

2016



Rapport de stage de Master M2 INFORMATIQUE

Sujet : Réalisation d'une application de gestion des commandes

Du 29 février au 2 septembre 2016

Stagiaire :

Loïc NOEL
N° étudiant : 32005337
loicnoel12@gmail.com
Tel : 0692 27 74 63

Responsable Pédagogique :

Frédéric MESNARD
frederic.mesnard@gmail.com

Tuteur d'entreprise :

Philippe STAMPA
philippe.stamp@veolia.com
Tel : 0262 90 25 24

Date d'envoi du document : 06/06/2016

Remerciements

Je tiens tout d'abord à remercier tout le personnel de Veolia Eau Réunion pour son accueil chaleureux, son soutien tout au long de mon stage et les diverses connaissances qu'ils ont partagé avec moi durant toute cette période.

Je tiens ensuite à remercier tout particulièrement les membres du service informatique, à savoir mon tuteur en entreprise M. Philippe STAMPA ainsi que M. Jérôme LINCAR pour leur disponibilité, leurs précieux conseils et leur bonne humeur au quotidien.

Je remercie également le personnel du service approvisionnement qui a également été très disponible pour répondre aux différentes questions que je me posais, dans le but de réaliser un outil performant, efficace et adapté à leur besoins spécifiques.

Enfin, je tiens à remercier les enseignants du Master 2 informatique qui m'ont permis de venir compléter ma formation d'ingénieur avec leurs cours. Ces connaissances complémentaires m'ont permis d'être encore plus performant lors de mon stage en entreprise et de trouver des solutions auxquelles je n'aurai peut-être pas pensé auparavant.

Résumé

Dans ce rapport de stage sont présentées les différentes étapes de conception et de réalisation d'une application web pour Veolia Eau Réunion. Cette application a pour but de permettre aux agents du service approvisionnement de gérer les commandes passées par toute la société. Elle doit implémenter les processus de suivi de commande classiques comme la gestion et la génération des bons de commandes ou encore la gestion des états pour permettre aux agents de connaître rapidement le statut d'une commande. L'application doit également permettre d'automatiser les relances aux fournisseurs et aux transitaire, dans le cas où elles sont en retard ou ne possèdent pas de date de mise à disposition. Cette application viendra remplacer celle qui est déjà utilisée par le service. Cette ancienne application, basée sur Microsoft Access 2003, est une solution provisoire, réalisée par un ancien membre du service approvisionnement. La solution étant obsolète et ne montant pas en charge, le but ici est de la remplacer pour permettre de gagner en productivité, gérer une plus grande quantité de commandes de manière plus fluide en utilisant un outil moderne, personnalisé et évolutif. Ce stage se déroule du 29 février au 2 septembre 2016 et n'est pas encore terminé au moment du rendu de ce rapport. Ainsi, celui-ci ne présente que l'évolution jusqu'à la version 1.2 de l'application.

Abstract

In this report are presented the different parts of the conception and realisation of a web application for the company Veolia Eau Réunion. This application has for objective to allow the agents of the provisioning service to manage the orders placed by the staff of the whole company in Reunion Island. It has to implement the typical order tracking processes, like the management and creation of the order forms, the management of the state of the orders. It also has to automate the sending of reminder emails to the supplier and the forwarder when the order is late or doesn't have a provisioning date. This application is going to replace an older one used by the service. This older application, based on Microsoft Access 2003, is a temporary solution, created by a former member of the provisioning service. The application is obsolete and not designed to deal with a big amount of orders. The aim of this internship was to replace it in order to increase productivity, manage a higher number of orders more fluently, using a modern, customized and evolutive tool. This internship is taking place from February 29th to September 2nd 2016 and is still not over when delivering this report. Thus, in this one are only presented the evolution of the application until version 1.2.

Table des matières

1.	Introduction	5
2.	Environnement	5
2.1.	L'entreprise	5
2.2.	L'équipe projet	6
2.3.	Communication au sein de l'équipe projet.....	7
2.4.	Communication hors équipe projet.....	7
2.5.	Ressources fournies et/ou à utiliser	7
2.6.	Contraintes particulières	7
3.	Conception et réalisation de l'application	8
3.1.	Description, résultats attendus et état d'avancement.....	8
3.2.	Étude du besoin.....	8
3.2.1.	Contexte	8
3.2.2.	Processus de commande	8
3.2.3.	Entretien avec le responsable approvisionnement.....	9
3.2.4.	Étude d'une copie de l'application	10
3.2.5.	Entretien avec la responsable Qualité Sécurité et Environnement	10
3.2.6.	Choix de développement.....	10
3.3.	Choix des technologies	11
3.3.1.	Choix du langage de programmation	11
3.3.2.	Choix du framework PHP	14
3.3.3.	Choix du système de gestion de bases de données (SGBD)	16
3.3.5.	Choix du framework CSS	19
3.3.4.	Développement et déploiement de l'application	20
3.4.	Rédaction du cahier des charges	22
3.5.	Modélisation de l'application.....	22
3.5.1.	Diagramme de cas d'utilisations	22
3.5.2.	Modélisation de la base de données de l'application	24
3.5.3.	Modélisation des classes de l'application	29
3.6.	Réalisation des maquettes	30
3.6.1.	Tableau de bord	31
3.6.2.	Commandes	32
3.6.3.	Destinataires.....	35
3.	Etablissement des versions.....	37
3.1.	Version 0.1 (prototype)	37
3.2.	Version 0.2 (version alpha).....	37
3.3.	Version 0.3 (version beta).....	38
3.4.	Version 1.0.....	38
3.5.	Version 1.1.....	38
3.6.	Version 1.2.....	38
4.	Phase de réalisation	39
4.1.	Prototype (version 0.1)	39
4.2.	Version Alpha (0.2).....	40
4.3.	Version Bêta (0.3).....	40
4.4.	Version 1.0.....	41
4.5.	Version 1.1.....	42
4.6.	Version 1.2.....	43
5.	Décomposition des tâches et sous-tâches	45
6.	Conclusion	46
7.	Sources	47
8.	Table des figures	49
9.	Lexique	50

1. Introduction

Dans le cadre de ma dernière année de Master Informatique à l'Université de La Réunion, je dois effectuer un stage de fin d'étude d'une durée de 6 mois. Ce stage vise à clôturer mon cursus. Il me permet d'être formé au sein d'une entreprise dans le but d'acquérir des connaissances sur un secteur d'activité, tout en me permettant de mettre en pratique les connaissances théoriques que j'ai acquise lors de mon cursus.

Dans ce rapport, je présente mon environnement de travail ainsi que la mission principale que j'ai réalisée au sein de la société Veolia Eau Réunion, à savoir la réalisation d'une application web de gestion des commandes. En effet, Veolia possède déjà un outil qui lui permet de gérer ses commandes, mais il s'agit d'une solution provisoire que l'entreprise a décidé de remplacer.

2. Environnement

2.1. L'entreprise



Figure 1 : Logotype Veolia Eau

Veolia est une entreprise multinationale présente sur 5 continents et comptant environ 174 000 salariés. En 2013, elle présente un chiffre d'affaire d'environ 24,96 milliards d'euros et célèbre ses 160 ans d'histoire cette même année. Anciennement appelée Vivendi Environnement ou encore Compagnie Générale des Eaux à son arrivée à La Réunion, Veolia conçoit et déploie des solutions pour la gestion de l'eau ainsi que la gestion de l'énergie et des déchets. Elle permet ainsi à ses clients de rester compétitifs sur ces mêmes domaines. Comme l'indique son slogan : « Ressourcer le monde », Veolia a pour objectif d'accompagner les industriels, les villes et leurs habitants dans la gestion optimisée des ressources pour augmenter le rendement économique et réduire l'impact de l'homme sur l'environnement.

L'entreprise Veolia est présente à La Réunion depuis 1976, et fête donc ses 40 ans de présence sur le département en 2016. La société s'occupe essentiellement du marché de l'eau potable et de l'assainissement dans plusieurs communes telles que Saint-Denis, Le Port, La Possession, L'Etang-Salé, Saint-Louis et Saint-Pierre.

Son siège se situe à Saint-Denis, où j'effectue mon stage. C'est ici que sont gérées les commandes passées pour l'ensemble de la société Veolia Eau Réunion.

2.2. L'équipe projet

L'équipe du projet est constituée de quatre personnes :

- Philippe STAMPA, responsable informatique de Veolia,
- Jérôme LINCAR, responsable informatique adjoint,
- Ludovic ALEXIS, responsable approvisionnement,
- Moi-même, Loïc NOEL, stagiaire ingénieur en informatique.

Mon tuteur en entreprise pendant ce stage est M. Philippe STAMPA. Les étapes de développement de l'application seront validées par M. Ludovic ALEXIS et M. Philippe STAMPA. Mes choix technologiques quant à eux, sont soumis à l'approbation de mon tuteur de stage en entreprise.

Les membres du service informatique sont là pour me permettre d'avoir accès à l'infrastructure informatique de l'entreprise, me conseiller sur les technologies à utiliser, m'apporter des pistes de réflexion et m'aider dans la réalisation de certaines fonctionnalités, notamment celles en rapport avec l'accès aux fichiers.

La figure ci-après renseigne plus de détails concernant les relations hiérarchiques au sein de l'entreprise.

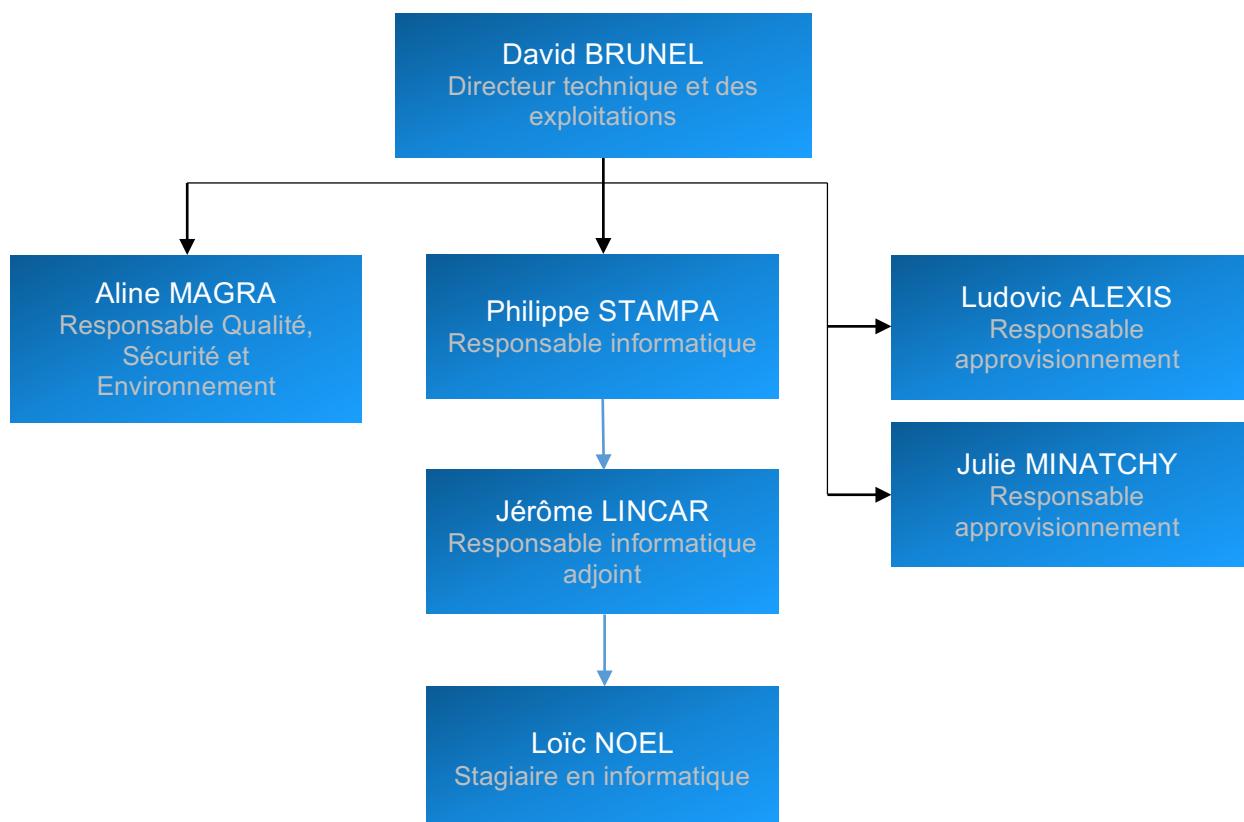


Figure 2 : Organigramme

Il s'agit ici d'un organigramme simplifié. La structure organisationnelle de Veolia étant beaucoup plus complexe, cette figure a pour but de mieux situer les liens de hiérarchies entre les personnes qui sont intervenues dans le projet.

2.3. Communication au sein de l'équipe projet

Les membres de l'équipe projet se trouvant au même étage de l'établissement, la communication verbale est le moyen de communication privilégié. L'équipe étant très disponible, il est aisément d'obtenir des réponses rapides à des questions ponctuelles ainsi que d'obtenir des entretiens voire même des réunions. Autrement, il m'a été possible de demander des informations par mail à l'ensemble de l'équipe projet. Pour des questions simples et nécessitant une réponse rapide, notamment les questions concernant certains détails de l'application Access déjà en place ou portant sur des points de procédure lors de la gestion des commandes. Lorsque les personnes concernées n'étaient pas présentes, il a été possible de les appeler par téléphone.

2.4. Communication hors équipe projet

J'ai voulu intégrer au projet des notions comme la qualité et le respect de l'environnement, ce qui a nécessité que je me rapproche de la responsable Qualité Sécurité et Environnement de Veolia. J'ai également dû me rapprocher d'autres personnes du service approvisionnement, pour récupérer leurs besoins et leurs éventuelles idées pour me permettre d'améliorer mon application. La communication hors de l'équipe projet se fait de la même manière qu'à l'intérieur de l'équipe projet. La communication sous forme d'entretiens et de réunion est privilégiée, cependant les mails et le téléphone sont aussi utilisés pour échanger avec les membres ne faisant pas partie de l'équipe projet.

2.5. Ressources fournies et/ou à utiliser

Plusieurs ressources m'ont été fournies dans le cadre de mon stage. Dans un premier temps, un PC a été mis à ma disposition, avec le système d'exploitation Windows. Connecté au réseau de Veolia, celui-ci m'a permis d'accéder au dossier contenant les informations du service approvisionnement ainsi que l'application développée sous Microsoft Access (disponible uniquement sous le système d'exploitation de Microsoft).

J'ai également eu un accès en RDP au serveur sur lequel j'allais développer l'application, ainsi qu'un accès au service IIS installé sur ce serveur.

2.6. Contraintes particulières

Le service informatique voulant pouvoir effectuer une maintenance de l'application de manière centralisée en cas de besoin et ne souhaitant pas installer l'outil sur tous les postes, la contrainte principale a été de réaliser une application web. Celle-ci, une fois installée sur un serveur, pourra être maintenue facilement et ne nécessitera pas d'être redéployée à chaque mise à jour. De plus, la réalisation d'une application web sur un serveur interne à l'entreprise entre dans la démarche de respect de l'environnement qui sera abordé dans la section 3.2.6.

3. Conception et réalisation de l'application

3.1. Description, résultats attendus et état d'avancement

La phase de conception de l'application est la plus importante du projet. Elle consiste à recueillir les besoins des agents, comprendre l'outil déjà en place, modéliser l'application et préparer son développement, de manière à ce qu'elle s'intègre correctement dans le système informatique de Veolia. A l'issue de cette phase de conception, nous devrons avoir un cahier des charges fonctionnel. Nous devrons également avoir des documents de comparatifs concernant les langages et les frameworks disponibles et expliquant les critères de choix d'une technologie par rapport aux autres. Nous pourrons également prévoir plusieurs autres documents qui expliqueront le fonctionnement de l'application, les technologies utilisées ainsi qu'un document résumant la phase de modélisation de la base de données.

La phase de réalisation quant à elle, sera essentiellement centrée sur l'implémentation des versions de l'application, les difficultés rencontrées pour chacune d'entre elles et les solutions trouvées aux problèmes posés.

3.2. Étude du besoin

3.2.1. Contexte

Les agents du service approvisionnement de Veolia utilisent un outil obsolète et réalisé par un agent non-spécialiste de l'informatique. Cet outil est en fait une application réalisée grâce au logiciel Microsoft Access 2003.

L'application fonctionne mais la logique qui est utilisée dans l'interface est assez complexe. De même, la modélisation de la base de données est clairement à revoir, car elle ne permet pas de stocker les commandes et les informations qui leurs sont relatives de manière optimisée. En effet, il s'agit d'une solution provisoire qui a été développée dans le but de faciliter le travail de l'ancien responsable approvisionnement. Cependant, après le départ de l'ancien responsable, l'application est restée dans le service car étant le seul outil informatique disponible pour gérer les commandes.

3.2.2. Processus de commande

Dans le but de mieux appréhender le besoin, j'ai dû me pencher sur l'application déjà présente dans le service. En effet, pour remplacer celle-ci de manière efficace et faire gagner du temps aux personnel du service approvisionnement, les premières versions de mon application devront implémenter les mêmes fonctionnalités que l'outil déjà utilisé.

Ainsi donc, dès le premier jour, les agents du service approvisionnement m'ont montré l'interface de l'application et m'ont expliqué comment celle-ci fonctionne dans les détails. Pour cela, ils m'ont expliqué toute la démarche de saisie des commandes, de génération des bons de commande et d'envoi de mails de relances aux fournisseurs et aux transitaires.

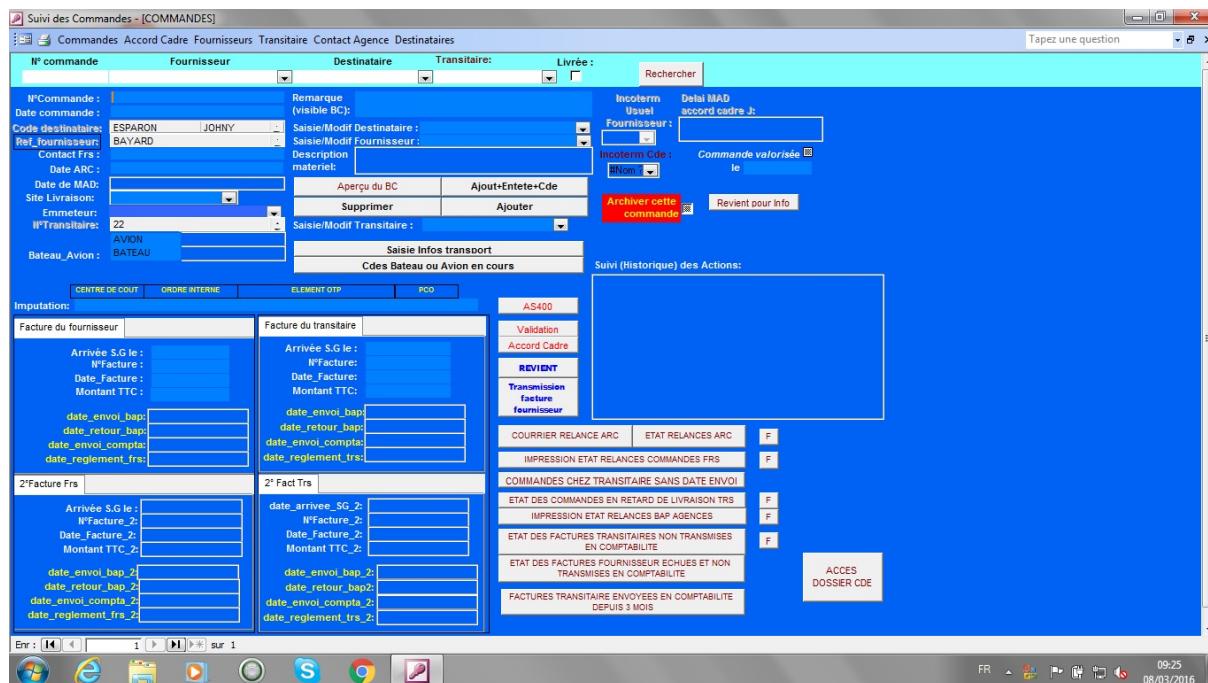


Figure 3 : Interface principale du formulaire de la base commande sous Microsoft Access

Ci-dessus, une capture d'écran de l'interface principale de l'application. Celle-ci va à l'encontre de nombreux critères d'ergonomie évidents (comme par exemple éviter surcharge cognitive), rendant encore plus difficile la prise en main et la compréhension de la logique générale.

De plus, cette application étant spécifique au service, elle comporte de nombreux termes techniques. Ainsi, pour une personne n'étant pas habituée à la logique, au vocabulaire et autres abréviations (telles que MAD pour Mise à Disposition), la compréhension ainsi que la prise en main de l'outil relèvent de la première vraie difficulté du projet. J'ai donc dû me familiariser avec ce fonctionnement pour pouvoir proposer une alternative plus intuitive et efficace.

3.2.3. Entretien avec le responsable approvisionnement

Dans le but de comprendre un peu mieux le fonctionnement de l'application ainsi que les besoins du service, je me suis entretenu avec le responsable approvisionnement. Il fait partie des principaux utilisateurs de l'outil. Celui-ci m'a expliqué ses attentes concernant les résultats du projet, à savoir un outil fonctionnel, qui implémente, à minima, les mêmes fonctionnalités que celui déjà en place, et qui automatise les tâches telles que les relances par mails, pour ainsi permettre d'être plus efficace et d'éviter les oubli. En effet, ces oubli entraînent par la suite des retards dans les commandes et peuvent avoir un enjeu crucial en terme d'efficacité pour l'entreprise.

Les personnes faisant partie du service approvisionnement sont les principaux utilisateurs de l'application Access, il est donc nécessaire de dialoguer au maximum avec le responsable ainsi que les autres personnes du service pour comprendre comment s'organise leur outil, comment il doit être utilisé et dans quel contexte. De plus il est intéressant de repérer des points d'éventuelles anomalies pour ensuite pouvoir proposer des améliorations. Cela permettra également de repérer les idées éventuelles qui pourraient permettre d'améliorer le produit final.

3.2.4. Étude d'une copie de l'application

Dans un troisième temps, j'ai étudié une copie de l'application réalisée sous Microsoft Access. La prise en main du logiciel aide à la compréhension détaillée de son fonctionnement. De plus, l'application n'étant pas totalement verrouillée, il est possible de regarder à l'intérieur pour analyser le contenu des tables ainsi que les requêtes et avoir une idée plus précise de son fonctionnement. A noter également que le fait que cette application ne puisse pas être totalement verrouillée relève d'un problème important, notamment pour l'intégrité des données, la sécurité et la stabilité du logiciel. De plus, un des premiers constats lors de l'analyse en détail du contenu de l'application est que celle-ci est composée de nombreuses tables non-utilisées qui viennent l'encombrer et l'alourdir.

3.2.5. Entretien avec la responsable Qualité Sécurité et Environnement

Il s'agit ici d'une démarche personnelle. J'ai demandé un entretien avec la responsable Qualité Sécurité Environnement (QSE) de Veolia. En effet, Veolia est une entreprise qui a une forte implication au niveau de la qualité et de l'environnement. Le but de cet entretien était donc de rassembler le maximum d'informations sur les exigences et initiatives potentielles de Veolia en matière de qualité et de développement durable. Par la suite, ces informations seront analysées pour me permettre de proposer des axes et d'orienter mon développement de manière à respecter ces exigences et de m'inscrire dans la logique de ces initiatives.

Suite à mon entretien avec la responsable QSE, j'ai réussi à mieux cerner les initiatives de Veolia Eau Réunion en matière d'environnement. En effet, Veolia possède une politique qualité axée sur 4 points clés : le management de la qualité de service, la protection de l'environnement, la réduction de la consommation d'énergie et la sécurité. Une application développée au sein de cette entreprise devrait, dans l'idéal, s'inscrire dans les mêmes initiatives pour l'aider à atteindre les objectifs fixés. Cependant, il faut trouver les points sur lesquels axer le développement pour que celui-ci entre dans la logique qualité de Veolia.

3.2.6. Choix de développement

Toutes les initiatives prises pour le développement de l'application sont issues de la notion de Green IT.

Ainsi donc, pour intégrer l'application dans la démarche de qualité et de protection de l'environnement de Veolia, j'ai choisi de développer l'application en respectant plusieurs règles ayant attrait au Green IT à savoir :

- L'utilisation de bonnes pratiques de programmation usuelles
- L'utilisation d'un langage de programmation adapté
- L'optimisation de la quantité de données envoyée sur le réseau
- L'optimisation du serveur et du moteur MySQL

L'enjeu essentiel ici se situera au niveau de l'optimisation de l'application et la réduction du temps d'utilisation de celle-ci par les agents.

Ces bonnes pratiques permettront de réduire la charge CPU des équipements en la répartissant du côté client et serveur. Le temps d'utilisation de équipements sera également réduit pour les préserver plus longtemps. En effet, la production de matériel informatique étant plus polluante que leur utilisation, les préserver, dans le but de les garder plus longtemps, réduit l'empreinte carbone des entreprises. Il faut tout de même garder à l'esprit que l'application restera un outil parmi de nombreux autres au sein de l'entreprise. Ainsi son impact, pour peu qu'il puisse être mesuré, ne fera pas faire d'économie importante à Veolia en matière d'utilisation d'énergie à court terme. L'impact que cette application pourra avoir se verra sur le long terme.

3.3. Choix des technologies

3.3.1. Choix du langage de programmation

Le langage de programmation utilisé va beaucoup influer sur le projet et la manière dont celui-ci sera développé, en fonction des avantages et des inconvénients du langage. Il convient de le choisir en considérant les besoins de l'entreprise, pour éviter de devoir changer de langage en cours de projet, ce qui constituerait une perte de temps considérable. De plus, un langage optimisé et facile à apprendre permettra d'avoir une meilleure optimisation de la charge du CPU, ce qui aura pour conséquence de préserver le matériel et de faciliter la maintenance, ce qui est en cohérence avec l'approche Green IT du projet.

Pour choisir un langage de programmation adéquat, il convient de comparer les langages disponibles entre eux. Il existe cependant une grande quantité de langages avec lesquels il est possible de réaliser une application. Il convient donc de limiter le nombre de langages pris en compte dans le cadre d'une comparaison.

3.3.1.1. Choix des langages à étudier

Un des critères limitant les choix des langages est lié au fait que l'application doive être accessible dans un navigateur, via l'intranet de Veolia, comme expliqué dans la partie 2.6 concernant les contraintes particulières de développement. Cependant il existe encore beaucoup de technologies web à comparer.

Au vu de la grande quantité de langages disponibles, il nous faut choisir les principales technologies à comparer. Les langages choisis compteront parmi les plus connus et les plus utilisés. Pour cela, nous nous baserons sur les statistiques du site w3techs.com.

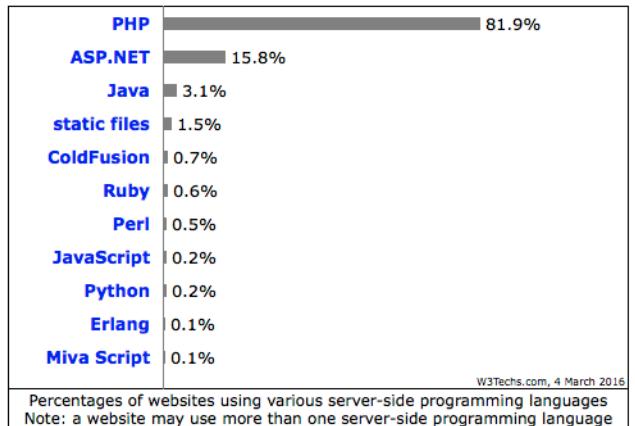


Figure 4 : Répartition des sites web par technologies (source : w3techs.com)

Ainsi donc, après analyse de l'histogramme ci-dessus, nous avons retenu uniquement les langages les plus répandus et les plus connus sur le web en général. Ceux-ci sont au nombre de 5 : PHP, JavaScript, Java, C# (ASP.NET) et Ruby (on Rails).

Choisir des langages très utilisés permet de bénéficier d'un meilleur support au moment du codage de l'application, et donc de développer une application plus rapidement. Cela permet également d'obtenir un outil plus robuste en suivant les conseils de développeurs plus expérimentés et de faciliter la maintenance ou l'évolution du produit par des personnes extérieures au projet. En effet, plus ces personnes auront accès à des ressources variées, plus il leur sera facile de trouver des réponses à leurs problèmes.

De plus, JavaScript a été préféré à Perl dans notre classement. En effet, JavaScript côté serveur est une technologie montante et qui s'avère très efficace, de par la nature même du langage. Une grande communauté se construit autour de JavaScript ainsi que de nombreux projets Open Source orientés web, par rapport à Python et Perl. De plus, grâce à certaines technologies comme Apache Cordova, il est possible d'étendre le site web réalisé à une application mobile. Il est donc possible par la suite de voir notre application fonctionner nativement sur tablette ou smartphone en ne la codant qu'une seule fois en JavaScript, ce qui offre de grandes perspectives en terme d'évolution de l'application.

3.3.1.2. Comparaison des 5 langages de programmation web retenus

Une fois les langages retenus, il a fallu comparer ceux-ci entre eux en fonction de plusieurs critères en rapport avec le projet.

PHP

PHP est un acronyme récursif qui signifie « PHP : Hypertext Processor ». Il s'agit d'un langage de script Open Source très utilisé et spécialement conçu pour le développement web. Il est plus souvent utilisé côté serveur qui va interpréter le code PHP et générer la page HTML en conséquence. Sa syntaxe s'inspire du C, du Java et de Perl. Il est peu typé.

PHP permet, entre autre, de collecter des données de formulaires, créer des pages web de manière dynamique ou d'envoyer ou recevoir des cookies.

Il peut être utilisé sur les systèmes d'exploitation les plus répandus tel que Linux, Solaris, OpenBSD, Microsoft Windows ou Mac OSX. Il est également présent sur de nombreux serveurs web comme Apache ou IIS.

Il permet de programmer de deux manière différentes, à savoir en procédural ou en orienté objet. De plus, il est possible d'utiliser des objets Java comme des objets PHP de manière transparente dans une application PHP. Il supporte un grand nombre de bases de données dont entre autre MySQL.

Il est également possible de générer divers documents à la volée avec PHP, comme des images ou des fichiers PDF.

Enfin, le langage permet de communiquer avec d'autres protocoles comme LDAP, IMAP, SNMP, NNTP, POP3 ou encore HTTP.

C#

Le C# est un langage de programmation orienté objet commercialisé par Microsoft depuis 2002. Il est utilisé au sein du Framework .NET de Microsoft. Ce langage est utilisé dans un environnement ASP.NET dans le cas d'une utilisation pour le web.

Le C# est un langage typé dérivé du C++. Il comporte un ramasse-miettes et un système de gestion d'exceptions. Il y a beaucoup de similarité entre C# et Java.

Il est, à l'origine, destiné à être essentiellement utilisé sur Windows. Cependant, il existe de solutions alternatives, telles que Mono, pour utiliser C# sur des systèmes d'exploitation comme Linux ou Mac OSX.

L'avantage de ce langage est qu'il est fortement couplé avec les outils Microsoft et permet ainsi de mieux interagir avec eux. Par exemple, il est possible de générer des documents Microsoft WORD via la technologie OLE Automation.

Le langage possède de nombreux composants pour communiquer avec d'autres protocoles comme IMAP et POP3 ou encore pour l'utilisation de sockets.

Dans le cadre de notre développement, l'utilisation du C# peut être très avantageux de par son couplage fort avec Microsoft et ses outils en général, les serveurs de Veolia étant principalement équipés de Microsoft Windows Server 2008.

Ruby

Ruby est décrit comme un langage de script orienté objet. Il a pour but de combiner le meilleur des langages de programmation procéduraux et fonctionnels pour les adapter dans le monde des langages script.

Il est utilisé dans Apache pour générer des pages web et dans PostgreSQL où des commandes Ruby sont exécutées sur le serveur de base de données.

Son interpréteur fonctionne sur de nombreux systèmes d'exploitation tels que les systèmes Linux, Microsoft Windows ou encore Mac OSX.

Ruby possède de nombreuses bibliothèques de fonctionnalités, appelés des gems, qui peuvent être adjointes au langage. Le langage possède également un gestionnaire de paquets appelé RubyGems qui permet d'installer ces gems. Parmi elles, on retrouve des bibliothèques pour permettre de communiquer avec le protocole POP3 pour l'envoi de mails, MySQL pour la gestion des bases de données. Il existe également des bibliothèques qui permettent de générer des documents PDF.

JavaScript

JavaScript est un langage de programmation web orienté prototype, contrairement aux autres langages de programmation. Ce paradigme permet, entre autre, de modular les prototypes à volonté en leur ajoutant des attributs et des méthodes. Il s'agit d'un langage interprété.

JavaScript a longtemps été un langage destiné à être téléchargé et exécuté chez le client. Cependant, ces dernières années ont vu la montée de nouvelles API et plateformes telles que NodeJS développé par Google, qui permettent d'utiliser la puissance de JavaScript à la fois chez le client et sur le serveur. Ainsi les développeurs peuvent coder la totalité de leur application dans un seul langage pour permettre une meilleure coordination du client et du serveur.

Les frameworks implémentant JavaScript sur le client et le serveur exploitent, entre autre, les sockets et permettent également au serveur d'envoyer des informations vers le client sans que celui-ci ait besoin d'envoyer de requête au préalable. Cela permet d'introduire une dimension « temps réel » et monitoring dans les applications qui n'est pas disponible avec les autres langages de programmation.

Les interpréteurs JavaScript sont disponibles sous plusieurs systèmes d'exploitation comme Linux, Mac OSX ou Windows.

À noter quand même que, bien que la technologie JavaScript côté serveur soit très intéressante en terme de possibilités, elle reste néanmoins une technologie assez jeune dans ce domaine et encore peu utilisée au sein des entreprises. En effet, nombreux frameworks et plateformes implémentant JavaScript côté serveur viennent de passer à leur version stable il y a environ un an ou deux.

3.3.1.3. Langage retenu : PHP

Le choix du langage s'est finalement porté sur PHP. En effet, il s'agit d'un langage facile d'apprentissage, accessible sur la plupart des systèmes d'exploitations et très populaire sur le web, ce qui permet un meilleur support et une meilleure maintenance. De plus, il s'agit d'un langage déjà éprouvé depuis plusieurs années et donc assez robuste pour répondre aux besoins de l'entreprise, qui veut s'appuyer sur des technologies matures et fiables pour fonctionner de manière optimale. Enfin, il est assez facile d'apprentissage, ce qui permettra à de futurs développeurs de maintenir ou de faire évoluer rapidement l'application.

3.3.2. Choix du framework PHP

Un Framework n'est pas indispensable pour la création de notre application web. Cependant, pour que l'application soit robuste, facile à faire évoluer et réalisable en un temps minimum, un framework représente un outil idéal.

3.3.2.1. Choix du framework à étudier

Il existe une grande quantité de frameworks PHP. Chacun présentant des avantages et des inconvénients, il a fallu réduire le choix de ces frameworks pour permettre ainsi de ne pas perdre trop de temps sur le comparatif et passer plus rapidement à la phase de modélisation.

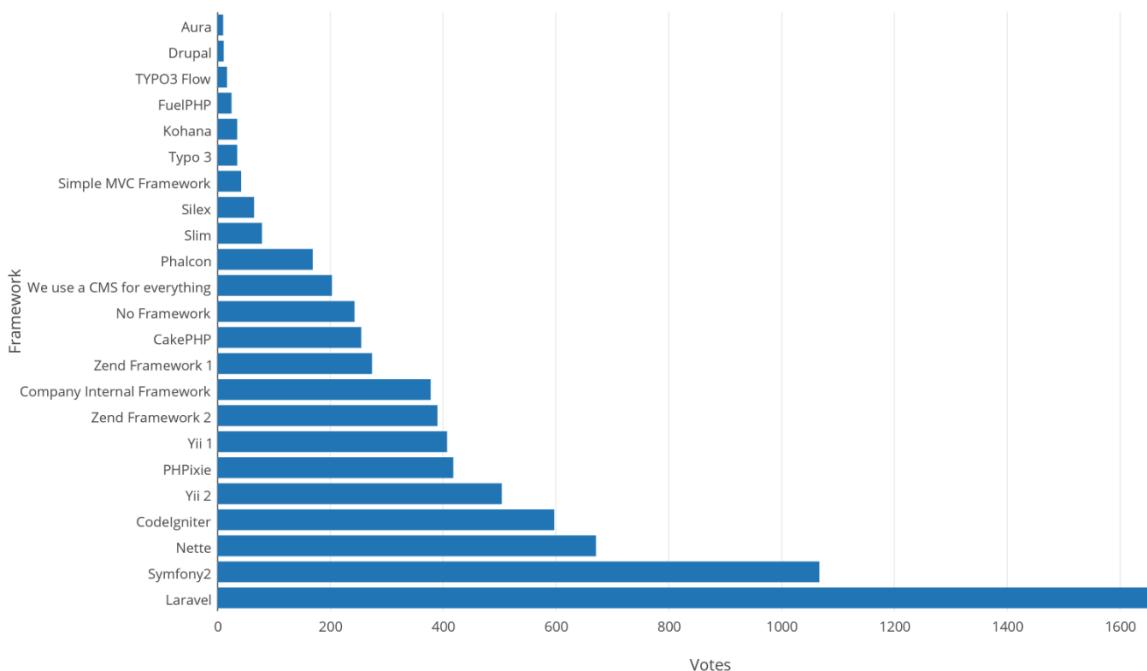


Figure 5 : Popularité des frameworks PHP en entreprise (source : sitepoint.com)

La figure ci-dessus présente les frameworks les plus populaires en entreprise. Nous allons donc ici nous concentrer sur les 2 frameworks les plus populaires : Laravel et Symfony, ainsi que deux moins populaires, mais sur lesquels j'ai plus d'expérience en tant que développeur et donc une meilleure visibilité : CakePHP et Zend. Une fois encore, la popularité de l'outil est essentielle pour une meilleure maintenance de l'application réalisée ainsi qu'une phase de codage plus aisée. De plus, une meilleure maîtrise du framework utilisé (comme CakePHP dans mon cas) représente un avantage pour le développement, en me permettant de gagner du temps et d'avoir un code plus cohérent par rapport au fonctionnement du framework.

Laravel

Laravel est un Framework web open-source écrit en PHP. Il respecte le modèle MVC (modèle-vue-contrôleur). C'est un Framework orienté objet distribué sous licence MIT.

Laravel est un outil qui, dans sa conception, se base sur le meilleur de plusieurs autres Frameworks pour développer son propre système et être plus efficace. Il possède également des composants qui lui sont propres.

Il fournit entre autre :

- Un système de routage des vues
- Un créateur de requêtes SQL et une gestion de version de Base de données
- Un ORM performant
- Un système d'authentification pour les connexions
- Un système de cache
- Une gestion de sessions

Symfony 2

Symfony est un Framework PHP qui a été lancé en 2005. Il est aujourd'hui stable et reconnu. Il est également orienté objet, respecte le modèle MVC et est développé sous licence MIT.

C'est un Framework très utilisé et reconnu internationalement. Il a été développé par la société SensioLabs qui l'utilise et le maintien régulièrement.

Il est considéré comme un ensemble d'outils rassemblant des composants préfabriqués, rapides et faciles à utiliser.

Un des avantages de Symfony est de proposer une évolutivité et une maintenance efficace en permettant à d'autres développeurs de prendre en main rapidement le projet sans avoir participé à son élaboration. Il existe également un nombre important de ressources sur le web pour rendre la maintenance encore plus facile. Enfin, il est très flexible car il permet de n'utiliser que certains de ces modules sans forcément avoir à utiliser tout le Framework. Laravel possède beaucoup de composants issus du Symfony.

CakePHP

Le projet Cake PHP a démarré en 2005. Il s'agit d'un Framework web libre écrit en PHP et distribué sous licence MIT. Il suit lui aussi le concept MVC et imite le fonctionnement de Ruby on Rails.

CakePHP a l'avantage de permettre aux développeurs de réduire le couplage entre leurs modules et présente de nombreuses fonctionnalités de base.

Il inclut notamment dans ses fonctionnalités :

- L'intégration des commandes CRUD pour l'utilisation simplifiée des bases de données SQL
- Un dispatcheur d'URL pour permettre d'avoir des adresses aisément lisibles
- La gestion de la sécurité
- La gestion des droits, des sessions et du cache

Zend

Le Framework Zend est un projet PHP gratuit fourni par la société Zend. Un de ses buts principaux est de permettre d'industrialiser la façon de coder en PHP. Il suit également le modèle MVC pour une plus grande simplicité des sites construits.

Au niveau des fonctionnalités, le Framework zend met l'accent sur la sécurité en protégeant des injections SQL ainsi que contre les attaques de types cross-site-scripting (XSS). Il est de plus doté d'outils de chiffrement. Il permet également de simplifier les URL et propose des fonctions courantes comme le moteur de recherche, l'accès aux bases de données, l'accès à des services externes comme Google, l'authentification ou encore les sessions.

3.3.2.2. Framework retenu : Laravel

Parmi ces frameworks PHP, tous présentent des avantages et des inconvénients certains. Cependant, ces avantages et inconvénients n'ont pas été assez discriminants pour permettre de choisir un framework parmi tous les autres. Ainsi, pour choisir celui qui serait le plus intéressant à utiliser, j'ai encore une fois basé mon choix sur la popularité de l'outil. Ainsi, mon choix s'est porté sur Laravel. En effet, celui-ci est un framework PHP très populaire en entreprise et réputé facile d'utilisation. Il présente également un ORM très puissant, appelé Eloquent, qui permet de récupérer les éléments présents en base de données de manière plus rapide et efficace en les associant à des classes PHP de manière automatique, et proposant des requêtes optimisées pour la base de données. Laravel possède également un système de migration de base de données. Il va permettre ainsi de développer et déployer plus rapidement l'application.

3.3.3. Choix du système de gestion de bases de données (SGBD)

Une fois le langage et le framework choisis, la question de la base de données à utiliser a elle aussi été très importante. Toujours dans l'optique d'une optimisation de l'outil, il faut choisir le système de gestion de bases de données le plus efficace possible. Son adéquation avec les besoins du programme impacte directement le temps de développement et la stabilité du système.

3.3.3.1. Choix des SGBD à comparer

Il convient donc de le choisir encore une fois en fonction du besoin, mais aussi des contraintes de maintenabilité et des critères de performance. Pour cela, une première étape consiste à étudier la popularité des solutions disponibles.

Rank			DBMS	Database Model	Score		
Mar 2016	Feb 2016	Mar 2015			Mar 2016	Feb 2016	Mar 2015
1.	1.	1.	Oracle	Relational DBMS	1472.01	-4.13	+2.93
2.	2.	2.	MySQL	Relational DBMS	1347.71	+26.59	+86.62
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1136.49	-13.73	-28.31
4.	4.	4.	MongoDB	Document store	305.33	-0.27	+30.32
5.	5.	5.	PostgreSQL	Relational DBMS	299.62	+10.97	+35.19
6.	6.	6.	DB2	Relational DBMS	187.94	-6.55	-10.91
7.	7.	7.	Microsoft Access	Relational DBMS	135.03	+1.95	-6.66
8.	8.	8.	Cassandra	Wide column store	130.33	-1.43	+23.02
9.	10.	10.	Redis	Key-value store	106.22	+4.14	+9.17
10.	9.	9.	SQLite	Relational DBMS	105.77	-1.01	+4.06

Figure 6 : Classement de popularité des SGBDs (source : db-engines.com)

La figure ci-dessus représente le classement des 10 systèmes de gestion de base de données les plus utilisés. On retrouve ainsi Oracle en tête de liste, suivi de MySQL, Microsoft SQL Server, MongoDB et PostgreSQL. Ce seront donc les 5 SGBDs retenus pour notre comparatif, en vue de choisir le meilleur pour notre application. Parmis eux, les 3 premiers sont des bases relationnelles, tandis que MongoDB est une base orientée document (NoSQL) et PostgreSQL est une base relationnelle objet.

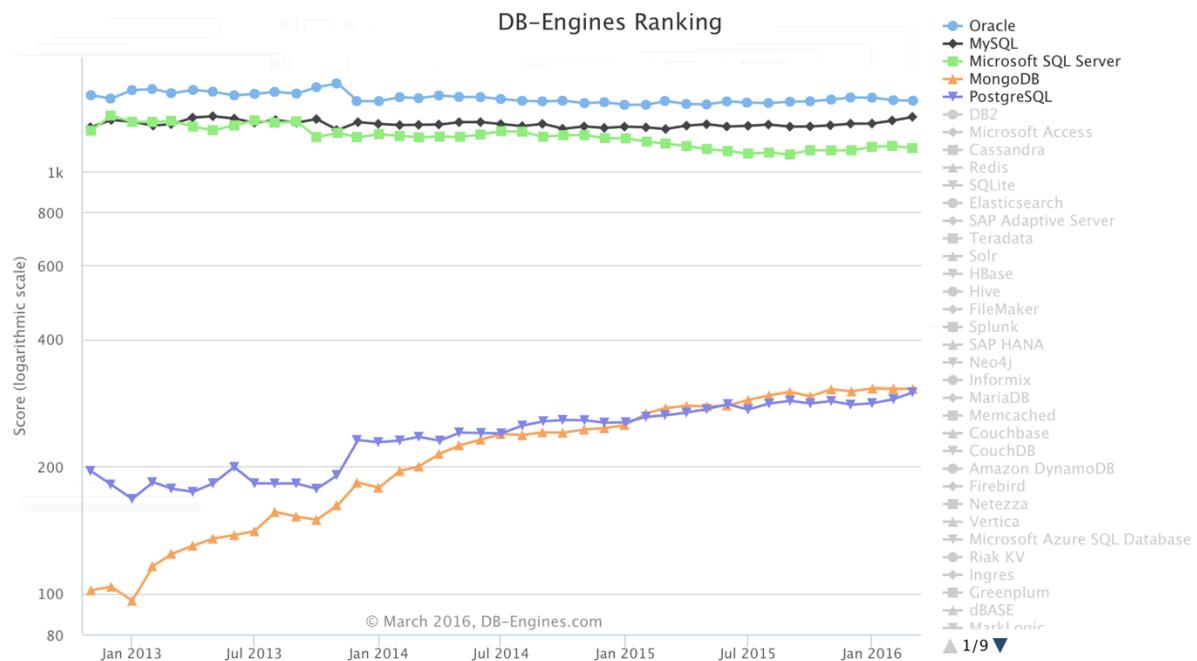


Figure 7 : Evolution de la popularité des SGBDs (source : db-engines.com)

Le graphique de la figure 7 illustre un fossé certain entre les 3 premiers SGBDs et les deux derniers, à savoir PostgreSQL et MongoDB en terme de popularité. Cela s'explique en partie par le fait que ces deux derniers SGBDs soient récents et ont un fonctionnement assez différent des autres SGBD. Ce fonctionnement sera détaillé dans la présentation des solutions dans la partie suivante.

3.3.3.2. Présentation des SGBDs

Oracle Database

Oracle Database est un système de gestion de base de données relationnelle objet. Il est développé par Oracle Corporation a été distribué pour la première fois en 1980. Il est implémenté en C et C++ et est disponible sur la plupart des systèmes d'exploitation (Linux, Solaris, OSX, Windows). Il est également compatible avec une grande variété de langage de programmation, dont PHP. Il respecte le principe des transactions ACID.

MySQL

MySQL est le deuxième SGBDs le plus populaire d'après le classement réalisé par le site db-engines.com. Il s'agit d'une base de données relationnelle open-source sous licence GPLv2. Elle est également développée par Oracle Corporation, autrefois par MySQL AB et Sun Microsystems. La première version a été distribuée en 1995. MySQL est implémenté en C et C++ et est disponible sous FreeBSD, Linux, Solaris, OSX et Windows. Il supporte également une grande variété de langages, dont PHP et respecte également le principe des transactions ACID.

Microsoft SQL Server

Microsoft SQL Server est un système de gestion de base de données développé par Microsoft et sorti en 1989. Disponible sous licence commerciale, une licence gratuite est également disponible mais est limité en terme de fonctionnalités. Actuellement dans sa version 2014, Microsoft SQL Server est développé en C++ et supporte moins de langage qu'Oracle et MySQL. Il respecte lui aussi le principe des transactions ACID.

MongoDB

MongoDB est un système à part, comparé à ceux présentés ci-dessus. En effet, il s'agit d'une base de données orientée document, faisant partie de la mouvance NoSQL. Cette tendance vise à gérer les bases de données de manière transparente, sans avoir recours, comme son nom l'indique, au SQL classique, dans un souci de simplicité. Ce sont des bases de données prévues pour des applications dites Big Data. Dans MongoDB, ce principe s'applique par la manipulation d'objets au format BSON (JSON Binaire).

Conçu par MongoDB Inc, la première version de MongoDB a été publiée en 2009, ce qui en fait le plus récent de tous les SGBDs étudiés. Il est open-source et disponible sous licence AGPL v3. Implémenté en C++, il supporte une grande variété de langages modernes et anciens, dont entre autre PHP, et est disponible sous Linux, OSX, Windows et Solaris.

PostgreSQL

PostgreSQL est un SGBD qui tend à monter en notoriété depuis plusieurs années déjà. Il est souvent vu comme l'alternative à MySQL.

PostgreSQL est vu comme une base de données relationnelle assez particulière car orientée objet. Il possède des extensions objet permettant de définir des fonctions en base de données et d'appliquer les principes d'héritage. Ce type de base de données est très utiles lorsque les données stockées sont très complexes et que le stockage des relations objets tels que l'héritage présentent un intérêt certain.

Sa première version est sortie en 1989, sous licence BSD. Il est implémenté en C, supporte moins de langages que les autres. PHP ne fait pas réellement partie de sa liste de langages compatibles, cependant PHP intègre la librairie « pgsql » pour permettre de réaliser des applications avec PostgreSQL.

3.3.3.3. Choix du SGBD

Pour choisir les SGBD à comparer, il a fallu comparer entre elles plusieurs solutions. Cependant, de par leur très grande popularité, deux SGBD ont retenu mon attention : MySQL et PostgreSQL. Dans les deux cas, il s'agit de SGBD très populaires et très répandus. Cependant, ils se différencient de par ce qu'ils implémentent. MySQL est le SGBD le plus répandu. Celui-ci est connu pour être très robuste et constitue une valeur sûre du web. PostgreSQL est quant à lui, vu comme la nouvelle alternative à MySQL. Il est également très robuste. Cependant MySQL implémente des bases de données de type relationnel, tandis que PostgreSQL implémente des bases de données de type objet. Ces dernières bases de données sont très intéressantes dans le cas de structures objet complexes nécessitant de faire intervenir des notions comme l'héritage. Dans notre application, la notion d'héritage est très peu présente car les objets restent assez cloisonnés. Ils présentent plus de relations d'appartenance entre eux que de relations d'héritage. Ainsi donc, MySQL sera plus approprié, d'autant plus que la plupart des informations présentes sur internet font référence à MySQL en relation avec Laravel plus que PostgreSQL, ce qui facilitera la maintenance.

3.3.5. Choix du framework CSS

Les feuilles de style en cascade, ou en anglais, Cascading StyleSheet (CSS), forment un langage informatique décrivant la manière dont les éléments d'une interface HTML doivent être affichés. Introduit dans les années 1990, CSS est un langage utilisé par tous les navigateurs. C'est le langage standard pour la réalisation d'interfaces web riches.

L'utilisation de framework CSS permet, comme dans le cas du framework PHP, d'améliorer la maintenabilité du code, son évolution, et plus généralement le design de l'application, la rendant ainsi plus agréable à utiliser. De plus, la plupart de ces frameworks implémentent des notions comme le design responsive qui permettent à l'application d'être universelle et de s'adapter à tout type d'écran. Nous choisirons donc de comparer 4 frameworks, une fois encore parmi les plus connus, pour permettre de gagner du temps à la fois pour la production, la maintenance et l'évolution de l'application.

Il est difficile d'obtenir des statistiques concernant la popularité d'utilisation des frameworks CSS. Cependant, plusieurs noms reviennent très souvent sur les sites consacrés au design. Parmi eux, on retrouve : Bootstrap, UIKit, Foundation et Skeleton.

Bootstrap

Bootstrap est un framework front-end gratuit pour le développement web. Il contient plusieurs outils utiles à la réalisation de sites interactifs. Il permet entre autre de concevoir plus facilement un design responsive qui va permettre d'adapter l'affichage de l'application à tout type d'écran. Il fait partie des frameworks les plus populaires. Bootstrap est un framework très récent. Il a été conçu en 2010 par deux développeurs de chez Twitter : Mark Otto et Jacob Thornton. Son but était de diminuer les coûts de maintenances dus aux incohérences entre les différentes bibliothèques existantes.

Il est conçu pour être compatible avec tous les navigateurs majeurs tels que Google Chrome, Firefox, Safari ou encore Opera. Il fonctionne également en mode dégradé sur des navigateurs plus anciens.

Le concept de Bootstrap est basé sur les grilles. Chaque élément de l'interface se situe à l'intersection d'une ligne et d'une colonne. Cette grille sert d'armature à la totalité de l'interface et est également très pratique en terme de design responsive.

Le framework est publié sous licence MIT.

UIKit

UIKit est un framework de développement front-end de site web. Il a été développé par la société YOOTHEME. Il a pour avantage d'être plus léger et plus modulaire que les autres. Il est également facile à personnaliser et peut être étendu avec des thèmes.

Comme Bootstrap, UIKit fonctionne sur le système de grilles pour permettre d'adapter plus facilement l'interface réalisée à tous types d'écrans. Il est compatible avec tous les navigateurs récents et est distribué sous licence MIT.

Foundation

Foundation est un projet open-source créé par la société de design Zurb. Il fournit une grille de design HTML responsive ainsi que des composants d'interfaces standardisés (boutons, champs de texte, formulaires etc.) et réutilisables. Foundation utilise SASS comme préprocesseur CSS par rapport aux autres qui utilisent LESS, ce qui modifie la syntaxe standard des éléments à l'intérieur du framework. Il inclut également des composants et plugins JavaScript.

Foundation utilise, comme la plupart des frameworks CSS, le système de grille pour permettre de développer des applications responsives. Il est publié sous licence MIT.

Skeleton

Skeleton est également un projet open-source développé par Dave Gamache. Il permet de prototyper rapidement des interfaces et accélère également le développement des interfaces web en général, et ce quelle que soit leur dimension. Cependant, Skeleton est différent des autres solutions car il s'agit d'un kit de développement. En effet, il est très épuré. En effet, il ne donne qu'une grille responsive par rapport à ses éléments HTML. Il n'inclut pas de thème par défaut, il permet simplement, comme son nom l'indique, d'avoir un squelette d'interface autour duquel construire son propre design.

3.3.4. Développement et déploiement de l'application

L'application devra être déployée sur un serveur en intranet. Cependant, le déploiement de l'application sur un serveur après une période de développement sous un autre environnement comporte des risques, notamment au niveau de la compatibilité de certains modules. Il peut même arriver parfois que l'application soit inutilisable sur certains serveurs car leurs administrateurs n'ont pas, par choix, activé certains modules comme la réécriture d'URL.

Cela risque d'allonger le temps de développement pour des détails techniques et limiter le temps consacré à la maintenance et aux améliorations du produit. Ainsi donc, dans le but de limiter l'impact que peut avoir ce genre de problèmes sur le projet, il convient de choisir une technologie adaptée qui va, dans l'idéal, nous permettre de développer dans les mêmes conditions. Pour cela, j'ai étudié et comparé plusieurs solutions : Docker, Vagrant. Nous verrons ensuite la solution adoptée.

3.3.7.1. Docker

Docker est un logiciel libre pour l'automatisation du déploiement d'applications. Basé sur le principe des conteneurs logiciels. Docker peut-être utiliser pour « empaqueter une application et ses dépendances dans un conteneur virtuel, qui pourra être exécuté sur n'importe quel serveur Linux ». Il permet donc de créer un conteneur pour l'application. Ce conteneur ne contient pas uniquement l'application, mais tout son environnement avec elle. Cela permet ainsi au développeur de créer son application dans un environnement contrôlé au sein du conteneur, puis de déployer celle-ci facilement en récupérant le conteneur et en l'importer dans une autre installation de Docker.

Le problème principal de Docker est qu'il doit fonctionner sous un environnement Linux. Or, les serveurs de Veolia fonctionnent tous sur du Windows. Il est tout à fait possible d'installer une machine virtuelle Debian sur les serveurs de Veolia. Cependant, compte tenu de la taille de l'application, il s'agirait ici d'une solution démesurée. De plus, pour que cette technologie devienne intéressante à utiliser au sein du système d'information, il faudrait également migrer les autres applications web de l'entreprise dans des containers.

3.3.7.2. Vagrant

La deuxième technologie que j'ai étudiée concernant le déploiement de l'application est Vagrant. En effet, Vagrant est un logiciel libre et open-source qui permet de créer et gérer des environnements de développement virtuels standards. Il peut être considéré comme un « wrapper » autour d'autres logiciels de virtualisation tels que VMWare ou VirtualBox.

Vagrant va permettre d'avoir un environnement de travail standard (appelé Homestead pour Laravel) sous lequel il sera possible de développer l'application. Ainsi, au moment du déploiement, il suffira de charger le même environnement dans Vagrant (qui sera préalablement installé sur le serveur) et par la suite d'y copier les fichiers. En utilisant cette technologie, l'application pourra fonctionner dans les mêmes conditions en développement et en production.

Cependant, Vagrant n'est pas non plus installé de base sur les serveurs de Veolia qui utilisent Microsoft Windows Server 2008 R2, ce qui implique encore de passer par une phase d'installation, et de test. De plus, Vagrant ne serait utilisé que pour cette application, et nécessiterait de migrer les applications intranet déjà présentes chez Veolia sous un environnement Vagrant. Enfin, la société utilise de la virtualisation de type 1 sur ses serveurs, et l'utilisation de Vagrant nécessite l'utilisation d'une virtualisation de type 2. Les deux types de virtualisations ne sont pas incompatibles, cependant, cela rajouterai une couche de virtualisation sur le serveur pour une seule application, ce qui est potentiellement démesuré.

3.3.7.3. Solution adoptée : développement sur le serveur de production

La solution adoptée repose sur l'utilisation des outils déjà en place sur le serveur intranet de Veolia. En effet, des solutions comme Vagrant ou Docker, malgré le fait que celles-ci soient optimisées, présentent deux inconvénients majeurs. Le premier vient du fait que ces technologies ne soient pas installées sur les serveurs de Veolia. Ainsi il faudra les installer, les configurer, les intégrer dans la logique de l'entreprise ce qui peut s'avérer assez délicat et démesuré pour une telle application. De plus, ces technologies prennent beaucoup de place au niveau du stockage du serveur par rapport à l'application développée, dont les fichiers pèseront entre 30 et 100 Mo maximum.

Des applications web (comme le site intranet) étant déjà en place sur le serveur, la technologie PHP et MySQL y étant également déjà installé et fonctionnant avec le serveur http IIS7, il est plus judicieux d'utiliser ces technologies déjà en place. L'avantage est de permettre d'intégrer l'application dans le pool des sites intranet de Veolia et ainsi de garder une certaine cohérence au niveau des applications web. De plus, cela facilitera les procédures en cas de migration des sites.

Afin de limiter les problèmes de compatibilité, le développement se fera directement sur le dossier du serveur. Pour cela, Laravel sera installé dans le dossier racine du nouveau site. Ce dossier sera partagé et le reste du développement se fera depuis une autre machine ayant accès à ce dossier. Cela permet de limiter au maximum l'impact du développement sur la sollicitation du serveur. Nous économiserons donc des ressources matérielles au niveau du serveur, et resterons ainsi dans l'approche Green IT qui a pour but de préserver les équipements.

3.4. Rédaction du cahier des charges

La rédaction du cahier des charges est une partie importante de la réalisation du projet car il va permettre de fixer les limites de celui-ci. Il va également permettre de mettre par écrit les fonctionnalités attendues dans l'application.

Une fois celui-ci terminé, il a été validé par le responsable informatique et le responsable approvisionnement. Aucune difficulté particulière n'a été rencontrée pour la rédaction de ce cahier car une étude préalable a été réalisée en amont et les besoins ont été clairement définis au début du stage.

3.5. Modélisation de l'application

3.5.1. Diagramme de cas d'utilisations

La connaissance des fonctionnalités à implémenter est essentielle pour établir le diagramme de cas d'utilisations de l'application. Une fois encore, l'étude réalisée pour la compréhension de l'application et l'écriture du cahier des charges ont permis d'avoir des éléments solides pour lister les fonctionnalités à implémenter et faciliter la réalisation de ce diagramme de cas d'utilisation. Dans cette section, nous allons détailler les différents diagrammes de cas d'utilisation réalisés.

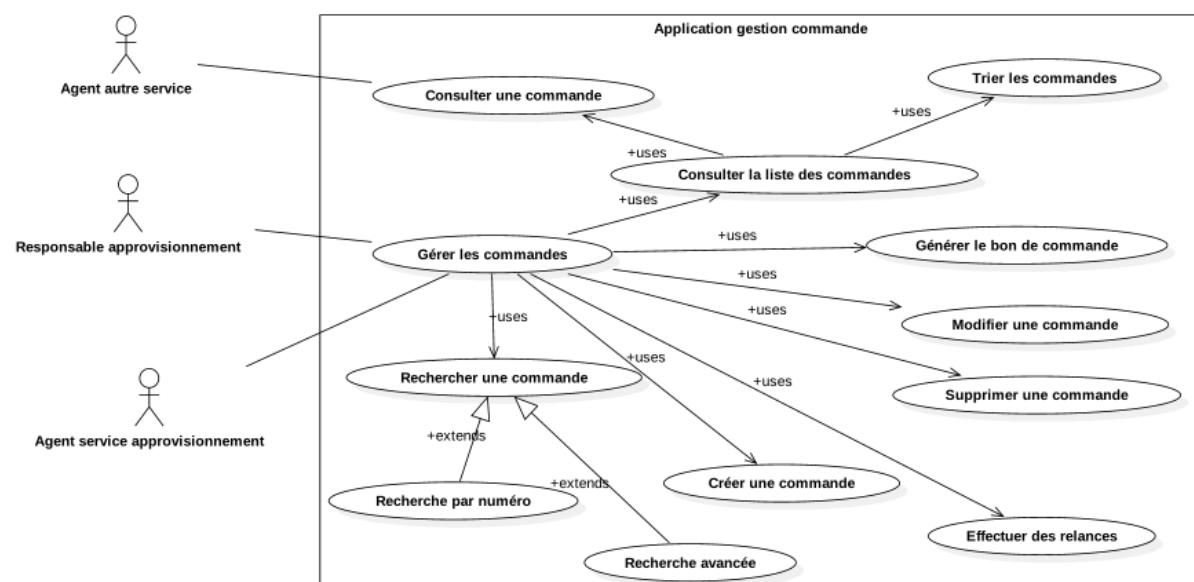


Figure 8 : Diagramme de cas d'utilisation (commandes)

La figure ci-dessus représente le diagramme de cas d'utilisation principal, à savoir celui concernant la gestion des commandes. Comme nous pouvons le constater, les deux acteurs principaux de ce diagramme sont le responsable approvisionnement et l'agent du service approvisionnement. L'application doit donc pouvoir leur permettre de gérer les commandes, et pour cela implémenter des fonctionnalités comme la consultation de la liste des commandes, en fonction de leur état, la recherche ainsi que toutes les fonctionnalités CRUD (Create Read Update Delete) en général.

Les agents d'un autre service sont également des acteurs de ce diagramme, bien que secondaires. Leur rôle ici se résume uniquement à la consultation de l'état de leurs commandes.

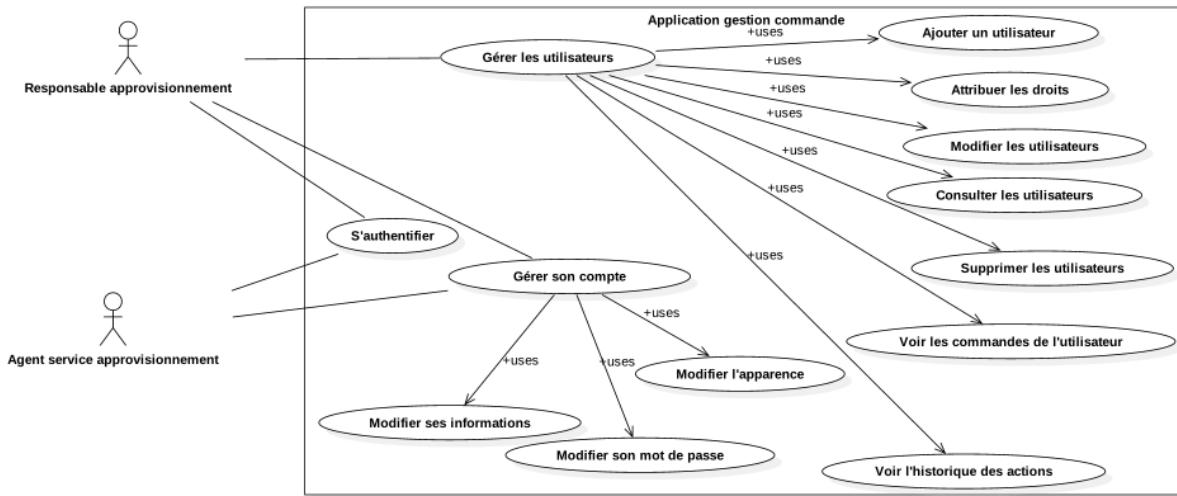


Figure 9 : Diagramme de cas d'utilisation (Utilisateurs)

L'application doit également permettre de gérer les utilisateurs comme le montre le précédent diagramme de cas d'utilisation. En effet, une des faiblesses de l'ancien outil venait du fait qu'il ne possédait pas de système d'authentification. Ainsi, il fallait, à chaque commande traitée, entrer le nom de l'agent. Ainsi donc, notre application proposera de s'authentifier afin de gagner du temps sur ce genre de détails lors de la saisie des commandes et d'augmenter la sécurité au cœur même de l'application en n'attribuant des droits d'administrateurs qu'à certains utilisateurs.

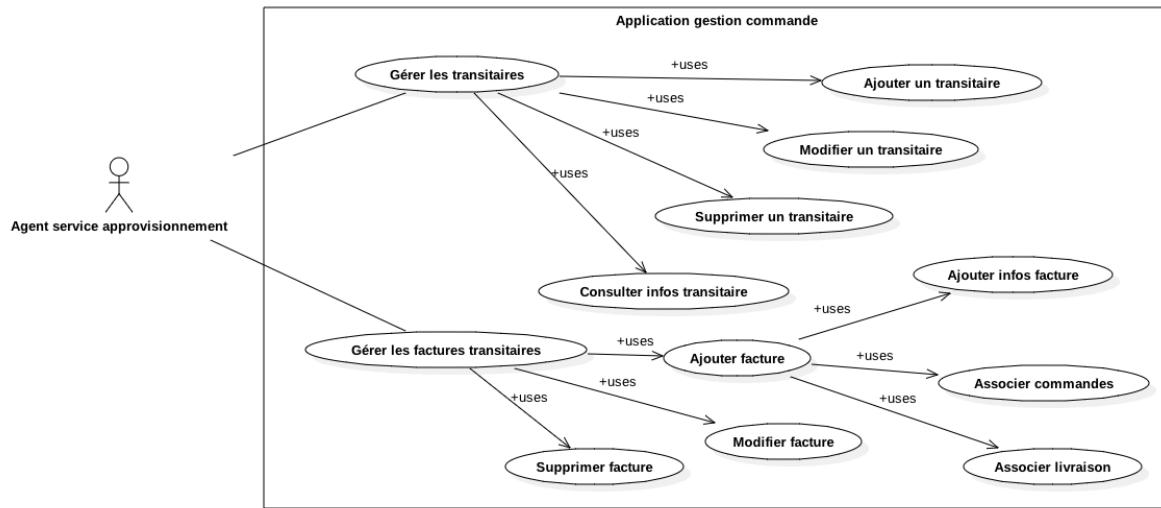


Figure 10 : Diagramme de cas d'utilisation (Transitaires)

Ce dernier diagramme montre les fonctionnalités attendues dans le cas des transitaires. Ce diagramme est différent des autres de par la manière dont sont gérées les factures. En effet, il faut également associer à celles-ci les commandes et les livraisons.

3.5.2. Modélisation de la base de données de l'application

La modélisation de la base de données est également une tâche très importante car il s'agit du cœur de l'application réalisée. Une application étant déjà en place, il en existe déjà une. Il va donc falloir prendre en compte le schéma de cette base de données actuelle et le modifier si nécessaire pour en obtenir une nouvelle qui permettra à l'outil d'être plus performant, tout en restant conforme aux informations que les commandes doivent comporter en temps normal.

Au cours de la phase de réalisation, plusieurs versions de la base de données ont été réalisées. En effet, certains processus (comme par exemple la saisie des factures transitaires) sont plus compliqués que les autres et nécessitent une optimisation de la base de données ainsi que de l'application pour gagner en efficacité.

Il est également à noter que le framework Laravel optimise la gestion des migrations d'applications grâce à son système également appelé migration. En effet, lorsqu'une application est migrée d'un serveur à un autre, il est souvent nécessaire de remettre en place la base de données. Ici, Laravel nous permet de réinstaller la base de données en exécutant une ligne de commande. Cette ligne de commande va permettre d'exécuter les migrations et permettra ainsi de gagner en temps et va limiter les oubli et les problèmes de compatibilité, dans le cas d'une réinstallation.

Nous allons, dans cette partie, lister les principales versions de la base de données et commenterons les principaux changements de modélisation qui sont intervenus.

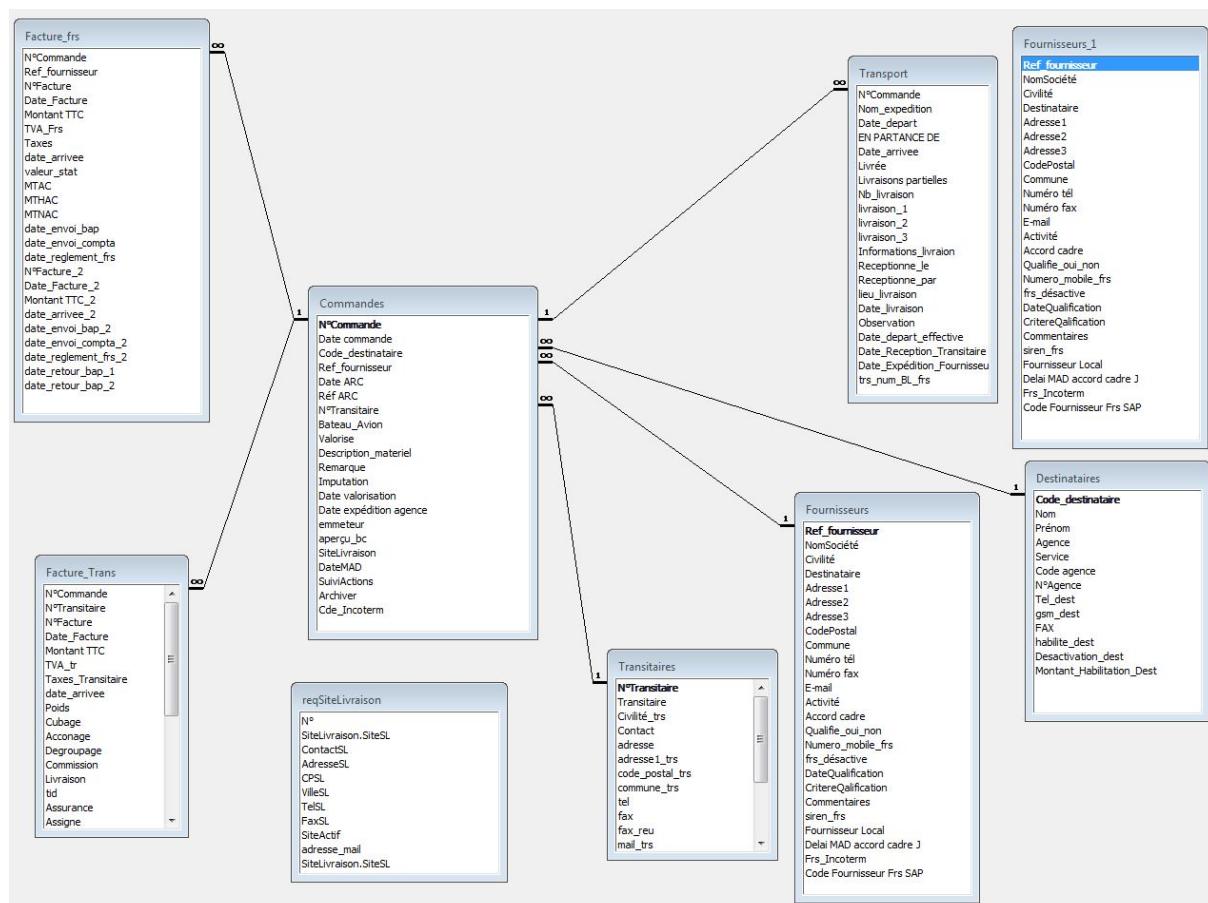


Figure 11 : Schéma de la base sous Microsoft Access

La figure ci-dessus représente la base de données réalisée sous Microsoft Access. Nous pouvons constater que quasiment toutes les tables sont reliées à la table « Commandes » et ne possèdent aucune relation entre elles. Cela nous indique que les requêtes effectuées sur

la base de données concernent essentiellement les commandes, et que la gestion des autres tables comme « Facture_Trans » (pour les factures transitaires) est totalement découpée de la table « Transitaires » ce qui nous montre bien le côté provisoire de la solution.

La base de données elle-même comportant un nombre assez important de tables, dont une bonne partie était en fait des tables provisoires, mises dans la base Access par défaut, il a fallu écarter celles-ci pour pouvoir garder celles qui nous intéressent. Cela nous permet de mieux appréhender le véritable squelette de l'application.

Une fois les tables utiles rassemblées, nous pouvons remarquer qu'il en existe 9 au total. Ces tables sont cependant très complètes avec une grande quantité d'attributs. Il est possible que ces attributs soient trop nombreux et doivent être séparés pour être mis dans de nouvelles tables.

Nous notons également la présence de caractères comme les lettres accentuées qui sont fortement déconseillées dans les noms des champs d'une base de données. En effet, ceux-ci peuvent engendrer des problèmes d'encodage. Il s'agit cependant d'une particularité de Microsoft Access, qui autorise ces lettres dans sa base. Cependant, en cas de migration de la base Access vers la base MySQL, il nous faudra faire très attention à ce genre de problèmes de compatibilité.

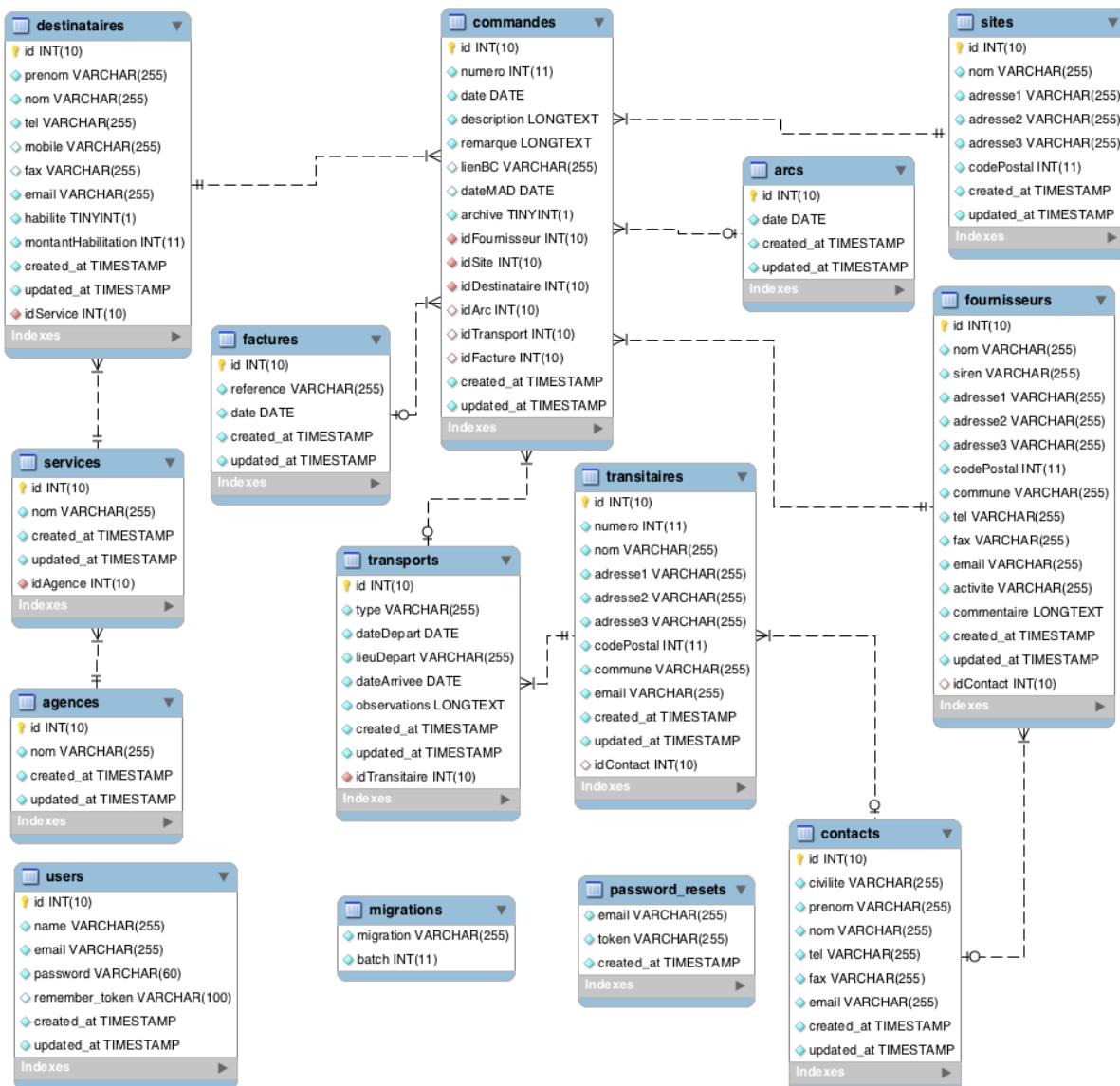


Figure 12 : Première version de la base de données

Cette première version de la base de données apporte des modifications notables par rapport à la version sous Access. Entre autre, on note l'apparition de relations entre les différentes tables, avec l'ajout de clés étrangères. Ces relations permettent à notre application d'être plus cohérente et plus rapide lors de l'exécution des requêtes, notamment au niveau des jointures. Toutes les requêtes sont réalisées de manière automatique par le framework, grâce à l'ORM Eloquent, qui va permettre de les standardiser, de les simplifier et d'assurer l'intégrité de la base de données. Cependant, ces clés étrangères doivent être standardisées pour que l'ORM puisse correctement effectuer les requêtes.

La table « commandes » reste la table centrale de l'application, qui référence la plupart des autres tables, comme la table « fournisseurs », « destinataires » ou encore la table « arcs » qui va stocker les accusés de réception. Cette dernière table n'apparaissait pas dans l'ancienne base, et va aider à optimiser considérablement les requêtes effectuées. Elle ne contient que des dates pour cette première version, cependant d'autres informations risquent d'apparaître dans les prochaines versions de la base de données.

Enfin, on note l'apparition de 3 tables : « users », « password_resets » et « migrations ». Il s'agit de tables par défaut, incluses dans le framework Laravel. Les deux premières tables permettent de gérer les utilisateurs dans le cas d'une application implémentant de l'authentification. La dernière table permet de garder une trace des migrations réalisées.

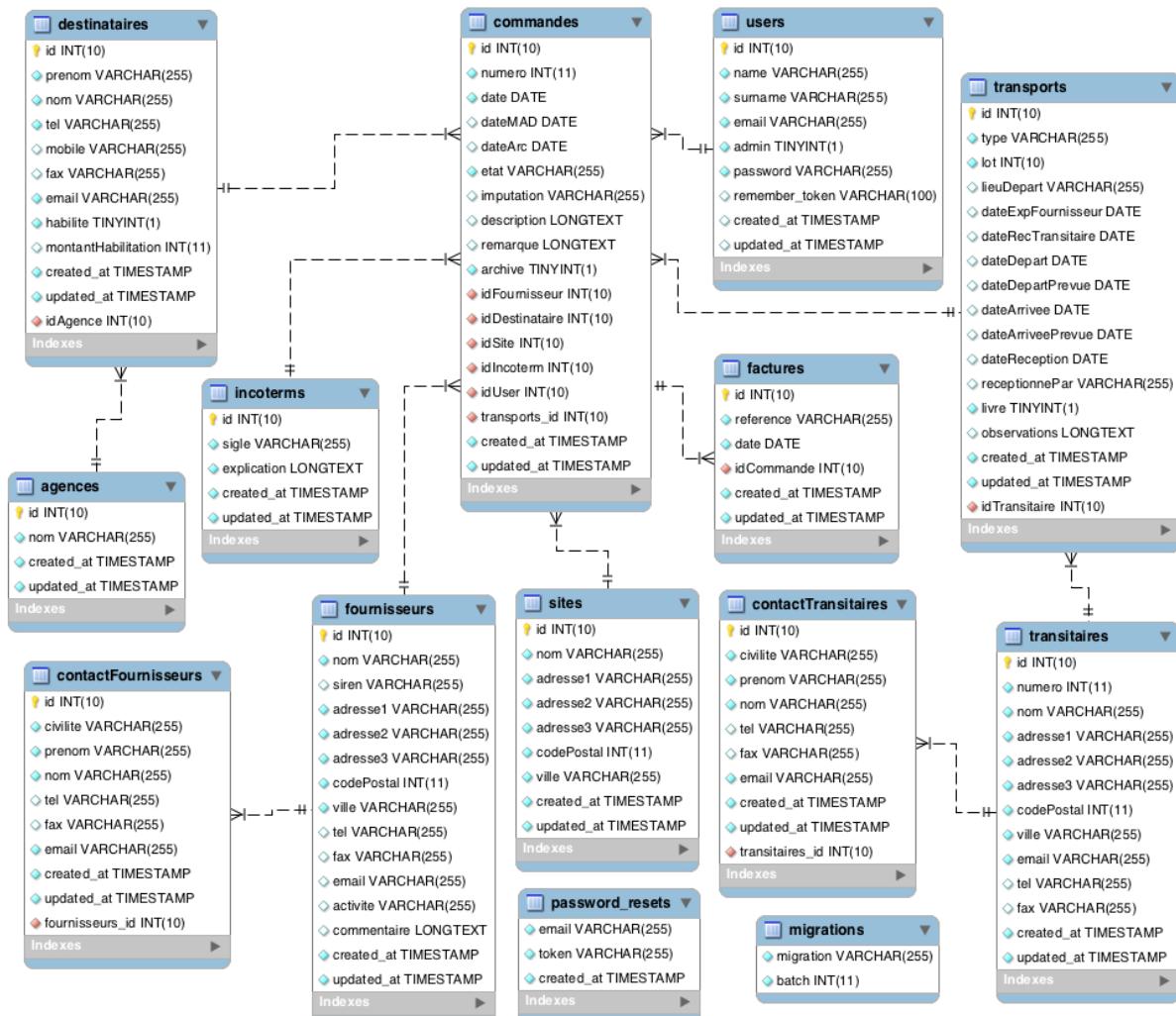


Figure 13 : Deuxième version de la base de données

La deuxième version de la base de données se veut déjà beaucoup plus complète. En effet, au fur et à mesure de mon évolution dans l'entreprise, j'ai eu le temps d'assimiler encore un peu plus le processus que j'automatisé. Ainsi, il m'a été possible d'affiner la modélisation de la base. Pour cela, j'ai séparé les contacts transitaires et fournisseurs dans le but d'optimiser mes requêtes de recherche et de tri.

La table ARC (accusé de réception) a également disparue et a été remplacée par un champ « dateArc » à l'intérieur même de la table commandes. En effet, cette table devait, à l'origine, évoluer pour contenir plus d'informations. Cependant, celle-ci n'ayant finalement pas besoin de contenir plus d'informations, le champ date qu'elle contenait a été intégré à la table commande. La table « incoterm » est également apparue. C'est une partie essentielle pour la mise en place d'une commande. En effet, l'incoterm (contraction du terme anglais International Commercial Terms) correspond à un ensemble de termes normalisés qui définissent les droits et devoirs des acheteurs et des vendeurs qui participent à des échanges, aussi bien nationaux qu'internationaux. La première version de la base de données étant essentiellement une version destinée à faire fonctionner un premier prototype, cette table n'y figurait pas.

Enfin, la table « transport » a également été ajoutée. Celle-ci va permettre de stocker les informations de suivi du transport de la commande.

Les autres tables restent assez similaires à la première version.

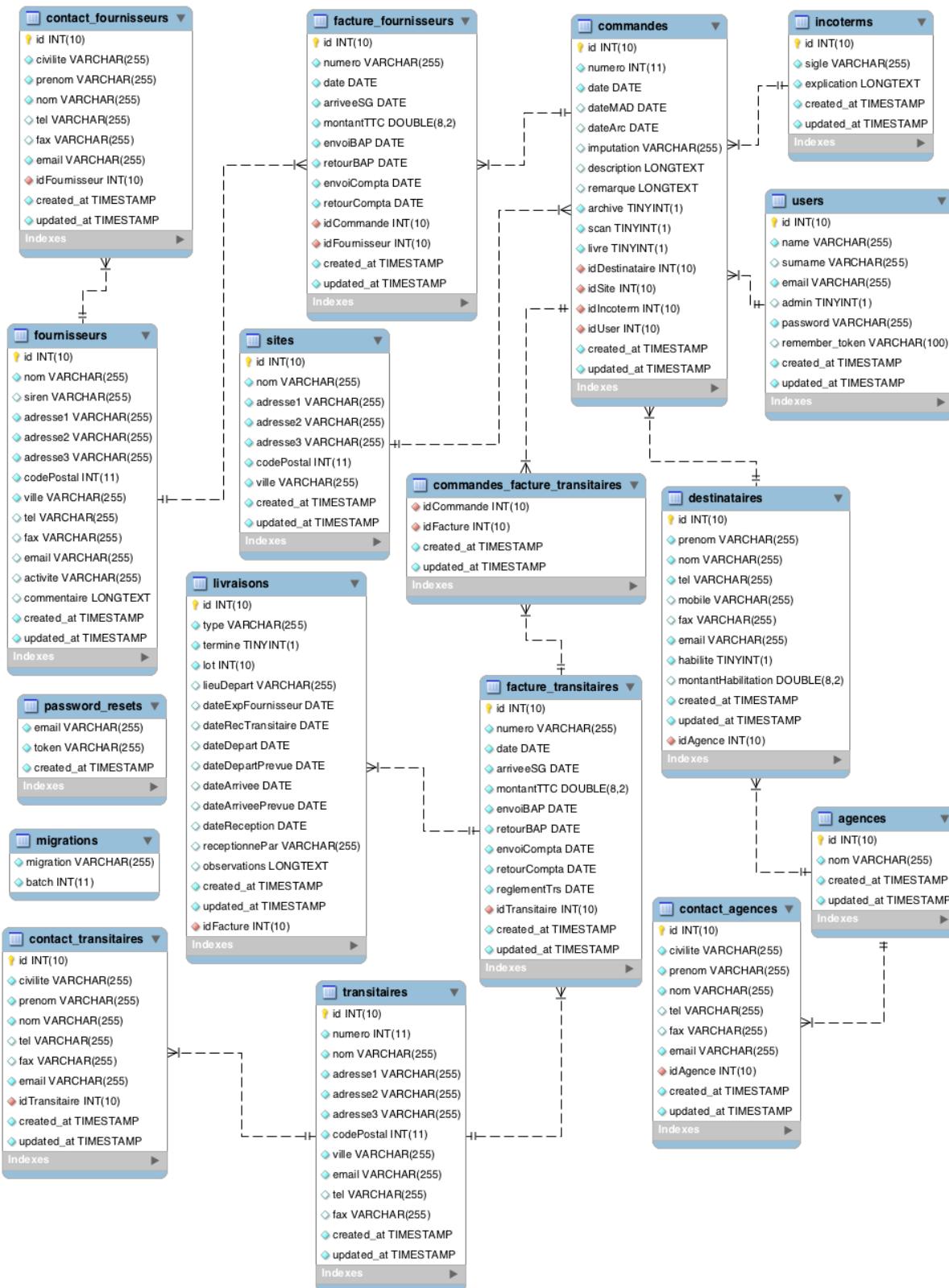


Figure 14 : Dernière version de la base de données

Cette dernière version de la base de données prend en compte plusieurs changements demandés par le service, à savoir, entre autre, la prise en compte du processus d'association des commandes aux factures transitaires. Ce processus sera expliqué plus en détail un peu plus loin dans le présent rapport.

Cette version apporte également de nombreuses améliorations en terme de nom des tables et de noms d'attributs, pour les rendre plus compatibles aux exigences de Eloquent et faciliter un peu plus les requêtes.

Il est à noter qu'une migration de cette ancienne base de données vers la nouvelle est à prévoir car, bien que non obligatoire, celle-ci est souhaitée par le service approvisionnement. Elle n'a pas encore été réalisée au moment où ce rapport est rédigé, cependant, au vue des nouvelles fonctionnalités et des nouvelles tables qui sont apparues, il est possible que cette migration ne soit pas possible en raison de l'incompatibilité des anciennes informations ou du manque d'informations concernant

3.5.3. Modélisation des classes de l'application

Les classes à implémenter font partie de la logique de l'application. Celles-ci représentent les entités utilisées dans l'application, et plus précisément leurs attributs ainsi que les méthodes qu'ils utilisent pour traiter les données. Ces classes s'appuient essentiellement sur les entités réelles manipulées par les agents du service approvisionnement (factures, commandes, fournisseurs etc.) pour les représenter au mieux.

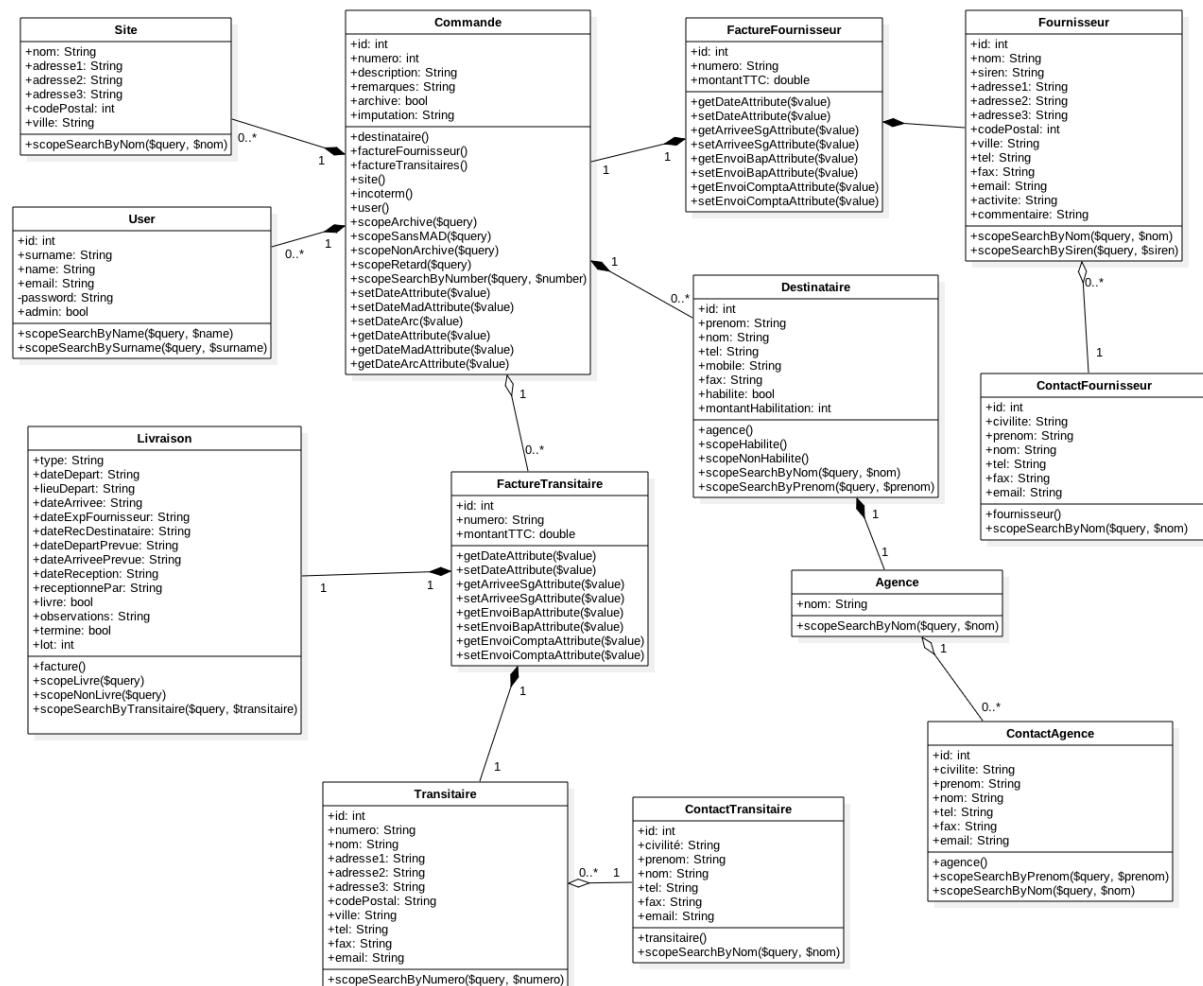


Figure 15 : Première version du diagramme de classes

Cette version du diagramme de classes, illustrée sur la figure ci-dessus, est la version actuelle de la modélisation. Celle-ci a cependant peu évolué par rapport à la base de données. On

constate qu'ici aussi, la classe Commande, bien qu'importante, n'est pas totalement au centre de l'application et que d'autres liens sont apparus entre les autres classes.

Nous constatons également que la plupart des fonctions présentes dans les classes sont des accesseurs, ainsi que des fonctions qui permettent d'effectuer des requêtes spécifiques, notamment pour les recherches. Ces fonctions sont appelées « scope ».

En effet, Laravel est construit sur un modèle MVC, couplé à une ORM puissante appelé Eloquent, qui permet de lier une table avec un modèle simplement en les nommant de la même manière. Ainsi, chacun des objets modélisés possède les mêmes attributs que ceux des tables présentes dans la base de données. Le modèle (représenté par une classe PHP) va permettre de gérer la visibilité de chacun des attributs (protected, private ou public) et va également permettre d'implémenter des fonctions. Les fonctions sont essentiellement destinées à effectuer des requêtes spécifiques avec la base de données. De plus, parmi les fonctions qu'il est possible d'implémenter, certaines d'entre elles sont particulièrement efficace pour la gestion des relations d'appartenance, basées sur les clés étrangères stockées en base de donnée. Ainsi, Eloquent permet, grâce à une simple fonction implémentée dans le modèle, de récupérer un objet qui lui est lié comme s'il s'agissait d'un simple attribut.

3.6. Réalisation des maquettes

Les maquettes présentent de nombreux avantages pour le développement d'une application. Tout d'abord, elles sont une bonne manière de montrer aux clients à quoi ressemblera potentiellement la version finale de l'application. Elle va leur permettre d'émettre une première critique de l'application, non seulement au niveau de l'aspect globale, mais également sur le contenu et la visibilité des fonctionnalités. Elles permettent donc d'avoir une vision un peu plus concrète de ce qui va certainement être réalisé, et permet de s'assurer que chaque partie comprenne bien ce qu'il se passe et soit d'accord sur la direction globale que prend le projet. Elles nous permettent également d'avoir un plan d'organisation précis des interfaces et de gagner du temps lors de l'élaboration de celles-ci. Enfin, au cas où certaines interfaces ne peuvent pas être réalisées, elles permettent à d'autre développeurs de réaliser les vues manquantes.

Les maquettes ont été réalisées uniquement pour certaines vues, le but ici n'étant pas de connaître chaque section de l'application dans le détail, mais de donner une vision globale de l'interface, qui sera reprise par la suite dans les autres vues.

3.6.1. Tableau de bord

Le tableau de bord est une des solutions que j'ai proposées pour améliorer l'application et permettre aux agents du service d'être plus productifs. En effet, ce dernier n'est pas une partie essentielle de l'application. Il vient cependant rajouter une partie très pratique pour la gestion des commandes et a été approuvé par le service approvisionnement.

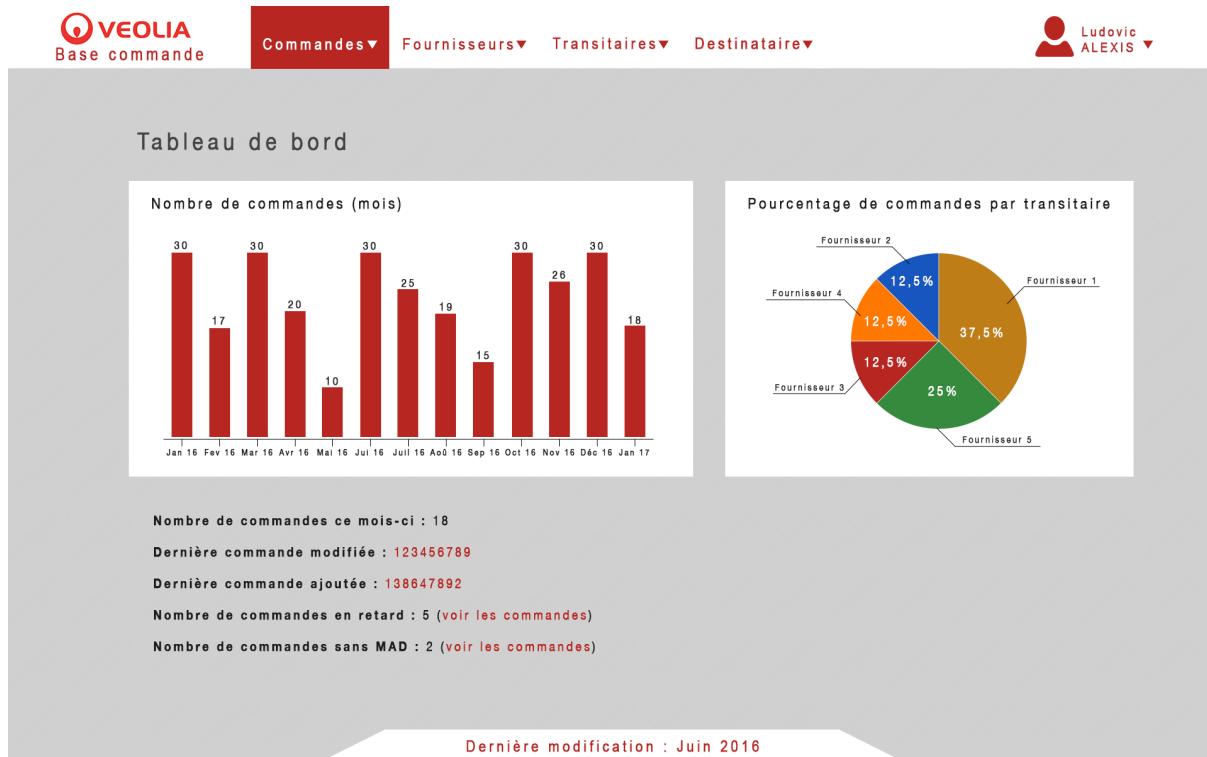


Figure 16 : Maquette du tableau de bord

La figure ci-dessus représente l'interface du tableau de bord de l'application. Ce tableau de bord présentera diverses statistiques, sous forme de texte ou de graphs (histogrammes, camemberts etc...). Les informations présentées sur cette maquette sont là à titre indicatif. Les informations à afficher sur cette page sont encore à déterminer avec le responsable approvisionnement. Cette maquette n'a pour but que de donner une tendance générale du tableau de bord.

La priorité de ce stage reste essentiellement le développement des autres interfaces qui permettent de réaliser toute la procédure d'enregistrement des commandes. Le tableau de bord sera parmi les dernières interfaces réalisées.

3.6.2. Commandes

Il s'agit ici d'une partie importante de l'application. La liste de commande et la page de consultation d'une commande font partie des interfaces les plus importantes de l'application. Un soin particulier leur a été apporté, notamment au niveau de leur organisation. En effet, il s'agit de pages qui peuvent potentiellement présenter une grande quantité d'information.

Résumé

- 1 commande(s) non-archivée
- 2 commandes sans date de MAD
- 1 commandes en retard
- 1 une commande sans BC

[Ajouter une commande](#)

Rechercher une commande

Numéro : [Rechercher](#)

>> [Recherche avancée](#)

Non-Archivées (1)	Archivées (1)	Sans MAD (2)	Retard (1)	Sans BC (1)
Liste des commandes				
Numéro	Date	Description		
123456	12/10/2016	Ipsum dolor amet	Consulter	Supprimer
123456	12/10/2016	Ipsum dolor amet	Consulter	Supprimer
123456	12/10/2016	Ipsum dolor amet	Consulter	Supprimer
123456	12/10/2016	Ipsum dolor amet	Consulter	Supprimer
123456	12/10/2016	Ipsum dolor amet	Consulter	Supprimer
123456	12/10/2016	Ipsum dolor amet	Consulter	Supprimer

Dernière modification : Juin 2016

Figure 17 : Maquette de la liste des commandes

Sur la figure ci-dessus, nous pouvons voir l'interface de la liste des commandes. Au vu du nombre important d'informations qu'il est possible de montrer, il a fallu rester simple au niveau de l'interface pour ne montrer que les informations utiles et éviter la surcharge cognitive de l'utilisateur.

Ainsi, nous pouvons noter que l'interface est composée de 3 sections : le résumé, la barre de recherche et la liste des commandes en elle-même. La partie « résumé » contient des informations sur la quantité de commandes dans chaque catégorie et permet à l'agent d'avoir un aperçu rapide de l'état de ses commandes en général.

Il est également à noter que l'interface respecte la règle de lecture naturelle de l'écran par l'utilisateur, à savoir la lecture en Z. Les informations importantes sont donc situées en haut à gauche, en haut à droite et au milieu de l'écran, permettant ainsi de mieux cerner le contenu de la page dès le premier coup d'œil.

Date	Nom du fichier	Actions
12/10/2016	fichier1.pdf	Consulter Supprimer
12/10/2016	fichier2.pdf	Consulter Supprimer
12/10/2016	fichier1.doc	Consulter Supprimer
12/10/2016	fichier2.doc	Consulter Supprimer
12/10/2016	fichier1.xls	Consulter Supprimer
12/10/2016	fichier2.xls	Consulter Supprimer

Figure 18 : Maquette de la liste des fichiers (consultation de commandes)

La figure ci-dessus illustre la vue de consultation d'une commande. Nous pouvons y voir le numéro de la commande avec, en dessous, les différentes actions disponibles pour cette commande (Générer le BC, Modifier, archiver, supprimer la commande). La vue possède également plusieurs onglets. Ces onglets correspondent aux différentes catégories d'informations sur la commande. En effet, toujours dans le but de réduire la charge cognitive de l'utilisateur, et ainsi lui permettre d'être plus efficace, j'ai dû séparer les informations à afficher.

Ici nous voyons l'onglet document qui est un onglet différent des autres. En effet, celui-ci permet une gestion simple des documents présents dans le dossier de la commande. Il est en effet possible de les supprimer ou de les consulter. Cette gestion des fichiers est complètement indépendante de la base de données. Elle va essentiellement être utilisée pour consulter les bons de commandes générés par l'application, ainsi que les documents scannés qui seront chargés sur le serveur.

La maquette de l'interface d'ajout de commande est une fenêtre intitulée "Ajouter une commande". Elle se présente comme suit :

- Barre de menu et identification :** En haut à gauche, le logo "VEOLIA Base commande". En haut à droite, un bouton avec l'avatar d'un utilisateur nommé "Ludovic ALEXIS".
- Barre d'onglets :** Une barre rouge contenant les liens "Commandes", "Fournisseurs", "Transitaires" et "Destinataire", tous suivi d'un petit triangle indiquant qu'ils sont déroulables.
- Formulaire principal :**
 - Champs obligatoires :** "Numéro" (blanc), "Fournisseur #" (avec un curseur rouge), "Site #" (avec un curseur rouge), "Destinataire #" (avec un curseur rouge), et "Transitaire #" (avec un curseur rouge).
 - Champs facultatifs :** "Date" (blanc), "Date de MAD" (blanc), "Imputation" (blanc), "Description" (grand champ blanc), et "Remarques" (grand champ blanc).
 - Boutons :** "Ajouter" (bleu), "Effacer" (jaune) et "Annuler" (gris).
- Détails en bas :** Un message "Dernière modification : Juin 2016" et une légende "Figure 19 : Maquette de l'ajout de commande".

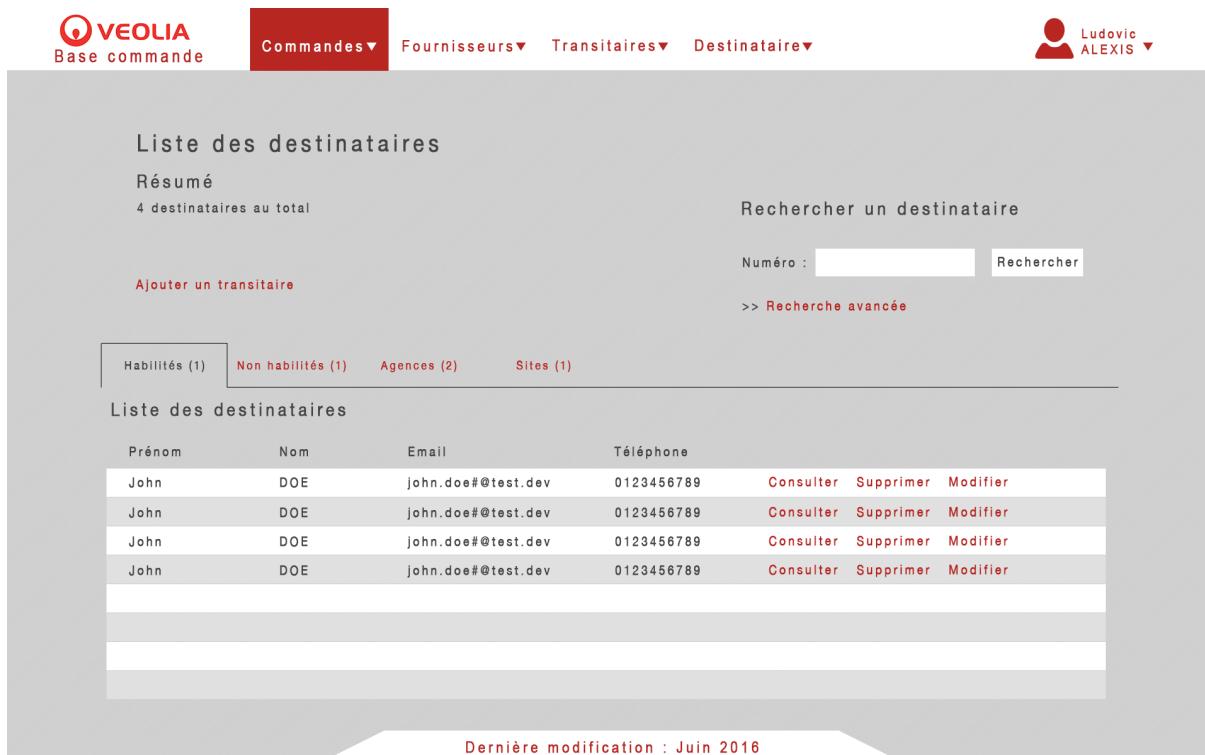
Figure 19 : Maquette de l'ajout de commande

L'ajout de commande passe par l'interface ci-dessus. Elle doit permettre d'ajouter rapidement une commande dans la base, tout en permettant à l'utilisateur nous seulement d'entrer les informations relatives à la commande, mais aussi de la lier aux éléments de la base (site, destinataire, transitaire et fournisseur).

Encore une fois, nous sommes ici devant une interface simple et épurée pour permettre à l'utilisateur de ne pas se perdre. Nous réutilisons également l'expérience utilisateur en mettant les éléments à remplir dans l'ordre auquel sont habitués les utilisateurs et en utilisant un système de formulaire classique pour permettre à l'utilisateur de rentrer les informations.

A noter, la présence du bouton « Effacer » qui va permettre de réinitialiser tout le formulaire, ainsi que la gestion des champs obligatoires qui va afficher un avertissement en rouge sous les champs obligatoires lorsque ceux-ci sont vides, dans le but de prévenir les messages d'erreurs lancés par l'application et incompréhensibles pour l'utilisateur.

3.6.3. Destinataires



La maquette de la liste des destinataires est une interface web qui se présente comme suit :

- Barre de navigation :** En haut à gauche, il y a le logo "VEOLIA Base commande". Ensuite, une barre rouge contenant les liens "Commandes▼", "Fournisseurs▼", "Transitaires▼" et "Destinataire▼". À droite de cette barre, il y a un profil utilisateur avec l'avatar d'un homme et le nom "Ludovic ALEXIS ▼".
- Titre et résumé :** Le titre "Liste des destinataires" est en haut à gauche. En dessous, il y a un résumé indiquant "4 destinataires au total".
- Recherche :** Un champ de recherche "Rechercher un destinataire" avec un bouton "Rechercher" et un lien "Recherche avancée".
- Filtrage :** Des filtres sont disponibles : "Habilités (1)", "Non habilités (1)", "Agences (2)" et "Sites (1)".
- Liste des destinataires :** La liste est présentée dans une table avec les colonnes "Prénom", "Nom", "Email" et "Téléphone". Chaque ligne contient les informations pour un destinataire (John DOE) et des boutons pour "Consulter", "Supprimer" et "Modifier".
- Dernière modification :** En bas de la liste, il y a une indication "Dernière modification : Juin 2016".

Figure 20 : Maquette de la liste des destinataires

L'interface de consultation des destinataires reste dans la même lignée que celle des commandes. La maquette ci-dessus nous montre la liste des destinataires. Cette liste présente le nom, prénom, l'email (cliquable) et le numéro de téléphone du destinataire. Elle propose également 3 actions : consulter la fiche du destinataire, le supprimer ou le modifier.

Au-dessus de la liste, on retrouve les mêmes informations que pour la liste des commandes à savoir le résumé, qui contiendra les informations pertinentes concernant les destinataires.

Le but ici est de garder une cohérence dans l'interface de l'application, encore une fois en vue d'optimiser son utilisation et de faire en sorte que l'utilisateur ne soit pas perdu.

Veolia Base commande

Commandes ▾ Fournisseurs ▾ Transitaires ▾ Destinataire ▾

Ludovic ALEXIS ▾

John DOE

[Modifier le destinataire](#) | [Supprimer le destinataire](#)

Email :	Agence :	Montant de l'habilitation :
john.doe@test.dev	NORD	2000 €
Tel :	Mobile :	Fax :
0293847563	0382234567	0283928374

Liste des commandes passées

Numéro	Date	Description	Consulter	Supprimer	Modifier
123456	12/10/2016	Lorem ipsum dolor amet	Consulter	Supprimer	Modifier
123456	12/10/2016	Lorem ipsum dolor amet	Consulter	Supprimer	Modifier
123456	12/10/2016	Lorem ipsum dolor amet	Consulter	Supprimer	Modifier
123456	12/10/2016	Lorem ipsum dolor amet	Consulter	Supprimer	Modifier
123456	12/10/2016	Lorem ipsum dolor amet	Consulter	Supprimer	Modifier
123456	12/10/2016	Lorem ipsum dolor amet	Consulter	Supprimer	Modifier

Dernière modification : Juin 2016

Figure 21 : Maquette de l'interface de consultation du destinataire

Cette interface présente les principales informations concernant le destinataire ainsi que la liste des commandes qui lui sont attribuées. On y retrouve son mail, sous forme de lien cliquable, qui va permettre d'envoyer un mail en un seul clic à la personne concernée, ce qui va faire gagner du temps à l'utilisateur.

La liste des commandes passées par le destinataire est également disponible. Elle est semblable à la liste des commandes dans la section « commandes » de l'application.

Il est également possible de modifier ou de supprimer le destinataire via les options situées sous son nom.

4. Etablissement des versions

Les fonctionnalités étant nombreuses dans cette application, une bonne organisation est essentielle pour pouvoir mener à bien le projet en un temps minimum et sans oublier de fonctionnalités. Pour cela, chaque fonctionnalité est listée et un planning des versions a été mis en place. Chaque version possède une date de livraison prévue, ainsi qu'une date de livraison officielle qui va permettre d'avoir une idée du retard pris sur la livraison. Ce planning permet également de prévoir des phases de test qui seront importantes pour assurer la qualité de l'application. Les versions ainsi que leur date de livraison seront listées ci-dessous.

4.1. Version 0.1 (prototype)

Il s'agit de la version qui servira de démonstration aux membres de l'équipe projet. Elle servira également à prendre en main l'outil Laravel. Les fonctionnalités qui devront être implémentées seront forcément basiques mais doivent être également assez complètes pour pouvoir effectuer une première démonstration.

Parmi ces fonctionnalités on retrouve :

- L'affichage de la liste des Commandes
- L'affichage de la liste des Fournisseurs
- L'affichage de la liste des Transitaires
- L'affichage de la liste des Destinataires
- La classification des commandes (Archivées, non archivées, Sans MAD, en retard)
- L'ajout/suppression de commandes dans la base

La date de livraison de cette version a été fixée au 29 mars 2016, soit un mois après le début du stage.

4.2. Version 0.2 (version alpha)

Cette version sera la version alpha et devra me permettre d'effectuer de premiers tests sur l'application.

Elle devra comporter les fonctionnalités suivantes :

- Recherche par numéro de commande
- Consultation des fiches commandes
- Modification/archivage des commandes
- Ajout/modification/suppression de fournisseurs
- Ajout/modification/suppression de transitaires
- Ajout/modification/suppression de destinataires

De plus, les URL utilisées pour chaque section devront être définitivement arrêtées. Cette version sera prévue pour le 8 avril 2016.

4.3. Version 0.3 (version beta)

La version beta est la version quasi-complète, qui implémente toutes les fonctionnalités principales. Il est possible que celle-ci comporte encore quelques erreurs mais elle peut permettre de

- Création des dossiers de stockage des documents concernant les commandes
- Consultation des fiches fournisseurs
- Consultation des fiches transitaires
- Consultation des fiches destinataires
- Recherche avancée de commande

Cette version est prévue pour le 15 avril 2016.

4.4. Version 1.0

La version 1.0 doit permettre à l'utilisateur d'effectuer la plupart des fonctions principales de l'application sans erreurs, à savoir :

- L'authentification des utilisateurs
- La consultation du tableau de bord
- La gestion des factures
- Création des bons de commande à partir des modèles Word
- Ajout des fichiers (scan) des documents dans le dossier correspondant (via l'interface)
- Recherche des fournisseurs
- Recherche des transitaires
- Recherche des destinataires
- La gestion des erreurs

Cette version de l'application est prévue pour le 29 avril 2016.

4.5. Version 1.1

La version 1.1 vient maintenant comme une amélioration. C'est ici que nous allons mettre de la valeur ajoutée à ce qui existait déjà dans l'entreprise. Pour cela, il va falloir mettre en place le système de relances par mail. Ce système devra repérer les commandes en retard ou sans date de mise à disposition, pour ensuite les mettre dans la bonne liste et permettre d'envoyer le mail de relance correspondant. L'amélioration du design fait également partie des améliorations notables de cette version. En effet, toutes les fonctionnalités étant mise en place dans la version 1.0, il est plus facile de se pencher sur le design et les derniers détails d'organisation des vues. Cette version est prévue pour le 6 mai 2016.

4.6. Version 1.2

Cette version doit contenir des améliorations, concernant essentiellement le tableau de bord mais également des fonctionnalités comme la gestion des comptes utilisateurs par l'administrateur. Nous donnerons également plus de place aux requêtes AJAX pour optimiser la bande passante et récupérer uniquement les informations nécessaires. Cette version est prévue pour le 14 mai 2016.

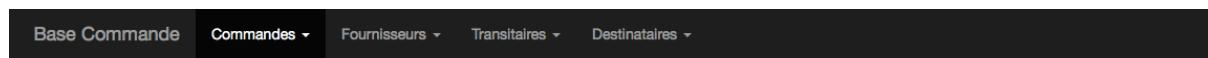
5. Phase de réalisation

Cette partie du rapport se concentre essentiellement sur la mise en place des fonctionnalités. Elle est décomposée par versions. Nous verrons tout au long de cette partie que le planning des versions n'a pas pu être respecté entièrement à la lettre, notamment à cause des demandes et des nouvelles informations qui sont arrivées en cours de développement, dont entre autre après la validation de la première version.

5.1. Prototype (version 0.1)

Cette version prototype a été réalisée en parallèle du choix des technologies, pendant la phase de conception. Le prototype est arrivé très rapidement dans le temps. En effet, celui-ci m'a permis d'utiliser le framework Laravel avant de le valider définitivement comme étant le framework de développement final de l'application. Il m'a également permis de tester Bootstrap un peu plus, pour ainsi me familiariser avec les éléments qu'il propose. Par la suite j'ai pu, grâce à cette première version, améliorer ma maquette de manière à la rendre un peu plus réaliste. En effet, il est difficile de proposer des maquettes qui soient à 100% fidèles au résultat final, notamment à cause de la grande différence de tailles d'écrans.

Ce prototypage m'a donc permis d'avoir une idée de l'agencement possible au niveau de l'interface.



Liste des commandes

Résumé

2 commande(s) non archivées
0 commande(s) sans date de MAD
0 commande(s) en retard

Ajouter une commande

Recherche de commande

Numéro Recherche
>> Recherche avancée

Non-archivées (2) Archivées (0) Sans MAD (0) En retard (0)

Commandes non-archivées

Numéro	Date	Description	Actions
11111	29/03/2016	test	Consulter Modifier Supprimer Archiver
222222	29/03/2016	test	Consulter Modifier Supprimer Archiver

Figure 22 : Vue de consultation des commandes (prototype)

Sur la figure ci-dessus, nous pouvons voir la toute première interface mise en place dans l'application. Très simple, elle utilise le thème par défaut de Bootstrap pour mettre en place le cadre général de la vue ainsi que de l'application en général. En effet, toutes les autres vues vont suivre la même logique au niveau de l'organisation.

5.2. Version Alpha (0.2)

Dans cette version, la base de données a subi une modification importante, notamment au niveau des livraisons. En effet, il a fallu modifier les clés étrangères pour permettre à la base de données de prendre en compte le fait qu'une livraison puisse être composée de plusieurs lots.

De plus, les contacts sont séparés en contacts fournisseurs et contacts transitaires pour améliorer leur recherche lorsque des requêtes sont effectuées en base de données. De plus, plusieurs contacts fournisseurs (respectivement transitaires) peuvent être associés à un fournisseur (respectivement transitaire).

Au niveau de l'interface, des librairies JavaScript et CSS pour l'utilisation de sélecteurs de dates sont ajoutées. En effet, les sélecteurs de dates ne sont pas implémentés par défaut dans tous les navigateurs, or ceux-ci sont très pratiques pour la normalisation des dates en vue de leur enregistrement en base de données, ainsi que leur saisie (en un seul clic).

5.3. Version Bêta (0.3)

Cette version représente une version de test pour le design. En effet, sur cette version, le design prévu sur la maquette a été implémenté sur une partie de l'application. Les premières icônes ont également été ajoutées grâce à l'utilisation de la librairie FontAwesome.

Des problèmes sont apparus aussi pour l'accès aux fichiers sur le réseau des disques partagés de Veolia. En effet, Microsoft ne permet pas à son service IIS d'accéder aux disques partagés pour des raisons de sécurité. Il a donc été impossible de stocker les bons de commande dans le même dossier que celui dans lequel ils étaient stockés auparavant. Cette version stocke donc les fichiers sur le même serveur qui contient l'application.

La version bêta comprend également la mise en place des premières versions de chargement de documents. Ce chargement permet de stocker les fichiers sur le serveur et d'afficher le contenu du dossier. Les fichiers sont triés en fonction de leur extension et des icônes adaptées sont affichées. Cela permet encore une fois à l'utilisateur de mieux se repérer au sein de l'application et de voir ses documents du premier coup d'œil.

5.4. Version 1.0

La version 1.0 présente des améliorations majeures par rapport aux anciennes versions. Entre autre, il s'agit d'une version totalement fonctionnelle, dans laquelle il est possible de dérouler entièrement tout le processus d'enregistrement de commande, y compris l'enregistrement de documents sur le serveur. La fonction de génération des bons de commandes est encore absente.

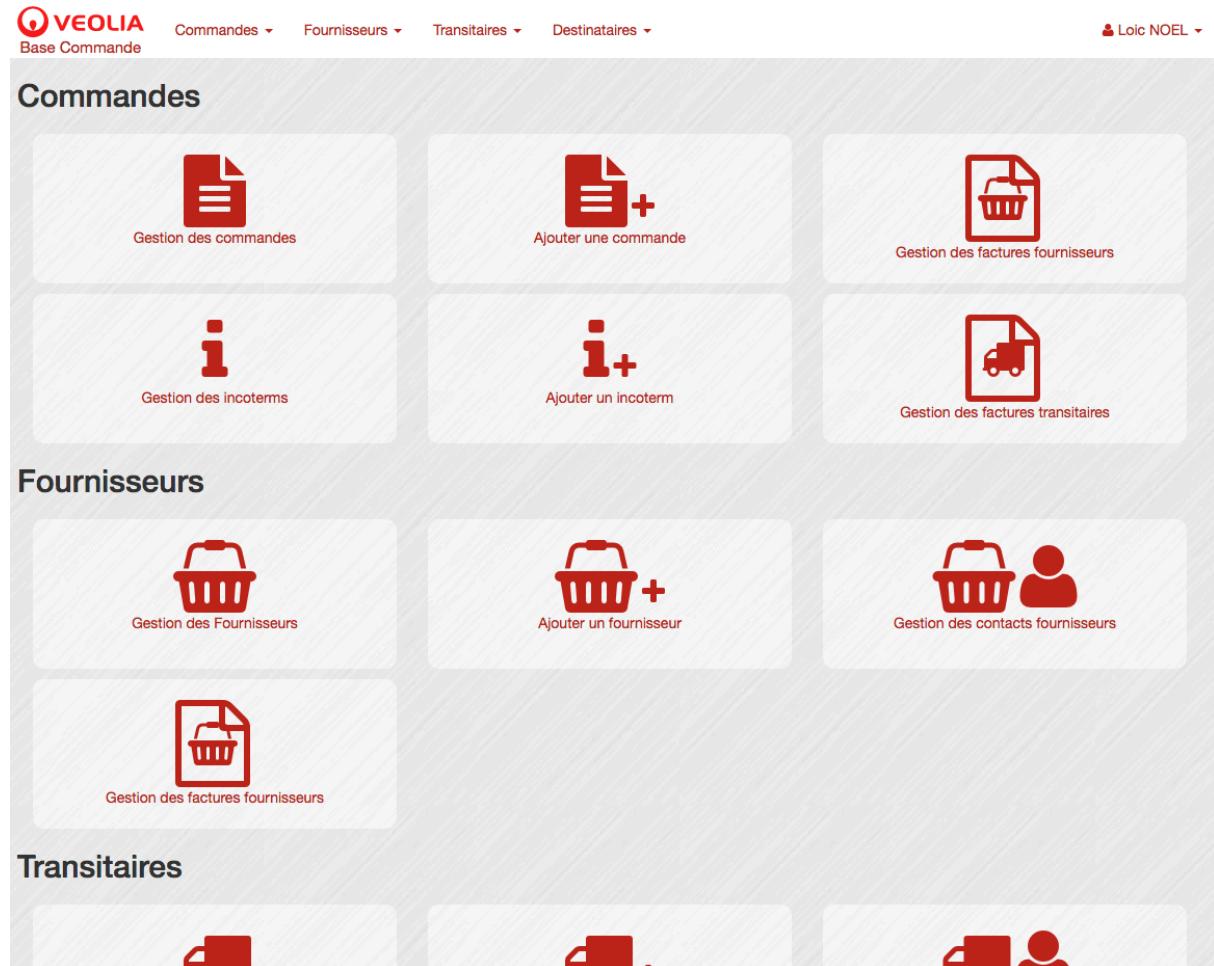


Figure 23 : Page principale de l'application

Ci-dessus, la page principale de l'application qui a suivi un chemin différent de celui qui avait été prévu initialement. En effet, dans la partie 3.6.1, nous pouvons voir que sur les maquettes, cette première page consistait en un tableau de bord contenant de nombreuses informations sur l'application et les commandes en elles même, notamment sous forme de graphs et de statistiques. Il a cependant été décidé que ces graphes constituerait une page à part, et que la page de démarrage devrait uniquement permettre aux agents d'accéder rapidement à la page de l'application dont ils ont besoin. Pour cela, l'interface reste simple, composée de carrés blancs transparents dans lesquels on trouve des icônes (les plus explicites possibles) ainsi qu'un texte indiquant la fonction implémentée. Ces ensembles de carrés représentent des liens vers les pages principales de l'application. Ces pages sont classés en fonction de leur catégorie.

Résumé

2 commande(s) non archivées
1 commande(s) sans date de MAD
4 commande(s) en retard

Ajouter une commande

Recherche de commande

Numéro Recherche [>>> Recherche avancée](#)

[En cours \(2\)](#) [Archivées \(4\)](#) [Sans MAD \(1\)](#) [En retard \(4\)](#) [Sans BC \(?\)](#)

Commandes non-archivées

Numéro	Date	Description	Actions
333222	28/04/2016	Nouveau non modifié	Consulter Modifier Supprimer Archiver
111111	27/04/2016	dedede	Consulter Modifier Supprimer Archiver

Base Commande v1.0 - Dernière modification : mai 2016

Figure 24 : Vue de consultation des commandes (v1.0)

Ci-dessus, nous pouvons voir l'interface de la consultation des commandes qui a subi une modification mineure par rapport à la version bêta. En effet, ici le design réalisé pour la maquette a été validé et appliqué, moyennant quelques légères modifications. Parmi elles, nous pouvons noter la réduction de la taille de la barre de navigation pour permettre d'afficher plus d'informations à l'écran et permettre à l'utilisateur de tout voir sans faire défiler la page vers le bas. Nous notons également l'ajout définitif des icônes qui aident au repérage des différentes catégories et fonctionnalités. Enfin, la barre de navigation contient maintenant le nom de l'utilisateur. En effet, la fonction de connexion et de déconnexion est maintenant disponible.

5.5. Version 1.1

Cette version comporte des modifications légères par rapport à la version 1.0 au niveau de la base de données. Il s'agit surtout de modification dans les migrations pour adapter certains champs de la base de données au comportement voulu pour l'application.

Il a également fallu régler certains problèmes de droit d'accès aux documents après l'upload sur le serveur. En effet, PHP utilise un système de stockage temporaire des fichiers avant de les stocker dans le dossier correspondant. Ainsi, le fichier hérite des droits du dossier dans lequel il a été stocké. Or ce dossier n'étant pas accessible par défaut, le fichier devient inaccessible lui aussi. La solution a été de modifier le dossier temporaire de PHP.

Enfin, la génération de documents Excel grâce à un modèle prédéfini a également été ajouté et fonctionnel. L'utilisateur peut choisir de générer le document excel, qui sera stocké sur le serveur et également téléchargé sur son poste. Il pourra alors y apporter des modifications

si nécessaires et charger la version PDF sur le serveur. Celui-ci lui permettra ensuite de fusionner ce bon généré avec les documents scannés

5.6. Version 1.2

La version 1.2 de l'application comprend une modification majeure au niveau de l'ajout de facture transitaire. En effet, après la présentation et la validation de la version 1.1, le responsable approvisionnement m'a fait part d'une nouvelle demande concernant le processus d'enregistrement des factures transitaire. En effet, il arrive qu'il choisisse de regrouper plusieurs commandes ensemble chez un même transitaire, pour ensuite faire livrer la totalité chez Veolia. Son problème principal est de lier la facture établie par le transitaire, ainsi que la livraison, à la commande.

Pour adapter l'application et répondre à son besoin, il a fallu modifier la base de données (comme expliqué en fin de partie 3.5.2) et implémenter le processus permettant de lier facture, commandes et livraison en modifiant le processus d'ajout de facture déjà présent. Celle-ci présente maintenant plusieurs étapes qui vont être expliquées ci-dessous.

The screenshot shows a web-based application interface for Veolia's 'Base Commande'. At the top, there is a navigation bar with links for 'Commandes', 'Fournisseurs', 'Transitaires', and 'Destinataires'. On the right side of the header, there is a user profile icon and the name 'Loic NOEL'. The main content area is titled 'Etape 1 : Ajouter une facture transitaire'. Below the title, there is a form with several input fields:

- Transitaire:** A dropdown menu set to 'Transitaire1' with a '+' button to 'Ajouter un transitaire'.
- Arrivée SG le *:** An input field for arrival date.
- Numéro de facture *:** An input field for invoice number.
- Date de la facture *:** An input field for invoice date.
- Montant TTC *:** An input field for total amount.
- Date envoi BAP *:** An input field for BAP delivery date.
- Date retour BAP *:** An input field for BAP return date.
- Date envoi comptabilité *:** An input field for accounting delivery date.
- Date règlement trs *:** An input field for payment date.

At the bottom of the form, a note states '(Les champs suivis d'un * sont obligatoires)'. Below the note are three buttons: '+ Ajouter' (red), 'Effacer' (white), and 'Annuler' (white).

At the very bottom of the page, there is a footer note: 'Base Commande v1.2 - Dernière modification : mai 2016'

Figure 25 : Interface d'ajout de facture transitaire (Etape 1)

La figure précédente représente l'étape 1 de l'ajout de facture transitaire. C'est une étape simple au cours de laquelle il faut renseigner toutes les informations obligatoires de la facture. Cette étape reste la même que celle des anciennes versions et est venue logiquement se placer en premier.

Etape 2 : Associer des commandes

Numéro de commande : 12 |

Recherche en cours...

(C)

Commandes ajoutées :

Numéro	Date	Description	
123456	18/4/2016	test	<input type="button" value="Retirer"/>

+ Ajouter Effacer Annuler

Base Commande v1.2 - Dernière modification : mai 2016

Figure 26 : Interface d'association de commandes à la facture transitaire

La deuxième étape représentée sur la figure ci-dessus consiste à associer les commandes à la facture. Il s'agit ici de la première fonctionnalité implémentée en AJAX dans l'application. Le principe ici devait rester simple : l'utilisateur tape le numéro de la commande qu'il veut associer dans la barre de recherche. Il s'agit d'une barre de « live search », qui exécute une recherche à chaque fois que le numéro dans la barre est modifié. Le numéro des commandes commençant par ce numéro s'affichent ensuite dans la zone des résultats à droite de l'écran. Une image GIF animée permet d'indiquer à l'utilisateur que la recherche est en cours dans la base de données. Une fois celle-ci terminée, l'image animée disparaît et les résultats sont affichés dans la même zone. Les numéros des commandes apparaissent alors sous forme de liens. Il est ensuite possible d'ajouter ces commandes à la liste des commandes concernées par la facture. Pour cela, il suffit de cliquer sur le lien correspondant à la commande.

Ce type de barre de recherche instantanée permet à l'utilisateur de limiter le nombre de clics à effectuer pour trouver une commande et lui permet de gagner du temps. En effet, en fonction du nombre de commandes à mettre dans la liste, il peut parfois y avoir un nombre important de requêtes à faire. Ce système limite ainsi le nombre d'actions requises de la part de l'utilisateur et rend le travail moins rébarbatif.

Cependant, le problème principal posé par ce genre de recherches est que si l'utilisateur connaît le numéro de sa commande, par exemple 1234567, et tape ce numéro dans la barre de recherche, ce sont 7 requêtes qui sont effectuées alors qu'une seule aurait suffi. Pour remédier à ce problème, j'ai mis en place un retardateur JavaScript. Lorsque l'utilisateur aura fini de taper le numéro, ce dernier sera mis en mémoire par l'application qui attendra 1500ms. Une fois ce délai dépassé, l'application comparera le contenu de la barre de recherche avec celui qu'elle a en mémoire. S'il s'agit du même contenu, cela voudra dire que l'utilisateur aura fini de taper le numéro de commande et la requête pourra être exécutée. Autrement, le contenu de la barre sera mis en mémoire et l'application attendra de nouveau 1500ms avant d'effectuer le contrôle.

Une troisième étape doit venir compléter les deux premières, à savoir l'association d'une livraison. C'est une étape simple comme l'étape 1 dans laquelle il faut simplement renseigner les informations concernant la livraison.

6. Décomposition des tâches et sous-tâches

Nous nous attarderons, dans cette dernière partie, sur le diagramme de gantt effectif du projet. Pour des raisons pratiques, seule les grandes étapes sont représentées, surtout pour les phases de réalisation. Il est à noter que plusieurs modifications ont été apportées aux modélisations pour chaque version de l'application, mais que ces étapes de modélisation ne sont pas affichées ici, car sous-entendues dans la tâche de réalisation de chaque version.

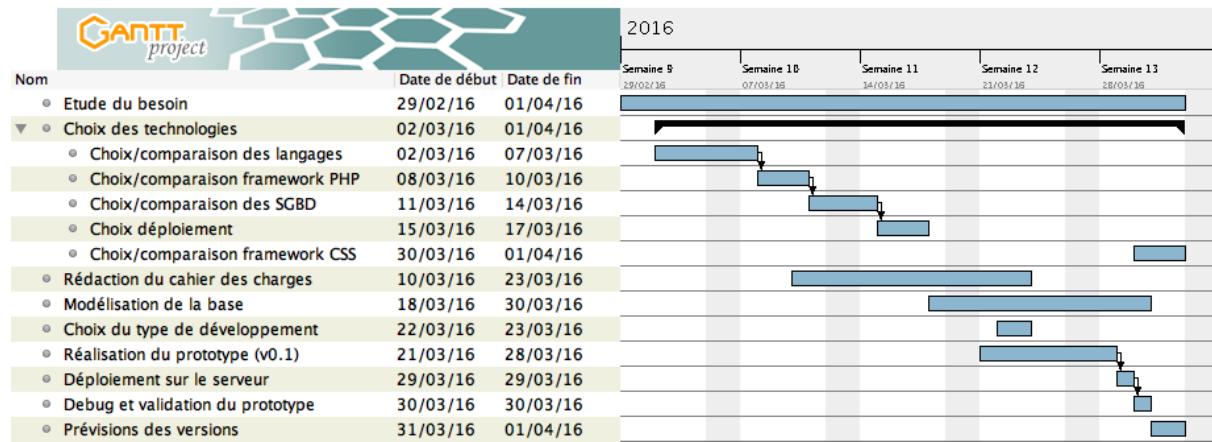


Figure 27 : Décomposition des tâches du 29 février au 4 avril

La partie du stage qui a eu lieu du 29 février au 4 avril (sur la figure ci-dessus), est essentiellement une phase de conception et de réalisation du prototype. Cette phase s'est déroulée sur un peu plus d'un mois avec l'étude du besoin qui a eu lieu tout au long de la période. En parallèle de cette étude du besoin, une étude des technologies, avec comparatif et choix de chacune d'entre elle a été menée, ainsi que la rédaction du cahier des charges. La réalisation du prototype et sa validation en fin de premier mois ont permis de donner plus rapidement un aspect concret au projet.

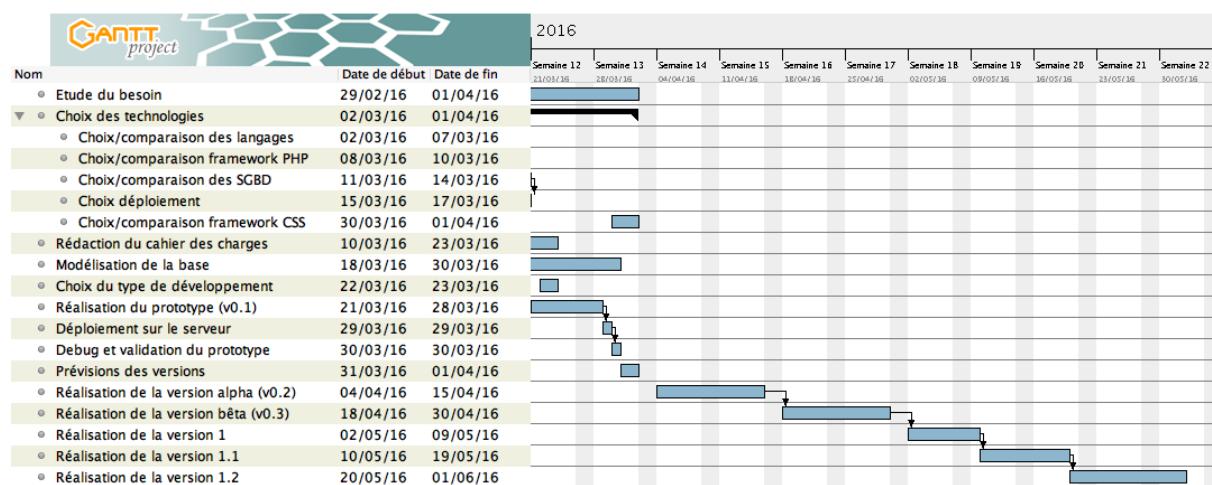


Figure 28 : Décompositions des tâches du 21 mars au 1 juin

La deuxième partie de ce stage, représentée sur le diagramme ci-dessus, est consacrée essentiellement à la réalisation des versions, du 4 avril au 1^{er} juin. A l'heure actuelle, la version 1.2 est encore en développement.

7. Conclusion

Ce stage m'est apparu comme une expérience très satisfaisante et enrichissante. Faire partie d'une équipe aussi dynamique et accueillante, travailler dans d'aussi bonnes conditions et pouvoir mettre en pratique le savoir que j'ai acquis sont autant de choses positives que j'en retire.

J'ai pu, grâce à ce stage, améliorer mes compétences sociales et élargir mes connaissances, en étant confronté à des professionnels de plusieurs corps de métier, comme la communication, le bâtiment, la qualité ou encore la distribution de l'eau potable, cœur de métier de Veolia Eau, avec qui j'ai pu échanger. J'ai également pu en apprendre plus sur les actions menées par un service informatique au sein d'une entreprise, leur rôle, passant de la maintenance à l'élargissement du réseau, ou encore à l'élaboration d'outils.

Le projet en lui-même a également représenté une bonne expérience pour moi. En effet, la programmation d'application web est une tendance de plus en plus adoptée par les entreprises de par ses nombreux avantages. De plus, j'ai déjà pu me former, par le biais de stage précédents, à l'élaboration de site web. Ainsi cette application arrive comme la conclusion de ma formation, avec une mise en pratique de tous les aspects de la programmation que j'ai pu rencontrer au cours de mon cursus.

Le résultat de ce stage est donc une application réalisée dans le respect de plusieurs règles de bonne pratiques de programmation web, qui permettent d'optimiser l'application et de faire entrer dans le projet dans une dimension Green IT. Parmi ces bonnes pratiques, on retrouve l'optimisation des données envoyées, l'utilisation de framework ou encore l'utilisation d'un design simple et adapté au web. Pour compléter ces bonnes pratiques et autres démarches adoptées, l'accent a été mis sur le choix de plusieurs outils connus et faciles d'apprentissage pour permettre une maintenance plus efficace. En effet la maintenance d'une application représentant la plus grande partie de son temps de vie, et les gains pour une application de cette taille n'étant visibles que sur le long terme, il est important de permettre à l'application d'être maintenue facilement et rapidement.

L'application en elle-même est fonctionnelle et implémente les mêmes fonctionnalités que l'ancienne application pour l'instant. Toute la partie de relance des mails est également prête à être déployée, mais n'a pas encore été validée par le service approvisionnement.

De plus il est nécessaire de rappeler qu'à la date de rendu de ce rapport, le stage n'est pas encore terminé, puisqu'il se termine le 2 septembre 2016. Ainsi donc, pendant les mois restant, d'autres améliorations seront apportées à l'application pour lui permettre d'être encore plus performante, comme par exemple l'introduction de script JavaScript dans les pages, le découpage plus fin des feuilles de script CSS ou encore l'utilisation de requêtes AJAX pour réduire la quantité de données circulant sur le réseau.

8. Sources

Voici la liste des sources utilisées pour réaliser ce document.

PHP :

- <https://fr.wikipedia.org/wiki/PHP>
- <http://php.net>

C# :

- https://fr.wikipedia.org/wiki/C_sharp
- <https://msdn.microsoft.com/en-us/library/kx37x362.aspx>
- [https://msdn.microsoft.com/fr-fr/library/bb470252\(v=vs.100\).aspx](https://msdn.microsoft.com/fr-fr/library/bb470252(v=vs.100).aspx)

Ruby :

- <https://www.ruby-lang.org/fr/>
- <https://fr.wikipedia.org/wiki/Ruby>
- <http://www.linuxdevcenter.com/pub/a/linux/2001/11/29/ruby.html>

JavaScript :

- <https://fr.wikipedia.org/wiki/JavaScript>
- <https://developer.mozilla.org/fr/docs/Web/JavaScript>
- <https://www.javascript.com>

Adobe Cold Fusion :

- https://en.wikipedia.org/wiki/Adobe_ColdFusion
- <http://www.adobe.com/products/coldfusion-standard.html>

Frameworks PHP :

- <https://www.linkedin.com/pulse/7-best-php-frameworks-2015-winspire-web-solution>
- <http://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/>

Laravel :

- <https://laravel.com>
- <https://en.wikipedia.org/wiki/Laravel>
- <https://openclassrooms.com/courses/decouvrez-le-framework-php-laravel>

Symfony :

- <https://symfony.com>
- <https://fr.wikipedia.org/wiki/Symfony>
- <https://openclassrooms.com/courses/developpez-votre-site-web-avec-le-framework-symfony2/symfony2-un-framework-php>

Cake PHP :

- <http://cakephp.org>
- <https://en.wikipedia.org/wiki/CakePHP>
- <https://www.grafikart.fr/formations/cakephp>

Zend :

- <http://framework.zend.com>
- <https://github.com/zendframework/zf2>
- http://www.z-f.fr/page/quest_ce_que_le_zf

Comparatif Bases de données :

- <http://www.sitepoint.com/mysql-compared-with-postgresql/>
- https://www.wikivs.com/wiki/MySQL_vs_PostgreSQL
- <http://db-engines.com/en/ranking>
- <http://www.silicon.fr/bases-donnees-nosql-pas-ombre-sgbdr-96184.html>
- <http://www.sitepoint.com/sql-vs-nosql-differences/>
- <http://www.silicon.fr/bases-donnees-oracle-leader-percee-nosql-135094.html>

Oracle Database :

- <http://docs.oracle.com/database/121/index.htm>
- https://fr.wikipedia.org/wiki/Oracle_Database

MySQL :

- <https://www.mysql.fr>
- <https://fr.wikipedia.org/wiki/MySQL>

Microsoft SQL Server :

- <https://www.microsoft.com/fr-fr/server-cloud/products/sql-server/overview.aspx>
- https://fr.wikipedia.org/wiki/Microsoft_SQL_Server

PostgreSQL :

- <http://www.postgresql.org>
- <https://fr.wikipedia.org/wiki/PostgreSQL>

Mongo DB :

- <https://www.mongodb.org>
- <https://fr.wikipedia.org/wiki/MongoDB>

Comparatif frameworks CSS :

- <http://www.sitepoint.com/5-most-popular-frontend-frameworks-compared/>
- http://www.cssnewbie.com/best-free-css3-frameworks-2015/#.VvuxaWMc_8c
- <http://www.sitepoint.com/top-10-front-end-development-frameworks/>
- <http://responsive.vermillion.com/compare.php>
- <http://usablica.github.io/front-end-frameworks/compare.html>

Docker :

- <https://www.docker.com>
- <https://fr.wikipedia.org/wiki/Docker>

Vagrant :

- <https://www.vagrantup.com>
- <https://fr.wikipedia.org/wiki/Vagrant>

Apache :

- <http://www.apache.org>
- https://fr.wikipedia.org/wiki/Apache_HTTP_Server

IIS :

- <http://www.iis.net>
- https://fr.wikipedia.org/wiki/Internet_Information_Services

Éco-conception web :

- <https://checklists.opquast.com/ecoconception-web/>

9. Table des figures

Figure 1 : Logotype Veolia Eau	5
Figure 2 : Organigramme	6
Figure 3 : Interface principale du formulaire de la base commande sous Microsoft Access....	9
Figure 4 : Répartition des sites web par technologies (source : w3techs.com)	11
Figure 5 : Popularité des frameworks PHP en entreprise (source : sitepoint.com).....	14
Figure 6 : Classement de popularité des SGBDs (source : db-engines.com)	16
Figure 7 : Evolution de la popularité des SGBDs (source : db-engines.com)	17
Figure 8 : Diagramme de cas d'utilisation (commandes)	22
Figure 9 : Diagramme de cas d'utilisation (Utilisateurs).....	23
Figure 10 : Diagramme de cas d'utilisation (Transitaires)	23
Figure 11 : Schéma de la base sous Microsoft Access.....	24
Figure 12 : Première version de la base de données.....	26
Figure 13 : Deuxième version de la base de données	27
Figure 14 : Dernière version de la base de données	28
Figure 15 : Première version du diagramme de classes	29
Figure 16 : Maquette du tableau de bord	31
Figure 17 : Maquette de la liste des commandes.....	32
Figure 18 : Maquette de la liste des fichiers (consultation de commandes).....	33
Figure 19 : Maquette de l'ajout de commande	34
Figure 20 : Maquette de la liste des destinataires	35
Figure 21 : Maquette de l'interface de consultation du destinataire	36
Figure 22 : Vue de consultation des commandes (prototype)	39
Figure 23 : Page principale de l'application.....	41
Figure 24 : Vue de consultation des commandes (v1.0)	42
Figure 25 : Interface d'ajout de facture transitaire (Etape 1)	43
Figure 26 : Interface d'association de commandes à la facture transitaire	44
Figure 27 : Décomposition des tâches du 29 février au 4 avril	45
Figure 28 : Décompositions des tâches du 21 mars au 1 juin.....	45

10. Lexique

Bon de commande (BC) : Le bon de commande est le document qui permet de signifier l'intention d'engager une transaction commerciale. Il existe deux types de bons de commandes au sein de Veolia. Le premier est le bon de commande issu du carnet de bon de commande, mis à la disposition des agents habilités. Ce bon de commande permet à l'agent de signaler son besoin au service approvisionnement. Le deuxième est le bon de commande généré par l'application. Le bon de commande de l'agent est généralement joint au bon de commande généré par l'application. Ce dernier permet d'officialiser l'intention de faire la demande auprès du fournisseur.

Date MAD : La date de MAD (pour date de mise à disposition) correspond à la date à laquelle le fournisseur s'engage à mettre la commande à disposition de Veolia. Il arrive que cette date de MAD ne soit pas respectée pour diverses raisons. La commande va alors nécessiter une relance.

Date Arc : La date Arc (pour date d'accusé de réception) correspond à la date à laquelle le fournisseur reçoit la commande.

Imputation : L'imputation est un outil comptable, présenté sous la forme d'un code alphanumérique à 27 caractères. L'imputation comporte plusieurs informations. Grâce à l'imputation, il est possible de savoir à qui est attribué la commande, et donc sur quel budget sera prélevé le coût de la commande. L'imputation permet également de connaître à quelle catégorie appartient la commande. Il peut s'agir de dépenses générales, de sous-traitance etc. Enfin, la dernière partie de l'imputation concerne les chantiers auxquels sont affectés les commandes.

Incoterm : L'incoterm (contraction de l'anglais International Commercial Terms) sont des termes normalisés pour les échanges nationaux et internationaux. Ces incoterms permettent de définir les « droits et devoirs » de chacun des parties lors des échanges.

ORM : L'Object Relationship Mapping (ou ORM) est une technique de programmation qui permet de gérer une base de données orientée comme une base orientée objet, et ce de manière transparente pour le développeur. Dans notre cas, l'ORM permet également d'interagir plus rapidement avec la base de données, en passant par des requêtes standardisées et optimisées.

RDP : Le Remote Desktop Protocol est un protocole d'accès à distance à des serveurs exécutant Microsoft Terminal Services.

IIS : Internet Information Services (IIS) est un serveur Web développé par Microsoft et présent sur ses systèmes d'exploitation Windows Server.