

Institut G4

Master 1, chef de projet en système d'information
Spécialité : informatique et management de projet.

Rapport d'alternance 2016-2017

Développement d'un système de paiement mobile Au sein d'INFINITY SPACE S.A.S

Mouhamadou SAMAKE
Samake.mouhamadou@hotmail.com

Tuteur : Cédric Atangana

PDG d'Infinity Space au 45 Rue Frédéric
Joliot Curie, 13013 Marseille

Remerciements

Mes remerciements vont à tous les corps administratifs de **l'Institut G4**, à savoir **VALET Lila** et **GIRAUDO Virgile** ainsi qu'à tous les enseignants pour leur soutien moral et conseils pour l'élaboration de ce travail.

Je remercie en particulier mon encadreur professionnel **Mr Cédric ATANGANA, PDG d'INFINITY SPACE S.A.S**, qui a été pour moi un guide mais aussi une source de motivation tout au long cette première année d'alternance. Je lui exprime ma profonde reconnaissance pour ses conseils et ses orientations pour la réalisation de ce travail.

Je remercie **Mme Annicelle Reine KUNGNE, Directrice Financière et co-fondatrice d'INFINITY SPACE S.A.S**, pour son soutien perpétuel et son excellent management envers l'ensemble de l'équipe de développement.

Je tiens, aussi à remercier infiniment, mes collègues en entreprise, pour la dimension très humaine qu'ils ont apporté à ce travail et leur précieuse contribution à l'accomplissement des tâches qui m'ont été confiées.

Enfin je remercie toutes les personnes ayant contribué de près ou de loin pour la réussite de ce travail.

Synthèse :

Au cours de mon année 2016-2017 au sein de l'**institut G4**, j'ai pu intégrer l'entreprise **INFINITY SPACE** afin d'acquérir ma première véritable expérience professionnelle au côté de mon tuteur : Cédric Atangana. Ceci qui m'a permis d'évoluer professionnellement, d'évoluer en compétences dans le domaine du développement web en général et enfin d'apporter une plus-value à mon entreprise.

Durant cette première année d'alternance j'ai participé au développement du projet phare de l'entreprise, de la conception à la réalisation, ce projet étant vigoureux et si complexe, demandait beaucoup d'implication et d'autonomie, car l'entreprise ayant le statut startup, et le nombre d'effectif étant minimum.

Ce projet (WeCashUp), est la mise en place d'une API (Applicative programming Interface) de paiement mobile, aidant toutes personnes non-bancarisées (qui n'ont pas une carte bancaire) de leurs permettre de faire des achats sur internet, sans carte bancaire qu'avec leur téléphone mobile.

Dans cette optique, j'ai été amené à faire dans un premier temps une étude approfondie du marché, c'est-à-dire, le public visé, les principaux clients et les différents opérateurs télécom qui font mobile money, de chaque pays concerné.

Dans la seconde partie, je me suis basé sur l'analyse des données, que peut transmettre ce public, la gestion des flux de données, et l'élaboration du cahier de charge, afin de bien mener ce projet.

Pour réaliser tout cela j'ai dû effectuer des planifications de mes différentes tâches pour un meilleur suivi de mon avancement et une meilleure estimation de la durée de mes tâches au fur et mesure de l'avancement du projet dans le temps.

Cette année a été très formatrice pour moi en m'apportant des compétences techniques sur diverses technologies, qui me permettront de pouvoir mieux découper et estimer la durée des différentes tâches dans mes futurs projets qui me semble être l'une des clés du succès pour la réussite d'un projet.

Abstract :

During my study year 2016-2017 at the **G4 Institute**, I joined a FinTech company called **INFINITY SPACE** to acquire my first professional experience at the side of my professional tutor: Cédric ATANGANA. This allowed me to grow professionally, develop skills in web development in general, and finally to bring value added to my company.

During this first year of study I participated to the development of WeCashUp, the lighthouse project of the company, from the design to the production, this project has been vigorous and complex from the beginning, required a lot of involvement and autonomy because the company is still young and small.

This project (WeCashUp), is the implementation of an API (application programming Interface) for mobile payment, helping all the unbanked people (no credit card) around the globe to make payments on the internet without credit card with their mobile phones.

I was led to do drive a comprehensive market study concerning the project's primary audience, the main customers and the various telecom operators that build mobile money in each targeted country (in Africa specifically).

The second part of my work was about the analysis of the data produced by the end users, management of the data flow and the setup specifications necessary to carry out this project.

To do this I had to perform tasks planning to track my progress and to have a better estimate of the duration of my tasks.

This year was full of rich experiences for me, I acquired new technical skills on various technologies, I learnt how to organize my work and estimate the duration of the different tasks in my future projects which seems to be one of the key elements for a successful project.

Table des matières

Introduction :	5
1- Présentation de l'entreprise :	6
2- Travaux réalisés :	10
2.1. Méthode de travail:	11
a) Outils utilisés :	11
b) Réalisation :	17
2.1.1. Inscription :	18
2.1.2. Rédaction de la documentation de l'API de WeCashUp :	20
a) Qu'est-ce qu'une API ?	20
b) L'API WeCashUp :	20
2.1.3. La documentation :	21
a) Pourquoi ?	21
b) Que doit-elle contenir ?	21
2.1.4. Mise en place de la documentation de l'API de WeCashUp :	23
a) Ruby :	24
b) Slate :	24
c) Installation de Ruby et exécution de Slate (via une invite de commande) :	26
2.1.5. Déploiement sur Google App Engine :	26
2.1.5.1. Qu'est-ce que c'est Google App Engine:	26
2.1.5.2. Datastore:	27
2.1.5.3. Objectify:	28
2.1.5.4. Déploiement:	28
2.1.6. Formation des nouvelles personnes(stagiaires) :	31
Conclusion :	32
Liste des Compétences professionnelles :	33

Introduction :

Dans le cadre de ma formation Master 1 à l'**institut G4**, spécialité Informatique et Management de projet informatique, j'ai signé un contrat d'alternance de deux ans comme développeur web au sein de l'entreprise **INFINITY SPACE**, afin de pouvoir obtenir mon diplôme du titre RNCP de niveau I, intitulé chef de projet en système d'information.

Dans le cursus de ma première année d'alternance 2016-2017 au sein de l'**INFINITY SPACE**, j'ai été amené à participer au développement d'un système de paiement mobile (WeCashUp). Ce projet qui me permettra non seulement de mettre en pratique les connaissances acquises durant la formation mais aussi de me familiariser avec le monde de l'entreprise afin de pouvoir acquérir les compétences techniques et managériales.

Mon choix pour cette entreprise était basé sur plusieurs facteurs : son domaine de compétence, étant l'informatique me permettrait de mieux cerner mon projet professionnel et ainsi de mesurer mon affinité par rapport à ce domaine, mais aussi, son statut de start-up, j'y ai trouvé une opportunité afin de découvrir le monde entrepreneurial notamment celui de l'innovation.

Au sein de l'entreprise j'ai eu à m'occuper dans un premier temps de l'inscription de nos différents clients (les e-commerçants, les opérateurs télécoms ...) en même temps le traitement des transactions, le calcul des commissions sur des différentes transactions, et aussi la mise en place de la documentation de l'API de WeCashUp pour aider aux e-commerçants d'intégrer la solution WeCashUp sur leur site.

Pour ce faire, j'ai dû cartographier tous les opérateurs de mobile money en Afrique et analyser leur mode de fonctionnement. Ce travail a ainsi permis d'étoffer l'étude de marché de l'entreprise et son positionnement.

Ainsi, dans une première partie je vais présenter l'entreprise : sa genèse, son produit, ses dirigeants. Ensuite, j'évoquerai le déroulement de l'alternance avec les différentes activités réalisées mais aussi les enseignements et les difficultés rencontrées. Enfin, l'alternance m'ayant permis de mieux connaître l'entrepreneuriat m'a amené à m'intéresser à ce domaine et fera donc l'objet de la dernière partie qui traitera de l'entrepreneuriat en France et d'une comparaison par rapport à l'Afrique.

1- Présentation de l'entreprise :

L'entreprise INFINITY SPACE est une société par action simplifiée (SAS) au capital de 1.000€ qui a été fondée le 23 Mars 2015 par 7 jeunes diplômés issus de formations complémentaires : Ingénierie, administration, finance et Marketing.

De nos jour INFINITY SPACE est composée des jeunes expérimentés de 16 personnes venant de 12 pays sur 4 continents, avec différents horizons tels que l'ingénierie, l'informatique, les finances, l'administration, la gestion, le marketing, Les médias numériques, Juridique, Recherche et développement, Développement des affaires, etc. 6 personnes en interne et les autres en externe qui travaillent en collaboration avec l'équipe de développement, parmi ces 6 personnes deux des co-fondateurs, deux salariés en temps plein et un alternant y travaillent : son CEO **M. Cédric ATANGANA** et sa CFO (Chief Financial Officer), **Mlle Annicelle Reine KUNGNE**, + 3 stagiaires.

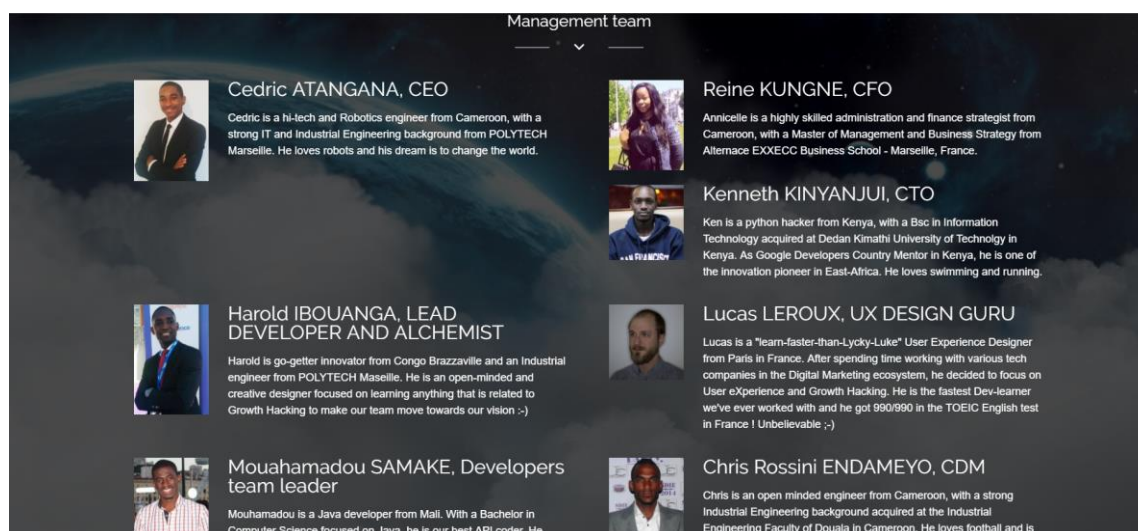


Figure N°1 : team.

Source : <http://www.infinityspace.fr/>.

INFINITY SPACE est une FinTech d'innovation sociale né à Douala (Cameroun), développé à Nairobi (Kenya) et basé à Marseille (France).

INFINITY SPACE est une FinTech startup innovante basée à Marseille, plus précisément dans la pépinière de Marseille innovation.

Marseille Innovation est un important centre d'entreprises et d'innovation (CEEI) de la région PACA qui opère 3 pépinières et hôtels d'entreprises, 1 incubateur international sur la Méditerranée et 1 programme d'accélération de PME. Plus de 160 startups sont accompagnées au quotidien.

Créé en 1996⁽¹⁾ par Christian Rey, Marseille Innovation est le plus grand centre européen d'entreprises et d'innovation (CEEI) de la région PACA : **3 pépinières et hôtels d'entreprises pour héberger et accompagner des start-up** en phase de démarrage dans les

domaines du numérique, du multimédia, de l'audiovisuel, des objets connectés et plus généralement des sciences de l'ingénieur. Depuis toujours Marseille innovation cultive une double ambition : **être à la fois accélérateur de croissance et attracteur de talents**. Ce sont plus de **100 start-up** qui nous font confiance aujourd'hui et plus de 500 depuis sa création.



Figure N°2 : Hôtel technologique de château gombert.

Source : google maps.

Une pépinière dédiée aux start-up du numérique et des sciences de l'ingénieur.

Sur le Technopôle Marseille Provence à Château-Gombert au cœur des écoles d'ingénieurs et des laboratoires de recherche, la pépinière héberge et accompagne des start-up issues des secteurs du numérique et des sciences de l'ingénieur. Elle offre un cadre privilégié, alliance réussie de l'innovation technologique et de la qualité de vie, un véritable lieu d'échange entre l'entreprise, la recherche, les écoles dans un cadre de vie fortement doté en services.

INFINITY SPACE développe www.wecashup.com, un système de paiement mobile universel qui permet aux personnes non-bancarisés dans le monde de payer en ligne avec leur téléphone portable comme moyen de paiement sans carte bancaire.

En effet, son marché couvre près de 2 milliards de personnes non-bancarisés dans le monde, localisés en Amérique latine (300 millions), Asie du Sud (500 millions), Europe de l'Est (200 millions) et en Afrique (800 millions) et qui ne peuvent pas faire d'achats sur internet parce qu'ils n'ont pas ni comptes bancaires classiques, ni de cartes bancaires.

Son focus stratégique pour le moment se tourne donc vers l'Afrique, son plus gros marché et continent présentant le plus gros challenge.

INFINITY SPACE s'est alors fixé pour mission principale de challenger l'impossible, faire des impossibles des possibles.

La startup INFINITY SPACE construit à cet effet un écosystème e-business pour aider les entreprises à accroître leurs activités à l'international et pour permettre aux personnes exclues du système bancaire traditionnel de faire des achats en ligne avec leur téléphone portable comme moyen de paiement sans carte bancaire ni compte bancaire.

Pour permettre à toutes ces personnes de faire les paiements en ligne par leurs téléphone mobile, la jeune startup travaille avec les opérateurs télécom qui font mobile money.

Mobile money : Le Mobile Money est un service de portefeuille mobile, disponible dans de nombreux pays, qui permet à ses utilisateurs d'épargner, envoyer et recevoir de l'argent sur leur téléphone portable.

Le transfert **d'argent par téléphone mobile** est né au Kenya qui l'ont nommé **M-PESA**.

M : comme mobile et **PESA** : comme l'argent en swahilie.

Lancé en 2007 au Kenya par l'opérateur SAFARICOM (Filiale du groupe VODAFONE), dont le succès considérable a eu un retentissement mondial. Aujourd'hui, presque tout le monde utilise M-PESA au Kenya et notamment les personnes **sans compte bancaire**. On estime qu'environ 80 % des paiements par téléphone mobile réalisés dans le monde sont faits en Afrique de l'Est ! (Kenya, Tanzanie, Lesotho, Mozambique notamment).

Ceci dit, plus de 11 services de **Mobile Money** existent à ce jour en Afrique : Barithi Airtel, Etisalat, Millicom, MTN, Ooredoo, Orange, STC, Vodafone, Tigo-cash, Glo-Money etc. (2)

Les paiements effectués avec le Mobile Money sont sécurisés et simples et c'est donc une alternative populaire aux comptes bancaires. Le Mobile Money est disponible sur les Smartphones et aussi sur les téléphones numériques.

La jeune startup a interconnecté tous ces opérateurs télécom pour en faire une API (Application programming Interface), afin de faciliter le paiement à toutes ces personnes concernées, n'importe où, où ils se retrouvent chez n'importe quel opérateur, ils peuvent de même faire un split-paiement (de payer une somme par groupe).

Dans le cas de l'Afrique, les problèmes les plus importants auxquels ces PME sont confrontées dans la vente en ligne sont :

1. - l'infrastructure d'adresse physique (système d'adressage de domicile) est quasi-inexistant et les livraisons pour le e-commerce sont difficiles à réaliser
2. - plus de 80% de la population africaine (800 millions de personnes) est très faiblement bancarisée et ne peut pas acheter en ligne.

Donc, pour trouver une solution rapide à ces deux questions, INFINITY SPACE a regroupé les technologies adéquates pour aider les gens à acheter facilement et se faire livrer à temps. Cela va considérablement aider les PME Française et internationales à se concentrer sur le cœur de leur métier sans se soucier de la façon d'atteindre plus de clients, comment ils seront payés et comment ils vont livrer des marchandises à des clients qui sont très loin du lieu de leurs magasins physiques.

INFINITY SPACE propose deux activités principales :

1. WeShopUp : Une Marketplace d'achat participatif où les gens achètent en groupe et se font livrer individuellement (une plateforme de crowdpurchasing c'est-à-dire crowdfunding + e-commerce), développé au début de la création de l'entreprise, stoppé par les problèmes dont je vous ai expliqué au page précédente, plus 55% des utilisateurs de cette Platform se trouvaient en Afrique d'où le problème de le problème de la livraison intervient, ce qui a pousser la jeune startup de mettre en place une API pour résoudre ce dernier .

2. WeCashUp : Une plateforme de paiement mobile universelle qui permet aux populations non bancarisées de payer en ligne par Mobile Money, avec leur téléphone portable, et sans carte de crédit. La plateforme de paiement mobile est dédiée principalement aux pays sous ou non bancarisés. En effet, il existe beaucoup de système de paiement mobile suivant différents opérateurs téléphoniques. Souvent, dans un pays on peut avoir par exemple quatre opérateurs téléphoniques qui font du paiement mobile (exemple : Airtel Money, pour l'opérateur téléphonique Airtel / OrangeMoney pour l'opérateur téléphonique Orange / MobileMoney pour l'opérateur MTN / MoovMoney pour l'opérateur téléphonique Moov...) mais ils ne communiquent pas entre eux. Ainsi, une personne utilisant ce système doit posséder à elle seule les quatre cartes Sim de ces opérateurs. Ce qui n'est pas pratique d'autant plus que dans certains pays, on peut quelque fois atteindre douze opérateurs. Ainsi, WeCashUp est la solution qui offre de l'interopérabilité à ces systèmes. De ce fait, en détenant une seule carte Sim d'un opérateur quelconque, une personne peut transférer et recevoir de l'argent de tous les autres opérateurs et même faire des achats sur n'importe quel site d'un marchand qui a embarqué la solution WeCashUp.

2- Travaux réalisés :

A mon arrivé, l'INFINITY SPACE venait juste de lancer son projet phare, WeCashUp, la plateforme de paiement mobile, pour donner une brève définition de WeCashUp de façon technique, WeCashUp est comme le contrôleur, son architecture ressemble à l'architecture MVC autrement dit l'intermédiaire entre le client et le provider, je vous montrerai un peu plus bas un schéma pour plus de détail. La technologie WeCashUp est conçu d'une façon à détecter le blanchissement d'argent ou encore détection de fraude, WCU est la combinaison de plusieurs technologie différente ce qui fait sa particularité, voici un schéma de WeCashUp dans son ensemble :

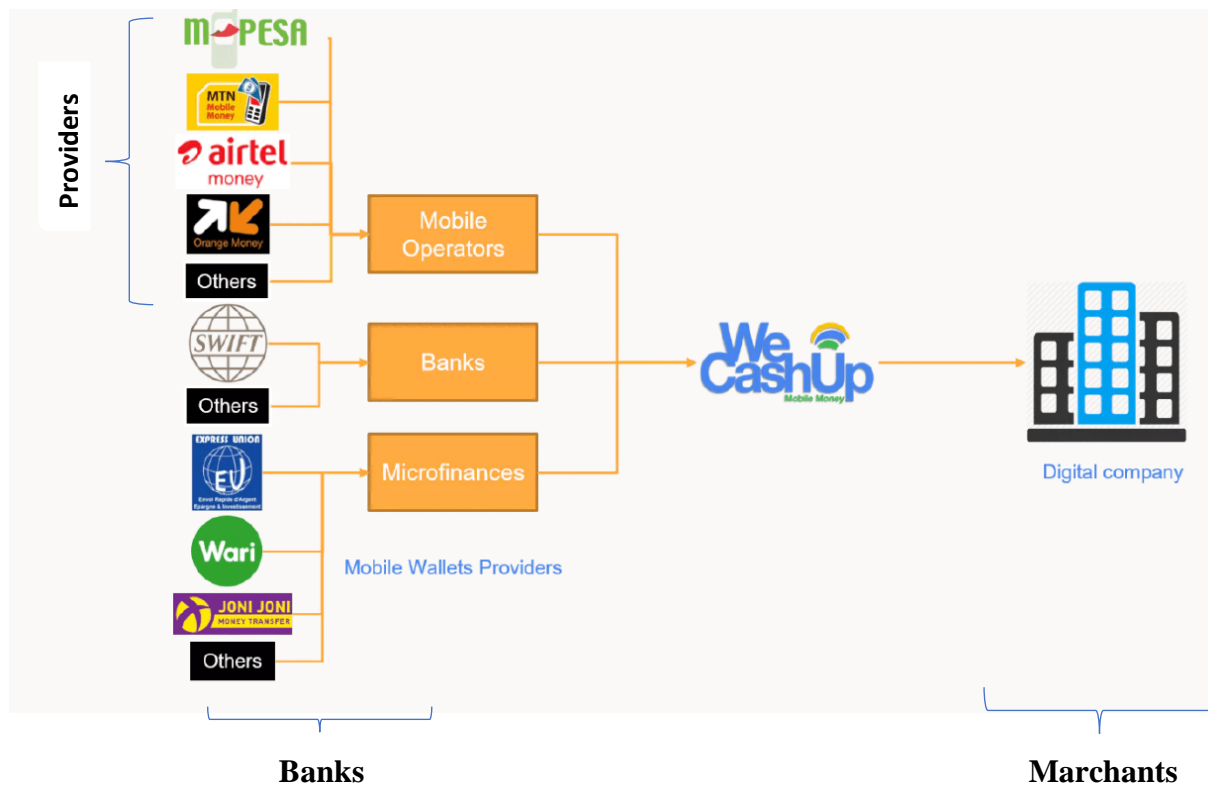


Schéma N°1 : wecashup-Archtype.

Banks : IFINITY SPACE n'est pas une institution financière, donc, pour pouvoir collecter ses fonds, elle travaille avec plusieurs Banks, dont une partie de ses transactions sont transférées directement dans son compte auprès de ces Banks, qui gagnent un pourcentage sur les transactions mensuelles. Les **Microfinances** sont à la fois une banque et opérateur télécom, c'est-à-dire, elles peuvent faire leurs propres transactions et en même temps garder ses fonds, il est beaucoup plus facile pour INFINITY SPACE, de travailler avec les microfinances, car elles seront à la fois leur client et en même temps leur banque.

Customers : sont tous les clients qui viendront sur le site du marchand pour un besoin quelconque, WeCashUp ne gère pas ces clients, elle s'occupe tout simplement à vérifier leurs informations, dont les marchands ont envoyé au serveur de WeCashUp.

Marchant : sont tous les sites d'e-commerce qui ont intégré la solution de WeCashUp, l'intégration de WeCashUp est aussi facile qu'à intégrer un seul bouton qui ressemble à ceci :



Figure N°3 : bouton de paiement

Source : www.wecashup.cloud/developers/docs/en/#test-phone-numbers

Mobile Money : est désormais la marque de INFINITY SPACE, c'est pourquoi au lieu de *paye by WeCashUp*, ils l'ont nommé : **Pay by Mobile Money** car WeCashUp est le nom de la plateforme et Mobile Money est la marque. Les marchands intégreront ce bouton sur leur page de paiement. Les clients qui n'ont pas de carte bancaire, peuvent alors effectuer leur paiement via ce bouton. Une fois que le client clique sur **Pay by Mobile Money**, il lui sera demandé quelques informations, sans entrer dans les détails, mais WeCashUp à son tour se charge de vérifier ces informations (numéro de téléphone, montant payé, code secret ...), auprès de Providers et informer le marchand de la crédibilité de ce client, et le marchand informe son client, si l'opération a été réussie ou pas.

Providers : Sont tous les opérateurs télécom qui font le paiement par mobile.

Certes, INFINITY SPACE reste une startup pour le moment, mais elle s'est dotée d'une méthode de travail qui se présente comme suit :

2.1. Méthode de travail:

a) Outils utilisés :

Travailler en équipe devient de plus en plus facile avec l'utilisation de Git et de ses nombreuses plateformes créées par les utilisateurs.

Héberger vos projets un peu partout sur la toile (Mega, FTP perso, MediaFree, FileFactory Etc.) et de devoir mettre à jour les liens de téléchargement à chaque mise à jour ! Lorsqu'on se lance dans des projets de développement de grande envergure, il devient alors nécessaire de recourir à un gestionnaire de source. Comme une partie de WeCashUp est développée avec la technologie JAVA, donc il nous fallait un éditeur ou encore un IDE (Environnement de Développement Intégré) et aussi un outil de versionning, pour éviter toute perte des données, ou pouvoir revenir sur la version précédente.

La team WeCashUp utilise « **Bitbucket** » comme gestionnaire de source et le client Git nommé « **Source Tree** ».

Pourquoi gestionnaire de source **Bitbucket** ?

Voici en quelques lignes ce qui caractérise principalement ce gestionnaire de sources :

- Création/ gestion des dépôts privés et publics (dépôts = projet).
- Partage de projets avec d'autres personnes.
- Travail collaboratif avec d'autres développeurs.
- Accepter ou rejeter des « Pull Request » (soumissions d'ajouts de fonctionnalités, correctifs,) d'autre développeurs ayant bifurqués vos projets.
- Modification en ligne ou depuis le client Git « **Source Tree** » de votre code source (complet ou indexé).
- Navigation dans vos projets grâce à l'historique de commit.
- Gestion de versionning et des branches du projet.

Il est surtout primordial d'avoir l'assurance qu'un projet est toujours disponible depuis Internet et de pouvoir revenir sur des versions antérieures à n'importe quel moment !

Le principal avantage à tirer de cet outil (**Bitbucket**) par rapport à son concurrent(GitHub) tient essentiellement dans la possibilité d'héberger autant de dépôts privés que vous voulez.

Accéder aux paramètres du compte en cliquant en haut à droite sur votre avatar puis « Manage Atlassian Account » afin d'accéder à ceci :

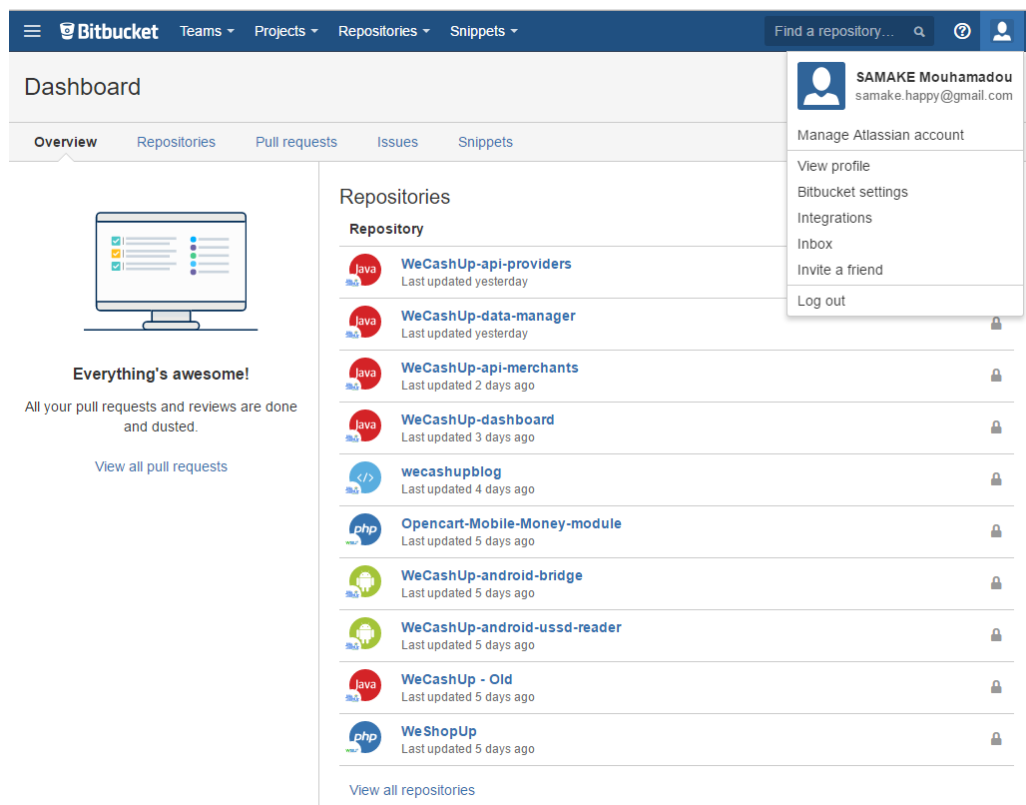


Figure N°4 : Bitbucket

Source : <https://bitbucket.org/product>

Le menu latéral de gauche vous permettra de paramétrer en profondeur votre compte.

Le menu en haut à gauche donne accès principalement :

- Dashboard : vue d'ensemble (projets et publics), demande d'ajouts, signalement de bugs.
- Repositories : Créer un dépôt, importer un dépôt.
- Create : créer directement un dépôt.

Pourquoi *Source Tree* ?

SourceTree : est un logiciel client développé par Atlassian, tout comme Bitbucket, permet aux utilisateurs d'accéder sur une interface utilisateur simple. SourceTree peut également être accéder à des services comme GitHub et BitBucket d'Atlassian qui prend en charge Mercurial et Git depuis un certain temps. SourceTree peut être aussi utilisé avec Stash, un Git Repository Management.

La plupart des utilisateurs accèdent à Git via ligne de commande et la plupart des environnements de développement offrent des plugins qui permettent aux utilisateurs d'interagir avec Git. Atlassian dit que 70% de ses clients sont des utilisateurs Windows et que probablement beaucoup de gens ne sont pas habitués à travailler à partir de la ligne de commande. C'est pour cela que SourceTree peut être très utile à ces utilisateurs, compte tenu du fait que ce logiciel prend en charge plus de commandes et offre plus de fonctionnalités que de nombreux plugins sur la plupart des IDE. En plus de cela il est gratuit 😊.

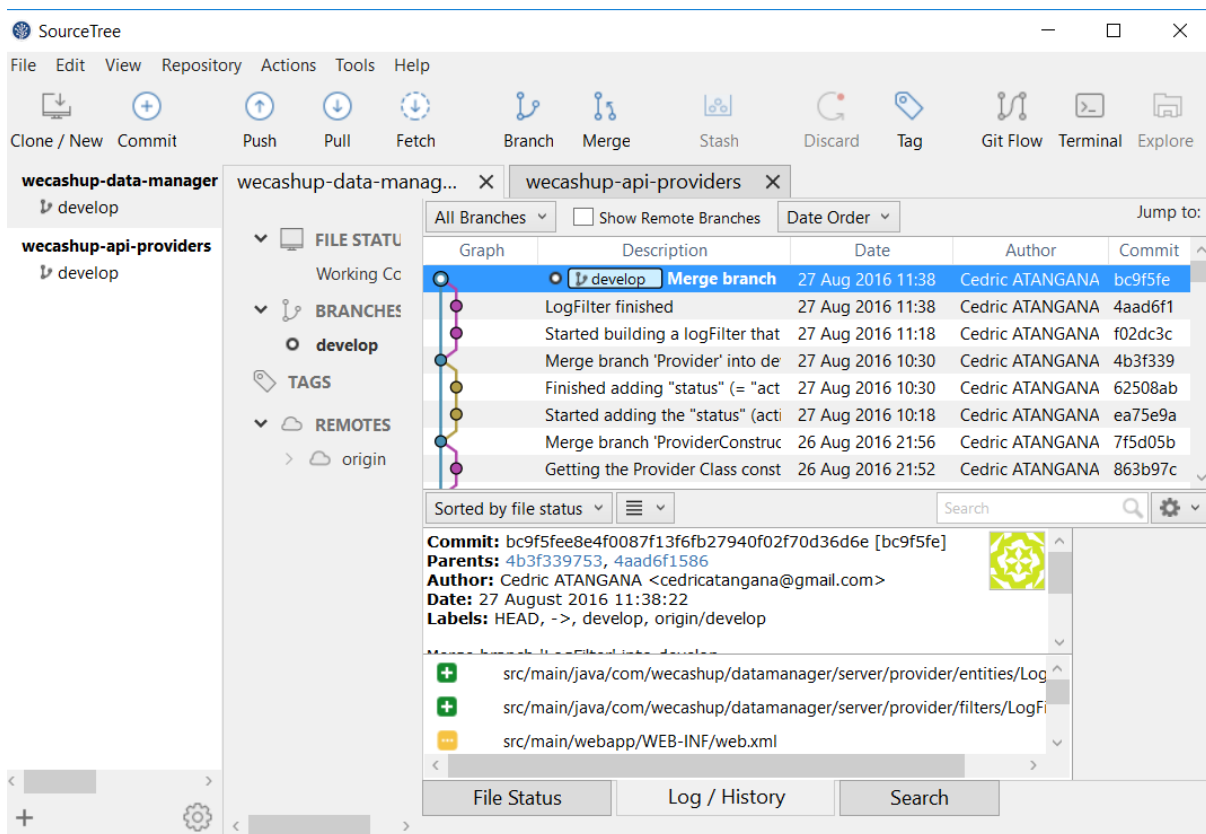


Figure N°5 : SourceTree

Source : <http://www.crazyws.fr/dev/sourcetree-un-client-git-et-mercurial-pour-windows-et-mac-N73D6.html>

La team WeCashUp travaille avec IDE Eclipse for java EE developpers, plus précisément Eclipse Néon 2, il est fluide et facile à utiliser, et vient avec plusieurs fonctionnalités tel que : J2EE, Maven, JSP

J2EE : Java Enterprise Edition, La plateforme Java Entreprise (Java EE) est un ensemble de spécifications coordonnées et pratiques qui permettent des solutions pour le développement, le déploiement, et de la gestion des applications multi-tiers centralisées sur un serveur. Construite sur la plateforme de Java 2 édition standard (Java SE), la plateforme Java EE ajoute les possibilités nécessaires pour fournir une plateforme complète, stable, sécurisée, et rapide de Java au niveau entreprise.

La plateforme Entreprise fournit un ensemble de services permettant aux composants de dialoguer entre eux :

- HTTP et HTTPS
- Java Transaction API (JTA)
- Remote Method Invocation/Internet Inter-ORB Protocol (RMI/IIOP)
- Java Interface Definition Language (Java IDL)
- Java DataBase Connectivity (JDBC)
- Java Message Service (JMS)
- Java Naming and Directory Interface (JNDI)
- API JavaMail et JAF (JavaBeans Activation Framework)
- Java API for XML Processing (JAXP)
- Java EE Connector Architecture
- Gestionnaires de ressources
- Entreprise Java Beans (EJB)
- Java Server Pages (JSP)
- Servlet
- Java API for XML Web Services (JAX-WS, anciennement JAX-RPC)
- SOAP with Attachments API for Java (SAAJ)
- Java API for XML Registries (JAXR)

JSP : Java Server Pages, comme son nom l'indique, les JavaServerPages (JSP) vous permettent d'insérer des petits bouts de code Java (scriptlets) directement dans du code HTML. Une page JSP est un document qui contient deux types de texte :

- Des données statiques (qui peuvent être exprimées en n'importe quel format texte tel que le HTML, le WML, et le XML) ;
- Des éléments de JSP, qui déterminent comment la page construit le contenu dynamique.

Maven ⁽³⁾ : Maven est un outil de gestion de projet qui comprend un modèle objet pour définir un projet, un ensemble de standards, un cycle de vie, et un système de gestion des dépendances. Il embarque aussi la logique nécessaire à l'exécution d'actions pour des phases bien définies de ce cycle de vie, par le biais de plugins. Lorsque vous utilisez Maven, vous décrivez votre projet selon un modèle objet de projet clair, Maven peut alors lui appliquer la logique transverse d'un ensemble de

plugins (partagés ou spécifiques). Voici une image qui m'a aidé à mieux cerner l'architecture de maven sous Eclipse :

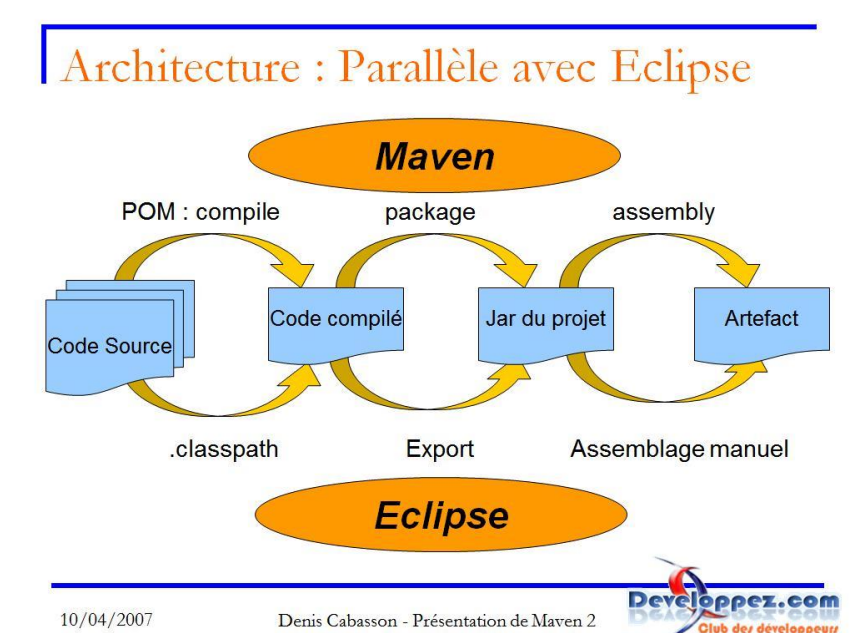


Figure N°6 : architecture maven

Source : Google search

J'ai utilisé les dépendances de Maven afin de me faciliter les imports de certaines bibliothèques, comme j'ai expliqué dans la page précédente, je n'ai pas eu besoin d'installer **maven** car Eclipse Néon vient avec, tout ce que j'ai eu à faire était de définir le nom de la bibliothèque dont j'avais besoin à partir du fichier pom.xml de **maven**, comme tel :

```
<!-- https://mvnrepository.com/artifact/com.googlecode.libphonenumber/libphonenumber -->
<dependency>
  <groupId>com.googlecode.libphonenumber</groupId>
  <artifactId>libphonenumber</artifactId>
  <version>8.3.2</version>
</dependency>
<dependency>
  <groupId>com.googlecode.libphonenumber</groupId>
  <artifactId>geocoder</artifactId>
  <version>2.65</version>
</dependency>
<dependency>
  <groupId>com.googlecode.libphonenumber</groupId>
  <artifactId>carrier</artifactId>
  <version>1.55</version>
</dependency>
```

Figure N°7 : expleMaven.

Comme vous pouvez le remarquer, c'est dans la balise <dependency></dependency> qu'on définit le nom de la bibliothèque dont on a besoin ; le <groupId></groupId> prend le nom global de la bibliothèque soit en ligne, ou soit dans un de vos projets qui se trouvent sur votre machine ; et le

<artifactId></ artifactId > prend le nom spécifique de la partie que vous avez besoin dans la bibliothèque.

NB : Au niveau de l'import d'un de vos projets qui se trouve dans votre machine, vous aurez besoin de taper un code supplémentaire, avant cela, vous avez à exporter votre projet en fichier .jar, pour ce faire, je procède comme suit :

Faites clic droit sur le projet à exporter, ensuite cliquez sur exporter, vous serez dirigé vers une boîte de dialogue qui ressemble à ceci :

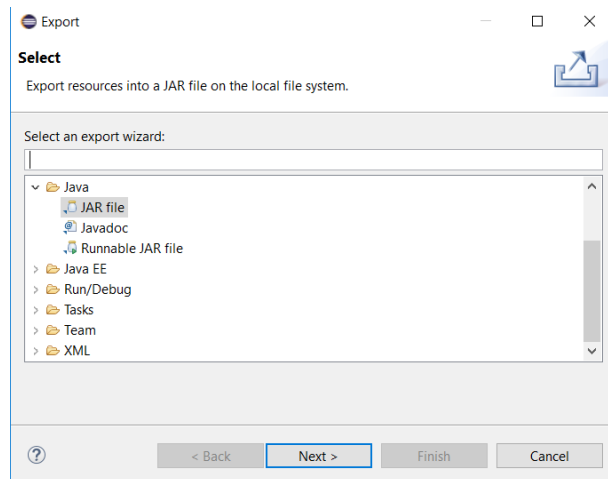


Figure N°8 : Eclipse.

Sélection JAR file et cliquez sur Next, une autre boîte de dialogue s'ouvre :

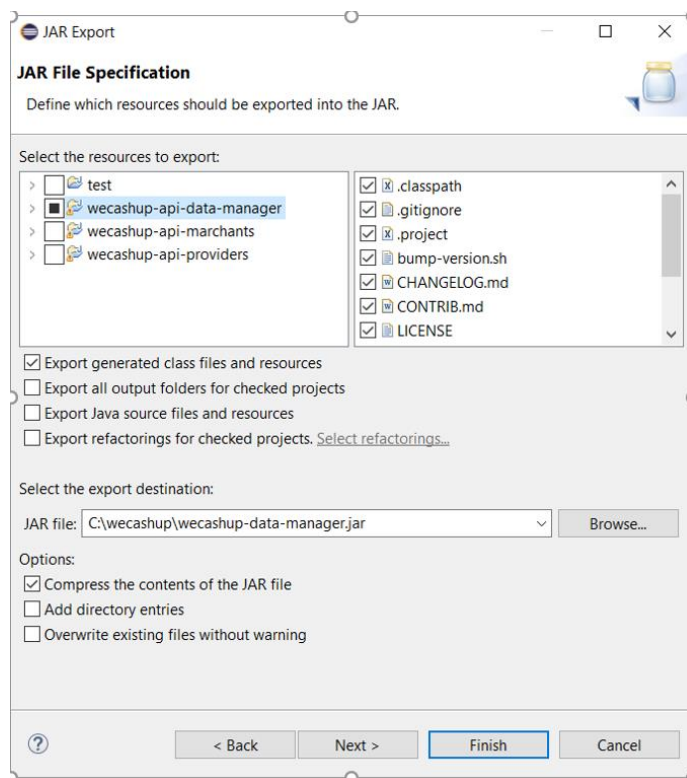


Figure N°9 : éclipse

Cochez le nom de projet à exporter en haut à gauche et donnez le chemin où enregistrer, pour finir cliquez sur finish, ensuite ok jusqu'à la fin.

Une fois que le projet est exporté, afin de l'indiquer le chemin d'accès et le nom du fichier, taper cette ligne de commande dans la console :

```
C:\Users\Annicelle>mvn install:install-file -Dfile=C:\wecashup\wecashup-data-manager.jar -DgroupId=com.wecashup -DartifactId=datamanager -Dversion=1.0 -Dpackaging=jar
```

- **Mvn install : install-file** : permet d'exécuter maven sur votre machine.
- **-Dfile=C:\wecashup\wecashup-data-manager.jar** : est le chemin d'accès au fichier spécifique que vous venez d'exporter.
- **-DgroupId=..... -DartifactId =..... -Dversion=.... -Dpackaging=jar** : le nom du groupId, artifactId et la version que vous avez défini le fichier pom.xml, laissez packaging comme tel, car il indique l'extension du fichier.

Rassurez-vous que cette commande a bien été exécutée avec succès, pour cela, vérifiez que la mention « Build success » apparaît dans la console.

```
C:\Users\Annicelle>mvn install:install-file -Dfile=C:\wecashup\wecashup-data-manager.jar -DgroupId=com.wecashup -DartifactId=datamanager -Dversion=1.0 -Dpackaging=jar
[INFO] Scanning for projects...
[INFO] -----
[INFO] Building Maven Stub Project (No POM) 1
[INFO] -----
[INFO] --- maven-install-plugin:2.4:install-file (default-cli) @ standalone-pom ---
[INFO] Installing C:\wecashup\wecashup-data-manager.jar to C:\Users\Annicelle\.m2\repository\com\wecashup\datamanager\1.0\datamanager-1.0.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 0.342 s
[INFO] Finished at: 2017-05-04T17:49:53+02:00
[INFO] Final Memory: 5M/15M
[INFO] -----
```

Figure N°10 : Cmd commande maven.

b) Réalisation :

WeCashUp ayant été présenté, cette partie consistera à présenter les réalisations techniques.

Dans un premier temps j'ai été amené à faire les inscriptions des Marchands, des clients et des providers (opérateurs télécom, banques et microfinances). Ensuite il a été question de mettre en place les mécanismes techniques qui nous garantissent l'intégrité des données venant de chacune de ces entités (traitement de la transaction, calcul des commissions sur les différentes transactions). Puis, la rédaction et le déploiement en ligne de la documentation pour permettre aux développeurs des marchands d'intégrer plus facilement l'API WeCashUp. En fin j'ai été en

charge de la formation des nouvelles personnes qui ont rejoint l'équipe sur le mode de travail (charte de codage, utilisation de Git...) de l'INFINITY SPACE.

2.1.1. Inscription :

Avant de me lancer dans la création des classes correspondantes à ces entités en JAVA, j'ai récolté toutes les informations que peuvent avoir ces entités dans un fichier Excel, afin de savoir quelle variable va contenir quel type de donnée et quelle est sa longueur ou sa capacité. Ce dernier ressemble à ceci :

OBJECT : Merchant registration				
Parameters			Type	Example
merchant_id	Automatic		key	
merchant_uid			String	bfdhdfhjfcjfhc456353jghf
merchant_legal_name			String	Ibouanga
merchant_email			String	wecashup@hotmail.fr
merchant_password			String	FDgde97'L_é_
Localization				
merchant_physical_address			String	171 avenue de luminy /quartier gobongo 2
merchant_postcode			String	13009
merchant_town			String	Brazzaville
merchant_country			String	Congo
merchant_phone_number			String	+237695640900
merchant_website			String	www.wcu.fr
Legal Representative				
merchant_legal_rep_firstname			String	Ibouanga
merchant_legal_rep_lastname			String	Harold
merchant_legal_rep_birthdate			date	17/08/1993
merchant_legal_rep_physical_address			String	13 avenue d'aix en provence
merchant_legal_rep_postcode			String	13080
merchant_legal_rep_town			String	Brazzaville
merchant_legal_rep_country			String	Congo

Figure N°11 : Merchant Object

Ces informations m'ont permis d'aller plus vite lors de la création des différentes classes, comme vous avez peut-être remarqué, chaque paramètre possède un nom, un type, et un exemple, qui compose bien évidemment chaque classe, tout en respectant les règles de la modélisation et les relations entre ces classes (association, composition, agrégation ...).

Intégrité de données : n'est autre que la cohérence des données lors de la transmission, de leur traitement, ou de la conservation, du côté utilisateur ou même venant du serveur de WeCashUp.

De manière générale, l'**intégrité des données** désigne l'état de **données** qui, lors de leur traitement, de leur conservation ou de leur transmission, ne subissent aucune altération ou destruction volontaire ou accidentelle, et conservent un format permettant leur utilisation (4).

Par exemple : un marchand menant ses activités au Mali, s'est inscrit avec la devise malienne, si par x-raison un de ses client, envoie une transaction avec une autre devise, cette transaction ne doit

pas passer, ou les données d'une carte Sim, dont cet opérateur télécom n'est pas opérationnel au Mali etc.

Pour ce faire, lors de la création de mes classes, je fais en sorte que, chaque classe qui est composée par une autre, ou qu'elles sont liées par une relation quelconque, doit embarquer le **uid** de ce dernier, à travers ce **uid**, j'obtiens toutes les informations complémentaires de cette classe, afin de faire mes différentes vérifications et de valider la transaction.

Uid : est une chaîne de caractère dont je génère, à chaque appellation du constructeur concerné.

Traitement de la transaction : Une transaction est créée lorsqu'un client vient sur le site du Marchand et clique sur le petit bouton vert (**figure N°3**), il sera dirigé vers le box de paiement de WeCashUp :

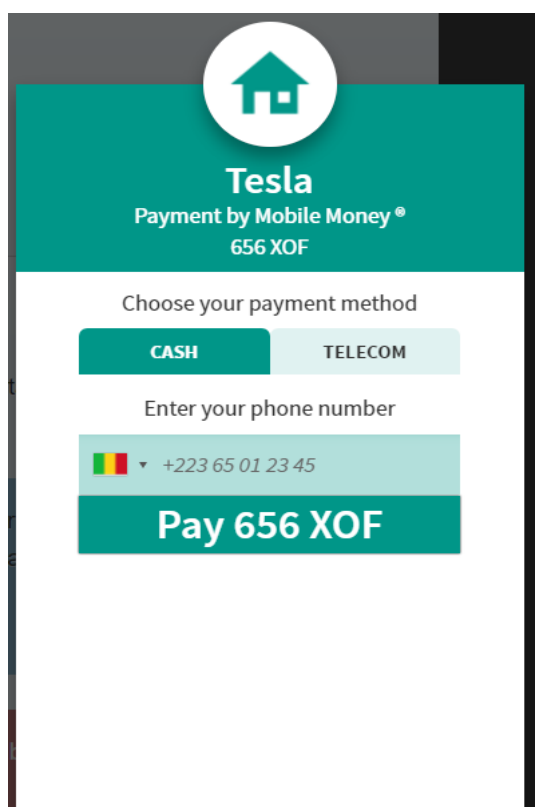


Figure N°12 : WeCashUp box payment

Source : <https://www.wecashup.cloud/developers/docs/en/#test-phone-numbers>

Une fois choisi sa méthode de paiement, et fourni son numéro de téléphone, ensuite **Pay**, à partir de ce moment, la transaction sera créée.

Pour le moment WeCashUp propose deux méthodes de paiement qui sont :

CASH : méthode cash permet aux utilisateurs qui ne possèdent pas, soit un compte mobile money, soit de l'argent dans leur compte, d'aller dans un point de vente de son opérateur télécom, et déposer

le montant à payer sur le site, en fournissant le code de confirmation qui lui a été généré par le serveur de WeCashUp (voir annexe2).

TELECOM : permet aux user de payer facilement sans déplacement, il leurs sera demandé uniquement de fournir le code de confirmation qu'il lui a été fourni par son opérateur dans le message de confirmation d'envoi de l'argent (voir annexe3).

Au niveau du traitement de la transaction, dans le backend, je récupère ces informations (méthode, numéro, montant...), je vérifie tout d'abord son numéro de téléphone afin de détecter son opérateur télécom, et je m'assure que cet opérateur autorise ce montant à travers l'une de mes classes Java (AmountAuthorized) afin de diriger la transaction vers le serveur adéquat.

Calcul de commission : la commission est le frais que INFINITY SPACE, prend sur toutes les transactions par rapport aux grilles tarifaires.

Ces grilles tarifaires sont définies dans un fichier JSON, ce qui me facilite le calcul de la commission, à l'arrivée d'une transaction, je charge ce fichier à travers de l'opérateur télécom et le montant de la transaction, qui me retourne un pourcentage à calculer sur la transaction dû. Comme INFINITY SPACE n'est pas une institution financière, elle travaille alors avec les Banques pour collecter ses fonds ; qui ont aussi une commission sur la transaction, ce qui me laisse à calculer la commission sur des différentes manières. Pour faire cela, je crée un objet Transaction dont je set son type par le nom de la commission concernée, et le montant, par la valeur de la transaction, multipliée par le pourcentage correspondant.

2.1.2. Rédaction de la documentation de l'API de WeCashUp :

a) Qu'est-ce qu'une API ?

Une API, interface de programmation applicative (Application Programming Interface) est un ensemble normalisé de classes, de méthodes ou de fonctions qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels. Elle est offerte par une bibliothèque logicielle ou un service web, le plus souvent accompagnée d'une description qui spécifie comment des programmes consommateurs peuvent se servir des fonctionnalités du programme fournisseur. Cette description est la documentation de l'API.

b) L'API WeCashUp :

WeCashUp est une plateforme de paiement mobile dédiée principalement aux pays sous ou non bancarisés. En effet, il existe beaucoup de système de paiement mobile suivant différents opérateurs téléphoniques.

Souvent, dans un pays on peut avoir par exemple quatre opérateurs téléphoniques qui font du paiement mobile (exemple : Airtel Money, pour l'opérateur téléphonique Airtel / OrangeMoney pour l'opérateur téléphonique Orange / MobileMoney pour l'opérateur MTN / MoovMoney pour l'opérateur téléphonique Moov...) mais ils ne communiquent pas entre eux. Ainsi, une personne

utilisant ce système doit posséder à elle seule les quatre cartes Sim de ces opérateurs. Ce qui n'est pas pratique d'autant plus que dans certains pays, on peut quelque fois atteindre douze opérateurs.

Ainsi, WeCashUp est la solution qui offre de l'interopérabilité à ces systèmes.

De ce fait, en détenant une seule carte Sim d'un opérateur quelconque, une personne peut transférer et recevoir de l'argent de tous les autres opérateurs et même faire des achats sur n'importe quel site d'un marchand qui a embarqué la solution WeCashUp.

2.1.3. La documentation :

a) Pourquoi ?

Une API est publique ; donc tout le monde doit pouvoir la comprendre. De ce fait, sa documentation doit être compréhensible et accessible à tout le monde via un navigateur web. Pour pouvoir l'implémenter un développeur doit connaître comment les entités sont décrites (leurs attributs) ; les fonctions et ce qu'elles renvoient.... Ce document sert de lien entre les concepteurs de l'API et les utilisateurs.

Ainsi d'une part, il s'agit d'un document qui en interne va permettre à toutes les personnes participant l'implémentation de l'API d'avoir les mêmes appellations de variables etc..., et d'autre part ce document servira de tutoriel à tout développeur voulant embarquer le système sur son site.

De ce fait, lorsque la documentation d'une API est bien faite (claire, concise et lisible), une entreprise peut se passer du support du produit qui générerait des frais supplémentaires à cette dernière.

b) Que doit-elle contenir ?

✓ Les entités :

Dans le cadre d'une API web, une entité est un objet caractérisé par des attributs.

Formaliser toutes les entités dans la documentation d'une API permet à toute personne manipulant cette API de connaître tous les objets et les attributs sur lesquels l'on agit.

On peut manipuler les entités (sur le web) en utilisant des requêtes HTTP :

- POST à une URL donnée : pour créer une entité.
- GET à une URL donnée : pour récupérer des entités.
- PATCH à une URL donnée : pour mettre à jour des entités.
- DELETE à une URL donnée : pour supprimer des entités existantes.

Exemple d'une entité client :

Clients

Objet client

Variables	Description	Requis
customer_uid	L'identifiant du client	oui
customer_phone_number	Le numéro de téléphone du client	oui
customer_firstname	Le nom du client	optionnel
customer_lastname	Le prénom du client	optionnel
customer_birthdate	La date de naissance du client	optionnel
customer_currency	La devise du client	oui
customer_status	Le statut du client (blocked, deleted, activated)	oui
customer_balance	La balance du client	oui
customer_email	L'email du client	oui

Figure N°13 : Entité client.

Source : <https://www.wecashup.cloud/developers/docs/fr/?html#clients>

✓ **La description des fonctions :**

Pour pouvoir implémenter une API, un développeur doit non seulement connaître les entités et leurs attributs mais aussi toutes les fonctions (leur signature : ce qu'elles prennent en paramètre, ce qu'elles renvoient comme résultat...) auxquelles on fait appel pour pouvoir manipuler ces entités.

Notant que connaître les paramètres et les types de retour des fonctions est très important pour l'utilisateur de l'API. Ainsi, il pourra envoyer les bonnes requêtes avec les bons paramètres à la bonne adresse.

✓ **Les codes d'erreur :**

Les codes d'erreurs sont très importants lorsqu'on veut décrire une API. Ceux-ci permettent de communiquer dans un langage universel.

Ainsi, il est recommandé de les mettre dans sa documentation dans le but de permettre à l'utilisateur de pouvoir interpréter les erreurs affichées afin de localiser la source du problème. De cette manière, le problème peut être facilement identifié et résolu. Les codes d'erreurs peuvent être :

- 200 OK : la requête HTTP GET, PATCH ou DELETE a été traitée avec succès.
- 201 Créé : la requête POST a été correctement traitée et a résulté en la création d'une nouvelle ressource.
- 204 Pas de contenu : le serveur HTTP a correctement traité la requête mais il n'y a pas d'information à envoyer en retour.
- 304 Non modifié : Document non modifié depuis la dernière requête.
- 400 Mauvaise requête la requête a été mal formulée.

- 401 Non autorisé la requête nécessite une identification de l'utilisateur.
- 403 InterditLe serveur HTTP a compris la requête, mais refuse de la traiter parce que l'auteur de la requête n'a pas assez de droits pour accéder à la ressource sollicitée.
- 404 Non trouvé le serveur n'a rien trouvé qui corresponde à la ressource demandée.
- 405 Méthode non autorisée la méthode utilisée par le client n'est pas supportée pour cette ressource.
- 410 Parti la ressource n'est plus disponible et aucune adresse de redirection n'est connue.
- 415 Format non supporté le serveur ne peut traiter la requête car son contenu est écrit dans un format non supporté.
- 422 Entité incompréhensible l'entité fournie avec la requête est incompréhensible ou incomplète.
- 429 Limite de nombre de requêtes atteinte le client a émis trop de requêtes dans un délai donné.
- 500, 502, 503, 504 Erreurs serveur problème survenu sur le serveur de WeCashUp (cas exceptionnel).

✓ **Versioning :**

Le versioning est très important pour une API. Ainsi il est préférable de définir au préalable une politique de versioning qui permettra à l'utilisateur de garder un œil sur des potentielles modifications de l'API.

Une politique de versioning est par exemple la politique de dépréciation. Lorsqu'on veut modifier une méthode, au lieu de la supprimer du jour au lendemain, il faut la signaler d'abord comme « deprecated », proposer la nouvelle solution pour que les gens puissent avoir le temps de se familiariser avec et ensuite lors du changement, supprimer la méthode dépréciée de l'API.

Bien sûr tout ceci doit être prédéfini dans la documentation de l'API ; parce que si l'utilisateur de l'API ne sait pas comment tout cela fonctionne, cela risque d'être embêtant.

2.1.4. Mise en place de la documentation de l'API de WeCashUp :

Pour mettre en place cette documentation, j'ai, dans un premier temps cherché plusieurs Template à proposer à mon tuteur et l'équipe de développement, et il m'a dit de partir sur la plus facile à comprendre, j'ai alors choisi **Slate**, qui est faite par le langage Ruby dont je vous présenterai dans la page suivante. Mon choix pour cette Template est plutôt son côté lisibilité, aussi elle donne la possibilité de présenter les différentes manières d'intégrer l'API dans des différents langage que vous souhaitez.

a) Ruby :

Ruby est un langage de programmation libre. Il est interprété, orienté objet et multiparadigme. Le langage a été standardisé au Japon en 2011 (JIS X 3017 :2011), et en 2012 par l'Organisation internationale de normalisation (ISO 30170 :2012).

b) Slate :

Slate est un outil de création de la documentation d'API, codé en langage Ruby. Il est complètement open-source.

Slate est un Template déjà prédéfinie. Son interface présente :

- Une zone pour un logo
- Une zone de recherche
- Une zone pour le sommaire (menu)
- Une zone pour le texte
- Une zone pour les exemples de codes dans différents langages
- Une zone pour les recherches.

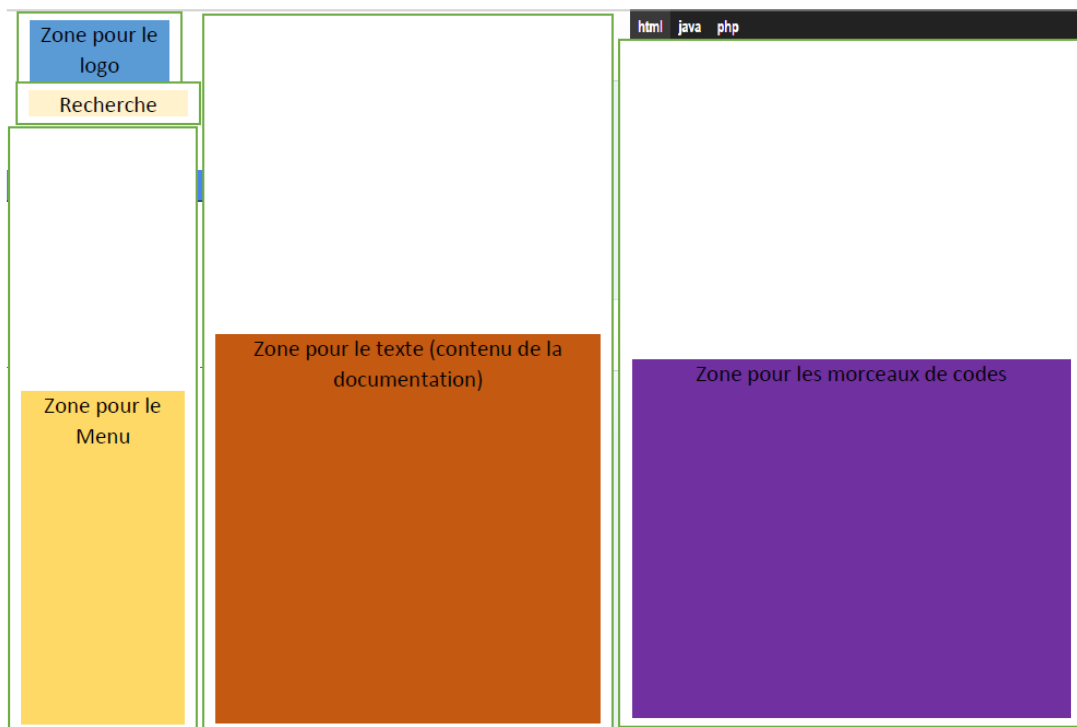


Schéma N°2 : Template Slate

Le document Slate contient un fichier .md où on peut rajouter notre documentation en fonction des différentes zones présentées ci-dessus.

Quelques notions élémentaires pour écrire dans le fichier .md :

- Un lien hypertexte : [le mot ou la phrase] (le lien)

- Titre de niveau 1 (Grand titre) : # Mon grand titre
- Titre de niveau 2 (titre moyen) : ## Mon moyen titre
- Titre de niveau 3 (petit titre) : ### Mon petit titre (ce titre n'apparaît pas dans le sommaire).

Notez que Les titres de niveau 1 et 2 vont s'ajouter automatiquement au Menu.

- Pour un paragraphe : Il suffit simplement d'écrire un texte (chaque paragraphe correspond à un retour à la ligne) Zone pour le logo Zone pour les morceaux de codes Zone pour le texte (contenu de la documentation) Zone pour le Menu Recherche
- Pour rajouter du texte dans la zone de code il suffit de rajouter le symbole « > » au début de la phrase.
- Pour rajouter du code dans la zone de code ; il suffit de rajouter les symboles suivants en fonction du langage en question :

```
```langage
Votre code ici
```
```

- Pour écrire en bas du Menu :

toc_footers:

- Votre texte ici (par exemple : `Sign Up for a Developer Key`)
- Pour rajouter les différents types de langages que vous allez utiliser dans la zone de code :

language_tabs:

```
Html
Java
PHP
```

- Rajouter un tableau (c'est le code qui a généré l'entité client présenté ci-dessus) :

Attributs | Description | Requis

----- | ----- | -----

`<code>customer_id </code>` | L' identifiant du client | oui

`<code>customer_phone_number </code>` | Le numéro de téléphone du client | oui

`<code>customer_name</code>` | Le nom du client | optionnel

`<code>customer_last_name</code>` | Le prénom du client | optionnel

`<code>customer_birthdate </code>` | La date de naissance du client | optionnel

`<code>customer_currency</code>` | La devise du client | oui

`<code>customers_status</code>` | Le status du client (blocked, deleted, activated) | oui

c) Installation de Ruby et exécution de Slate (via une invite de commande) :

Avant tout, il faut vous assurer que Node js soit installé sur votre machine. Si non, il faudra l'installer.

En effet, Node js est une plateforme logicielle qui contient une bibliothèque de serveur HTTP ; ce qui permet de faire tourner un serveur web sans avoir à utiliser un logiciel externe comme Apache par exemple.

Ensuite :

- Télécharger le setup sur <http://railsinstaller.org/en>
- Lancer le setup une fois le téléchargement terminé.
- Lancer un invité de commande
- Taper la commande **bundle** (Si la commande n'est pas reconnu, taper **gem install bundler**)
- Ensuite retapez **bundle** pour être sûr que l'installation s'est bien passée
- Puis déplacez-vous dans votre document slate avec la commande **cd**
- Tapez **bundle install** pour installer toutes les dépendances
- Puis exécutez **bundle exec middleman server**
- Votre documentation est disponible à l'adresse suivante : <http://localhost:4567>
- Si vous voulez l'exporter au format HTML, taper la commande **bundle exec middleman build clean** : cette commande générera un document nommé « build » qui contient la doc au format HTML.

2.1.5. Déploiement sur Google App Engine :

2.1.5.1. Qu'est-ce que c'est Google App Engine:

Google App Engine permet d'exécuter vos applications web sur l'infrastructure de Google. Les applications App Engine sont faciles à construire, faciles à maintenir, et supportent facilement la montée en charge de votre trafic et de vos besoins croissants de stockage de données. Avec App Engine, il n'y a pas de serveurs à maintenir, vous chargez juste vos applications, et elles sont aussitôt disponibles pour vos utilisateurs.

Google App Engine supporte les applications écrites dans plusieurs langages de programmation. Avec l'environnement d'exécution Java™ App Engine (JRE), vous pouvez construire votre application en utilisant les technologies standard Java, incluant la JVM, les servlets Java, et le langage de programmation Java - ou n'importe quel autre langage utilisant un compilateur ou un interpréteur basé sur une JVM, tel que le JavaScript ou le Ruby. App Engine comprend également un environnement d'exécution Python dédié, qui inclut un interpréteur Python rapide et une librairie

python standard. Les environnements Java et Python sont construits de sorte à assurer à vos applications qu'elles fonctionnent rapidement, dans un espace sécurisé, et sans interférences avec les autres applications du système.

Avec App Engine, vous payez seulement pour ce que vous utilisez. Il n'y a pas ni coût de mise en service, ni commissions récurrentes. Les ressources que vos applications utilisent, tel que le stockage et la bande passante, sont mesurées en giga-octets, et facturées à des taux compétitifs. Vous contrôlez le niveau maximum de ressources que peut consommer votre application, de sorte que votre budget ne soit jamais dépassé.

App Engine ne coûte rien pour démarrer. Toutes les applications peuvent utiliser jusqu'à 500 Mo de stockage et assez de CPU et de bande passante pour supporter une demande d'environ 5 millions de pages vues par mois, absolument gratuitement. Quand vous activez la facturation pour votre application, vos limites gratuites sont augmentées, et vous ne payez seulement que les ressources que vous consommez au-delà de ces limites gratuites (5).

Le rôle d'App Engine est de "masquer" le fonctionnement et la complexité des serveurs de Google. Lorsqu'un visiteur se connecte à votre site, il arrive sur le load balancer (répartiteur de charge) de Google, qui va chercher un serveur disponible et pas trop chargé pour gérer la demande de votre visiteur. Si votre site a besoin d'accéder à des données, ce qui est fréquent, il fera appel au Datastore (voir annexe4).

2.1.5.2. Datastore:

App Engine fournit un service performant de stockage distribué de données qui comprend un moteur de recherche et la gestion des transactions. En même temps que les serveurs web distribués augmentent avec votre trafic, la base de données grossit avec vos données.

Le datastore App Engine n'est pas comme une base de données relationnelle traditionnelle. Les objets de données ou "entités" ont des propriétés. Les recherches peuvent retourner des entités selon un filtre donné et les trier selon les propriétés de leurs valeurs. Les valeurs de ces propriétés peuvent être l'un des types de valeur de propriété supportés.

Les entités Datastore n'ont pas de schéma. La structure de données des entités est fournie et mise en place par le code de votre application. Les interfaces JDO/JPA et l'interface Python datastore incluent les dispositifs pour mettre en place la structure à l'intérieur de votre application. Votre application peut aussi accéder au datastore directement pour mettre en place une structure répondant plus précisément à vos besoins.

Le datastore est fortement consistant et utilise le contrôle d'accès simultané optimiste. Une mise à jour de l'entité se déroule dans une transaction qui est réessayé un nombre de fois fixe si d'autres applications tentent de mettre à jour la même entité simultanément. Votre application peut exécuter

plusieurs opérations concernant la base de données dans une seule transaction qui soit réussi entièrement, soit ne réussit pas du tout, assurant l'intégrité de vos données.

Le datastore implémente les transactions à travers son réseau distribué en utilisant des "groupes d'entités". Une transaction manipule des entités à l'intérieur d'un simple groupe. Les entités du même groupe sont enregistrées ensemble par souci d'efficacité d'exécution des transactions. Votre application peut attribuer des entités à des groupes quand les entités sont créées.

Google fourni également un ORM (**OBJECTIFY**) qui vous permet de mapper plus facilement vos objets dans le datastore, il est fluide et facile à construire vos requêtes.

2.1.5.3. Objectify:

Objectify s'occupe de la création d'entités et de clés et de la relation entre ces éléments. Mieux : il utilise automatiquement le service Memcache de Google qui permet de mettre en cache des données pour aller encore plus vite et vous éviter d'appeler systématiquement le Datastore. Objectify est suffisamment simple à comprendre et pourtant très puissant (6).

Objectify est une bibliothèque créée par un développeur tiers. Elle n'est pas installée automatiquement avec le plugin App Engine pour Eclipse. Heureusement, son installation et sa configuration sont simples. :)

Le projet Objectify est hébergé sur Google Code à l'adresse suivante : <https://github.com/objectify/objectify>

2.1.5.4. Déploiement:

Pour pouvoir envoyer la documentation de WeCashUp sur les serveurs de Google, il faut d'abord réserver un identifiant d'application. Nous pourrions ensuite dans un second temps déployer la documentation de l'API directement depuis Eclipse !

Je n'ai pas eu à réserver un identifiant d'application car l'entreprise avait déjà son identifiant, pour ceux qui veulent comprendre comment monter son site sur Google App Engine, voici un lien d'un tuto qui va peut-être vous aidez : <https://openclassrooms.com/courses/montez-votre-site-dans-le-cloud-avec-google-app-engine>.

Une fois créé votre identifiant d'application, le reste se passe sur Eclipse, Google nous fournit aussi un Plugin à installer sur Eclipse qui nous permet de créer notre projet jusqu'à le déployer, Hum Google a tout fait ☺. Pour installer ce plugin, dans la barre de menu d'Eclipse cliquer sur **Help/install New software...**, vous serez dirigé vers une boîte de dialogue, cliquez sur le bouton **add** au coin droit et vous serez sur une nouvelle page, dans la partie **Name** donnez un nom comme App Engine par exemple et **location** vous mettez ceci : <http://dl.google.com/eclipse/plugin/4.4>, ensuite cliquez sur **ok**, décochez certaines librairies dont vous n'avez pas besoin et laissez les autres cochées, ils sont

toutes cochées par défaut, une fois fini d'installer, vous allez apercevoir une petite icône de Google dans la barre de menu de votre Eclipse.

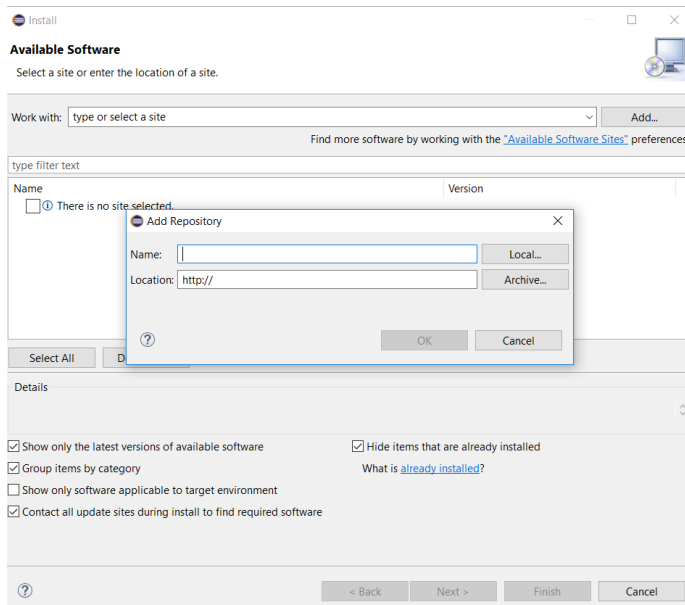
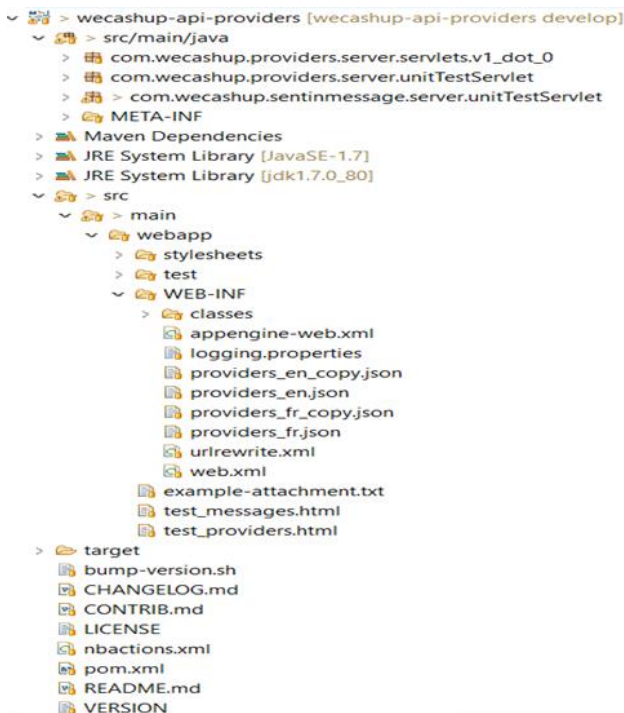


Figure N°14 : Install software

Une fois fini d'installer le plugin, j'ai alors importé le projet WeCashUp sur Bitbucket via sourceTree, Tout projet App Engine sur Eclipse, donne une architecture particulière qui est tout à différente des autres IDE comme NetBeans ...



FigureN°15 : Structure PJ App Engine

Selon la structure App Engine, les fichiers statiques sont stockés dans le dossier WEB-INF, et c'est dans ce dossier que je vais stocker mon fichier HTML, dont j'ai exporté par la commande que je vous ai donné au niveau de l'installation du Ruby, ensuite je copie l'identifiant de l'application que Google nous a fourni lors de la création du projet, et je le colle dans le fichier appengine-web.xml, entre les balise application comme ceci :

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE xml>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <application>wecashup-payment</application>
  <module>datamanager</module>
  <version>1</version>
  <threadsafe>true</threadsafe>

  <!-- Configure serving/caching of GWT files -->
  <static-files>
    <include path="*" />

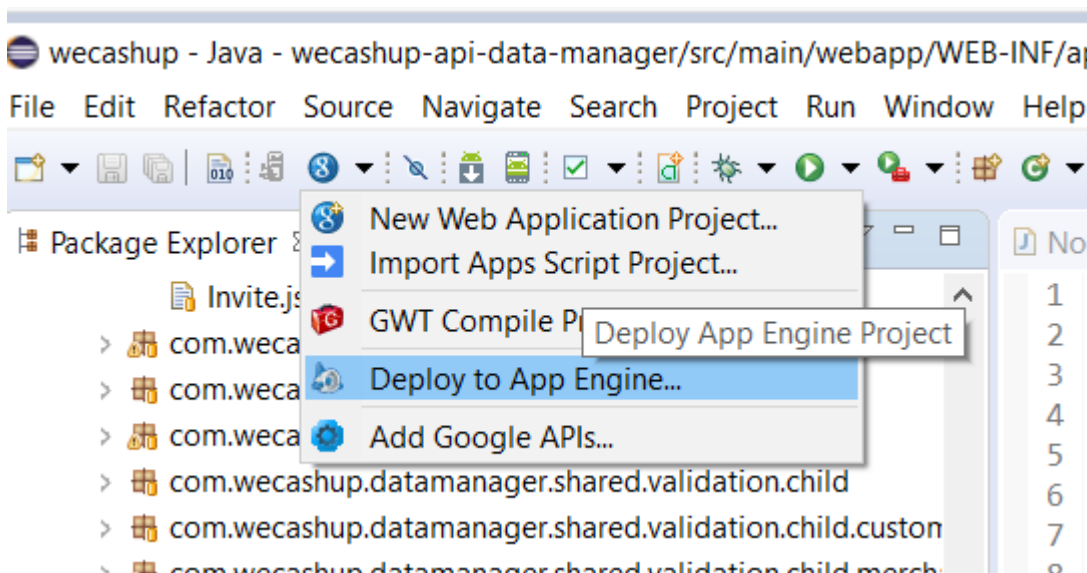
    <!-- The following line requires App Engine 1.3.2 SDK -->
    <include path="*.nocache.*" expiration="0s" />

    <include path="*.cache.*" expiration="365d" />
    <exclude path="*.gwt.rpc" />
  </static-files>

  <system-properties>
    <property name="java.util.Logging.config.file" value="WEB-INF/Logging.properties"/>
  </system-properties>
</appengine-web-app>
```

Figure N°16 : appengine-web.xml

La balise **module** prend le nom du projet dont je vais déployer, et la balise **version** pour la version du projet, ici j'ai déployé la première version, pour finir, un simple clic sur l'icône de Google et deploy to app engine :



L'avantage de Google App Engine est qu'on peut créer plusieurs services qui vont tourner sous le même projet (grâce à l'ID du projet) et créer des dépendances entres eux. Ces services peuvent être un site web, un blog et même un code java qui tourne.

2.1.6. Formation des nouvelles personnes(stagiaires) :

Lors de cette formation, j'étais amené à former les stagiaires sur le mode de travail de l'entreprise, plus précisément les outils à utiliser, la charte de codage, et le processus d'intégration la solution WeCashUp.

Désormais, l'entreprise dispose déjà un certain nombre de schéma à suivre pour bien mener cette formation, (voir annexe1). Pour les outils à utiliser, la plupart des stagiaires savent utiliser Eclipse. Parmi les difficultés rencontrées nous pouvons citer la prise en main de l'outils de versionning, qui a été plus au moins complexe pour les stagiaires, d'où l'utilité de SourceTree (client Git qui propose à la fois un mode d'interaction par ligne de commande ou par mode graphique).

La charte de codage consiste à coder le plus simplement possible, pour cela nous avons aussi un fichier Excel définissant les méthodes et les attributs de chaque classe doit passer avant de le valider. Le but est de tester la « robustesse » des classes. Puis on rajoute les jeux des tests qu'on veut faire passer à chacune de nos classes. Notant que nous avons tous les mêmes appellations de variables qu'on manipule et nous nous sommes tous d'accord sur la manière à laquelle on doit mettre en œuvre les jeux de test.

Le processus d'intégration est un schéma expliquant les différentes étapes qu'un développeur doit suivre, afin de faire l'intégration de WeCashUp sur le site de l'e-commerçant, dans l'entreprise tout le monde doit forcément pouvoir expliquer ce schéma, afin de répondre aux questions posées par les développeurs de nos clients.

Conclusion :

Cette première année d'alternance, fut pour moi un moment très intéressant et très enrichissant.

En effet, il m'a permis de mettre en application les connaissances acquises durant mon cursus à savoir la modélisation, conception et programmation orientée objet et surtout la mise en place d'une API ; les techniques de rédaction de document du cours de génie logiciel et les différents cours de management.

Il m'a également permis de m'imprégner et d'approfondir mes connaissances du monde de l'entreprise et ses difficultés ; le respect de la hiérarchie.

Enfin, il m'a permis d'améliorer certaines qualités comme la rigueur et la confiance en soi.

Cependant des perspectives d'améliorations de cette API restent envisageables, telles que l'ajout de plugin pour les CMS, et une application pour les utilisateurs finaux.

PORTEFEUILLE DE COMPETENCES

PROFESSIONNELLES

Liste des Compétences professionnelles :

1. Je suis capable d'élaborer un cahier des charges relatif à un problème concret.
2. Je suis capable de faire le suivi d'un projet avec l'outil de Microsoft Management Project.
3. Je suis capable de concevoir un modèle conceptuel et logique de données à partir de l'étude de l'existant d'un problème (analyse des données existantes).
4. Je suis capable de capturer les besoins fonctionnels dans un projet informatique afin de ressortir un diagramme de classe, de cas d'utilisation, de séquence en donnant une idée du squelette et de l'organisation.
5. Je suis capable de transformer ces diagrammes de classe en objets en JAVA avec toutes les relations qui leur lient (association, agrégation, composition).
6. Je suis capable de monter un site d'application web sur Google Cloud en langage JAVA.
7. Je suis capable de d'écrire et lire les données avec NoSQL.
8. Je suis capable de manipuler ces données avec Objectify.
9. Je suis capable de déployer un site sous GC en ligne.
10. Je suis capable de gérer l'outil de versionning <https://bitbucket.org/> et la gestion du coté client avec <https://www.sourcetreeapp.com>.
11. Je suis capable de faire le Frontend d'un site web avec la techno Twig.
12. Je suis capable de d'utiliser la techno Bootstrap : <http://getbootstrap.com/>
13. Je suis capable de d'utiliser la techno Fondation : <http://foundation.zurb.com/>
14. Je suis capable de résoudre un algorithme compliqué avec les données mathématique.
15. Je suis capable de créer les différentes versions du site en ligne et la redirection.
16. Je suis capable de monter un site d'application web avec JAVA/J2EE.
17. Je suis capable de faire la documentation d'un API Template Slate avec Ruby.
18. Je suis capable de faire une application mobile Android.
19. Je suis capable de monter un site en PHP avec l'ORM Propel.
20. Je suis capable de monter un site avec le PHP Framework Zend.
21. Je suis capable de monter un site avec CMS WordPress.
22. Je suis capable de faire le Dashboard d'un client ou un Marchand avec la techno JSP.
23. Je suis capable de faire la maintenance d'un ordinateur (détection des problèmes, installation des différentes cartes ...).
24. Je suis capable de créer une base de données sous Access.
25. Je suis capable de créer une base de données et d'interroger la base via des requêtes sous Oracle.
26. Je suis capable de mettre en place le câblage d'un réseau Ethernet.
27. Je suis capable de créer des routines SQL (procédures stockées et triggers).
28. Je suis capable de développer des applications orientées objet principalement en langage JAVA.

29. Je suis capable d'implémenter dans un système de gestion de base de données (MYSQL, MSSQL SERVER 2005) déjà conçue au préalable à l'aide de la méthode MERISE, ceci en utilisant le langage SQL.

30. Je suis capable de réaliser une application autonome à l'aide du langage DELPHI, synchronisant avec une base de données.

31. Je suis capable de programmer dans des langages de base tels que : PASCAL, C compte tenu de mon raisonnement logique et algorithmique.

32. Je suis capable d'intervenir dans la réalisation de la structuration et la mise en forme du frontend d'un site web à l'aide des langages HTML5 et CSS3.

33. Je suis capable de rendre plus dynamique l'interface de présentation d'un site web à l'aide du langage JavaScript ainsi que de son Framework JQuery.

34. Je suis capable d'intervenir dans le rendu responsif de l'interface d'un site web.

35. Je suis capable d'installer et de configurer un serveur apache sous les différentes plateformes : Windows 10, Linux (Ubuntu), Mac OSX.

36. Je suis capable de configurer les hôtes virtuels sur un serveur apache (permettant le lancement de plusieurs sites sur le même serveur).

37. Je suis capable d'interagir avec un serveur FTP via un client FTP (FileZilla).

38. Je suis capable de rédiger la documentation des applications que je réalise.

39. Je suis capable de rédiger des rapports relatifs à un projet, à un stage.

40. Je suis capable de m'exprimer clairement afin de résumer un projet sur lequel j'ai travaillé.

41. Je suis capable d'utiliser le langage de métadonnées XML pour décrire les données.

42. Je suis capable d'intervenir dans le développement du backend d'un site en utilisant le langage PHP (voir en PHP orienté objet)

43. Je suis capable d'utiliser un ORM (Object Relationnel Mapping) pour la génération du modèle relatif à la partie métier de l'application.

44. Je suis capable d'apprendre n'importe quel langage grâce à mon sens algorithmique.

45. Je suis capable de gérer les stagiaires qui travaillent avec l'un des langages dont j'ai déjà utilisé.

46. Je suis capable d'utiliser n'importe quel Framework aujourd'hui grâce à ma connaissance sur POO et MVC.

47. Je suis capable d'intégrer n'importe quel API sur un site grâce à la mise en place de l'API de WeCashUp.

48. Je suis capable de faire les différentes cartes de vœux avec l'outil de Microsoft PUBLISHER.

49. Je suis capable d'utiliser les différentes formules de calcul avec l'outil de Microsoft XCL.

50. Je suis capable de faire un Bilan d'entrée et sortie d'une PME grâce à ma formation de la comptabilité de gestion.

COMPETENCE PROFESSIONNELLE N°1

Monter une API avec Google cloud

Exercée en 2017

Projet d'alternance	API déployée
<p>A1. Besoin de l'entreprise</p> <p>Mettre en place une API pour les marchands</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur l'inscription des marchands et la connexion.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Informations = Cahier des charges fournis par le chef, Planning du projet.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Vérifié que les données transmises par les marchands sont sécurisées.</p> <p>B2. Résultat obtenu</p> <p>Succès</p> <p>B3. Ce que les autres ont dit de mes résultats</p> <p>De continuer à améliorer au fur à mesure que les données arrivent.</p>

COMPETENCE PROFESSIONNELLE N°2

DES requêtes en NoSQL/Objectify

Exercée en 2017

Projet d'alternance	Save, update, filtre ...
<p>A1. Besoin de l'entreprise</p> <p>Sauvegarder, update, recherche, indexation des données.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur l'inscription des marchands, client et la connexion.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Informations = Cahier des charges fournis par le chef, Planning du projet.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Fait les sauvegardes, les recherches et le filtre selon les besoins.</p> <p>B2. Résultat obtenu</p> <p>La recherche et le filtre sont partiellement finis.</p> <p>B3. Ce que les autres ont dit de mes résultats</p> <p>Mon tuteur et le reste de l'équipe étaient vraiment content de mon travail vu que c'était ma première fois de travailler avec objectify.</p>

COMPETENCE PROFESSIONNELLE N°3

Déploiement du projet en ligne

Exercée en 2017

Projet d'alternance	Réalisé
<p>A1. Besoin de l'entreprise</p> <p>De déployer la première version de projet et de dupliquer le service, un pour les marchand et l'autre pour le test.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur la gestion de la branche developpe.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Toujours me déconnecté avant de sortir pour mesure de sécurité.</p> <p>Avoir connaissance de Cloud.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Été chargé de déployer au fur à mesure qu'on développe, sur le service test.</p> <p>B2. Résultat obtenu</p> <p>Déployé avec succès.</p> <p>B3. Ce que les autres ont dit de mes résultats</p> <p>De contrôler le code qu'il n'y a pas d'erreur avant le déployer.</p>

COMPETENCE PROFESSIONNELLE N°4

Intégration des différents codes sur bitbucket/source tree

Exercée en 2017

Projet d'alternance	Codes vérifiés et pushés
<p>A1. Besoin de l'entreprise</p> <p>Récupération de tous les codes sur developpe, vérifier et de pusher sur master</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Récupération des différents codes, de les tester et pusher sur master avec source tree.</p>
<p>A2. Travail que globalement</p> <p>J'ai travaillé sur la gestion de la branche developpe et master.</p>	<p>B2. Résultat obtenu</p> <p>Je continue toujours à récupérer les données sur la branche developp et jusqu'à présent tout se passe bien.</p>
<p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Accès aux données sur bitbucket.</p> <p>Connaitre le principe de Git.</p>	<p>B3. Ce que les autres ont dit de mes résultats</p> <p>Les développeurs ne se sont jamais plaints de conflit, quand ils font un push ou pull.</p>

COMPETENCE PROFESSIONNELLE N°5

Algorithme de détection...

Exercée en 2017

Projet d'alternance	DéTECTÉ
<p>A1. Besoin de l'entreprise</p> <p>Parmi l'ensemble des données pour détecter la plus fiable, selon l'ensemble des critères qui varient.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur algorithme.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Accès aux données.</p> <p>Avoir une idée sur les matrices.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Adapter l'algorithme au changement des variable.</p> <p>B2. Résultat obtenu</p> <p>Réalisé.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>L'algorithme n'est pas encore mis en service.</p>

COMPETENCE PROFESSIONNELLE N°6

Un site web avec J2EE/JDBC

Exercée en 2017

Projet personnel	Encours
<p>A1. Besoin de l'entreprise Faire un site d'e-commerce</p> <p>A2. Travail que globalement Le projet complet.</p> <p>A3. Moyens et contraintes Matériels = Ordinateur Portable, accès internet. Cahier de charge et le tuto à suivre. Avoir une base solide en java classique.</p>	<p>B1. Chronologiquement, j'ai : Travaillé sur l'ensemble de projet, le Backend avec J2EE et le frontend avec la techno JSP.</p> <p>B2. Résultat obtenu Réalisé.</p> <p>B3. Ce que les autres ont dit de mes résultats : Parfait.</p>

COMPETENCE PROFESSIONNELLE N°7

Documentation de l'API/Template Slate avec Ruby

Exercée en 2017

Projet d'alternance	Réalisé
<p>A1. Besoin de l'entreprise</p> <p>Faire la documentation bien détaillée en français et en anglais.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur la version française.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Maitrise de HTML5, CSS3.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Adapter au changement et l'amélioration du projet.</p> <p>B2. Résultat obtenu</p> <p>La documentation est mise en place.</p> <p>B3. Ce que les autres ont dit de mes résultats</p> <p>De l'améliorer au fil d'avancement du projet.</p>

COMPETENCE PROFESSIONNELLE N°8

Application mobile Android

Exercée en 2016

Projet d'alternance	Facilité l'expérience utilisateur
<p>A1. Besoin de l'entreprise</p> <p>Faire une application mobile qui permet aux utilisateurs d'enregistre.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur la version française.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Avoir une base en Java.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Adapter au changement et l'amélioration du projet.</p> <p>B2. Résultat obtenu</p> <p>La documentation est mise en place.</p> <p>B3. Ce que les autres ont dit de mes résultats</p> <p>De l'améliorer au fil d'avancement du projet.</p>

COMPETENCE PROFESSIONNELLE N°9

Site web : vente aux enchères avec PHP et l'ORM Propel.

Exercée en 2016

Projet de la fac	Site en ligne.
<p>A1. Besoin de l'entreprise Faire un site web de vente aux enchères.</p> <p>A2. Travail que globalement J'ai travaillé sur la totalité de projet.</p> <p>A3. Moyens et contraintes Matériels = Ordinateur Portable, accès internet. Cahier de charge.</p>	<p>B1. Chronologiquement, j'ai : Faire en sorte qu'il y ait un gagnant.</p> <p>B2. Résultat obtenu Le site est déjà en ligne.</p> <p>B3. Ce que les autres ont dit de mes résultats De l'améliorer le projet.</p>

COMPETENCE PROFESSIONNELLE N°10

Site web : une plateforme pour les PME qui leurs permettent de vendre leur produit en ligne, sans avoir leur propre site, en PHP Framework Zend.

Exercée en 2017

Projet de l'institut G4	Encours de développement
<p>A1. Besoin de l'entreprise</p> <p>Faire une plateforme de vente pour les PME.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur l'inscription des particulier, galerie des mini-site, et la gestion de panier.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Cahier de charge.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Fait du développement.</p> <p>B2. Résultat obtenu</p> <p>65% développé.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p>

COMPETENCE PROFESSIONNELLE N°11

Mettre en place le Dashboard des marchands

Exercée en 2017

Projet d'alternance.	Dashboard effectué
<p>A1. Besoin de l'entreprise</p> <p>Permettre aux marchands de voir les transactions réelles, le montant global, et le nombre de leurs clients.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur le frontend avec la techno JAVA ET JSP.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Cahier de charge.</p> <p>Avoir une bonne base en langage JAVA</p>	<p>B1. Chronologiquement :</p> <p>Se rassurer que les données soient mises à jours automatiquement.</p> <p>B2. Résultat obtenu</p> <p>Aujourd'hui les marchands voient leur transaction réelle.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>Mon tuteur et mon chef d'entreprise était totalement satisfait de mon travail compte tenu de retour des marchands.</p>

COMPETENCE PROFESSIONNELLE N°12

Monter un site avec CMS WordPress.

Exercée en 2017

Projet d'un client personnel	Site déjà en ligne.
<p>A1. Besoin de l'entreprise</p> <p>Mettre en place un site d'information sur la maladie drépanocytose.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé la personnalisation du site.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Cahier de charge.</p> <p>Un niveau minimal en CSS et HTML5</p>	<p>B1. Chronologiquement :</p> <p>J'ai personnalisé le thème et l'ajout des articles.</p> <p>B2. Résultat obtenu</p> <p>Développé avec succès.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>Le client est satisfait de mon travail, qui m'avait même proposé d'autre site mais je n'ai pas eu le temps.</p>

COMPETENCE PROFESSIONNELLE N°13

Une application en respectant MVC.

Exercée en 2016

Projet tutoriel de la fac	Besoin réalisé.
<p>A1. Besoin de l'entreprise</p> <p>Au cours de la réalisation d'applications en PHP et JAVA dans le cadre scolaire il m'a été demandé de respecter une organisation de mon code selon le pattern MVC.</p> <p>A2. Travail que globalement</p> <p>Avantages d'une architecture MVC :</p> <p>L'application doit pouvoir être implantée facilement dans un autre langage que celui prévu au départ, vu que le modèle métier ne changera pas.</p> <p>La maintenance et l'évolutivité de l'application s'avèrent plus simple. Aussi la réutilisabilité des parties de l'application reste envisageable.</p> <p>A3. Moyens et contraintes</p> <p>Il faudrait au préalable avoir quelques notions en programmation orientée objet avant de se lancer dans cette architecture.</p> <p>Cette architecture nécessite réellement un temps important d'analyse du problème posé.</p> <p>Un modèle MVC ne saurait être parfait dès le départ mais peut servir de bonne base pour l'organisation future du code et peut aussi être modifié au fur et à mesure que l'on développe l'application.</p>	<p>B1. Chronologiquement :</p> <p>J'ai analysé le problème, capturer les besoins fonctionnels.</p> <p>J'ai ressorti le modèle métier de l'application sous forme de diagramme de classe compte tenu grand lien entre le MVC et la programmation orienté objet.</p> <p>J'ai imaginé les différentes vues que comportera l'application et imaginer la relation entre celles-ci et le modèle métier.</p> <p>C'est ainsi que j'ai obtenu les trois de mes applications : Modèle-Vue-Contrôleur.</p> <p>B2. Résultat obtenu</p> <p>A terme de mon travail j'ai conçu un modèle MVC répondant aux principales fonctionnalités des applications à réaliser.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>Mes professeurs trouvaient que j'ai atteint l'objectif visé de ce cours et que me faudra juste persévérer à l'avenir dans la pratique de cette architecture.</p>

COMPETENCE PROFESSIONNELLE N°14

Etablir un cahier de charge relatif à un besoin concret.

Exercée en 2015

Projet de fin d'étude BTS	Testé et validé par le tuteur.
<p>A1. Besoin de l'entreprise</p> <p>Réaliser une application permettant le suivi des paiements des de travaux public.</p> <p>A2. Travail que globalement</p> <p>L'objectif était d'automatiser les demandes venant du ministère et suivi des paies.</p> <p>A3. Moyens et contraintes</p> <p>Je n'avais pas accès à certaines données.</p>	<p>B1. Chronologiquement :</p> <p>J'ai pris connaissance du besoin de l'entreprise tout en dialoguant avec tous les postes de travail concernés par l'application.</p> <p>Ainsi j'ai pris des notes sur les spécifications et les fonctionnalités attendues.</p> <p>Afin élaboré un cahier de charge et de valider ce dernier auprès du service de la comptabilité afin de pouvoir apporter des modifications si besoin.</p> <p>B2. Résultat obtenu</p> <p>Le travail n'est pas totalement fini.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>Mon tuteur a pris un autre stagiaire qu'il puisse continuer avec le projet, vu que mon délai de stage était à terme.</p>

COMPETENCE PROFESSIONNELLE N°15

Rendre responsif le site enchère avec Bootstrap

Exercée en 2015.

Projet de la fac	Réalisé
<p>A1. Besoin de l'entreprise</p> <p>Le site devrait s'adapter face aux différents écrans utilisé par les user.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur le Frontend précisément rendre le site responsif.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Toujours me déconnecté avant de sortir pour mesure de sécurité.</p> <p>Avoir connaissance HTML5/CSS3.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>J'ai lu un tuto sur Bootstrap pour comprendre l'utilisation de grills.</p> <p>B2. Résultat obtenu</p> <p>Le site est responsif.</p> <p>B3. Ce que les autres ont dit de mes résultats</p> <p>Les professeurs étaient contents 😊 de mon travail.</p>

COMPETENCE PROFESSIONNELLE N°16

Rendre le une page responsive avec FOUNDATION

Exercée en 2015.

Projet de la fac	Réalisé
<p>A1. Besoin de l'entreprise</p> <p>Le but de projet était de faire nos CV avec HTML5/CSS3 et le rendre responsive avec FOUNDATION.</p> <p>A2. Travail que globalement</p> <p>J'ai travaillé sur le Frontend précisément rendre le site responsive.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Toujours me déconnecter avant de sortir pour mesure de sécurité.</p> <p>Avoir connaissance de HTML5/CSS3.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>J'ai lu un tuto sur FOUNDATION pour comprendre l'utilisation de grilles.</p> <p>B2. Résultat obtenu</p> <p>Travail rendu.</p> <p>B3. Ce que les autres ont dit de mes résultats</p> <p>Les professeurs étaient contents 😊 de mon travail.</p>

COMPETENCE PROFESSIONNELLE N°17

Interagir avec un server via FTP

Exercée en 2017.

Projet d'alternance	Travail fait
<p>A1. Besoin de l'entreprise</p> <p>Mettre en ligne la documentation de l'API sur le server OVH via FTP(FilZilla).</p> <p>A2. Travail que globalement</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>J'ai installé le client FTP (FileZilla).</p> <p>B2. Résultat obtenu</p> <p>En ligne.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>L'équipe m'a remercié compte tenu de retour des développeurs.</p>

COMPETENCE PROFESSIONNELLE N°18

Capable d'écrire diagramme de classe, cas d'utilisation, diagramme de séquence à partir d'un besoin concret avec UML

Exercée en 2016.

Stage de fin de formation L3	Travail fait
<p>A1. Besoin de l'entreprise</p> <p>Mettre en place une application Android.</p> <p>A2. Travail que globalement :</p> <p>On m'a demandé de faire cette application pour les clients afin de faciliter l'expérience utilisateur.</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Android studio.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Pour arriver à mettre en place cette application, j'ai fait la partie conception avec UML.</p> <p>B2. Résultat obtenu</p> <p>Le travail a été effectué avec succès, ce qui m'a permis de développer aisément.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>Le tuteur était satisfait.</p>

COMPETENCE PROFESSIONNELLE N°19

Capable de créer une base de données sous Oracle et d'interroger la base avec des requête selon les besoins

Exercée en 2015.

Projet tutoriel de fac	Travail fait
<p>A1. Besoin de l'entreprise</p> <p>Faire un requêteur universel, avec l'interface swing JAVA, qui peut interroger n'importe quelle Base de données.</p> <p>A2. Travail que globalement :</p> <p>J'ai travaillé sur l'ensemble de projet</p> <p>A3. Moyens et contraintes</p> <p>Matériels = Ordinateur Portable, accès internet.</p> <p>Avoir Oracle installé.</p>	<p>B1. Chronologiquement, j'ai :</p> <p>Utilisé Swing pour l'interface utilisateur, et de JDBC pour la connexion à Oracle.</p> <p>B2. Résultat obtenu :</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>De continuer à utiliser Oracle même avec mes projets, pour avoir plus de notion.</p>

COMPETENCE PROFESSIONNELLE N°20

Faire la conception d'une application avec Merise

Exercée en 2015.

Projet de fin d'étude BTS	Testé et validé par le tuteur.
<p>A1. Besoin de l'entreprise</p> <p>Réaliser une application permettant le suivi des paiements des de travaux public.</p> <p>A2. Travail que globalement</p> <p>L'objectif était d'automatiser les demandes venant du ministère et suivi des paies.</p> <p>A3. Moyens et contraintes</p> <p>Ordinateur, accès internet, JMERISE installé.</p>	<p>B1. Chronologiquement :</p> <p>J'ai pris connaissance du besoin de l'entreprise tout en dialoguant avec tous les postes de travail concernés par l'application.</p> <p>Ainsi j'ai pris des notes sur les spécifications et les fonctionnalités attendues.</p> <p>Afin de de faire la conception facilement.</p> <p>B2. Résultat obtenu</p> <p>La conception a été effectué avec succès.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>Tuteur de stage a évalué mon travail d'un travail professionnel.</p>

COMPETENCE PROFESSIONNELLE N°21

Rédiger la documentation des applications que je réalise.

Exercée en 2015.

A faire dans tous mes projets Java	Effectué
<p>Besoin de l'entreprise :</p> <p>Bien documenter ses applications en commentant son code est une très bonne pratique aussi bien au niveau scolaire que professionnel</p> <p>Travail que globalement :</p> <p>Le travail consistait à bien commenter son code lors des différents projets de développement abordé (JAVA, PHP).</p> <p>Moyens et contraintes :</p> <p>Ordinateur, IDE eclipse...</p>	<p>Chronologiquement, j'ai :</p> <p>J'ai créé les classes, attributs, et méthode tout en y ajoutant des commentaires de type Java doc qui me permettra de générer ma Java doc plus tard.</p> <p>Génération de la Java doc à partir de tous les commentaires ajoutés.</p> <p>Faire des fiches techniques résumant des points assez particuliers dans le développement de l'application.</p> <p>Résultat obtenu : Applications assez compréhensible grâce à la documentation</p> <p>Ce que les autres ont dit de mes résultats :</p>

COMPETENCE PROFESSIONNELLE N°22

Faire la conception d'une application avec Merise

Exercée en 2015.

Projet de fin d'étude BTS	Testé et validé par le tuteur.
<p>A1. Besoin de l'entreprise</p> <p>Réaliser une application permettant le suivi des paiements des de travaux public.</p> <p>A2. Travail que globalement</p> <p>L'objectif était d'automatiser les demandes venant du ministère et suivi des paies.</p> <p>A3. Moyens et contraintes</p> <p>Ordinateur, accès internet, JMERISE installé.</p>	<p>B1. Chronologiquement :</p> <p>J'ai pris connaissance du besoin de l'entreprise tout en dialoguant avec tous les postes de travail concernés par l'application.</p> <p>Ainsi j'ai pris des notes sur les spécifications et les fonctionnalités attendues.</p> <p>Afin de de faire la conception facilement.</p> <p>B2. Résultat obtenu</p> <p>La conception a été effectué avec succès.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p> <p>Tuteur de stage a évalué mon travail d'un travail professionnel.</p>

COMPETENCE PROFESSIONNELLE N°23

Faire une application avec le langage DELPHI

Exercée en 2015.

Projet de fin d'étude BTS	Testé et validé par le tuteur.
<p>A1. Besoin de l'entreprise</p> <p>Réaliser une application permettant le suivi des paiements des de travaux public.</p> <p>A2. Travail que globalement</p> <p>L'objectif était de faire l'application avec DELPHI et SQL SERVER.</p> <p>A3. Moyens et contraintes</p> <p>Ordinateur, accès internet, DELPHI 2012.</p>	<p>B1. Chronologiquement :</p> <p>J'ai pris connaissance du besoin de l'entreprise tout en dialoguant avec tous les postes de travail concernés par l'application.</p> <p>Ainsi j'ai pris des notes sur les spécifications et les fonctionnalités attendues.</p> <p>Afin illustrer les différentes entités à créer.</p> <p>B2. Résultat obtenu</p> <p>La conception a été effectué avec succès.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p>

COMPETENCE PROFESSIONNELLE N°24

Faire les cartes de vœux avec PUBLISHEUR

Exercée en 2014.

Formation	Travail fait
<p>A1. Besoin de l'entreprise Faire une carte de vœux de mariage</p> <p>A2. Travail que globalement</p> <p>A3. Moyens et contraintes Ordinateur, accès internet.</p>	<p>B1. Chronologiquement : J'ai réalisé mes tâches demandées.</p> <p>B2. Résultat obtenu La réalisation a été effectué avec succès.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p>

COMPETENCE PROFESSIONNELLE N°25

Mise en œuvre d'un réseau Ethernet.

Exercée en 2014.

Formation privé	Travail fait
<p>A1. Besoin de l'entreprise Connecter quelque bureau pour la transmission des données.</p> <p>A2. Travail que globalement</p> <p>A3. Moyens et contraintes Ordinateur, accès internet.</p>	<p>B1. Chronologiquement : J'ai réalisé mes tâches demandées.</p> <p>B2. Résultat obtenu La réalisation a été effectué avec succès.</p> <p>B3. Ce que les autres ont dit de mes résultats :</p>

Bibliographie/Webographie :

1. <http://www.marseille-innov.org/notre-histoire/>
2. <http://business-en-afrique.net/mobile-money-en-afrique-m-pesa-cie/>
3. <http://maven-guide-fr.erwan-alliaume.com/maven-guide-fr/site/reference/introduction.html>
4. [https://fr.wikipedia.org/wiki/Intégrité \(cryptographie\)](https://fr.wikipedia.org/wiki/Int%C3%A9grit%C3%A9_(cryptographie))
5. <http://brocoli.developpez.com/articles/presentation/google-app-engine/>
6. <https://openclassrooms.com/courses/montez-votre-site-dans-le-cloud-avec-google-app-engine/manipuler-le-datastore-avec-objectify>

Les Annexes :

Annexe1 : Git avec le client SourceTree

0

Pulling remote changes from the remote **develop** branch

```
$ git pull
```

1

Creating a new feature

creating a local newfeature branch from the local develop branch

Create and switch to the new branch "newfeature"

```
$ git checkout -b newfeature develop
```

Edit and commit changes on the **newfeature** branch

Edit some files in the local project then commit

```
$ git commit -am "Started newfeature"
```

Edit some files in the local project again and commit

```
$ git commit -am "Finished newfeature"
```

2

Incorporating a finished feature on **develop**

Finished features may be merged into the **develop** branch to definitely add them to the upcoming release:

Switch to the local branch 'develop'

```
$ git checkout develop
```

if there are changes on the remote server (origin), pull them down

```
$ git pull
```

Merge the 'newfeature' branch into the current one ('develop') and add a merging message 'adding newfeature'

```
$ git merge --no-ff newfeature --m "adding newfeature"
```

Delete the branch newfeature

```
$ git branch -d newfeature
```

Push the local develop branch (with the new feature) to the remote develop branch

```
$ git push origin develop
```

3

Creating a release branch

Release branches are created from the `develop` branch. For example, say version 1.1.5 is the current production release and we have a big release coming up. The state of `develop` is ready for the “next release” and we have decided that this will become version 1.2 (rather than 1.1.6 or 2.0). So we branch off and give the release branch a name reflecting the new version number:

Create and switch to a new branch "release-1.2"

```
$ git checkout -b release-1.2 develop
```

Bump (Update) project version to 1.2

```
$ ./bump-version.sh 1.2
```

Commit changes to the release-1.2 branch

```
$ git commit -a -m "Bumped version number to 1.2"
```

4

Finishing a release branch

a) Merge to `master`

Switch to branch 'master'

\$ git checkout master

Merge the release-1.2 branch to Master if everything is ready to go to production

\$ git merge --no-ff release-1.2 -m "merging release-1.2 to master"

Adding the release-1.2 tag to the master branch after merging it with the new release branch

\$ git tag -a 1.2 -m "release-1.2"

b) Merge to **develop**

Switch to branch 'develop'

\$ git checkout develop

Merge the release-1.2 branch to develop to include it in the next release

\$ git merge --no-ff release-1.2 -m "merging release-1.2 to develop"

c) Delete **release-1.2**

\$ git branch -d release-1.2

5

Creating the hotfix branch

Hotfix branches are created from the master branch. For example, say version 1.2 is the current production release running live and causing troubles due to a severe bug. But changes on develop are yet unstable. We may then branch off a hotfix branch and start fixing the problem:

Create a new branch "hotfix-1.2.1" from master and switch to it

\$ git checkout -b hotfix-1.2.1 master

Bump (Update) project version to 1.2.1

\$./bump-version.sh 1.2.1

Commit changes to the hotfix-1.2.1 branch

\$ git commit -a -m "Bumped version number to 1.2.1"

After fixing all the bugs, commit changes to the hotfix-1.2.1 branch

\$ git commit -m "Fixed severe production problem"

6

Finishing a hotfix branch

a) Merge to **master**

Switch to branch 'master'

\$ git checkout master

Update the local branch 'master' with the remote one

\$ git pull

Merge the 'hotfix-1.2.1' branch with 'master'

\$ git merge --no-ff hotfix-1.2.1 -m "merging hotfix-1.2.1 to master"

Tag the 'master' branch with hotfix-1.2.1

\$ git tag -a 1.2.1 -m "hotfix-1.2.1"

b) Merge to **develop**

Switch to branch 'develop'

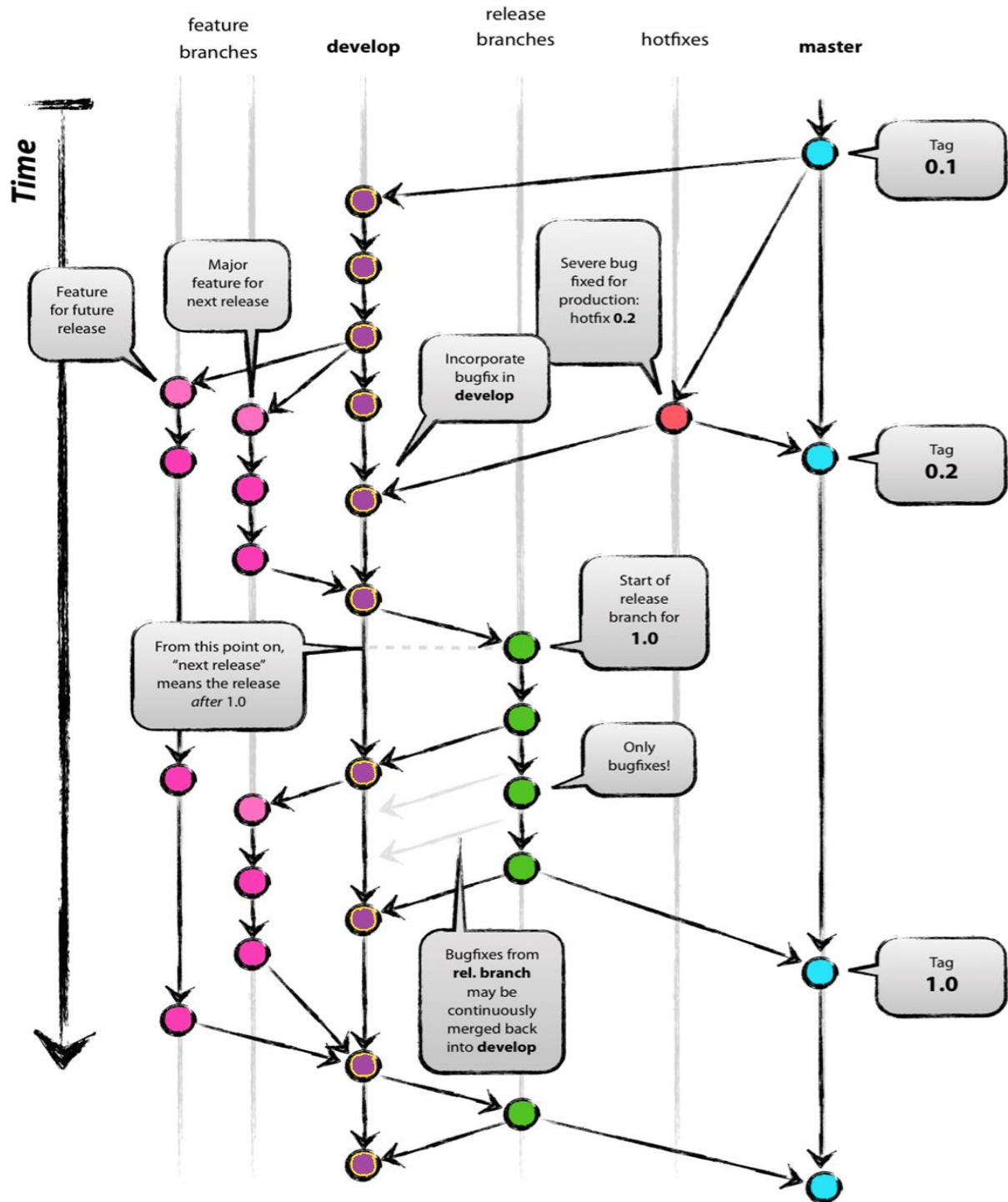
\$ git checkout develop

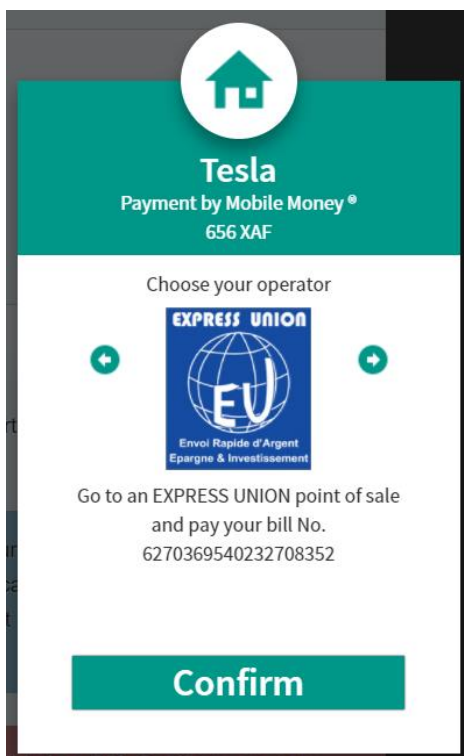
Merge the last hotfix-1.2.1 branch to develop to include it in the next release

\$ git merge --no-ff hotfix-1.2.1 -m "merging release-1.2 to develop"

c) Delete the **hotfix-1.2.1** branch

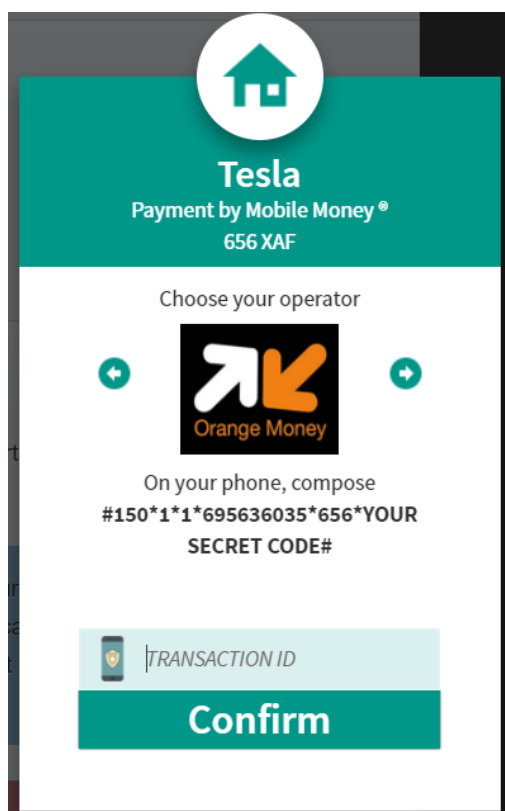
\$ git branch -d hotfix-1.2.1





The screenshot shows a mobile payment screen for Tesla. At the top, there is a house icon in a circle. Below it, the text reads "Tesla", "Payment by Mobile Money®", and "656 XAF". The main section is titled "Choose your operator" and features the "EXPRESS UNION" logo, which includes a globe and the text "EU", "Envoi Rapide d'Argent", and "Epargne & Investissement". Below the logo, the instructions are: "Go to an EXPRESS UNION point of sale and pay your bill No. 6270369540232708352". At the bottom, there is a large green button labeled "Confirm".

Annexe3 : Mode télécom



The screenshot shows a mobile payment screen for Tesla. At the top, there is a house icon in a circle. Below it, the text reads "Tesla", "Payment by Mobile Money®", and "656 XAF". The main section is titled "Choose your operator" and features the "Orange Money" logo, which includes an orange arrow and the text "Orange Money". Below the logo, the instructions are: "On your phone, compose #150*1*1*695636035*656*YOUR SECRET CODE#". At the bottom, there is a light blue box containing a smartphone icon and the text "TRANSACTION ID", followed by a large green button labeled "Confirm".

Annex4 : architecture App Engine.

Architecture

