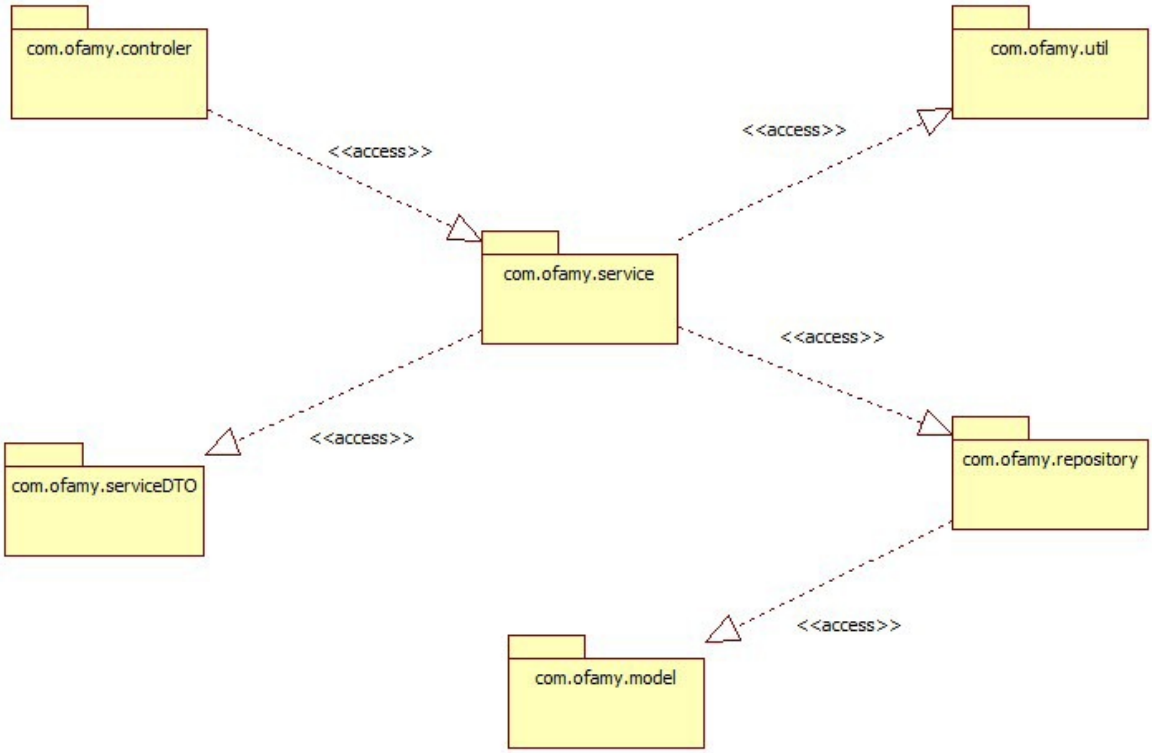

Figure 4 : Diagramme de paquetages

Conclusion

Tout au long de cette partie, nous avons décrit les différents aspects conceptuels de notre travail. Nous avons commencé par présenter l'architecture globale de l'application ensuite nous avons illustré l'architecture détaillée.

3 Journal de stage

Il est nécessaire de tracer un diagramme qui décrit la répartition des tâches du projet tout au long de ces quatre mois de stage, afin de donner une vue globale de la répartition du temps par rapport au travail demandé. La figure suivante représente le chronogramme des tâches.

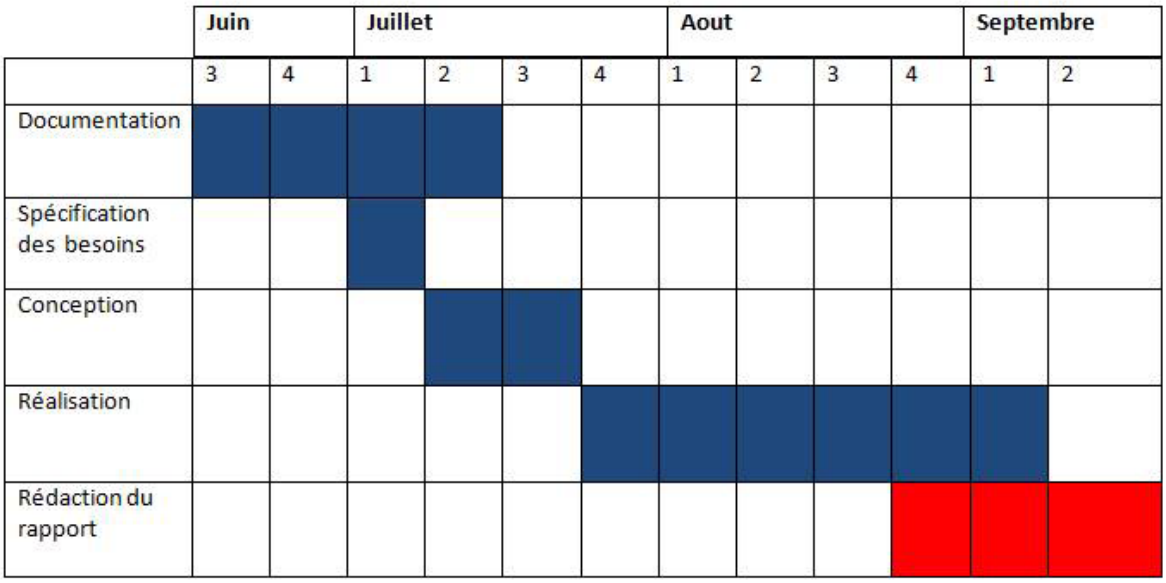

Figure 5 : Chronogramme du travail

4 Travail réalisé

4.1 Choix technologiques

4.1.1 Choix du Java EE

Plusieurs technologies capables de déployer une application web existent sur le marché. On peut citer PHP, .NET et Java EE. Toutes ces technologies offrent sensiblement les mêmes possibilités, mais utilisent un langage et un environnement bien adéquats à chacune. Le choix fixé sur Java EE pour certaines raisons :

- La richesse de la documentation
- La richesse des librairies, Framework et composants que cette technologie permet d'utiliser.

4.1.2 Choix de Framework Spring

Spring est un framework libre, qui permet de construire et définir l'infrastructure d'une application Java, dont il facilite le développement et les tests.

Spring est considéré comme un conteneur dit « léger », il prend en charger la création d'objets et la mise en relation d'objets par l'intermédiaire d'un fichier de configuration qui décrit les objets à fabriquer et les relations de dépendances entre ces objets.

Le gros avantage par rapport aux serveurs d'application est qu'avec, les classes n'ont pas besoin d'implémenter une quelconque interface pour être prises en charge par le framework. C'est en ce sens que Spring est qualifié de conteneur « léger ».

4.1.3 Neo4j

Neo4j est un système de gestion de base de données au code source libre orienté graphes, développé en Java. Neo4j vous permet de représenter les données connectées naturellement, en tant qu'objets reliés par un ensemble de relations, chacun possédant ses propres propriétés. La base de données de graphes, permet au développeur de commencer immédiatement à coder, car les données stockées dans la base font le parallèle direct avec les données elles-mêmes.

Pourquoi nous n'avons pas utilisé un SGBD-Relationnelles ?

Comparé aux bases relationnelles, la base de données de graphe Neo4j peut être jusqu'à plusieurs milliers de fois plus rapide pour traiter les données associatives, tout en simplifiant considérablement les requêtes qui peuvent s'étendre plus facilement à de larges ensembles de données, car elles ne nécessitent pas de recourir aux coûteuses jointures du monde SQL.

4.1.4 Spring MVC

La mise en pratique du patron de conception MVC II (Model View Controller) offre une meilleure structuration du tiers de présentation des applications Java EE en dissociant les préoccupations de déclenchement des traitements de la construction de la présentation proprement dite. Le framework Spring offre une implémentation innovante du patron MVC II par le biais du module nommé Spring MVC, qui profite des avantages de l'injection de dépendances employée par Spring. De plus, à partir de la version 4.0, Spring MVC intègre un support permettant de gérer la technologie REST dont nous avons besoin dans notre projet.

4.1.5 Spring Data Neo4j

Spring Data un projet supplémentaire de Spring créé pour répondre aux besoins d'écrire plus simplement l'accès aux données et d'avoir une couche d'abstraction commune à de multiples sources de données NoSQL[2]. Pour Spring Data Neo4j il est dédié à l'interfaçage avec Neo4j en fournissant des entités, repositories et des templates orientés graphes.

[2] NoSQL (Not only SQL en anglais) désigne une catégorie de systèmes de gestion de base de données (SGBD) qui n'est plus fondée sur l'architecture classique des bases relationnelles. L'unité logique n'y est plus la table, et les données ne sont en général pas manipulées avec SQL.

4.1.6 Spring Security

Spring Security est un framework permettant de sécuriser et de contrôler l'accès aux applications web qu'elles utilisent Spring ou non. Comme tous les frameworks issus de la communauté Spring, il repose sur le principe de la CoC (Convention Over Configuration). Il est donc facile à intégrer tout en offrant des options avancées aux développeurs voulant personnaliser la sécurité de leurs applications.

4.1.7 Spring Social

Spring Social est un module Spring qui va nous permettre de connecter notre application à plusieurs services en ligne, tel que les réseaux sociaux Facebook ou Twitter.

Tous ces choix nous donnent l'architecture suivante de la plateforme :

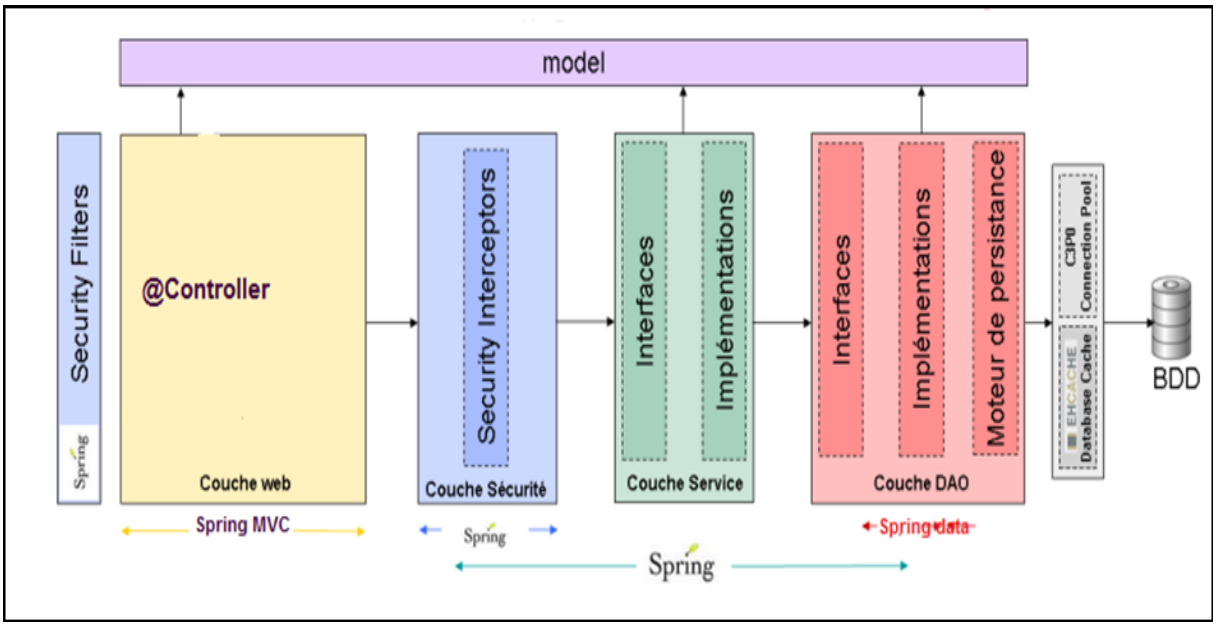

Figure 6 : Architecture de la plateforme

4.2 Réalisation

4.2.1 Timeline

Dans le timeline les offres qui apparaissent ne sont pas détaillées, et les boutons qui mènent aux détails sont clairement au-dessous.

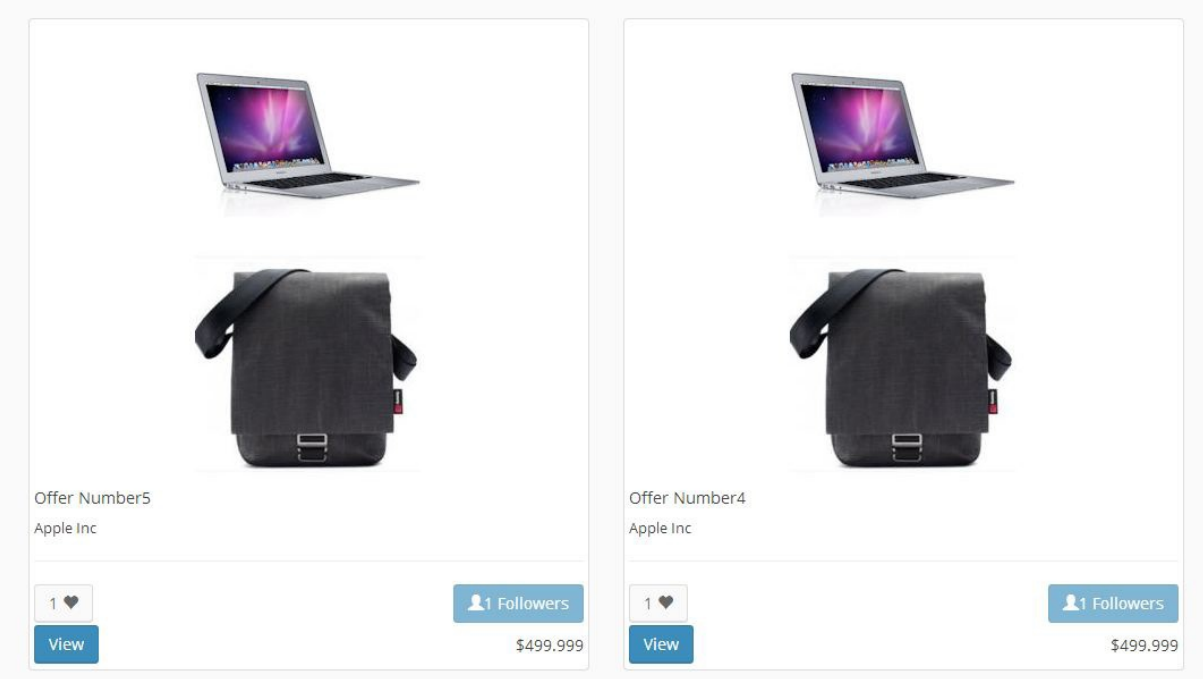

Figure 7 : Deux offres dans le Timeline

4.2.2 Profile

Lorsque l'utilisateur choisit de modifier son profile un formulaire s'affiche pour les données personnelles ainsi qu'une liste pour les trends, préférences location ...

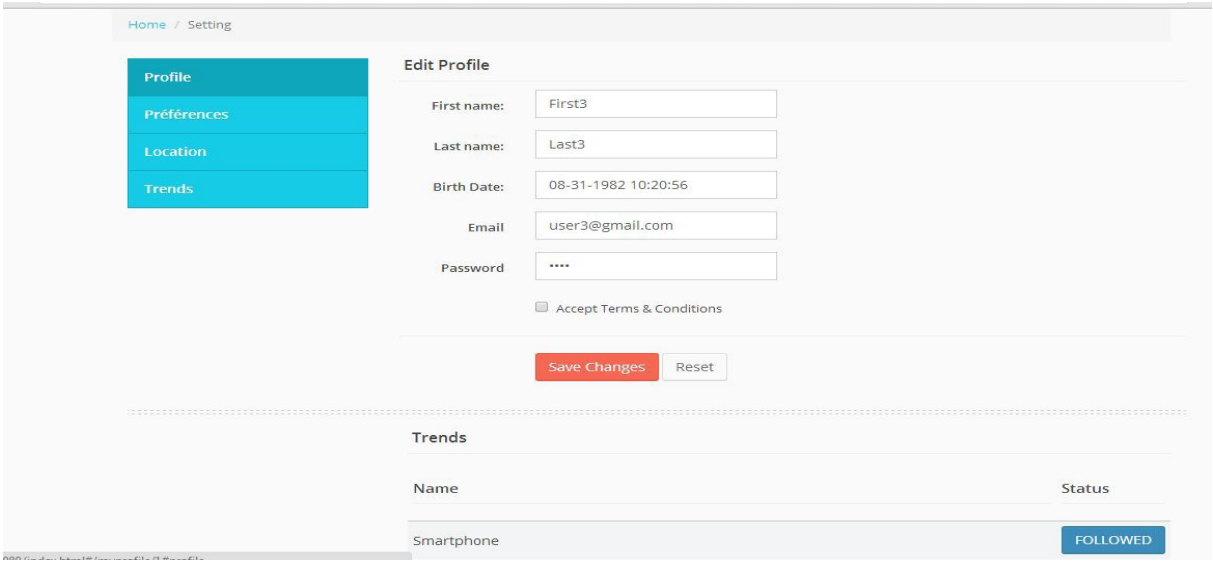

Figure 8 : Éditer profile

Lorsque l'utilisateur se fixe sur une offre choisit sous le trend « smartphone », l'interface montre des données supplémentaires sur l'offre.

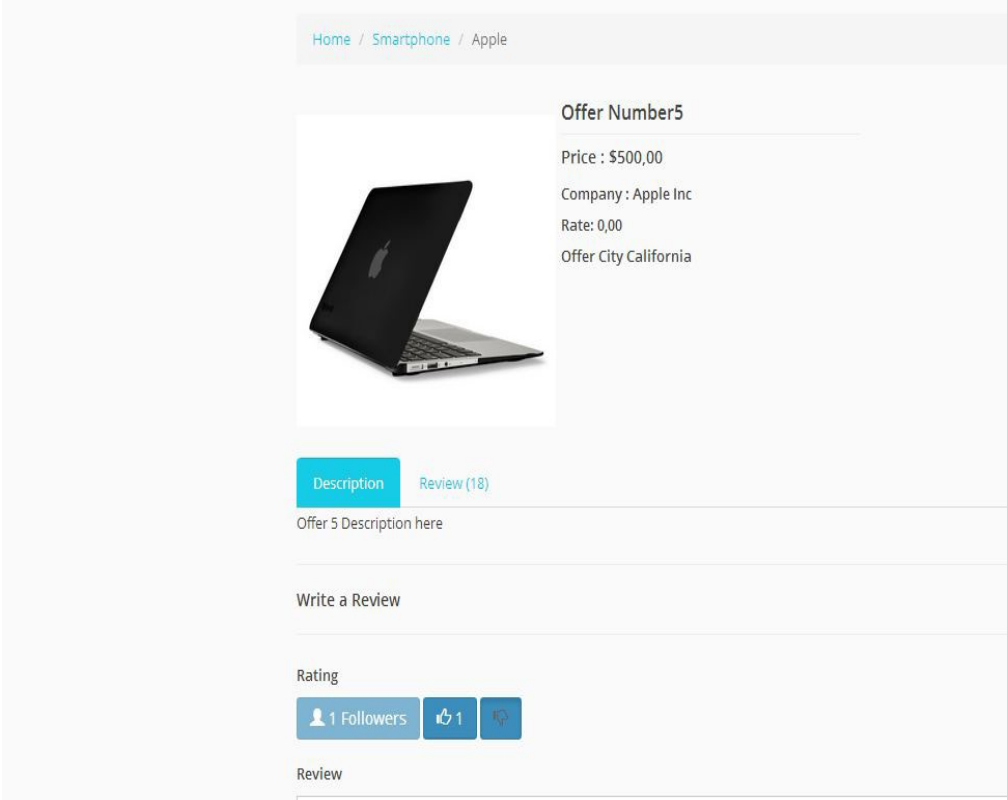

Figure 9 : Une offre détaillée

L'utilisateur choisit de se fixer sur l'un des suiveurs d'une offre consultée, la personne est déjà suivie par l'utilisateur.

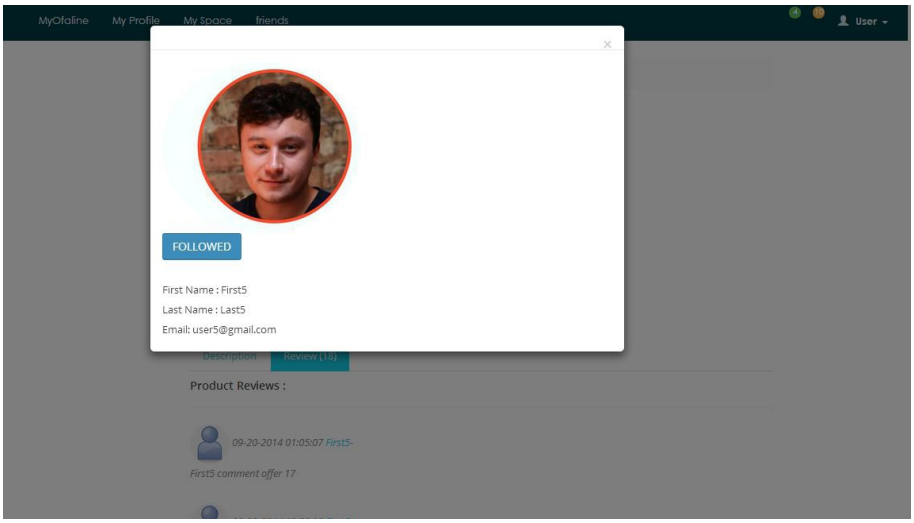

Figure 10 : Un autre utilisateur suiveur de l'offre

Une liste des personnes « followers » qui ne sont pas suivis par le user (indiqué par « unfollowed »)

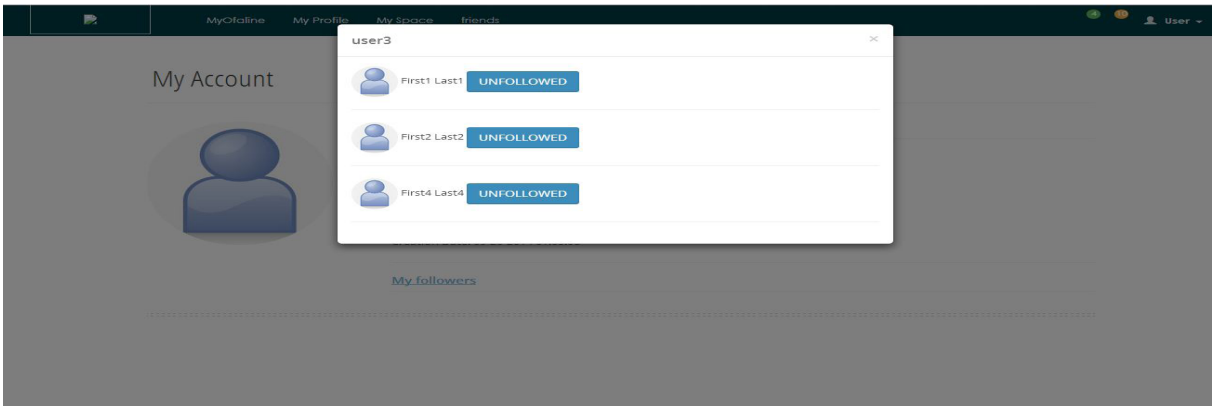

Figure 11 : La liste des personnes qui suivent l'utilisateur

5 Consolidation des acquis

DANS ce rapport nous avons mis en œuvre le bilan complet de notre travail qui se situe dans le cadre du stage d'immersion en entreprise. Notre mission était de développer la couche de la logique métier pour une plateforme de commerce personnalisé.

Ainsi, pour atteindre ces objectifs nous avons commencé par mettre le projet dans son cadre général en présentant l'organisme d'accueil et en détaillant les objectifs de ce projet.

Ensuite, nous avons effectué une étude théorique qui nous a permis de mettre l'accent sur les différentes technologies qui feront l'objet de notre projet. Par la suite, nous avons entamé la partie analyse et spécification des besoins permettant de tracer les objectifs à atteindre à partir des besoins fonctionnels et non fonctionnels. Puis, nous avons passé à l'étude conceptuelle de la future application et cela à travers la conception globale et détaillée de l'application. Enfin, nous avons terminé par la réalisation qui nous a permis de recenser les différents outils matériels et logiciels à utiliser pour développer l'application.

6 Conclusion

L'application réalisée reste un point de départ qui laisse des grandes possibilités d'extension et d'amélioration voir l'aspect intelligent du Framework dans le traitement des données personnelles pour dégager une image pour la personnalité du « cyberconsommateur » afin d'aider les « companies » à envoyer les offres personnalisées et minimiser le plus possible le temps de réponse pour donner aux clients l'impression que les données sont traitées en local.

Bibliographies

[1] M. Hunger, D. Montag, A. Kollegger, Good Relationships The Spring Data Neo4j Guide Book 2.0.0.RELEASE, 2010 – 2011

[2] Mark Pollack,Oliver, Gierke,Thomas Risberg,Jon Brisbin,Michael Hunger, Spring Data Modern Data Accesss for Entreprise Java

Webographies

[1] http ://fr.wikipedia.org/wiki/Big-data **consultée le 30/06/2014**

[2] http ://www.jmdoudoux.fr/java/dej/chap-j2ee-javaee.htm **consultée le 02/07/2014**

[3] http ://projects.spring.io/spring-data-neo4j/ **consultée le 12/07/2014**

[4] http://fr.wikipedia.org/wiki/NoSQL **consultée le 06/07/2014**

[5] http ://projects.spring.io/spring-framework/ **consultée le 03/07/2014**

[6] http ://www.neotechnology.com/neo4j-graph-database/ **consultée le 04/09/2014**

[7] http ://www.neo4j.org/ **consultée le 05/07/2014**

[8] http ://docs.neo4j.org/chunked/stable/index.html consultée le **15/07/2014**