Stagiaire : **Bastien JORGE**	*Branche :* **ISI/SSI** *Année :* **2016**
Responsable pédagogique UTT : **Marc LEMERCIER**	*Semestre :* **P16**

Université de technologie de Troyes

Analyse, Conception, Développement et Sécurisation de l'application client de gestion de rechargement de véhicules électriques

Ce stage s'est déroulé dans l'entreprise *Park'n Plug*, une startup informatique et électricité spécialisée R&D.

Il a consisté en une refonte d'un outil préexistant afin d'y ajouter des fonctionnalités, améliorer l'expérience utilisateur, et renforcer sa maintenabilité. J'ai pu, dans le cadre de la gestion du projet, faire un travail d'analyse et conception poussé, mettre en place des process de développement, concevoir et migrer une base de données complexe, et maquetter l'application.

Le deuxième volet concernait l'amélioration de la sécurité de l'ensemble du système d'information de *Park'n Plug*, que ce soit au niveau serveur, clients, postes de travail ou communications.

L'enjeu de cette intégration est double : concevoir une interface utilisateur efficace et intuitive qui sera au cœur du produit commercial, mais aussi permettre d'avoir l'assurance d'être dans un espace de travail et un SI sains afin de réduire les menaces pesant sur la sécurité.

Entreprise : **Park'n Plug**

Lieu : **Saint-André les Vergers**

Responsable : **Samy Jacquet**

Mots clés

1. Gestion de projets
2. Génie logiciel
3. Logiciels – Recherche
4. Sécurité des systèmes

utt université de technologie Troyes

park'n plug RECHARGE VÉHICULE ÉLECTRIQUE

Remerciements

Je tiens tout d'abord à remercier Pascal Toggenburger, Président Directeur Général, pour m'avoir offert l'opportunité unique de piloter un projet en autonomie durant mon stage, ainsi que celle d'avoir pu intégrer une entreprise au cadre aussi agréable.

Je remercie tout particulièrement Samy Jacquet, mon responsable et ami, qui m'a accordé sa pleine confiance sur la mission et qui m'a permis de progresser bien au delà de mes espérances.

Je veux également remercier Marc Lemercier pour son suivi et son aide précieuse durant le stage, ainsi que pour s'être assuré du bon déroulement de celui-ci. J'en profite également pour remercier Patrick Lallement et Alain Corpel pour avoir rendu ce stage possible.

Je souhaite aussi remercier l'équipe pédagogique de l'Université de Technologie de Troyes et les intervenants des diplômes Informatique et Systèmes d'Informations (ISI) et Sécurité des Systèmes d'Informations (SSI) pour les enseignements théoriques qu'ils ont pu m'apporter et qui m'ont aidé à la bonne réalisation de ma mission de stage, et plus particulièrement Rémi Cogranne pour son soutien dans l'élaboration d'une théorie de chiffrement adaptée.

Je veux remercier également mes collègues Étienne Pollet et Nhân-Quý Nguyễn pour leur accueil et l'ambiance très agréable à laquelle ils ont contribué durant ces six mois dans l'entreprise, de même que Jaufré Lallement, stagiaire de DUT à Park'n Plug. Enfin, je remercie Diane Pacaud, en stage de TN09 sur le même projet que moi, pour l'aide formidable qu'elle a su m'apporter.

Sommaire

I. Rapport

II. Annexes

Première partie

Rapport

Introduction

C'est avec enthousiasme que j'entame mon stage de fin d'études de 24 semaines dans le cadre de mon double cursus en Informatique et Systèmes d'Information (ISI) et Sécurité des Systèmes d'Information (SSI) au sein de l'Université de Technologie de Troyes. Ma formation ne m'a jamais autant plu, et c'est plein d'assurance que j'ai commencé cette aventure.

Ce stage s'est déroulé au sein de l'entreprise Park'n Plug, une startup de cinq personnes spécialisée dans la mise en place de solutions de recharge de véhicules électriques (*voir* **1 - Présentation de l'entreprise**), à son siège social de Rosières-près-Troyes (Aube).

J'ai été intégré à l'équipe de développement de l'entreprise au poste d'ingénieur logiciel, sur un **projet de conception d'une application de gestion de rechargement de véhicule électrique**. J'ai été affecté à la création de *Gestio* avec l'aide de Diane Pacaud, stagiaire de TN09 qui a partagé cette mission avec moi.

J'ai ainsi pu m'investir de manière autonome dans la réalisation de ce logiciel et dans chacune des phases de la conception : analyse de l'existant, rédaction du cahier des charges, maquettages, architecture et migration de base de données, choix d'implémentation, de technologies et de modes de travail, ou encore production de code. J'ai également pu travailler sur l'implémentation UX [1], aspect que j'apprécie tout particulièrement.

J'ai donc eu l'occasion de m'impliquer dans les phases de Recherche et Développement de la solution, ainsi que dans les phases de conception technique et de réalisation fonctionnelle.

Le second volet de mon poste consistait à **étudier et améliorer la sécurité globale du système d'informations**. À ce titre, j'ai pu réaliser une analyse des risques via un audit de sécurité, qui m'a permis de faire des préconisations d'amélioration. J'ai également été amené à mener d'autres actions annexes se rapportant à la sécurité, comme l'élaboration d'un chiffrement pour le protocole de communication ou la mise en place de contre-mesures suite à une attaque sur le SI.

Le choix de ce stage s'inscrit dans un projet professionnel précis : l'univers "StartUp". Dans un premier temps, je souhaiterais rester dans une structure à taille humaine, dans laquelle chacun est artisan de l'avenir de l'entreprise, et où de très nombreuses missions différentes peuvent être affectée à une même personne ; c'est cette recherche de polyvalence perpétuelle qui m'a attiré.

1. ***U**ser **X**perience*, ou Expérience Utilisateur, « actions et réponses qu'une personne trouve ou anticipe dans l'utilisation d'un logiciel, une application, un site internet voire tout produit »[1]

De par cette polyvalence, la mission proposée parvenait à couvrir la quasi-totalité de mon programme, aussi bien concernant mon parcours ingénieur ISI que celui du master SSI. C'est également ce tour de force qui a participé à mon envie profonde de rejoindre l'équipe de Park'n Plug.

Enfin, l'intérêt lui-même que je porte à la mission a été fondamental dans ma décision.

L'objectif de ce rapport est de présenter le déroulement de mon stage effectué au sein de Park'n Plug.

Dans un premier temps, nous allons nous intéresser à l'entreprise en elle-même : son positionnement sur le marché ainsi que son fonctionnement interne, atypiques mais qui présentent des particularités intéressantes.

Après une brève description du stage, nous verrons plus en détail son contexte, ainsi que les différents enjeux de ces six mois.

Enfin, nous nous intéresserons successivement aux deux missions sur lesquelles j'ai été positionné. Nous aborderons ainsi des éléments techniques et surtout des choix de conception.

À l'issue de ce document, nous effectuerons un bilan du stage : mon vécu personnel et la mise en rapport avec mon projet professionnel, ainsi que les difficultés auxquelles j'ai pu être confronté durant ces six mois.

1. Présentation de l'entreprise

Park'n Plug est une entreprise sur le secteur du véhicule électrique [2], très jeune, et basée sur le modèle startup. Malgré tout, ses relations étroites avec la société Toggenburger — dont elle est issue — lui permettent d'avoir un historique plus long que ces seules dernières années.

FIGURE 1. – Le rechargement d'un véhicule Zoé (Renault)

De plus, les valeurs que véhicule Park'n Plug sont un réel atout, aussi bien commercial qu'éthique, et ont rendu mon stage d'autant plus intéressant.

1.1. Park'n Plug, cinq ans de Recherche & Développement

C'est en 2011 que commence l'aventure de Park'n Plug : Pascal TOGGENBURGER décide cette année-là de lancer une entreprise spécialisée dans l'équipement de recharge pour véhicules électriques, afin de répondre aux besoins croissants dans l'habitat collectif

2. « *Véhicule Électrique* » est souvent abrégé en *VE*.

(copropriétés...) ou entreprises avec une flotte de VE. À l'époque, la société ne compte que deux employés.

FIGURE 2. – Logo de Park'n Plug

Fondée par deux entités — le capital étant détenu à la fois par Pascal devenu Président Directeur Général, et par la société Toggenburger (cf. *infra* **1.4 - Les liens avec l'entreprise Toggenburger**) — la startup peut rapidement compter sur diverses subventions (notamment celle de l'ADEME[3]) en plus de la levée de fonds initiale. Très vite, elle développe son premier produit qu'elle baptisera Nemo (cf. *infra* **1.6.1 - Le serveur Nemo**).

Quatre ans plus tard, en octobre 2015, la société change de locaux pour s'agrandir. Elle compte maintenant quatre ingénieurs dans son équipe, ainsi qu'un commercial situé sur Paris.

1.2. Le domaine d'activité : du service pour la recharge de véhicules électriques

Si Park'n Plug continue d'équiper les parkings en bornes de recharge pour véhicules électriques, la société a à présent diversifié ses activités. Son cœur de métier se situe maintenant principalement dans la prestation de services. Ces derniers sont de trois sortes :

- L'**assistance à maitrise d'ouvrage**, qui consiste en une mission d'audit des équipements électriques, ainsi que des besoins physiques et en énergie. Cela comprend également des études techniques des installations par le bureau d'études concernant le dimensionnement des infrastructures, l'optimisation énergétique, des préconisations...
- L'**équipement des infrastructures** en lui-même, qui est généralement sous-traité.
- La **facturation** et la répartition des coûts de charge. Il est en effet souvent vital pour les grosses structures de pouvoir dissocier les utilisateurs finaux des bornes de

3. Agence De l'Environnement et de la Maitrise de l'Energie

recharge afin de leur réattribuer les coûts des consommations selon leur utilisation du système. C'est précisément sur ce service que s'est portée ma première mission (cf. *infra* **2.1.2 - Le volet conception**).

1.3. Mon poste dans la Startup

Park'n Plug est une startup qui emploie aujourd'hui cinq personnes, chacune d'entre elle étant rattachée à un pôle (*voir* l'organigramme, Figure 3).

D'un point de vue purement géographique, l'entreprise possède deux agences :

- Le siège social, situé à Rosières-près-Troyes dans la Technopôle qui concentre le travail de R&D et de conception. C'est ici que j'ai été intégré.
- Des locaux sur Paris dans le 15$^{\text{ème}}$ arrondissement, qui accueillent les parties commerciale et marketing de l'entreprise.

Mon poste a été rattaché au pôle « Production - Recherche & Développement », dans un service qui n'était plus fonctionnel depuis le départ d'un des collaborateurs qui gérait la conception du logiciel *Gestio* (cf. *infra* **2.1.2 - Le volet conception**).

Mon poste lié à la sécurité était quant à lui inclus au même pôle, mais n'était rattaché à aucun service particulier.

J'ai travaillé sous la responsabilité d'un ingénieur systèmes, Samy JACQUET, ancien étudiant à l'UTT. étant cependant attaché au développement et à l'intégration du serveur Nemo (cf. *infra* **1.6.1 - Le serveur Nemo**), j'ai conservé une grande autonomie.

Au sein de mon équipe, une autre stagiaire — de TN09 cette fois — Diane PACAUD, a été intégrée sur la même mission de conception. Ainsi j'ai eu le plaisir de la superviser durant six mois.

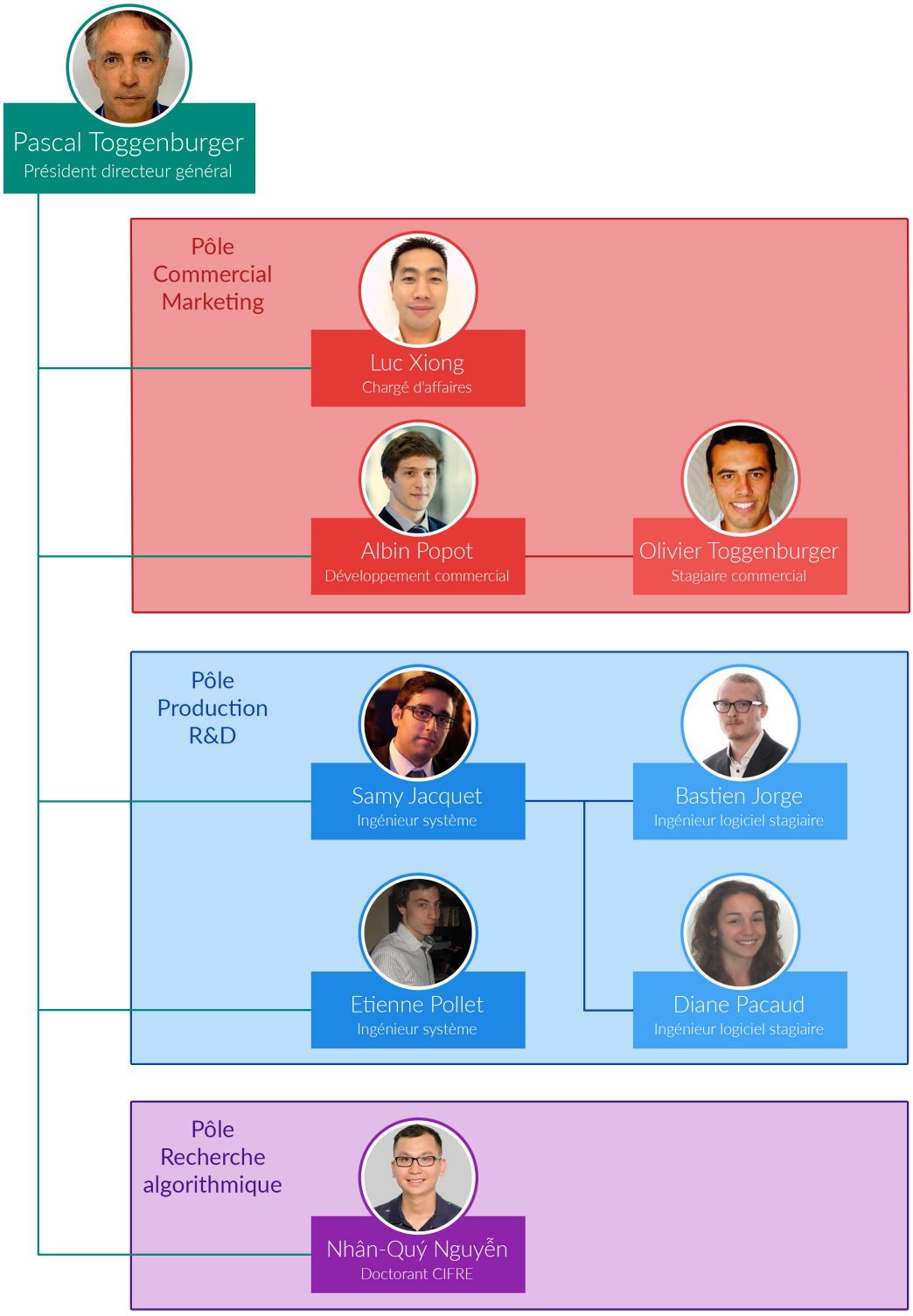


FIGURE 3. – Organigramme de la société Park'n Plug

1.4. La position de Park'n Plug par rapport à l'entreprise Toggenburger

Détenant une part du capital, l'entreprise Toggenburger a une forte influence sur Park'n Plug. Son investissement dans la startup permet principalement de pouvoir nous appuyer sur des partenaires de qualité, aussi bien en tant que prestataires, fournisseurs ou clients.

1.4.1. Origine

Fondée en 1957 par Jean TOGGENBURGER, l'entreprise qui prendra son nom se spécialise dans l'électricité. Ses activités se concentrent sur :

- L'équipement électrique
- La sécurité du bâtiment
- La maitrise de l'énergie

En 1988, Jean TOGGENBURGER cède son activité à son fils Pascal, l'entreprise emploie alors 35 collaborateurs.

FIGURE 4. – Logo de Toggenburger

En 2008, l'entreprise familiale rachète la *SARL Garnier*, autre société d'installation électrique qui se spécialisera dans l'installation de matériels photovoltaïques.

En 2011 on l'a vu, Toggenburger crée Park'n Plug.

Forte de son expertise électrique et de sa position dominante sur le marché aubois depuis plus de 60 ans, la société compte aujourd'hui près de 50 employés et un réseau de partenaires important.

1.4.2. Avantages pour Park'n Plug

Park'n Plug s'est construite et développée autour de partenariats commerciaux forts avec l'ensemble des acteurs de la filière du véhicule électrique (équipementiers, gestionnaires de réseaux, constructeurs de bornes, constructeurs automobiles...). Il s'agit de la principale influence que Toggenburger a pu avoir sur la startup. En effet, sans cet appui, jamais Park'n Plug n'aurait pu bénéficier aussi vite d'un tel maillage de partenaires, ni d'une telle importance (*voir* Figure 5).

Le second avantage concerne la renommée locale que possède Toggenburger. Grâce à elle, il est possible pour Park'n Plug de répondre à des appels d'offre plus sereinement avec un professionnalisme reconnu.

FIGURE 5. – Partenaires de Park'n Plug

Enfin, Toggenburger elle-même demeure un partenaire privilégié dans la mesure où c'est régulièrement par son intermédiaire que sont réalisées les prestations d'installation.

1.5. Le modèle économique : une philosophie ancrée dans la transition énergétique

Aurélien GAY et Marc GLITA tentent de donner une définition de la transition énergétique dans leur ouvrage [6] :

> *[Le]* principal objectif *[de la transition énergétique]* est le passage des énergies dites de « stock » (pétrole, gaz, charbon, nucléaire), dont l'épuisement des ressources est un enjeu majeur, aux énergies renouvelables dites de « flux » (éolien, solaire), en s'interrogeant principalement sur la réduction des émissions de gaz à effet de serre (dont une grande partie est liée aux énergies fossiles), la réduction des risques environnementaux liés à l'exploitation de ces ressources (marées noires, accidents nucléaires), et la réduction des pollutions et des déchets dangereux qu'elle engendre.

Cette notion, touchant à la fois à l'écologie et à une économie différente, Park'n Plug l'a faite sienne. Sa volonté d'offrir un meilleur accès au rechargement pour les véhicules

électriques (*voir* Annexe A page I) est motivée par l'envie de développer ce mode de transport dit « propre ».

La transition énergétique soutenue par l'état français (plus d'informations sur developpement-durable.gouv.fr) va permettre à Park'n Plug de se développer plus facilement et de faire connaitre ses différentes offres.

FIGURE 6. – Logo de la loi relative à la TEPCV

1.6. Les produits Park'n Plug

Aujoud'hui, hormis pour quelques véhicules très haut de gamme, l'autonomie du VE ne permet que des trajets courts, donc principalement en zone urbaine. Et si un particulier peut aisément recharger sa voiture dans son domicile personnel, en agglomération on trouve beaucoup de logements collectifs. De même, le secteur tertiaire (avec une flotte de véhicules destinés aux commerciaux ou aux techniciens) comme les commerçants (pour une recharge à destination de leur clientèle) sont confrontés aux problèmes de recharges simultanées.

Cette problématique est due à plusieurs éléments. La plupart de ces structures ne sont pas correctement dimensionnées pour supporter la recharge de plusieurs VE, que ce soit au niveau des câblages nécessaires ou de l'arrivée électrique elle-même. On constate également la difficulté de router les frais de recharge vers les utilisateurs lorsqu'elle est effectuée dans une zone commune — comme le parking par exemple.

C'est pourquoi au delà des services d'accompagnement pour le meilleur choix dans la solution la plus adaptée aux besoins du client, Park'n Plug propose deux produits (*voir* Annexe B page V) luttant contre les problèmes soulevés.

1.6.1. Le serveur Nemo

Nemo permet de distribuer l'énergie aux différents véhicules connectés à une même installation dans le respect de la puissance maximale supportée. Plus précisément, elle donne l'ordre aux bornes électriques connectées à un VE de lancer ou non la charge en fonction de la puissance disponible.

Ce système a largement évolué à travers le temps : à ses prémices simple automate programmable (*voir* Figure 7), il a sû s'adapter aux besoins de l'entreprise et est main-

tenant assemblé en laboratoire dans les locaux de Park'n Plug. Nemo est aujourd'hui basé sur un micro-ordinateur de type Raspberry Pi.

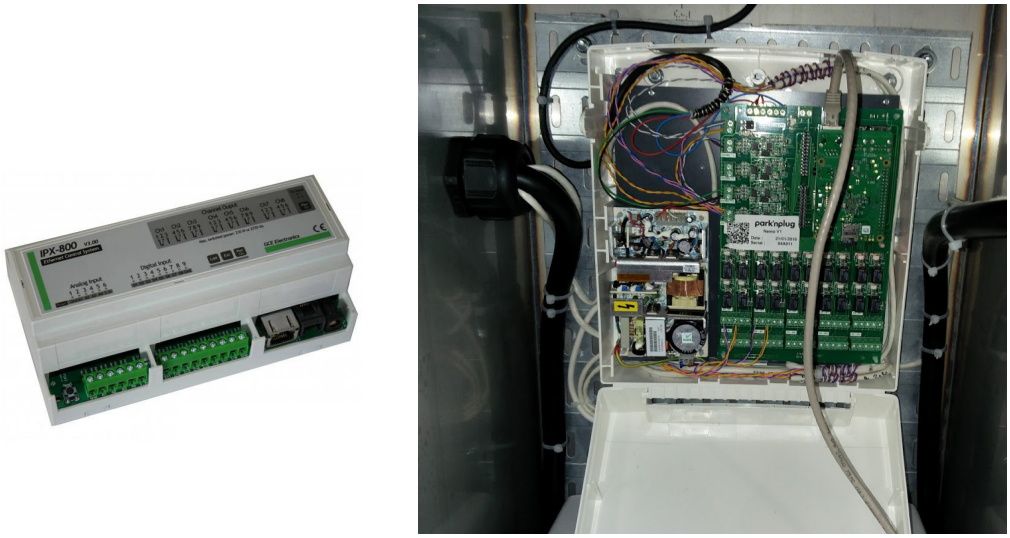


FIGURE 7. – Automate programmable IPX800-V3 vs Nemo dernière génération

Passer sur un système propre a apporté souplesse et intelligence. Nemo a gagné en prédictivité — la recharge des VE se lance de manière automatique en fonction des habitudes des utilisateurs — et communique plus facilement avec notre serveur central.

Elle est directement installée dans l'une des bornes de recharge de véhicule électrique équipée sur le parking (*voir* Figure 8), elle est donc invisible pour les utilisateurs.

Ce système résout donc l'un des deux problèmes soulevés en préambule, à savoir les recharges multiples simultanées.

1.6.2. Gestio : le système de gestion et de refacturation

Le second produit phare de Park'n Plug est une interface de gestion et de facturation.

En effet, lorsqu'il recharge dans sa copropriété, un particulier consomme l'énergie mise à disposition dans les parties communes. Il est donc important pour le gestionnaire de pouvoir répartir les coûts de charge à chacun des utilisateurs. C'est là qu'intervient Gestio, le produit sur lequel j'ai pu travailler.

Chaque utilisateur est identifié de manière unique (code, badge…) sur la Nemo, qui transmet alors les informations de consommation au système Gestio. Ce dernier se chargera par la suite d'établir une facture mensuelle aux différents utilisateurs finaux, ainsi qu'au gestionnaire afin de le recréditer en conséquence.

FIGURE 8. – La Nemo installée dans une borne de recharge

2. État des lieux du stage

Lors de mon arrivée à Park'n Plug, on m'a exposé les deux objectifs recherchés à travers mon poste : une conception logicielle afin de refondre l'outil *Gestionnary* (cf. *supra* **2.2 - Présentation du logiciel *Gestionnary***), et une recherche d'amélioration de la sécurité globale au sein de l'entreprise. Ces deux aspects m'ont amené à côtoyer différents domaines de l'ingénierie à travers les méthodes de travail de Park'n Plug.

C'est pour cela qu'il est intéressant de s'attarder sur le choix du sujet, ainsi que les outils sur lesquels ont porté mon stage.

2.1. Ma mission au sein de Park'n Plug

Le sujet du stage qui m'a été confié a été défini dans un premier temps de la façon suivante : « *Conception et sécurisation d'un serveur Node.js* ».

Pour entrer un peu plus avant dans les détails de la mission, il convient de diviser le sujet en deux parties : la sécurisation et la conception.

2.1.1. L'amélioration de la sécurité

Park'n Plug étant une startup relativement jeune, beaucoup d'éléments de sécurité étudiés durant mon master n'ont pas pu être mis en place en amont. Cela concerne aussi bien le Système d'Informations lui-même — notamment la sécurité côté serveur — que celle ayant trait à la conception — sécurité de mon application comme de celles déjà en production.

Par ailleurs, un audit ayant été effectué l'an passé, il a été jugé intéressant d'en réaliser un nouveau dans le cadre du stage. Ses buts : faire un état des lieux actualisé de la sécurité afin de constater le travail accompli depuis, et définir les nouvelles priorités.

2.1.2. Le volet conception

La conception, au sens large, a pris une part très importante de mon stage. En effet, l'entreprise avait besoin de refondre intégralement le logiciel d'administration utilisateurs *Gestionnary*. Il m'a fallu passer par les différentes étapes étudiées durant mes trois années à l'UTT, la phase de développement n'étant pas une tâche jugée prioritaire au début. Il s'agissait avant tout de partir sur des bases saines.

Ainsi, j'ai pu travailler sur de l'analyse, de la rédaction de cahier des charges, du maquettage ou encore sur la mise en place d'un *workflow* fonctionnel et efficace.

2.2. Présentation du logiciel *Gestionnary*

L'outil *Gestionnary* est un logiciel de gestion des consommations effectuées lors des recharges de véhicules électriques (*voir* Figure 9). C'est d'ailleurs bien plus qu'un simple outil de facturation comme présenté au client (cf. *supra* **1.6.2 - Gestio : le système de gestion et de refacturation**) : *Gestionnary* se veut un ERP multi-utilisateurs pour l'administration des installations et une interface de management des dépenses énergétiques.

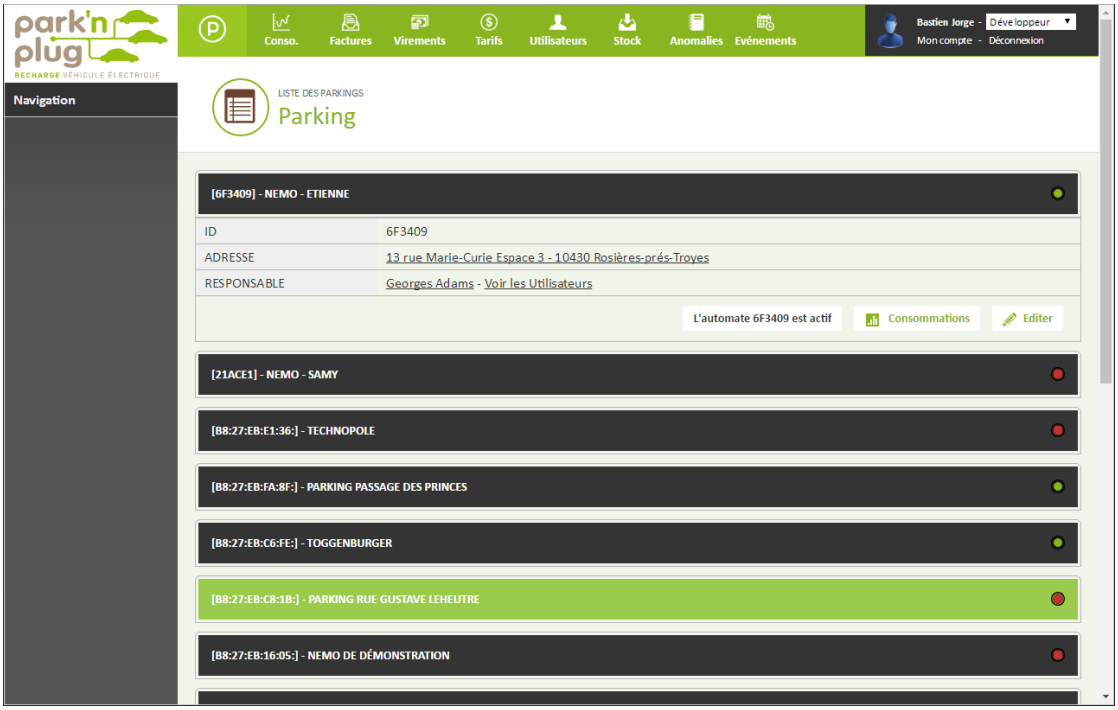


FIGURE 9. – Page d'accueil développeur du logiciel *Gestionnary*

À mon arrivée cependant, *Gestionnary* avait déjà été passé en utilisation restreinte, et pour plusieurs raisons il avait été fermé aux utilisateurs extérieurs : le projet n'était plus maintenu, l'évolution récente des serveurs Nemo l'avait rendu obsolète (*voir* Figure 10) et on y trouvait plusieurs failles de sécurité [4].

Forts de ces constatations, les ingénieurs de Park'n Plug ont alors pris le parti de fermer *Gestionnary* au public. Le projet est néanmoins resté en production du fait de la nécessité d'accéder à l'interface d'administration développeur, par laquelle les Nemo sont paramétrées à distance.

4. Plusieurs failles avaient déjà été identifiées : des injections SQL subsistaient, certaines ressources (comme les factures) étaient accessibles par tous... Il ne sera pas question de cette analyse dans le rapport, car antérieure à ma venue dans l'entreprise.

FIGURE 10. – Menu dépassé[5] du logiciel *Gestionnary*

Se limiter à la seule fonctionnalité de configuration des serveurs Nemo est très éloigné des objectifs de départ du logiciel. En effet, son utilisation se voulait plus ambitieuse :

- L'utilisateur final doit pouvoir gérer sa facturation et ses paiements.
- C'est également via cette interface qu'il doit consulter ses consommations et gérer ses badges.
- Le gestionnaire, responsable d'un parc automobile, devrait également pouvoir accéder à *Gestionnary* : pour ajouter ses utilisateurs, gérer la tarification...
- Ce devrait aussi être un point d'entrée pour le comptable qui aura alors une vue d'ensemble sur les paiements via le SI.
- Et enfin, c'est via *Gestionnary* que le réseau national d'installateurs agréés se verrait affecter des ordres de mission et pourrait effectuer les retours terrain.

2.3. Planification de la mission et méthodologie de travail

On l'a vu, de nombreuses fonctions étaient manquantes dans le *Gestionnary* d'origine. On comprend donc la nécessité de reprendre le projet laissé en l'état.

Finalement, la décision a été prise de le refondre dans son intégralité. En effet, pour ne pas repartir sur les mêmes travers que ceux par lesquels était passé *Gestionnary* — de lourdes modifications successives pensées séparément au lieu d'être un ensemble — nous avons choisi de repartir de zéro tout en préparant l'application aux évolutions futures.

C'est ici que je suis intervenu, avec l'aide de Diane.

FIGURE 11. – Refonte de *Gestionnary* vers *Gestio*

On m'a également demandé sans instructions plus précises d'améliorer la sécurité de tout le système : il m'a donc fallu effectuer un travail d'investigation afin d'identifier les différentes failles présentes dans le SI. Sur cette partie, j'étais seul et en complète autonomie.

Ma mission consiste à travailler sur la sécurité du système d'informations tout en concevant *from scratch* un logiciel de gestion intégré. Il m'a donc été nécessaire de seg-

5. Dépassé, car les items Factures, Virements, Stock, Anomalies et Evénements ne sont plus fonctionnels, et Tarifs et Conso. ne le sont que partiellement.

menter mes différentes tâches ; cela m'a permis de m'adapter au fonctionnement interne de l'entreprise et de répartir de façon efficace le travail à accomplir.

2.3.1. Fonctionnement interne

Park'n Plug étant une startup, son fonctionnement ne diffère que peu d'une entreprise plus traditionnelle, néanmoins on peut noter des particularités.

Des services relativement autonomes

On l'a vu, Pascal TOGGENBURGER est à la tête de deux entités, il doit donc partager son activité entre elles. De plus, son statut de dirigeant de Park'n Plug l'amène à devoir souvent être l'interlocuteur privilégié de partenaires ou de clients, notamment dans ses locaux parisiens.

Ainsi, sa présence au siège social étant limitée, l'équipe dispose de manière générale d'une grande autonomie.

Par ailleurs, hormis vis à vis du PDG, il n'existe pas de relation hiérarchique au sein des pôles informatiques. L'ambiance de travail est donc grandement améliorée car chacun connait ses objectifs et s'efforce de les atteindre au mieux.

Cela implique également beaucoup d'échanges afin d'éprouver les différentes théories, ou tout simplement pour trouver de l'aide auprès d'un collègue plus expérimenté — le fait d'être intégré à une équipe jeune et composée d'ingénieurs permet notamment d'avoir à disposition un large panel de connaissances, toutes technologies confondues.

L'importance de ces échanges au niveau local est renforcée par ceux effectués avec le reste de l'équipe sur Paris.

Des efforts de communication

La vie de l'entreprise Park'n Plug est rythmée de réunions régulières, dues au fait que les équipes commerciale et R&D sont physiquement séparées.

Ainsi, une *conf-call* est organisée tous les vendredis afin d'échanger sur les progrès effectués depuis le dernier appel. C'est aussi un moment privilégié pour poser les questions techniques ou commerciales qui auraient pu survenir dans l'esprit de l'une ou l'autre des équipes.

Par ailleurs, une réunion mensuelle plus formelle est organisée afin de tenir le PDG et le pôle marketing au courant des avancées effectuées et de définir les grandes lignes de conduite pour le mois à venir. C'est également l'occasion d'avoir une vision plus globale de l'entreprise, des projets sur lesquels elle se lance, des subventions dont elle a récemment disposé, des pays vers lesquels elle se dirige...

C'est notamment au travers de ces différentes réunions que nous faisions état de l'avancée du stage.

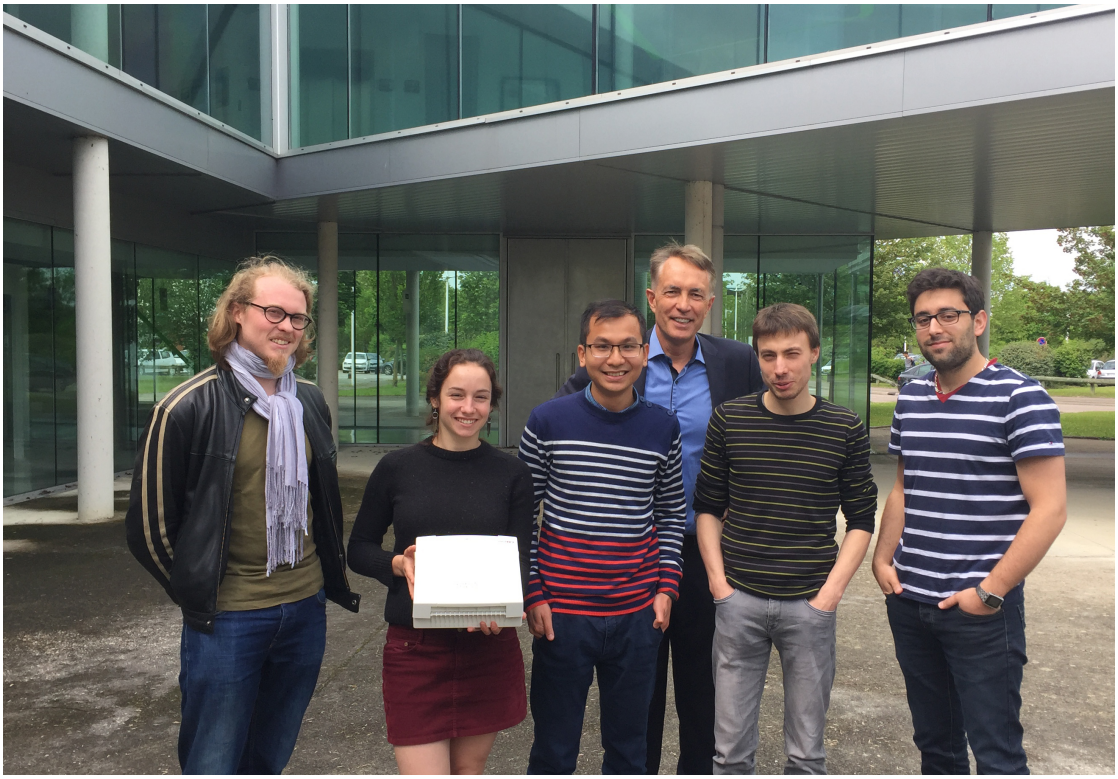


FIGURE 12. – L'équipe R&D de Park'n Plug

2.3.2. Organisation temporelle

Contrairement à mon stage de mi-cursus, je n'ai pas eu le loisir de séparer pleinement mes deux rôles puisqu'ils étaient étroitement liés. En effet, la documentation, la mise en place de process, la conception... sont autant de tâches touchant aussi bien la sécurité que la conception logicielle.

Ainsi, j'ai été amené à effectuer différentes missions (*voir* Figure 13) durant ces six mois de stage.

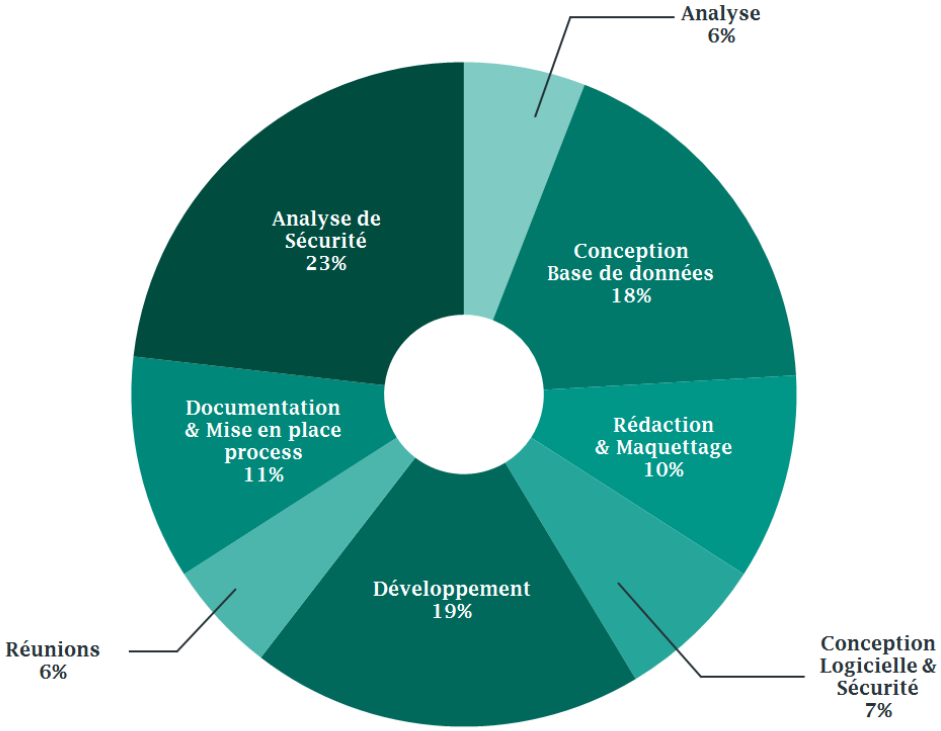


FIGURE 13. – Répartition du travail sur la durée du stage

3. Conception logicielle

La première mission de mon stage concerne le volet ingénieur. Il s'agit de conception logicielle, que l'on peut diviser en deux parties : l'opérationnel et le technique.

Ce sera également l'occasion d'aborder les différentes missions annexes que j'ai pu réaliser.

3.1. Opérationnel

L'un des aspect de ce TN30 n'a pas trait directement à la conception. Cela se concrétise par exemple par des choix réfléchis d'outils ou de techniques d'amélioration du flux de travail.

3.1.1. Choix de l'environnement de développement

À mon arrivée, j'ai eu la possibilité de choisir mon OS. Entre Windows 10 et une machine UNIX, j'ai préféré la première pour deux raisons : j'avais besoin de logiciels ne tournant que sous Windows — la virtualisation aurait été possible, mais lourde au vu de la configuration nécessaire pour les faire tourner — et Diane, stagiaire de TN09, était plus familière des machines Microsoft.

Cependant, il nous a fallu compléter cet environnement, car pour des soucis de compatibilité avec Node il est nécessaire de travailler également sous Linux (la limite de longueur des chemins sous Windows étant de 255 caractères [6], les routes complètes menant vers les dépendances étaient trop longues).

Notre choix s'est porté sur une console Cygwin (*voir* Figure 14), qui permet d'émuler des commandes Linux [7] (telles que git, zsh ou ssh), couplé à une machine virtuelle Debian. C'est sous cet OS que nous développions en Node.js.

Node.js a également été le langage sélectionné pour développer Gestio. Nous l'avons préféré malgré une autoformation nécessaire — les connaissances disponibles au sein de l'entreprise étaient très limitées sur ce langage — car nous devions effectuer en temps réel des mises à jour depuis le serveur. Le package `socket.io` a été installé pour remplir cette fonction, ce qui a conditionné le choix de Node.

6. Ce n'est aujourd'hui plus le cas depuis la *build 14352*, mais la limite existait encore lors des premiers jours de mon stage.

7. De même, il sera bientôt possible de se passer de cet outil, Microsoft ayant annoncé l'intégration native de bash sous Windows 10.

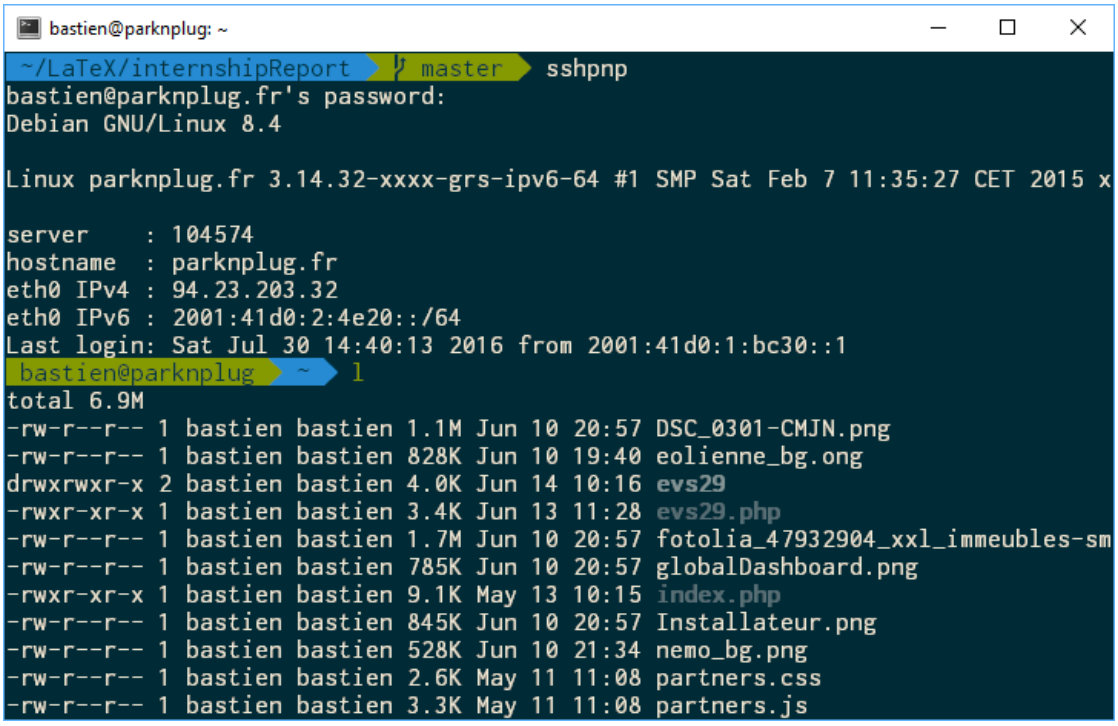


FIGURE 14. – Interface de Cygwin sous Windows

3.1.2. Encadrement

Ces différentes décisions n'ont pas été prises unilatéralement. Les choix ont été faits en accord avec l'encadrant au sein de l'entreprise, mais aussi avec les différentes personnes que j'ai pu avoir sous ma responsabilité.

En effet, j'ai eu la chance d'être chef de projet sur la conception logicielle de Gestio, et au delà de Diane, Jaufré LALLEMENT, venu de l'IUT pour une période de dix semaines, a rejoint mon équipe. J'ai essayé de faire en sorte de pouvoir les former au mieux selon leurs spécialités respectives tout en restant attentif à leurs suggestions.

Le management est un aspect de ma formation que je n'avais pas encore eu l'occasion d'expérimenter, et je me suis efforcé d'effectuer un suivi régulier tout en étant à leur écoute.

3.1.3. Mise en place d'outils de développement agile

J'ai également été sollicité sur la mise en place d'un *workflow* efficace. En effet, avant mon arrivée, le flux de travail n'était pas particulièrement défini, j'ai donc pu m'appuyer sur mon expérience personnelle pour définir les fonctionnalités nécessaires ainsi que les logiciels les plus adaptés.

Workflow global

Les trois contraintes qui ont été fixées sont les suivantes :

- Une solution **gratuite**, ce qui nous a rapidement dirigé vers l'*open-source*.
- Un système **auto-hébergé**, afin de s'assurer de la confidentialité des informations stockées.
- Une suite **compatible entre logiciels** qui permette la mise en place une CI [8] complète.

Ainsi, j'ai pu trouver et faire installer toute une série de logiciels libres (*voir* Figure 15), pendants de services plus connus : *Mattermost* (qui regroupe les mêmes fonctionnalités que *Slack*), *GoGs* (solution semblable à *GitHub* développée en Go), *Wekan* (identique à *Trello*), *Drone* (équivalent à *TravisCI*) et Dokku (*Heroku*-like) [9].

FIGURE 15. – Logiciels utilisés : Mattermost, GoGs, Wekan, Drone, Dokku

Le *workflow* est relativement simple :

- Le point d'entrée du processus de développement se trouve souvent sur Mattermost, notre espace de discussion interne. Divisé en *chans*, l'outil est favorable à l'émergence d'idées nouvelles. C'est également ici que seront posées les bases de la priorisation des tâches, sous forme de *draft*.
- Sur la même base que GitHub, GoGs permet la création d'*issues*. Ce système offre la possibilité d'ouvrir un ticket décrivant l'intervention à effectuer, d'en discuter pour apporter des informations ou idées supplémentaires, et enfin, d'affecter un responsable à la tâche.
- Ensuite, la tâche peut éventuellement être répartie entre plusieurs développeurs après avoir été divisée en sous-tâches par le responsable. C'est sur le support Wekan, outil de kanban, que seront déposées les différentes *cartes* de tâches.
- À la suite de cet étiquetage, intervient le développement. Il s'appuiera sur l'outil Git pour versionner (cf. *infra*, **3.1.3 - Workflow propre à git**).
- À l'issue du développement, une *pull request* [10] sera déposée sur GoGs. Si elle est

8. L'intégration continue, ou CI (*Continuous Integration* en anglais) est « un ensemble de pratiques utilisées en génie logiciel consistant à vérifier à chaque modification de code source que le résultat des modifications ne produit pas de régression dans l'application développée » (Source : putaindecode.io).

9. À l'heure actuelle, Drone et Dokku ne sont pas encore en service, car aucun développement n'a été suffisamment avancé pour en avoir besoin.

10. La *Pull Request* permet de tenir les autres développeurs au courant des changements effectués sur un dépôt. Une fois la requête envoyée, les utilisateurs concernés peuvent évaluer les changements, discuter de potentielles modifications, ou même proposer des mises à jour. (Source : github.com)

validée, elle sera fusionnée au projet et une notification automatique sera envoyée sur Mattermost via un bot (*voir* Figure 16).
- Drone va automatiquement faire tourner les tests unitaires mis en place dans le repo.
- S'ils sont valides, Dokku va déployer la branche en production.

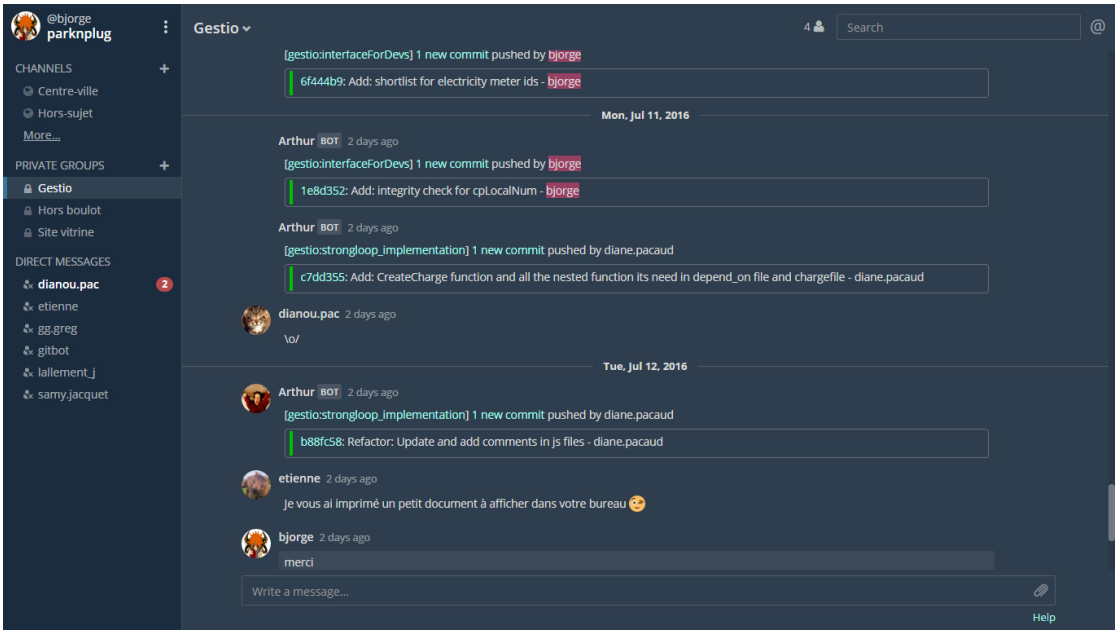


FIGURE 16. – Intervention du bot GoGs dans Mattermost

Un flux de travail particulier a également été mis en place concernant le développement pour des projets d'ampleur importante, tel que le mien.

Workflow propre à git

Toujours dans un souci d'amélioration du développement, j'ai optimisé le *workflow* du logiciel de *versionning*. Git fonctionne sur le principe des *commits*[11] et des *branches*[12], j'ai donc choisi d'avoir un flux basé sur trois branches (*voir* Figure 17) :
- ***dev***, la branche la plus sollicitée : elle contiendra tous les commits de développement. La plupart du temps, elle donnera naissance à diverses sous-branches (locales pour des modifications mineures, répliquées sur le serveur pour d'autres plus lourdes ou plus longues à mettre en place).

11. Les commits valident une série de modifications afin de créer un point de restauration dans le temps.
12. « Créer une branche signifie diverger de la ligne principale de développement et continuer à travailler sans se préoccuper de cette ligne principale. » (Source : git-scm.com).

- ***release***, qui sera un environnement de test interne : l'objectif est de la fusionner depuis la branche inférieure lorsque plusieurs améliorations ont été développées, afin de faire le recettage.
- ***master***, qui sera la branche principale — celle de production : il sera très rare d'effectuer un *commit* directement dessus, sauf dans le cadre d'un *hotfix*[13].

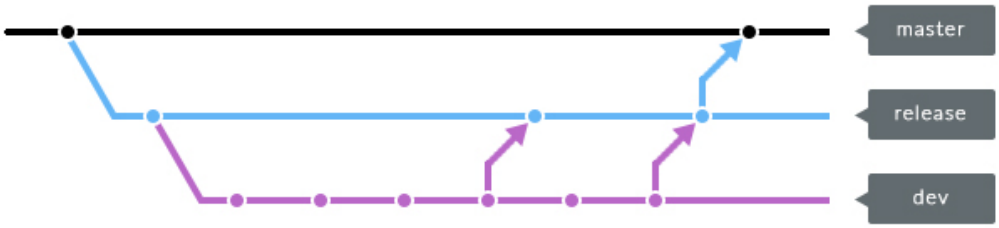


FIGURE 17. – Gestion des branches dans git

3.2. Conception

L'essentiel de mon travail touchant à la branche ingénieur a été principalement de la conception ; celle-ci est divisée en plusieurs étapes que nous étudierons successivement.

3.2.1. Analyses du besoin et de l'existant

La première étape de la conception consiste en une analyse du métier, de l'existant, ainsi que des besoins utilisateurs.

En cours d'IF05 (« *Qualité du logiciel* ») dispensé par Aurélien BENEL, nous avons eu la chance d'avoir pour client un ethnométhodologue[14], Christophe LEJEUNE. Ce dernier nous a expliqué que la doctrine édictée par sa discipline considérait une étude terminée lorsque l'analyste était capable d'effectuer pleinement et en autonomie les tâches du métier analysé.

C'est en gardant cette philosophie à l'esprit, que j'ai commencé à m'imprégner du métier, essayant d'avoir une compréhension idéale — dans le temps dont je disposais — des besoins utilisateurs et du fonctionnement de l'entreprise.

L'étude des besoins et de l'existant ayant déjà été détaillée plus haut (cf. *supra*, **2.2 - Présentation du logiciel *Gestionnary***), nous ne l'aborderons pas plus avant ici.

Le résultat de cette analyse a été formalisé en un cahier des charges afin de s'assurer que tous les acteurs avaient une bonne compréhension du projet.

13. Un *hotfix*, ou patch correctif, « est une section de code que l'on ajoute à un logiciel pour y apporter des modifications : correction d'un bug, traduction, crack. » (Source : Wikipedia).
14. L'ethnométhodologie insiste sur la profondeur du terrain et y consacre bien plus de temps que pour l'ethnologie traditionnelle. L'exigence de la compréhension est poussée beaucoup plus loin.

3.2.2. Rédaction du cahier des charges, documentations

Document interne destiné à poser les règles de construction depuis les fondations jusqu'aux détails les plus fins, le cahier des charges doit faire preuve d'une précision absolue. Chez Park'n Plug, il a également eu pour objectif d'assurer un suivi des propositions, car plusieurs versions successives ont été rédigées.

Composé de l'analyse, des exigences techniques et fonctionnelles, ou encore de documents graphiques (cf. *infra*, **3.2.3 - Maquettage**), le cahier des charges se veut un rappel de toutes les phases de la conception autant qu'un aperçu du rendu final.

Nous l'avons rédigé conformément aux apprentissages de l'UV IF10 (« *Conception centrée usage des systèmes intéractifs* ») enseignée par Pascal SALEMBIER. Les différentes étapes de la conception agile ont été respectées, avec un positionnement du point de vue "utilisateur final" et de nombreuses itérations jusqu'à obtenir un résultat satisfaisant (*voir* Annexe C page VII).

Une documentation a également été commencée, mais le développement du produit n'étant pas terminé, cette tâche n'est pas considérée comme prioritaire.

3.2.3. Maquettage

Parmi les éléments intégrés au cahier des charges, on trouve divers fichiers graphiques. Il s'agit de tout le travail de maquettage.

Il a été réalisé en trois temps :

- Un brouillon sur tableau blanc : il est en effet bien plus facile de travailler à plusieurs dans un premier temps sur un support qui propose autant de souplesse. Ce travail a été réalisé en collaboration avec les autres ingénieurs.
- Un maquettage *WYSIWYG*[15] en *drag-n-drop* via le logiciel Balsamiq (*voir* Figure 18) : l'avantage de ce système est d'avoir des rendus réalistes très rapidement, et de mettre en place des liens vers différentes pages afin de créer un semblant de navigation — et donc préparer la mise en place de scenarii utilisateurs (*voir* Annexe D page XV).
- Un maquettage HTML/CSS (*voir* Figure 19) : plus long, mais reflet de la réalité, il permet, entre autre, d'avoir des effets d'animation et sera réutilisé comme *layout* lors du développement final.

Le maquettage est une étape très importante du processus de conception, car c'est un des rares éléments graphiques compréhensibles par un non-technicien. Il va donc conditionner la suite du projet.

3.2.4. Conception de base de données

L'aspect sans doutes le plus délicat de la conception a été la refonte complète de la base de données. En effet, à mon arrivée, la base *MySql* comportait une quarantaine de tables, dont seulement 15 utilisées. Elle n'était pas normalisée et très peu de contraintes d'intégrité étaient mises en place, de même que pour les clés étrangères.

15. Le WYSIWYG (*What You See Is What You Get*) est « une interface utilisateur qui permet de composer visuellement le résultat voulu. » (Source : Wikipedia).

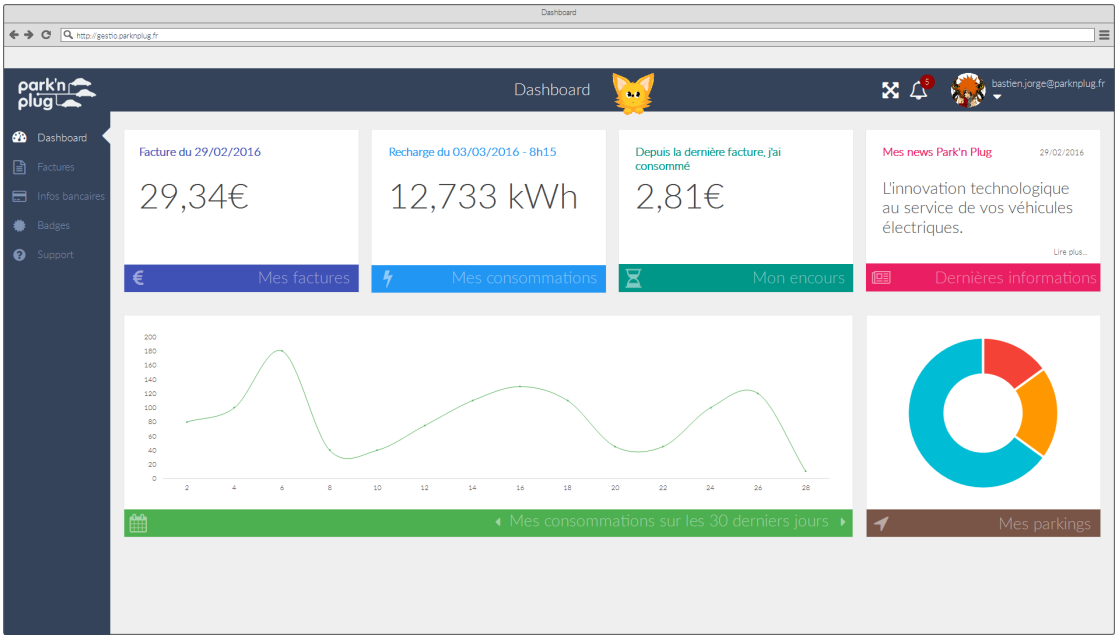


FIGURE 18. – Maquette Balsamiq de Gestio

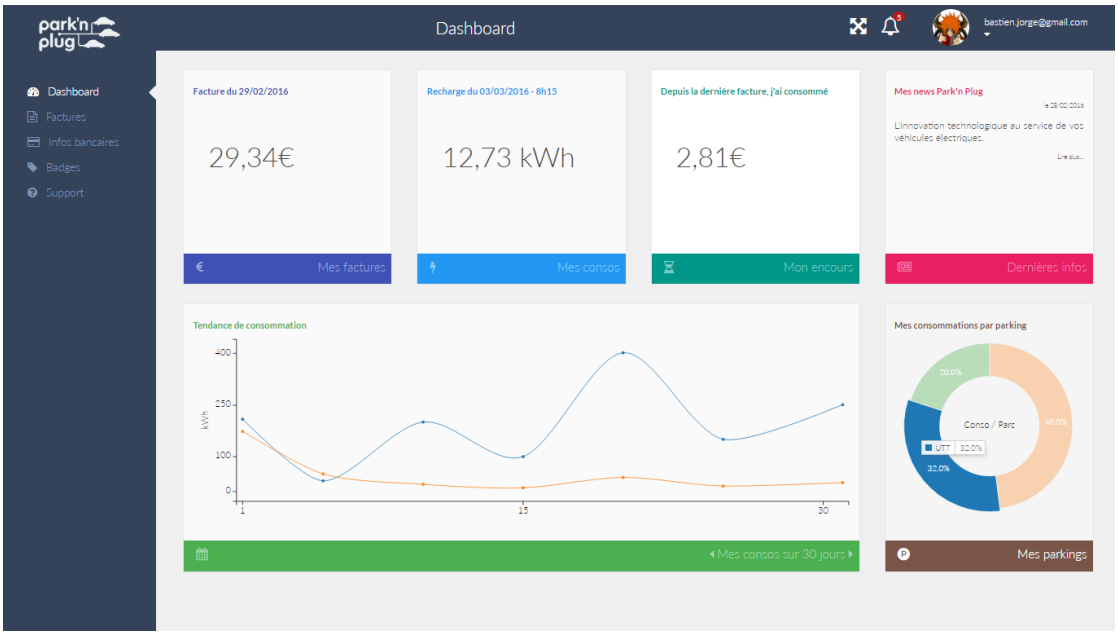


FIGURE 19. – Maquette HTML/CSS de Gestio

Il a donc été décidé de repartir sur des bases saines et d'en créer une nouvelle. Cela aura également l'avantage de permettre une meilleure maintenabilité dans le futur ; par ailleurs, la nouvelle base pourra être en mesure d'accueillir les futures améliorations du produit, car elle a été pensée ainsi dès la conception.

Elle contient maintenant 37 tables (*voir* Annexe C page VII) qui respectent la norme BCNF [16] ; leur nombre peut paraitre surdimensionné, mais cela permettra un gain de performance intéressant si la base se peuple rapidement une fois en service.

Le réel challenge a été la conservation des *datas* préexistantes. En effet, même si la base en production ne contenait que peu de données réelles, il n'était pas pensable d'en effacer le contenu. Ainsi nous avons dû mettre en place un script de migration des données afin de les conformer à la nouvelle forme de la base.

3.2.5. Développement Node.js

Dans leur livre *Node.js – Exploitez la puissance de javascript côté serveur*, Julien FONTANET et Olivier LAMBERT expriment l'intérêt de ce langage : selon eux, il permet de « fournir un moyen simple de produire des applications performantes et extensibles » [4]. C'était là la volonté de Park'n Plug ; par ailleurs, il était également spécifié dans les besoins que le serveur devrait envoyer régulièrement des mises à jour au client (cf. *supra*, **3.1.1 - Choix de l'environnement de développement**), ce que permet Node.

Le développement s'est fait en deux temps : la mise en place d'une API puis son implémentation dans une interface utilisateur. C'est la première étape qui sera la plus intéressante ici, car ce sont ses choix d'implémentation qui sont le plus pertinents dans le cadre de mes études.

La nécessité de se doter d'une API

Historiquement, la base de données était modifiée directement. À cet égard, il est utile de noter qu'une Nemo faisant remonter des informations communique avec un serveur, baptisé *Switch* ; ce dernier récupère la version de la Nemo afin de parser de façon adaptée le paquet d'informations envoyées. Il modifiera alors la base en conséquence en s'y connectant directement.

Avec la modification de la base de données, il a fallu recoder complètement le *Switch*. Il en ira de même dans le futur si nous sommes amenés à faire de nouveau évoluer sa structure.

Afin de pallier ce problème, il suffit de mettre en place un intermédiaire supplémentaire : l'API. Ce sera son fonctionnement interne qui sera recodé en cas de modification de la base.

D'autres avantages sont disponibles par la mise en place d'une API :

- Une sécurisation forte de par l'utilisation d'un module éprouvé (cf. *infra*, **3.2.5 - Le choix d'implémentation**)

16. La forme normale Boyce-Codd (BCNF) est une forme utilisée en normalisation de base, légèrement plus robuste que la troisième forme. Il s'agit d'un schéma relationnel permettant d'éviter la redondance de données. (Source : Wikipedia).

- La possibilité de n'ouvrir des fonctions qu'à un certain type d'utilisateur (administrateur, utilisateur final, switch...)
- Une forte maintenabilité
- Une souplesse accrue pour la mise en place de nouvelles connexions (le jour où une application smartphone sera développée par exemple)

Le système final peut sembler plus complexe (*voir* Figures 20 et 21) mais sera finalement plus efficace.

Le choix d'implémentation

Nous avons choisi *Loopback* comme framework d'API ; cela a été conditionné par plusieurs éléments :

- *Loopback* est un logiciel qui a été racheté par IBM. Si l'entreprise a souhaité se diriger vers ce produit c'est qu'il possède de nombreux atouts. Sa notoriété lui permet également d'avoir une communauté active de taille très intéressante, ce qui permet d'avoir accès à de l'aide rapidement (via *StackOverflow* par exemple).
- L'entreprise qui distribue *Loopback* a été fondée en 2012 : le produit est donc stable, et a plus de chances d'avoir une bonne persistance dans le temps qu'un framework jeune.
- *Loopback* embarque d'ores et déjà le framework *Express.js* que nous utilisons pour gérer l'infrastructure web de Gestio.
- Ayant déjà personnellement un peu travaillé sur *Loopback*, il m'a été plus facile de le prendre en main. Après différents essais d'autres frameworks, c'est donc lui qui a été préféré.

3.3. Tâches annexes

J'ai également été affecté à divers travaux annexes durant mon stage : il est intéressant de les mentionner ici.

3.3.1. Réalisation du site vitrine

Park'n Plug a eu la chance de faire une apparition dans le magasine Turbo diffusé sur M6. Il a donc fallu développer très rapidement un site vitrine actualisé, l'ancien étant alors dépassé. Cette mission m'a été attribuée, ainsi qu'à Jaufré, stagiaire de DUT.

3.3.2. Travaux graphique : création de la mascotte

Une autre tâche annexe a été effectuée avec Diane, à savoir la création d'une mascotte pour Gestio (*voir* Figure 22). Nous ne savons pas encore si elle sera utilisée, mais elle existe.

FIGURE 20. – Échanges sans API

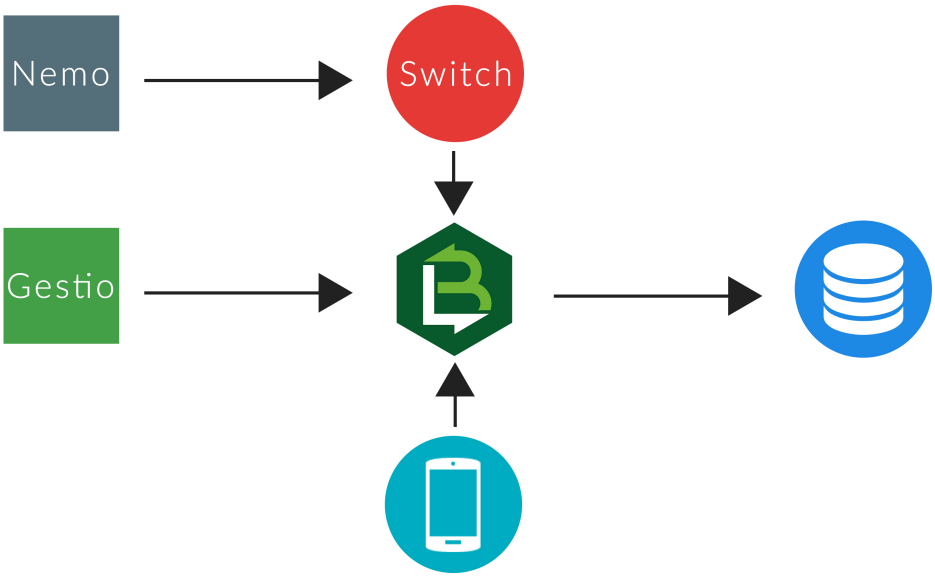


FIGURE 21. – Échanges avec une API

FIGURE 22. – Mascotte de Gestio

3.3.3. Etude du protocole eMIP

Afin d'agrandir son pool de consommateurs et fournir un service toujours plus complet, Park'n Plug souhaite être interopérable. En effet, dans le monde de la recharge de véhicules électriques, il existe deux types d'opérateurs : ceux de mobilité et ceux de recharge. Les premiers gèrent l'activité utilisateurs finaux, les seconds sont en charge de la gestion de l'infrastructure.

Park'n Plug endosse les deux profils, mais d'autres acteurs sont répartis partout en France et il peut être intéressant de proposer à nos clients la possibilité de se recharger chez d'autres opérateurs (on pourrait rapprocher ce processus de celui de l'itinérance en téléphonie mobile).

Il existe plusieurs protocoles permettant cette interopérabilité, dont le protocole eMIP, défini par l'entreprise Gireve.

Dans cette optique, j'ai donc participé à plusieurs réunions et formations concernant eMIP, afin de décider si, dans l'avenir, nous choisissons de nous diriger vers cette technologie, vers une autre, ou si nous en abandonnons l'idée.

4. Sécurité du Système d'Information

Le volet sécurité est la mission associée à mon master SSI. J'ai eu l'occasion de le mettre en pratique de nombreuses fois lors de mon stage.

4.1. Correction de défauts de sécurité dans la conception

Il s'agit ici de pistes d'améliorations que j'ai pu apporter concernant différents défauts de conception que j'ai observés au cours de mon analyse métier. Ce sont les erreurs les plus visibles, celles que j'ai pu constater avant même d'effectuer un travail d'analyse approfondi.

4.1.1. Collision de badges

Le premier défaut que j'ai pu relever concerne la collision d'identifiants des badges d'identification.

En effet, notre lecteur de puce utilise la technologie du RFID Mifare 1k. L'identification de l'id unique dans notre base de données est basé sur le CSN (*Card Serial Number*), encodé sur 4 octets. On se rend compte qu'il est donc possible que ce numéro de série ne soit pas unique[17]. Cette hypothèse n'avait jamais été envisagée, il a donc fallu prévoir un process en cas de collision d'id de badges. La seule solution sécurisée est la révocation pure et simple des deux badges avec bannissement immédiat de l'id.

4.1.2. Identification sans authentification

J'ai aussi pu relever un problème commun aux systèmes d'identification : aucune authentification n'était effectuée. En effet, jusqu'alors, un utilisateur pouvait relier son véhicule électrique à une borne et consommer de l'énergie uniquement avec une identification par badge ou par code. Aucun mot de passe n'était demandé, ce qui rendait l'usurpation d'identité des plus simples.

Il est maintenant nécessaire d'entrer un code associé à l'identification pour pouvoir lancer le rechargement.

17. Même si cette probabilité reste très basse — 4 octets permettent plus de quatre milliards d'id différents — on se rend compte qu'il est de plus en plus difficile pour l'ipV4, codée également sur 4 octets, de conserver des adresses distinctes.

4.2. Amélioration de sécurité de l'existant

Dans cette section, nous étudierons les propositions d'amélioration que j'ai pu faire concernant le système informatique de l'entreprise.

4.2.1. Sandboxing serveur

Dans un premier temps, j'ai envisagé d'améliorer la sécurité du serveur en proposant du sandboxing avec *Docker*.

FIGURE 23. – Logo de Docker

Il s'agit d'un logiciel automatisant les déploiements d'applications au sein de conteneurs. Il a le double avantage de proposer la sauvegarde de l'application déployée via un *snapshot* facilitant sa réinstallation future ou son export dans une autre machine, et de segmenter le logiciel en l'isolant au sein d'une machine virtuelle. La conséquence est qu'en cas d'intrusion, si l'utilisateur malveillant parvient à élever ses droits sur l'application, il ne pourra pas sortir de l'espace sandboxé et donc corrompre d'autres éléments du système.

4.2.2. Chiffrement du protocole de communication

À mon arrivée, les communications entre Nemo et Switch n'étaient pratiquement pas protégées. Les seules mesures mises en place étaient :

- L'obfuscation : le protocole utilisé ayant été créé pour nos besoins, il n'est pas évident de lire le message envoyé sans connaitre le codage du paquet.
- L'inversion de bits : afin de sécuriser *a minima* les échanges, chaque bit du message est inversé.

Les informations circulant par ce biais étant sensibles, j'ai jugé important de les chiffrer. Dans un premier temps, il m'a fallu lister les contraintes inhérentes au matériel avant de proposer un schéma de chiffrement.

Concernant le système, les limites sont au nombre de quatre :

- L'ip de la Nemo est inconnue du serveur. Elle ne peut donc permettre d'identifier le matériel avec certitude.
- La Nemo est *stateless*[18], il n'était donc pas envisageable d'établir une connexion sécurisée dans le temps.

18. « Un serveur stateless est un serveur qui traite chaque requête comme une transaction indépendante et sans relation avec une quelconque précédente requête. ». (Source : Wikipédia)

- Le paquet ethernet doit faire moins de 1500 octets : cette limitation repose sur l'utilisation de la couche UDP ; il n'est donc pas possible d'alourdir les échanges.
- La data mensuelle autorisée pour la Nemo est de 1Mo, cette limite étant imposée par notre abonnement.

Je me suis donc dirigé vers un échange de clé à intervalle régulier servant à chiffrer bit-à-bit la communication. En effet, les algorithmes de type RSA étaient trop lourds pour l'échange, de même que le chiffrement symétrique, alors que la limite du 1Mo de données par mois interdit formellement les protocoles comme SSH.

Une fois les limites analysées, il découle du cours de GS15 (« *Cryptologie et signature électronique* ») dispensé par Rémi COGRANNE que le protocole le plus adéquat pour l'échange de clé serait basé sur Diffie-Hellman [19]. Le renouvellement des clés se ferait à l'initiative de la Nemo à intervalle régulier (environ toutes les deux semaines, modulo un jour ou deux).

À l'heure actuelle, ce chiffrement n'a pas encore pu être implémenté par les ingénieurs, mais il est prévu pour le dernier trimestre 2016.

4.2.3. Améliorations de la base de données

On l'a vu (cf. *supra*, **3.2.4 - Conception de base de données**), la base de données a subi un *refactoring* complet. Cela a permis d'accroitre sa sécurité, mais ce n'est pas la seule amélioration que l'on a apporté.

En effet, certaines données caractérisées de sensibles — comme les RIB/IBAN des clients — se devaient d'être protégées des attaques extérieures. Une solution temporaire a été adoptée en attendant de pouvoir mettre en place une sécurité plus importante, par le chiffrement synchrone côté serveur à partir d'une clé stockée directement dans un fichier local.

Enfin, des *dumps* réguliers ont également été mis en place afin d'assurer la meilleure cohérence possible en cas de corruption de la base.

4.2.4. Attaque du serveur : sécurité système

Le dimanche 10 avril 2016 notre serveur a subi une attaque réussie. Elle a été très rapidement identifiée car le pirate a fait l'erreur de nous retirer les accès SSH.

Nous avons décidé de monter une partition *recovery* afin de reprendre la main sur la machine et d'investiguer. Après une rapide analyse, nous avons pu voir que seuls quelques éléments avaient été modifiés (*voir* Figure 24), notamment les fichiers `/etc/shadow`, `/etc/passwd`, `/etc/ssh` ainsi que le dossier `/root/.ssh/`. Heureusement, nous disposions d'un *backup* récent, et j'ai été en mesure d'effectuer la restauration. L'absence de modifications supplémentaires nous a mis à l'abri de *backdoors* [20].

19. Diffie-Hellman est une « méthode par laquelle deux agents [...] peuvent se mettre d'accord sur un nombre (qu'ils peuvent utiliser comme clé pour chiffrer la conversation suivante) sans qu'un troisième agent [...] puisse découvrir le nombre, même en ayant écouté tous leurs échanges. ». (Source : Wikipédia)

20. Une *backdoor*, ou porte dérobée, est une fonctionnalité inconnue de l'utilisateur légitime, qui donne un accès secret au logiciel, le transformant en cheval de Troie

```
root@rescue:/part2# find ./ -newermt "2016-04-10" -ls
2          4 drwxr-xr-x  22 root     root         4096 Apr 11 13:50 ./
1835009    4 drwxrwxrwt   7 root     root         4096 Apr 11 09:04 ./tmp
1835014    4 drwxrwxrwt   2 root     root         4096 Apr 11 09:04 ./tmp/.Test-unix
1835013    4 drwxrwxrwt   2 root     root         4096 Apr 11 09:04 ./tmp/.font-unix
1835010    4 drwxrwxrwt   2 root     root         4096 Apr 11 09:04 ./tmp/.X11-unix
1835011    4 drwxrwxrwt   2 root     root         4096 Apr 11 09:04 ./tmp/.ICE-unix
1835012    4 drwxrwxrwt   2 root     root         4096 Apr 11 09:04 ./tmp/.XIM-unix
5901298    4 -rw-r--r--   1 root     staff          15 Apr 10 02:18 ./usr/local/rtm/etc/rtm-ip
3932161    4 drwxr-xr-x  97 root     root         4096 Apr 11 13:55 ./etc
3932315    4 -rw-r--r--   1 root     root          222 Apr 11 09:04 ./etc/motd
3933870    4 -rw-r--r--   1 root     root           18 Apr 10 04:24 ./etc/passwd
3932592    4 drwxr-xr-x   2 root     root         4096 Apr 11 14:03 ./etc/ssh
3933482    4 -rw-r--r--   1 root     root          277 Apr 10 04:24 ./etc/shadow
5505029    4 drwxr-xr-x   2 root     root         4096 Apr 10 04:24 ./root/.ssh
5506729    4 -rw-r--r--   1 root     root           18 Apr 10 04:24 ./root/.ssh/authorized_keys
5506748    4 -rw-r--r--   1 root     root          416 Apr 10 04:24 ./root/.ssh/authorized_keys2
```

FIGURE 24. – Fichiers modifiés depuis l'attaque

L'étude du fichier `/etc/shadow` (*voir* Figure 25) contenant les mots de passe utilisateur chiffrés, nous a révélé d'autres informations. Le pirate a supprimé nos accès et a changé le mot de passe *root*, mais une trace de son passage a subsisté : l'en-tête du fichier est le suivant : `REDIS0006-ok@-`. Il a été généré par Redis, un système de base de données, sans doute mal protégé, et par lequel l'attaquant a pu éditer le fichier `/etc/shadow`.

```
root@rescue:/part2# cat /etc/shadow
REDIS0006-ok@-

root:$6$EHBHDvI5pLodfTEC$0vv1J17NJnsiLp8KTJvCWt3OM8cpZfebz2G9cS2rVMOY04fvbEpvAvjWxY60p.GKLeGpY/
42WCJDtJZbC.8su0:16738::::::
ram:$6$EHBHDvI5pLodfTEC$0vv1J17NJnsiLp8KTJvCWt3OM8cpZfebz2G9cS2rVMOY04fvbEpvAvjWxY60p.GKLeGpY/
42WCJDtJZbC.8su0:16738::::::


-F-+}-k
```

FIGURE 25. – La modification d'accès dans le fichier /etc/shadow

J'ai donc pris l'initiative de désinstaller Redis, car après vérification il avait été historiquement installé en tant que dépendance d'un service qui n'était plus utilisé. J'ai également supprimé l'utilisateur ajouté par le malware.

J'ai par la suite renforcé les règles pare-feu d'*iptables*, installé *PortSentry*[21] ainsi que *Fail2ban*[22], et enfin limité l'accès aux services web sensibles à une plage d'ip dans Apache.

Concernant l'attaque, il s'agissait vraisemblablement d'un programme automatique dont le but était de se répliquer afin d'infecter d'autres systèmes. En effet, les fichiers logs m'ont permis de tracer l'ip qui s'était introduit dans notre système, et il s'est avéré

21. « Portsentry est un programme de détection et de blocage de "scan de ports" (généralement programme qui scanne votre machine à le recherche de ports ouverts, en général dans le but de préparer une attaque). » (Source : wiki.debian-fr.xyz)

22. « Fail2ban lit les logs de divers services (SSH, Apache, FTP...) à la recherche d'erreurs d'authentification répétées et ajoute une règle iptables pour bannir l'adresse IP de la source. ». (Source : ubuntu-fr.org)

que la machine possédant cette adresse hébergeait un site personnel britannique basé sur *Wordpress* (connu pour ses nombreuses failles de sécurité). Les propriétaires ont été contactés pour les en avertir, sans réponse ; cela prête à penser qu'il s'agit donc d'une attaque lancée par un *script kiddie*[23].

4.3. RSSI

J'ai pu également effectuer différentes missions attachées aux attributions d'un Responsable de la Sécurité des Systèmes d'Informations : à savoir un audit de sécurité et un dossier de déclaration à la CNIL.

4.3.1. Audit de sécurité

Ne disposant que d'un temps limité pour l'analyse des risques, j'ai choisi de m'appuyer sur une méthode anglaise inventée par Siemens : *Cramm*[24] (***C****CTA*[25] ***R****isk* ***A****nalysis and* ***M****anagement* ***M****ethod*). Son approche est lourde lorsqu'elle est exhaustive, mais j'ai pu me rendre compte qu'elle était toujours efficace dans le cas d'une petite structure comme Park'n Plug en l'édulcorant. Composée de trois étapes (identification de l'existant, évaluation des menaces et des vulnérabilités, choix des remèdes), j'ai pu l'appliquer sans difficultés.

Je me suis également appuyé sur divers documents comme les recommandations OWASP [8] (Open Web Application Security Project) ou d'autres ouvrages de sécurité [5], [2] et [3], qui, s'ils ne sont pas des œuvres de référence, permettent malgré tout de mener à bien une mission aussi délicate qu'un audit de qualité.

Cartographie des *assets* de la société

La première étape de mon audit a été l'identification des actifs de la société (*voir* Table 1). J'ai choisi de les diviser en cinq catégories afin d'en faciliter le traitement :

- Le hardware : il s'agit de tout ce qui touche au matériel informatique présent dans le SI ainsi que les choix portant sur leur implémentation.
- Le code : pour juger de la solidité logicielle développée au sein de l'entreprise, une revue de code sommaire sera effectuée.
- Les process : ils regroupent les procédures, techniques ou non, normées ou non, propres à l'entreprise.
- L'informatique pure : il s'agit principalement de flux d'informations qui transitent par nos serveurs et qui n'entrent pas dans les catégories précédentes.
- L'humain : élément central de tout SI, il peut lui aussi être sujet à des défaillances.

23. "Script kiddie" est un ≪ terme péjoratif d'origine anglaise désignant les néophytes qui, dépourvus des principales compétences en matière de gestion de la sécurité informatique, passent l'essentiel de leur temps à essayer d'infiltrer des systèmes, en utilisant des scripts ou programmes mis au point par d'autres. ≫. (Source : Wikipedia)

24. Plus d'informations sur developpez.com

25. ***C****entral* ***C****omputer and* ***T****elecommunications* ***A****gency*, agence gouvernementale britannique

Catégorie	Assets
Hardware	Borne Câble Nemo Carte SIM Serveur OVH Redondance
Code	Python côté serveur Python côté embarqué Node.js Codec de communication
Process	Installation / Configuration de Nemo Accueil / Départ du personnel Déploiement de nouvelles versions Test d'une nouvelle borne électrique Accès serveur (root) Badgeage RFID PRA/PCA Politique de renouvellement des mots de passe Audits de sécurité Journalisation
Informatique	Base de données MySQL Serveur « *Switch* » API *Strongloop* dev.gestio Serveur SMTP Certificats gratuits (PKI) Environnement de développement (Git) Environnement web
Humain	Utilisateurs Administrateurs Éxtérieurs à l'entreprise Charte informatique

TABLE 1. – Assets de Park'n Plug

Afin de cerner les différents *assets*, je me suis intéressé à tous les éléments extérieurs interagissant avec le système, considéré comme une boite noire (*voir* Figure 26). Ce sont à partir de ces éléments que ma liste s'est créée et que j'ai pu simuler le cheminement d'une information d'un bout à l'autre du système.

L'identification des process en revanche, n'a pas été chose aisée pour la simple raison que presqu'aucun d'entre eux n'est documenté à ce jour.

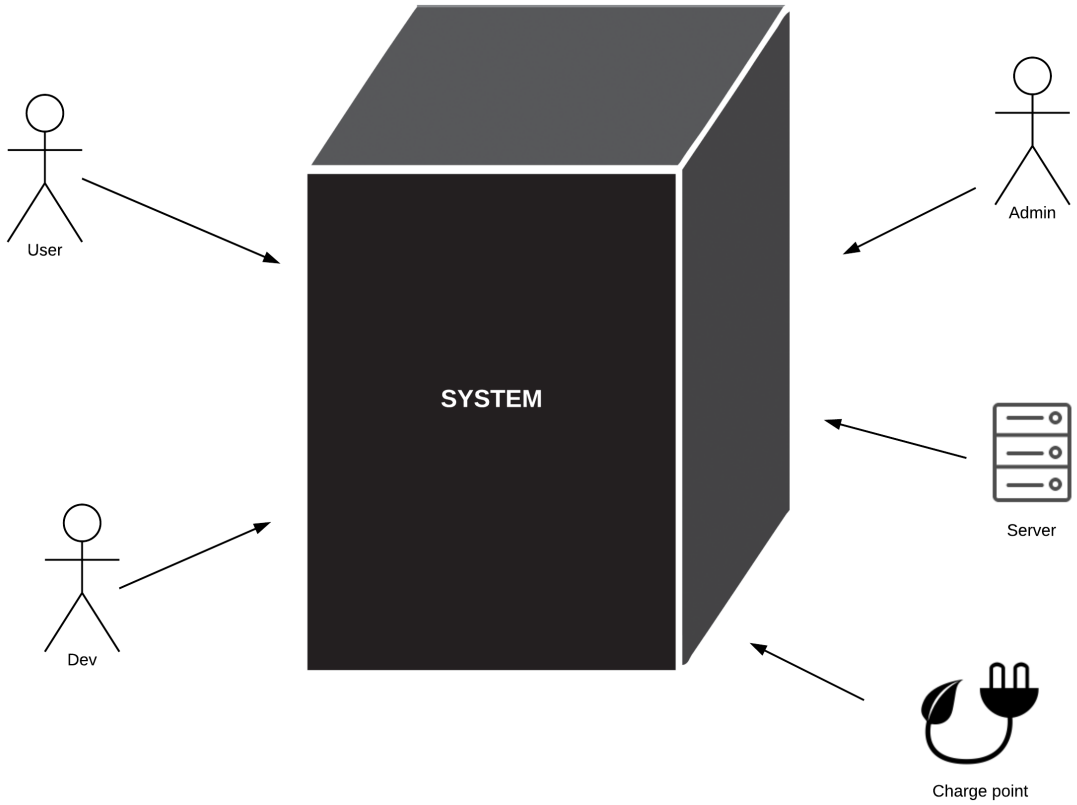


FIGURE 26. – Interactions avec le système, vu comme une boite noire

Analyse des menaces et vulnérabilités

Une fois les *assets* cartographiés, il convient d'identifier les menaces. Il m'a fallu procéder ainsi pour chacune des catégories listées ci-dessus.

Hardware. Les vulnérabilités hardware sont assez peu nombreuses. Même si, poussé à l'extrême, un scénario de détournement par un utilisateur malicieux est toujours possible lorsqu'il a un accès physique au matériel, de nombreuses précautions ont été prises.

L'intégralité des équipements sur site étant équipé d'un système anti-vandalisme, il est très difficile d'accéder aux différentes connectiques dans le but de le détourner. Cela concerne aussi bien les **bornes de recharge** elles-même que les compteurs électriques ou le serveur **Nemo**.

De même, les supports physiques des échanges sont relativement bien protégés : les **câbles** sont passés dans des gaines métalliques et la communication vers le serveur central à partir de la Nemo se fait en GPRS. Ce protocole n'est certes pas des plus sécurisés, mais une attaque de type *Man in the Middle* reste difficile de par la nécessaire identification au serveur.

Les potentiels risques que l'ont peut relever sont d'un autre ordre. Ils concernent notre serveur distant, hébergé chez OVH. Matériellement ce dernier offre une sécurité suffisante via l'offre à laquelle nous avons souscrit chez *SoYouStart*, cependant, le véritable problème que l'on peut soulever n'est pas négligeable : aucune **redondance** n'est mise en place.

Code. Le code est l'un des éléments centraux de la startup, mais il peut être divisé en quatre catégories :

- **Python côté serveur** : la plupart du code sur le serveur Park'n Plug est développé en Python. C'est lui qui va interpréter les requêtes venant des Nemo et les envoyer vers la base.
- **Python côté client** : ce code est exécuté continuellement à partir du moment où la Nemo est mise en production. Il interprète les badgeages, contrôle le lancement des charges... mais principalement communique avec le serveur central.
- **Node.js** : il s'agit de la base de développement de Gestio.
- **Codec de communication** : langage propriétaire d'échanges très léger, comme vu *supra* (**2.2 - Présentation du logiciel *Gestionnary***), son fonctionnement a été mis à jour.

La revue de code que j'ai pu effectuer n'a pu qu'être succincte face à l'étendue du travail de relecture et du peu de temps disponible — soient des années de développement à analyser en quelques jours.

Néanmoins, j'ai pu me concentrer sur la lisibilité du code ainsi que sa maintenabilité. L'implémentation de l'outil GoGs (cf. *supra*, **3.1.3 - Mise en place d'outils de développement agile**) a été d'une grande aide afin d'effectuer ce travail.

On constate sur la Figure 27, comme ailleurs, que les scripts sont lisibles, correctement commentés et indentés — il s'agit dans tous les cas d'une nécessité pour du python, principal langage de développement.

La seule menace que nous pourrions identifier touche un point assez délicat, et est contraire à la lisibilité du code. En effet, si un agent malicieux parvient à s'approprier les sources de nos programmes, il sera aisé pour lui de le déchiffrer. C'est pourquoi on pourrait préconiser d'utiliser le procédé d'*obfuscation*[26] dans le but de rendre le code

26. L'obfuscation, ou « code impénétrable d'un programme informatique est un code dont la compréhension est très difficile pour un humain tout en restant parfaitement compilable par un or-

run.py 12KB — Permalink | History | Raw

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import os, sys
sys.path.append(os.getcwd())
import socket, traceback, json, threading, time, subprocess, traceback, syslog, tools


###################
## Environnement ##
###################

UDP_IP = [redacted]                    # Ip d'écoute (locale)
UDP_PORT = [redacted]                  # Port d'écoute
LOGFILE = [redacted]                   # Fichier de log
ROUTING_TABLE_FILE = [redacted]        # Fichier de la table de routage
CONFIG_UPDATES_INTERVAL = 60           # Intervale de temps entre chaque tentative d'update du codec
MAX_THREADS = 100                       # Nombre maximal de threads dédiés à la communication

ROUTING_TABLE = {}          # Données de routage des requêtes
THREADS = []                 # Pile de threads d'exécution
THREAD_TIMEOUT = 2000       # Temps maximal d'éxécution d'un thread


############
## Outils ##
############

def log(message):
    syslog.syslog(time.strftime("[%y/%m/%d - %H:%M:%S] - ", time.localtime()) + message + "\n")
```

FIGURE 27. – Revue de code

illisible aux intrus, mais le développeur serait également en peine de se relire.

Les risques sont ici suffisamment peu importants pour décider de maintenir une lisibilité saine.

Process. Les différents process identifiés étant assez nombreux, il ne seront pas tous traités dans le présent rapport. Seuls ceux susceptibles d'être menacés seront étudiés.

L'**installation d'une Nemo** est l'un des moments les plus critiques lors de la mise en place d'un nouveau matériel. En effet, c'est le seul moment où un pirate serait assuré d'une demande de mise à jour vers nos serveurs, car c'est au premier branchement que la Nemo effectue une demande de configuration vers le serveur central. Un *Man in the Middle* pourrait donc se faire passer pour notre interface et répondre au matériel. L'équipe a donc fait le choix de pré-paramétrer la Raspberry Pi en amont afin de s'assurer de l'adéquation des informations récupérées.

Néanmoins, une attaque est toujours possible, et c'est pour cela que les techniciens doivent être très vigilants lors de l'installation. L'impact qui pourrait en découler serait grave, mais localisé sur les parkings touchés — et non au système complet — faussant les données envoyées par la Nemo (principalement des données de facturation).

L'**accueil du personnel** ne posait aucun problème avant ma venue dans l'entreprise, mais la question de son **départ** n'était soumise à aucun process clairement défini. Cela

dinateur ≫ (Source : Wikipedia)

peut avoir pour conséquence d'aboutir à des accès non supprimés, ce qui constituerait un grand danger pour le SI, notamment en cas d'embauche d'un collaborateur par un concurrent. Le risque encouru serait la perte de nos données, ou pire, la récupération d'informations sensibles par l'entreprise rejointe.

L'**accès au serveur** est aussi sensible car de manière générale, la plupart des ingénieurs se connectent avec un accès `root`, ce qui est bien évidemment très dangereux en cas de fausse manipulation. Les risques peuvent être très différents (effacement pur et simple du contenu du serveur, perte d'accès...) et quasiment infinis.

Le process du **badgeage RFID** sur une borne (et donc sur une Nemo) est le process propre à l'utilisateur final. Il s'agit de son identification sur notre système. Un risque a été identifié lors de cette phase de connexion : un utilisateur malicieux pourrait falsifier son badge en vue de subtiliser l'identité d'un autre acteur. Cela aurait pour conséquence de débiter un autre compte et donc de générer une perte financière sèche pour Park'n Plug qui devrait rembourser ces dépenses.

Les **plans de reprise d'activité** et **de continuité d'activité** sont des documents tout simplement inexistants au sein de l'entreprise, de même que la **politique de renouvellement des mots de passe**. Cependant, les risques sont négligeables de par la taille peu importante de l'entreprise.

La **journalisation**, quant à elle n'est que partielle et mériterait d'être approfondie. Les risques sont minimes, d'autant que le *logging* systématique n'aurait que peu de sens à l'heure actuelle, et ce, dû à la connexion au serveur en `root` de façon quasi-systématique.

Informatique. De même que pour les process, nous ne nous pencherons ici que sur les *assets* les plus significatifs.

À mon arrivée, la **base de données MySQL** était modifiable par tous les utilisateurs et toutes les ip. La conception du système l'exigeait alors, mais il s'agissait d'un réel risque pour l'entreprise. Une intrusion pouvait en prendre le contrôle, et l'altérer de manière définitive, avec un impact dramatique pour Park'n Plug.

L'interface **dev.gestio** est également soumise à des risques. Il s'agit de l'ancien espace de configuration des Nemo (Gestionnary), toujours utilisé. Son point fort est qu'il est fermé à un usage exclusif des développeurs : connexion nécessaire et accès uniquement depuis l'adresse ip de l'entreprise. Néanmoins, la vétusté du code (cf. *supra* **2.2 - Présentation du logiciel *Gestionnary***) le prédispose aux failles de sécurités.

Nous utilisons également des **PKI**[27] gratuites, afin de s'assurer qu'elles soient enregistrée pour éviter les erreurs de PKI auto-générées (*voir* Figure 28). Le danger avec ces open-PKI réside dans l'autorité de certification : on ne peut s'assurer de la sécurité mise en place de son côté pour protéger les clés privées enregistrées chez elle.

Humain. Que ce soit côté **utilisateurs** ou **administrateurs**, le risque zéro n'existe pas. Un agent malicieux peut toujours intervenir pour pénétrer le système. Mais le

27. Utilisées principalement pour sécuriser les connexions à nos sites web, l'≪ infrastructure de gestion de clés ou Public Key Infrastructure (PKI) délivre des certificats numériques permettant d'effectuer des opérations de cryptographie. ≫ (Source : certeurope.fr)

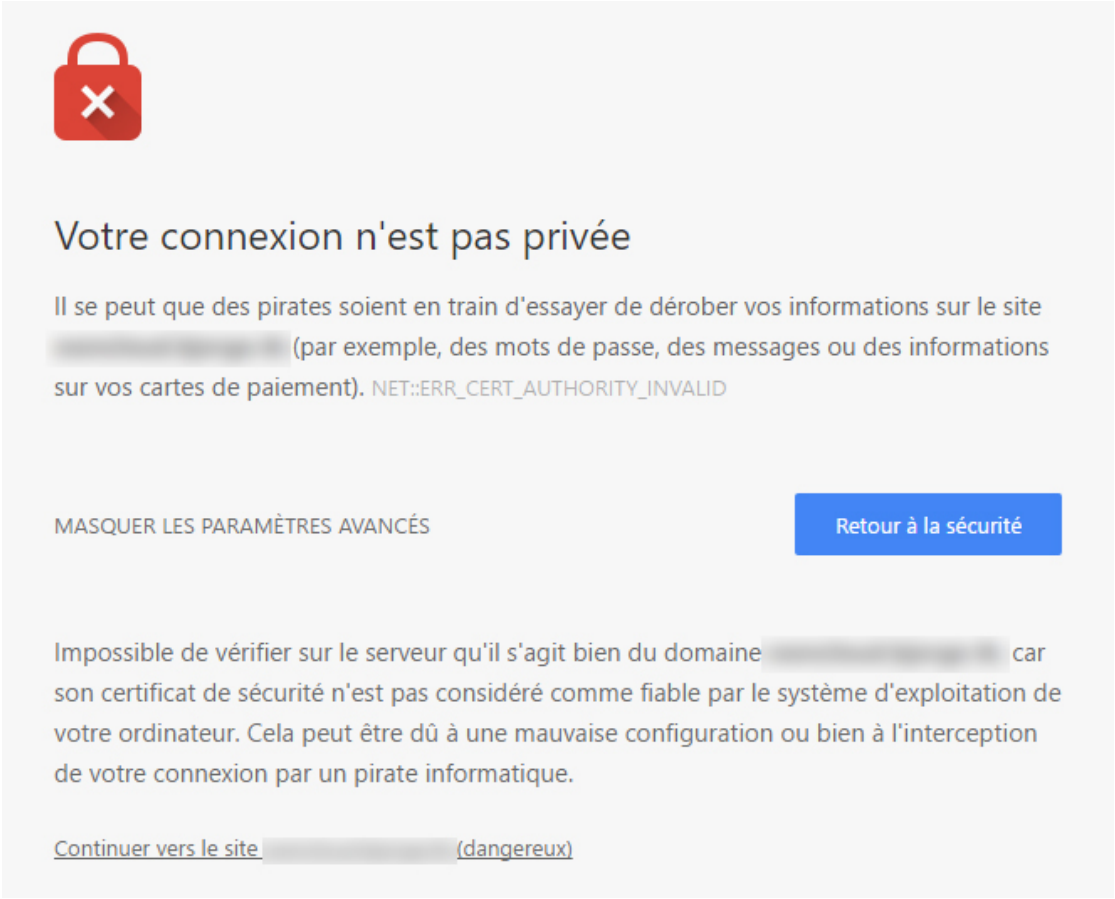


FIGURE 28. – Erreur lors d'une connexion avec une PKI auto-générée

risque principal côté administrateur réside dans l'erreur de bonne foi, laquelle peut être dévastatrice pour l'entreprise, alors que par ailleurs elle reste imprévisible.

Il est en revanche possible que le danger vienne des **agents extérieurs** à l'entreprise : techniciens de surface, ouvriers, voire même une personne lambda cherchant à s'introduire dans notre système. À l'aide de *social engineering*, il leur serait possible de pénétrer dans nos locaux et donc d'obtenir des accès physiques sur nos machines.

Enfin, le dernier risque identifié relève de la **charte informatique** : il n'y en a tout simplement pas et son absence peut porter atteinte à l'entreprise en cas de litige.

Évaluation du risque, calcul de l'impact et préconisations

L'étape suivant l'analyse des menaces et vulnérabilités est leur évaluation. Il s'agit de les identifier selon le modèle CIA [28], et les associer à une probabilité, puis les chiffrer pour évaluer les coûts pour l'entreprise.

Enfin, il s'agira d'effectuer des préconisations qui consistent en l'acceptation, la diminution, l'élimination ou le transfert [29] le risque.

Dans un premier temps, nous pouvons écarter certaines menaces car la probabilité est très faible, ou l'impact négligeable. Il est tout d'abord possible de mettre de côté les risques portant sur le matériel — à une exception près, cf. *infra* — et sur le code, car leur potentialité est extrêmement basse. Nous avons donc fait le choix de les accepter.

Le premier *asset* à relever est l'**absence de redondance**. Cela risque d'impacter directement la disponibilité : sans redondance, si le système tombe, aucun service ne sera assuré. S'agissant d'un serveur délocalisé chez un prestataire (OVH), nous disposons de services de qualité comme une assurance de remise en marche rapide ou du matériel de qualité. Mais le risque existe néanmoins et sa gravité financière dépend du temps nécessaire à la remise en fonctionnement. Il serait donc judicieux de mettre en place un second serveur permettant de prendre le relais en cas de défaillance du premier pour éliminer le risque.

L'**installation de la Nemo** présente peu de risques, mais ils touchent l'intégrité de l'information. Si le système est altéré à ce moment précis, il nous sera impossible d'identifier l'intrusion. La probabilité reste faible mais l'impact financier peut être important si la Nemo équipe un parking de grande envergure. Il est donc possible de transférer le risque en transférant contractuellement sa responsabilité à l'installateur. Ce sera à lui de prendre les précautions qui s'imposent afin de s'assurer qu'aucune menace ne se concrétise.

Le process de **départ du personnel** pose un risque de confidentialité : un accès après avoir quitté l'entreprise n'est jamais le bienvenu. Sa probabilité croît avec le nombre d'employés et son *turnover*, et le coût sera de plus en plus important à mesure que l'entreprise gagnera en ancienneté et développera de nouvelles technologies. Nous pouvons

28. La triade CIA signifie *Confidentiality, Integrity and Availability* (Confidentialité, Intégrité et Disponibilité) (*cf.* [9] pour plus d'informations)

29. Dans notre cas, le transfert de risque vers une assurance ne se présentera pas, dû à la taille de la structure.

éliminer ce risque en mettant en place une politique de clôture systématique des accès au départ du personnel.

Un autre risque facile à éliminer est **la connexion au serveur en tant que** `root` malgré son importance. Il contrevient aux trois principes CIA simultanément — ainsi qu'à celui de non-répudiation par la même occasion. Même si la probabilité d'altération des données est faible, il convient de la diminuer en interdisant tout accès `root` et en préconisant l'usage de la commande `sudo` — le risque sera toujours présent, mais nécessitera un niveau de concentration supplémentaire. Il reste en revanche difficile à évaluer la valeur du risque tant les conséquences d'une telle utilisation peuvent être aléatoires.

L'**absence de PRA et de PCA** ainsi que de **politique de renouvellement des mots de passe** sont, on l'a vu, un risque négligeable à l'heure actuelle grâce à la petite taille de Park'n Plug. Il est donc possible d'accepter ce risque.

Si l'interdiction de l'utilisation de l'utilisateur `root` est mise en place, une meilleure politique de **journalisation** peut être initiée. Le *logging* n'est pas un risque qui peut être associé à la triade CIA, mais on peut l'approcher de la traçabilité et de la non-répudiation.

Les menaces portant sur **MySQL** et **dev.gestio** ne sont plus d'actualité depuis la refonte de l'outil.

L'usage des **open-PKI** et les risques liés aux **agents extérieurs** concernent respectivement l'intégrité et à la confidentialité. Financièrement, le premier est négligeable. Le second l'est également quant à sa potentialité. Nous choisissons donc de les accepter.

Enfin, le dernier risque est celui de l'**absence de charte informatique** : il n'entre pas dans le cadre du CIA mais peut néanmoins avoir de lourdes conséquences financières en cas de problème juridique. Même s'il présente peu de chances de se concrétiser, nous préconisons cependant la mise en place d'une telle charte.

Comparaison avec l'analyse antérieure

On peut voir en comparant la première analyse (*voir* Annexe E page XXI) effectuée l'an passé que de nombreux risques ont été réduits (programmation, services web...). D'autres ont pu être annulés (système de paiement devenu manuel) alors que certains n'ont pas évolué (redondance ou charte informatique).

On constate finalement qu'avec l'apparition de nouveaux process, d'autres problématiques ont parallèlement vu le jour.

4.3.2. Déclaration à la CNIL

Cette tâche n'a pas pu être menée à bien par manque de temps à la fin du stage, mais sera réalisée dès la rentrée car j'ai eu la chance d'avoir eu une proposition d'emploi de la part de Park'n Plug.

Elle consiste à faire une déclaration simplifiée à la CNIL en leur faisant parvenir la nouvelle structure de la base de données, conformément à la Loi Informatique et Libertés.

Bilan du stage

Ce stage au sein de l'entreprise Park'n Plug a été très profitable pour moi. J'ai pu en tirer de nombreuses leçons et suis fier de continuer l'aventure au sein de cette équipe dynamique.

Enrichissement personnel

Avoir eu la possibilité de mener de bout en bout un projet de conception logicielle a été pour moi une réelle chance. La confiance qui m'a été accordée jusqu'à pouvoir assumer le statut de chef de projet sur Gestio m'a permis de bénéficier d'une expérience professionnelle extrêmement valorisante.

Il est en effet impossible d'acquérir certaines compétences (comme le management par exemple) en se réduisant au seul apprentissage théorique. La pratique reste dans ce cas une nécessité.

Au niveau personnel j'ai également beaucoup appris : que ce soit au niveau des méthodes de travail acquises à l'UTT et que j'ai pu éprouver en situation réelle, mais aussi au niveau de l'apprentissage de nouveaux langages et *frameworks*, de l'approfondissement des connaissances dans d'autres, ou encore dans la découverte d'un métier gravitant autour d'un univers très particulier.

L'ambiance de travail m'a aidé à m'épanouir chez Park'n Plug, au sein d'une équipe soudée et dans un contexte très agréable. J'ai ainsi pu saisir l'opportunité d'expérimenter de nouvelles technologies particulièrement intéressantes grâce au parcours différent de chaque collaborateur.

Enfin, j'ai pu découvrir l'aspect très pratique de la sécurité, principalement survolé pendant le semestre de master à l'UTT, par manque de temps.

Difficultés rencontrées

À mon arrivée, j'appréhendais un peu l'utilisation d'un nouveau langage. S'agissant d'une startup, nous ne disposions pas forcément de beaucoup d'expérience au sein de l'entreprise, le savoir étant alors difficile à transmettre. Par ailleurs, le fait de devoir encadrer d'autres stagiaires sur cette même technologie, peu maitrisée au début du stage, était également un challenge de taille.

Mais la formation de l'UTT, nous a appris à apprendre, et cela reste la formule idéale pour ce type de problématique.

Par ailleurs, j'ai été confronté à quelques difficultés en matière de sécurité. Il n'est pas aisé de mettre en place des solutions complexes avec une formation sur seulement un

semestre d'apprentissage.

Malgré tout, les ressources dont j'ai pu disposer (documents du SCD, expérience de mes collègues, rencontres de professeurs de l'UTT...) ont été largement suffisantes pour m'aider dans mes tâches au quotidien.

Réponse à la problématique

L'enjeu du stage était double : concevoir une solution permettant aux usagers Park'n Plug de gérer leur consommation, et améliorer la sécurité globale du SI.

L'objectif a été rempli en ce qui concerne la conception, même si j'aurais apprécié d'avoir plus de temps à dédier à la réalisation. En effet, l'urgence de la mise en ligne du site commercial a pris partiellement le pas sur le développement de Gestio.

En revanche, la sécurisation du système d'information n'a été que partielle. En effet, une partie des missions du RSSI consiste à émettre des recommandations, l'implémentation se faisant dans un second temps. Mais cela est également dû au manque de temps qui m'a contraint à prioriser les tâches (*ie.* la déclaration à la CNIL qui a été repoussée). Cependant, la mission a été réalisée avec application et l'objectif a été atteint dans les catégories concernées.

Deuxième partie

Annexes

A. Plaquette commerciale

L'Annexe A est la plaquette envoyée aux prospects afin qu'ils prennent conscience de la facilité d'accès au rechargement de véhicule électrique grâce à la solution proposée par Park'n Plug.

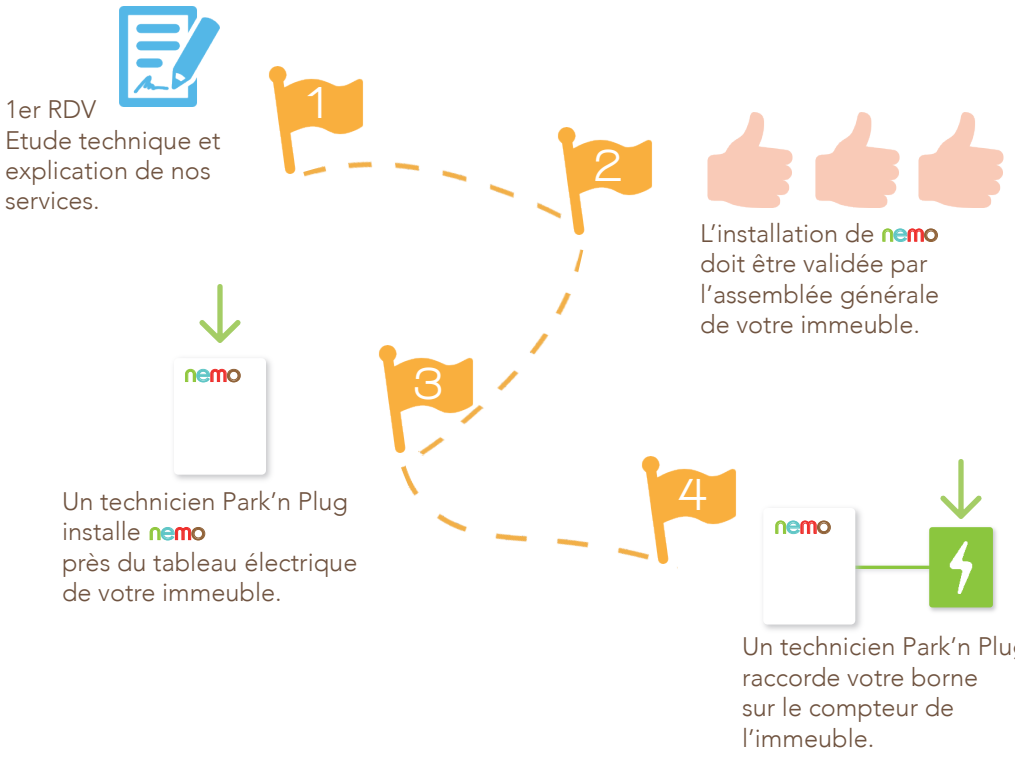
Rechargez votre
véhicule électrique
avec

park'n plug

RECHARGE VÉHICULE ÉLECTRIQUE

La recharge simple

Comment installer nemo dans votre résidence ?



1er RDV
Etude technique et explication de nos services.

L'installation de nemo doit être validée par l'assemblée générale de votre immeuble.

Un technicien Park'n Plug installe nemo près du tableau électrique de votre immeuble.

Un technicien Park'n Plug raccorde votre borne sur le compteur de l'immeuble.

Bénéficiez également du crédit d'impôt pour la transition énergétique (CITE)

park'n plug
RECHARGE VÉHICULE ÉLECTRIQUE

1, rue de l'abbé Roger Derry
75015 Paris

01 77 18 98 20

www.parknplug.fr
contact@parknplug.fr

B. Les produits Park'n Plug

L'Annexe B est une fiche produit fournie aux clients en fonction de la solution la plus adéquate pour leur structure.

FICHE PRODUIT NEMO

park'n plug
RECHARGE VÉHICULE ÉLECTRIQUE

CARACTERISTIQUES

10 sorties de commande de bornes de recharge
Gestion par microprocesseur 700 Mhz
Sauvegarde des données par carte Flash 8 Gb
Port USB
Port Ethernet
Ports RS485 (3)
Port RS 485 Modbus
Modem GPRS sur port RS 485
Alimentation : 85 à 264 V
Consommation : 25 W

Dimensions : 240 X 250 X 90 mm

ACCESSOIRES

Pack électrique
Chaque borne installée nécessite un pack électrique pour assurer l'asservissement et le comptage électrique.
Le pack est livré avec un cordon de raccordement de 1,5 m (5 fils)
Enveloppe plastique avec rail DIN 6 modules

Pack électrique 32A monophasé	ref Pack mono 32
Pack électrique 4X40A triphasé	ref Pack tri 40

Contrôle d'accès STID_LXC
L'usage des bornes du parking est sécurisé par un ou plusieurs lecteurs de badges avec claviers sur bus RS 485.

Lecteur de badges saillie antivandale	ref LXC
Badges compatibles VIGIK bleu ou autres coul.	ref CLE/B

SERVICES

Gestion des accès
Nous gérons les accès à la borne par badge ou code. Cela permet de sécuriser le point de charge, mais aussi d'identifier le client de la borne, utile aux services de gestion des consommations et de gestion d'énergie.

Gestion des consommations & service de facturation
Les consommations individuelles sont télé-relevées et transmise à notre cloud.
Nous pouvons émettre une facture relative aux coûts de charge individuels.
Cela permet au besoin de dédommager le gestionnaire d'immeuble.

Gestion d'énergie et pilotage intelligent ©
Grâce à notre pilotage intelligent, réduisez votre besoin en puissance globale de 40 %*.
Cela fonctionne grâce à notre algorithme breveté de prédiction des besoins de charge.

Supervision
Un dashboard interactif vous permet d'interagir en temps réel avec votre infrastructure (détections des pannes, ajout/suppression de compte, historique de consommations, limite de puissance...).

Agence commerciale principale
96, avenue de Suffren 75015 Paris
(Siret) 538 605 197 00010
SAS au capital de 310 000 €

Tél +33 1 77 18 98 20
Email : contact@parknplug.fr
Site : www.parknplug.fr

* Résultat moyen obtenu lors de tests de simulation en temps réel pour un nombre de VE compris entre 30 et 100 par parking

C. Cahier des charges

L'Annexe C est un extrait du cahier des charges. Ce document a été validé par Samy Jacquet, mon responsable, ainsi que par Pascal Toggenburger.

Sont inclus :

- Le sommaire, aperçu rapide du contenu
- Un extrait des exigences fonctionnelles et techniques du Cahier des Charges
- Un des diagrammes d'activité, afin de faciliter la compréhension du lecteur non technicien
- Le MCD de la base refondue ainsi qu'un extrait du MLD correspondant

CAHIER DES CHARGES

Plateforme Gestio

Release 1.0
Dernière édition le 29 juin 2016

Parkn'n'Plug

Table des matières

3. Exigences fonctionnelles

Le choix a été fait de classer les différentes fonctionnalités par type d'usager. Deux autres catégories de fonctionnalités émergent : celles qui sont communes à tous les utilisateurs, et les fonctions exécutées lors de tâches automatiques.

Il est important de noter que toutes les fonctionnalités listées ci-dessous ne seront pas forcément implémentées dans un premier temps, il s'agira de prioriser les tâches les plus importantes.

3.1. Fonctionnalités automatiques

Ces fonctionnalités sont automatiques car elles ne font pas intervenir les acteurs et sont donc déclenchées par des évènements précis.

3.1.1. Envoyer des alertes automatiques

Cette fonctionnalité va être déclenchée par différents type d'évènements. Par exemple, lorsque le contact entre une Nemo et *Gestio* sera perdu, cela va automatiquement envoyer un message aux utilisateurs concernés l'informant d'une maintenance à prévoir, une réparation à programmer au plus vite, etc. D'autres types pourront être développer comme un message automatique d'information lorsque le rechargement d'un véhicule est terminé etc.

3.1.2. Réinitialiser le mot de passe

Celle-ci est essentielle pour tout logiciel, toute application. En effet, il est courant d'avoir un utilisateur qui ne se souviens plus de son mot de passe ou qui souhaite le changer. Faire passer cette fonctionnalité en automatique va faire économiser beaucoup de temps à l'administrateur.

3.2. Fonctionnalités communes aux usagers

3.2.1. Consulter le Dashboard

Le Dashboard est la page d'accueil de l'utilisateur connecté. Elle permet d'avoir les informations primordiales d'un simple coup d'œil. Ce DashBoard s'adapte à chaque profil utilisateur pour avoir plus de pertinence.

3.2.2. Gérer "Mon Compte"

Via cet accès, l'utilisateur pourra :

- Modifier ses coordonnées personnelles (nom, prénom, mail, adresse, login)
- Gérer ses paramètres et informations bancaires
- Changer d'avatar
- Modifier son mot de passe *(ce processus relève plus des fonctionnalités automatiques de* Gestio*)*
- Changer la langue par défaut

3.2.3. Créer / Consulter un ticket (interface de Support & Assistance)

Le comportement de cette fonction sera également différent en fonction de l'acteur concerné, mais néanmoins sera transverse.
Il s'agit d'une interface permettant de faire remonter un problème quelconque (facturation, borne...) qui sera ensuite redirigé vers l'acteur concerné en fonction de la nature du problème.

3.2.4. Consulter ses alertes détaillées

Si les dernières alertes non traitées/non lues sont présentes sur le Dashboard, il est nécessaire de pouvoir retourner dans l'historique complet des notifications passées. Ce sera donc via une fenêtre dédiée que l'utilisateur pourra y accéder.

3.2.5. Se connecter / S'inscrire

Fonction élémentaire, la connexion se fait par association mail ou login avec un mot de passe. L'inscription se fait à la volée en renseignant son adresse mail, le paramétrage du mot de passe se fera alors par la suite.

3.3. Le client final

Il s'agit du profil client générique, à savoir l'utilisateur qui branche son véhicule électrique sur une borne de recharge.

3.3.1. Consulter son Dashboard

Le Dashboard va être le premier visuel du client lors de sa connexion à Gestio. Il devra donc comprendre les principales informations concernant l'utilisateur final à savoir : ses consommations instantanées, ses dernières factures, ses alertes (bornes indisponibles...) et des statistiques (graphiques, chiffres clés) évocatrices.

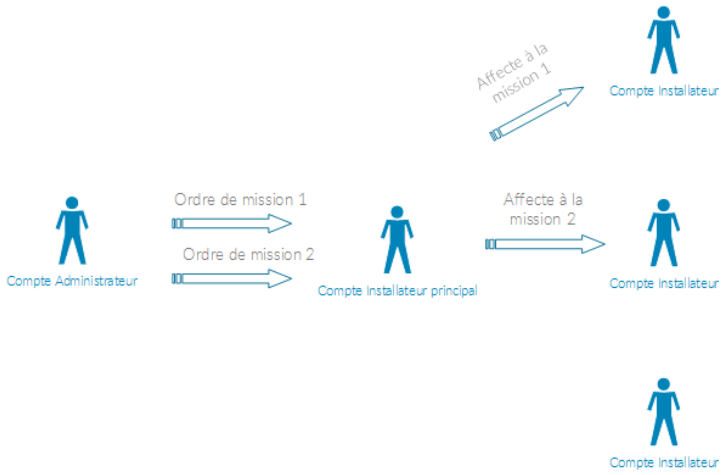


Figure 3.2. – Affectation d'une mission par le commercial

3.6.3. Gérer les commandes de Nemos

Cette fonctionnalité doit permettre au préparateur d'avoir un point de vue globale sur toutes les commandes de Nemos en cours (traitement des commandes en attente ou en cours) et les commandes traitées.
Les Nemos seront automatiquement créées dans la Base de Données lorsque le QrCode associé à la Nemo sera scanné. L'état de cette Nemo sera automatiquement défini comme "en attente". Ensuite, deux cas se présentent :

1. Tout est renseigné via l'interface, l'état de la Nemo passe alors automatiquement de "en attente" à "en installation".
2. Une partie seulement est renseignée (la Nemo n'est pas encore affiliée à une commande). L'état de la Nemo passe alors de "en attente" à "en cours".

Les profils "comptable" et "commercial" sont encore à l'état d'ébauche voir inexistants puisqu'ils n'ont pas été définis comme l'une des priorités du projet Gestio. En effet, *Gestio* doit déjà traiter de l'opérationnalité des Nemos, donc de leur configuration afin de pouvoir assurer le service client (paiement, gestion des consommations). Leurs profil et fonctionnalités ne sont donc pas représentatifs de leurs fonctions générales.

4. Exigences techniques

4.1. Exigences techniques côté serveur

4.1.1. Serveur applicatif

L'application *Gestio* se voulant légère et dynamique (notamment concernant les graphiques de charges qui doivent être évolutifs en temps réel), le choix a été fait de porter le développement sous *Node.js*. Concernant les frameworks, le choix reste libre, même s'il semble se diriger vers *Express.js*.

4.1.2. Serveur de données

Concernant la gestion des données, il faut intégrer *Gestio* dans l'existant, à savoir *MySQL*.

La migration des données devra être effectuée avant la mise en production car la base actuelle est déjà utilisée, mais n'étant pas optimisée elle sera remaniée.

Modèle conceptuel de données

La base de données sera conçue selon le modèle présenté en Annexe B page III.

Modèle logique de données

Le modèle logique de données est disponible en Annexe C pages V à XIV. Sauf mention contraire, tous les *int*s sont de type *unsigned*.

Commentaires de la base de données

Cette section vise à rendre compte du fonctionnement de la base de données ainsi que de la logique avec laquelle le modèle de données a été conçu. Globalement, la base s'organise autour de serveurs Nemo qui gèrent des sites. Sur ces sites sont concentrés des groupes de bornes que peuvent utiliser les clients. Leurs consommations sur ces bornes leur seront facturées. D'autres profils et entités s'organisent ensuite autour de ce processus. Ce modèle est composé de tables (préfixées par *"e_"*) et d'associations (préfixées par *"r_"*).

Avant toute chose, la table `e_pending_update` est à présenter du fait de sa spécificité. Cette table va stocker toutes les modifications faites en base de données mais qui n'ont pas encore été transmises à la Nemo. En effet, la Nemo ne communique pas continuellement avec le serveur et n'est donc pas à jour en temps réel. Cette table va stocker ces

modifications en attente. Elle est donc fortement liée à une Nemo et est optionnellement liée à un utilisateur, un relais, une borne ou un moyen d'authentification. Cette table comporte un type pour préciser la nature des MàJ (0 : update, 1 : add, 2 : delete).

De même existe une table `e_pending_invitation` qui possède un identifiant unique et un booléen caractérisant la validité de cette modification (validé ou non par l'administrateur ?). Cette table va concerner toutes les invitations de création de compte ou de délégations de droits d'une partie des utilisateurs (gestionnaires, clients, etc.). La table `r_invites` met donc en relation un utilisateur et un groupe d'utilisateurs comprenant un à plusieurs utilisateurs.

La Nemo possède un identifiant unique, toutes ses configurations et paramètres :

- un serial qui est l'identifiant unique de la Raspi composant la Nemo (associée au QR code),
- un statut (active ou non),
- la date du dernier contact avec la Nemo,
- une puissance maximale de la Nemo en Watt,
- le numéro IMEI du modem,
- l'iccid à savoir l'identifiant unique d'une carte SIM d'un modem,
- la consommation minimale à partir de laquelle on considère qu'il y a bien une conso par un utilisateur (en kW),
- le signal du modem qui exprime la qualité du réseau, de la réception du signal (%, de 0 à 30),
- la quantité maximale d'information sur la SIM par mois (max datarate en k octet),
- la marge de sécurité en % du modem,
- l'intervalle de temps en secondes entre deux consommations qui permet de savoir si la conso est un résidu ou bien une charge,
- l'intervalle de temps en secondes entre chaque relevé de consommations permanentes,
- les versions de software, de l'os, du codec et hardware de la Nemo (raspi),
- l'IP locale de la Nemo, le netmask et le gateway,
- l'intervalle de temps entre deux package envoyés de la Nemo au serveur à diviser par l'attribut ci-dessous.
- le ratio de génération de données en temps réel entre chaque communication (à multiplier par 10)
- le moyen de communication de la Nemo (0 : ethernet, 1 : modem)
- data en octet consommée depuis le début du mois,
- la version du codec utilisé,
- la date à laquelle a été testée la Nemo, la date en numéro de mois du dernier reset des données,
- la quantité moyenne de donnée par communication avec le modem ainsi que
- un paramètre auto-ajustant ou non l'intervalle entre deux communications.

Une ou plusieurs Nemo est reliée à un seul site et à un ou plusieurs installateurs (user spécifique).

Les relais Chaque relai (table `e_relay_mapping`) aura un type de condition d'activation (0 : ne s'active pas, 1 : activation sur authentification, 2 : activation lors de la mise en marche d'une borne), un numéro pour les identifier sur la carte électronique, sur la carte relai (`e_relay_board`). Cette carte est rattachée à une seule Nemo mais aussi à une borne (celle qu'elle va impacter).

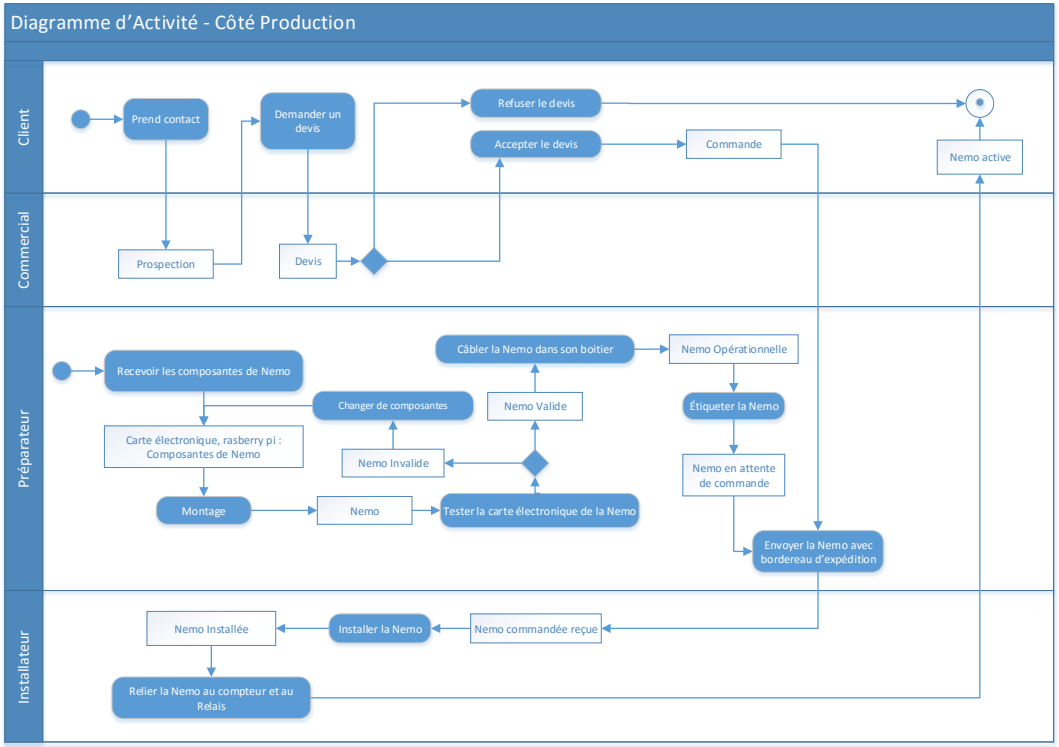
Cependant il est possible d'avoir plusieurs cartes par Nemo. En effet, en réalité, on a deux types de carte relai et de relai : internes et externes. Les relais externes sont mis en place lorsque les relais internes (une seule carte relai interne par Nemo) ne sont pas suffisants. Ces cartes, de manière générale, ont un type précis qui va impacter sur le nombre de relais sur une carte ou d'autres informations plus générales qui seront accessibles via `e_relay_board_type`. Cette carte relais externe va donc entrer en compte lorsque le *rc_id* de la table des relais n'est pas égal à 0. En effet, cet id de board implique que les relais sont internes.

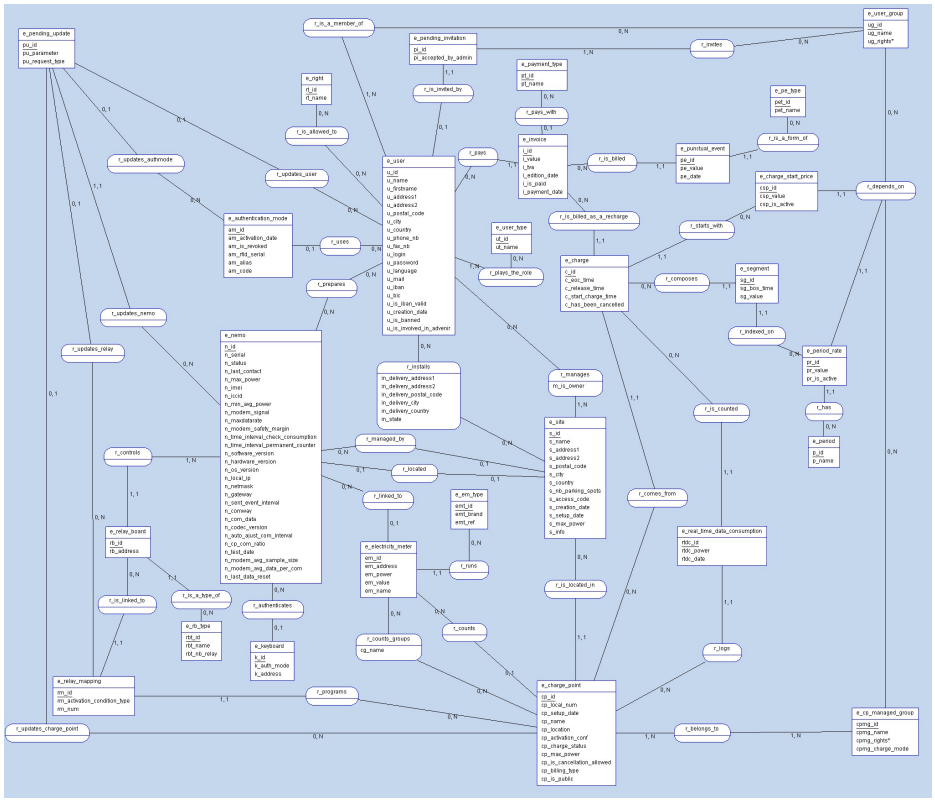
Les claviers On rappelle qu'une Nemo va prendre en charge des bornes. Pour accéder à ces bornes et leur donner des instructions, des claviers peuvent être mis à disposition. Généralement, il y aura plusieurs claviers par Nemo. Celle-ci sera chargée de recevoir les informations transmises via ce clavier. La table `e_keyboard` contiendra donc :

- un identifiant unique,
- un mode d'authentification (0 : désactivé, 1 : badge + choix de la borne par défaut, 2 : code + choix de la borne par défaut, 3 : uniquement le choix de la borne, 4 : badge ou code + choix de la borne),
- une adresse locale pour pouvoir le localiser,
- l'identifiant de la Nemo à laquelle un clavier est relié.

L'utilisateur La table `e_user` possède un identifiant unique ainsi que des informations personnelles :

- un nom, prénom, adresse complète (adresse, code postal, pays), numéro de téléphone, de fax, email,
- ses identifiants sur le site : langue, username, password, date de création du compte, validité du compte (est-il banni ?),
- un paramètre sur son engagement sur "advenir",
- ses informations bancaires : IBAN, BIC, validité de l'IBAN,
- l'identifiant de l'invitation en attente (si tel est le cas) d'un autre utilisateur l'ayant invité à rejoindre la plateforme Gestio.





C. Modèle Logique de Données

L'Annexe C est la représentation logique de la nouvelle version de la base de données de Gestio (cf. 2.1 - Activité de l'entreprise : de la production d'une Nemo page 4).

TABLE C.1. – *e_user*

Attribut	Contrainte	Type	Défaut	Clé
u_id	Unique - Not null	int(4)		Primary
u_name		varchar(50)		
u_first_name		varchar(50)		
u_address1		varchar(100)		
u_address2		varchar(100)		
u_postal_code		int(4)		
u_city		varchar(50)		
u_country		varchar(50)		
u_phone_nb		int(5)		
u_fax_nb		int(5)		
username	Unique	varchar(30)		
password		varchar(64)		
u_language		varchar(2)		
email	Unique - Not null	varchar(100)		
u_iban		varchar(34)		
u_bic		varchar(11)		
u_iban_is_valid	Not null	boolean	0	
created	Not null	datetime	now()	
u_is_banned	Not null	boolean	0	
u_is_involved_in_advenir	Not null	boolean	0	
emailVerified	Not null	boolean	0	
realm		varchar(512)		
credentials		text		
challenges		text		
verificationToken		varchar(512)		
status		varchar(15)		
lastUpdated		datetime		
u_u_invited_by		int(4)		Foreign → e_user

D. Maquettes

L'Annexe D est le document de maquettes envoyé pour validation à l'équipe commerciale.

On y trouve l'interface de l'utilisateur final sur plusieurs onglets.

Dashboard

http://gestio.parknplug.fr

park'n plug

Dashboard

5

bastien.jorge@parknplug.fr

- Dashboard
- Factures
- Infos bancaires
- Badges
- Support

Facture du 29/02/2016

29,34€

Mes factures

Recharge du 03/03/2016 - 8h15

12,733 kWh

Mes consommations

Depuis la dernière facture, j'ai consommé

2,81€

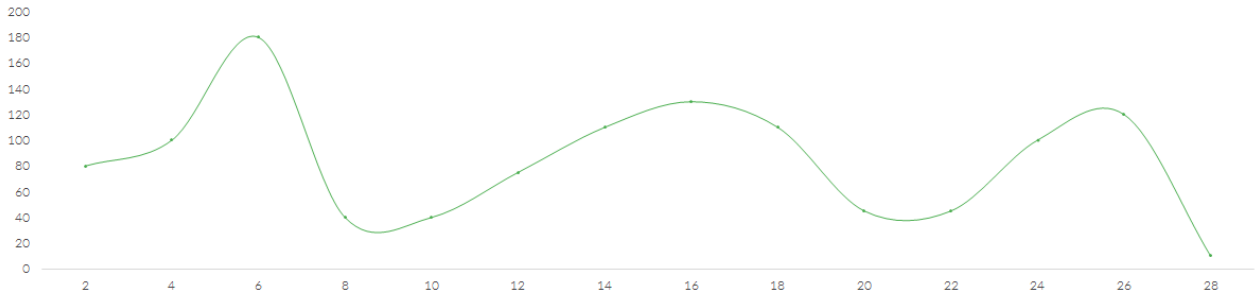
Mon encours

Mes news Park'n Plug 29/02/2016

L'innovation technologique au service de vos véhicules électriques.

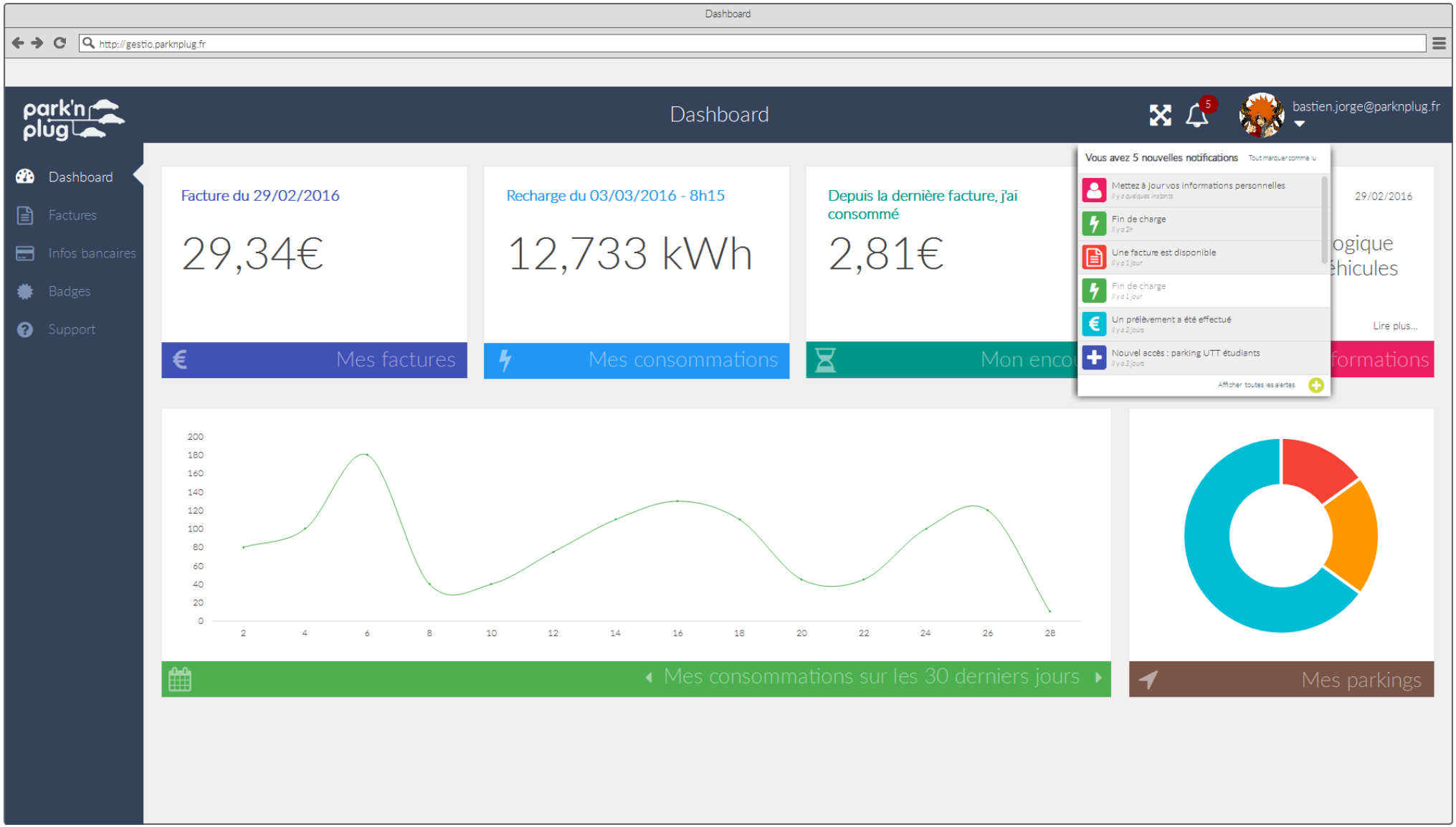
Lire plus...

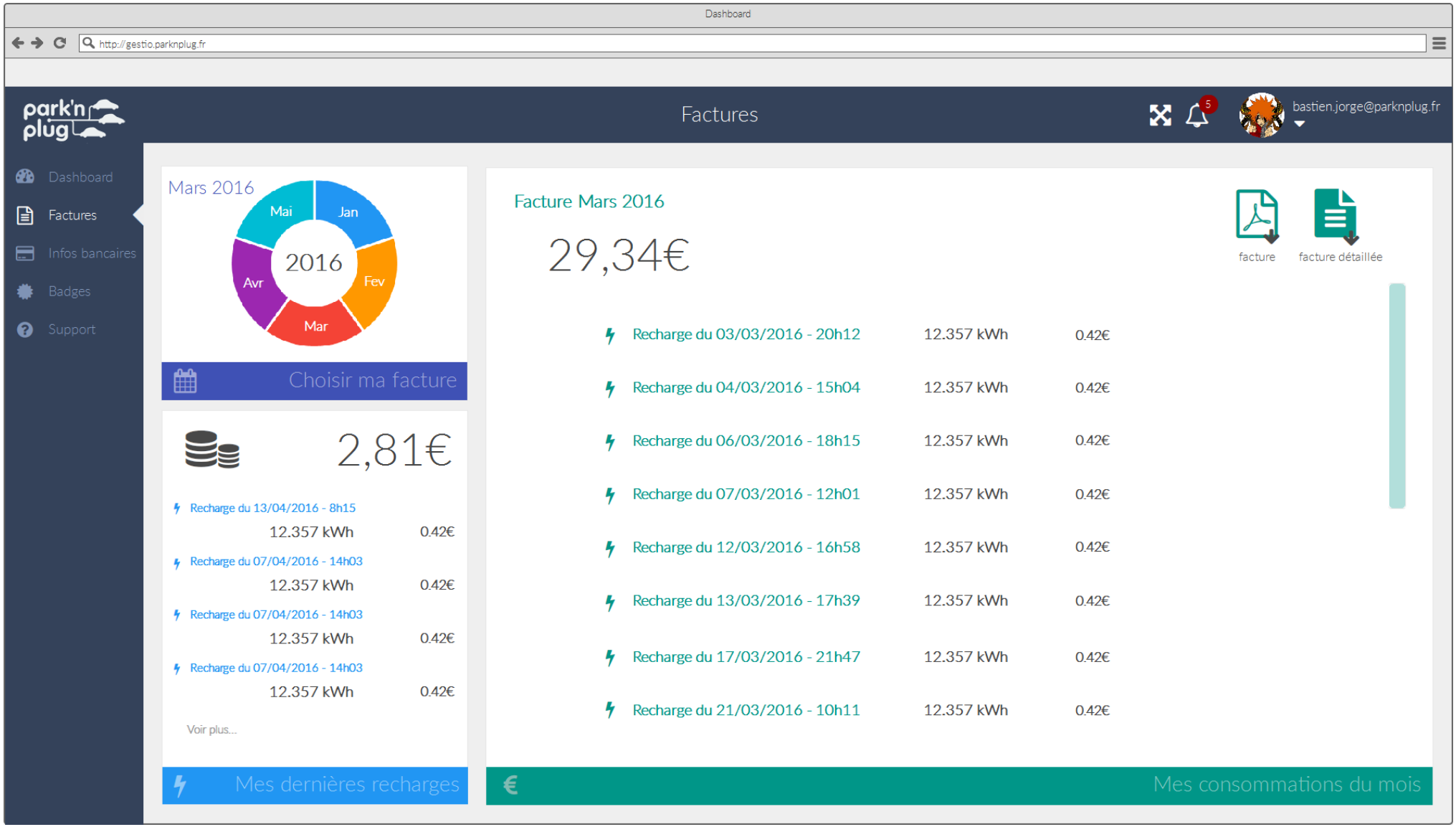
Dernières informations



Mes consommations sur les 30 derniers jours

Mes parkings


Dashboard
http://gestio.parknplug.fr
park'n plug
Dashboard
bastien.jorge@parknplug.fr
Dashboard
Factures
Infos bancaires
Badges
Support
Facture du 29/02/2016
29,34€
Mes factures
Recharge du 03/03/2016 - 8h15
12,733 kWh
Mes consommations
Depuis la dernière facture, j'ai consommé
2,81€
Vous avez 5 nouvelles notifications
Tout marquer comme lu
Mettez à jour vos informations personnelles
Il y a quelques instants
Fin de charge
Il y a 2h
Une facture est disponible
Il y a 1 jour
Fin de charge
Il y a 1 jour
Un prélèvement a été effectué
Il y a 2 jours
Nouvel accès : parking UTT étudiants
Il y a 2 jours
Afficher toutes les alertes
29/02/2016
Lire plus...
200
180
160
140
120
100
80
60
40
20
0
2
4
6
8
10
12
14
16
18
20
22
24
26
28
Mes consommations sur les 30 derniers jours
Mes parkings


Dashboard
http://gestio.parknplug.fr
park'n plug
Factures
5
bastien.jorge@parknplug.fr
Dashboard
Factures
Infos bancaires
Badges
Support
Mars 2016
Mai
Jan
2016
Avr
Fev
Mar
Choisir ma facture
2,81€
Recharge du 13/04/2016 - 8h15
12.357 kWh
0.42€
Recharge du 07/04/2016 - 14h03
12.357 kWh
0.42€
Recharge du 07/04/2016 - 14h03
12.357 kWh
0.42€
Recharge du 07/04/2016 - 14h03
12.357 kWh
0.42€
Voir plus...
Mes dernières recharges
Facture Mars 2016
29,34€
facture
facture détaillée
Recharge du 03/03/2016 - 20h12
12.357 kWh
0.42€
Recharge du 04/03/2016 - 15h04
12.357 kWh
0.42€
Recharge du 06/03/2016 - 18h15
12.357 kWh
0.42€
Recharge du 07/03/2016 - 12h01
12.357 kWh
0.42€
Recharge du 12/03/2016 - 16h58
12.357 kWh
0.42€
Recharge du 13/03/2016 - 17h39
12.357 kWh
0.42€
Recharge du 17/03/2016 - 21h47
12.357 kWh
0.42€
Recharge du 21/03/2016 - 10h11
12.357 kWh
0.42€
Mes consommations du mois

Dashboard

http://gestio.parknplug.fr

park'n plug

Mon compte

5

bastien.jorge@parknplug.fr

- Dashboard
- Factures
- Infos bancaires
- Badges
- Support

Informations personnelles

Login: bastienj

Prénom: Prénom

Nom: Nom

Adresse: Adresse 1

Adresse 2: Adresse 2

Code Postal: Code Postal

Ville: Ville

Pays: Pays

Informations de contact

Téléphone: 03.25.00.01.02

Fax: 03.25.00.01.02

Mail: bastien.jorge@parknplug.fr

Valider

Mon profil

Client depuis le

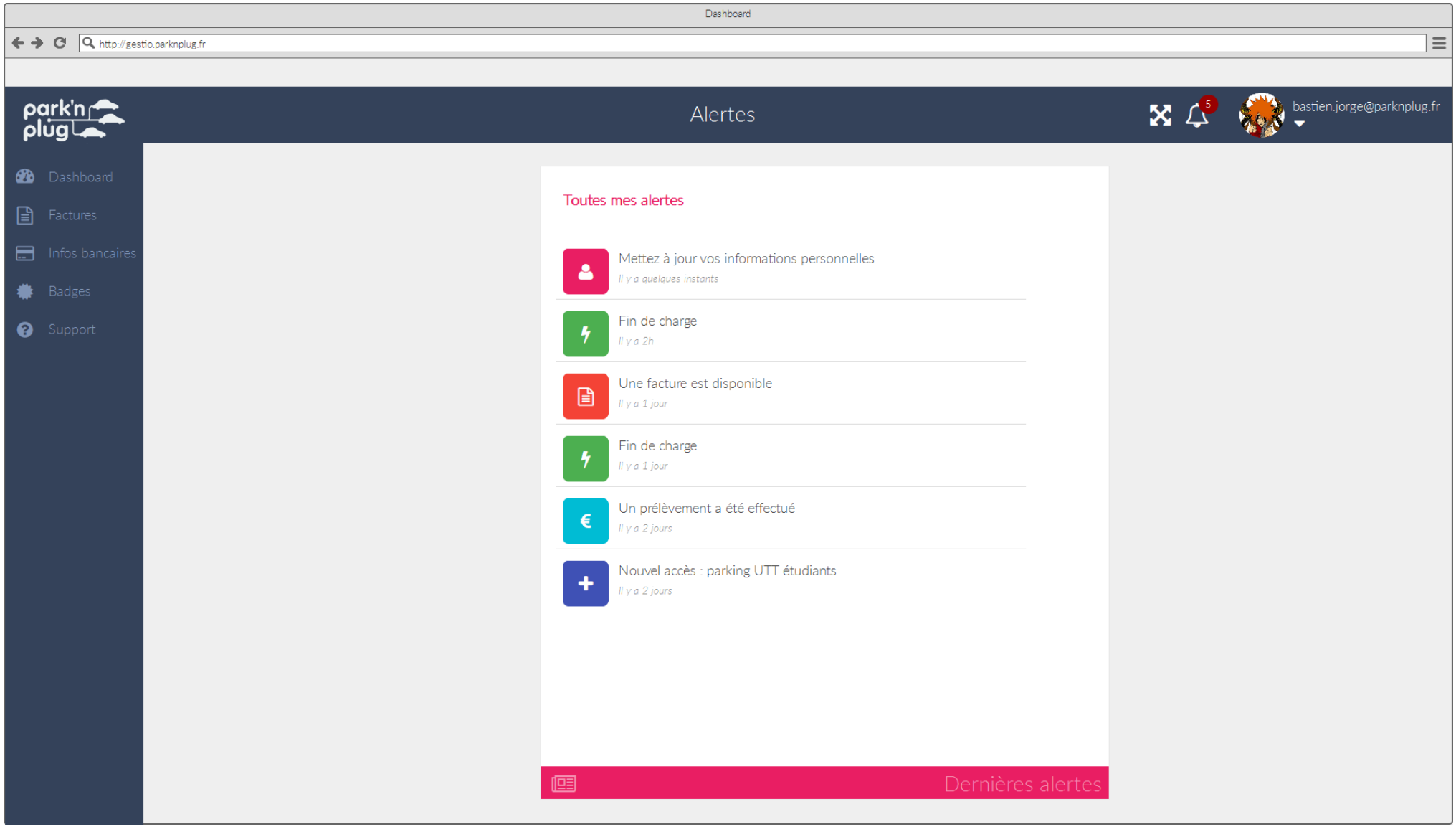
18/03/2016

Date d'inscription

Changer d'avatar

Upload

Image perso


Dashboard
http://gestio.parknplug.fr
park'n plug
Alertes
5
bastien.jorge@parknplug.fr
Dashboard
Factures
Infos bancaires
Badges
Support
Toutes mes alertes
Mettez à jour vos informations personnelles
Il y a quelques instants
Fin de charge
Il y a 2h
Une facture est disponible
Il y a 1 jour
Fin de charge
Il y a 1 jour
Un prélèvement a été effectué
Il y a 2 jours
Nouvel accès : parking UTT étudiants
Il y a 2 jours
Dernières alertes

E. Analyse des risques de 2015

L'Annexe E est la version simplifiée de l'analyse de risques menée en 2015 par Samy Jacquet. Elle a été allégée de l'aspect matériel et électronique, non pertinent dans le cadre de ma propre analyse.

4		Système de paiement	Redondance Charte informatique	Laboratoire Sensibilisation
3	Réceptions des données NEMO Service mail Développement		Système / Users Programmation Service web Propriété intellectuelle Concurrence Renouvellement des mdp Politique d'audit	Poste de travail Salariés
2	Journalisation	Stockage des données PRA/PCA	Lieu de travail	
1				
Risque Potentialité	1	2	3	4

Table des figures

Bibliographie

[1] Benjamin Monnereau. *UX, le guide : définition, enjeux, méthodes et 34 outils.* `http://www.benjamin-monnereau.com/guide-ux/`, Mis à jour le 25 mai 2016 [Consulté le 5 juin 2016]. [En ligne].

[2] Alexandre Fernandes-Toro. *Management de la sécurité de l'information.* Eyrolles, Clermont-Ferrand, 16 février 2012, 322p. ISBN 978-2212126976.

[3] Alexandre Fernandes-Toro. *Sécurité opérationnelle – Conseils pratiques pour sécuriser le SI.* Eyrolles, France, mars 2015, 352p. ISBN 978-2212139631.

[4] Julien Fontanet et Olivier Lambert. *Node.js – Exploitez la puissance de javascript côté serveur.* Éditions ENI, Saint-Herblain, 9 juillet 2014, 231p. ISBN 978-2746089785.

[5] Bernard Foray. *La fonction RSSI – Guide des pratiques et retours d'expérience.* Dunod, Vottem, janvier 2011, 328p. ISBN 978-2100556939.

[6] Aurélien Gay et Marc Glita. *Le système électrique européen : Enjeux et défis.* Presse des Mines, Paris, 9 novembre 2012, 162p. ISBN 978-2911256929.

[7] Bernard Lauxerrois. *Migration de Données. D'un Système d'Information à l'autre : la démarche complète.* Éditions ENI, Saint-Herblain, 9 novembre 2009, 350p. ISBN 978-2746051928.

[8] OWASP Foundation. *OWASP Top 10 – 2013. Les Dix Risques de Sécurité Applicatifs Web les Plus Critiques.* `http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013%20-%20French.pdf`, [Consulté le 23 mai 2016]. [En ligne].

[9] Stan Gibilisco, Matthew Haughn, Margaret Rouse. *Security management – Confidentiality, integrity, and availability (CIA triad).* `http://whatis.techtarget.com/definition/Confidentiality-integrity-and-availability-CI` Mis à jour en novembre 2014 [Consulté le 15 mai 2016]. [En ligne].

À Propos

Ce document a été réalisé en LaTeX sous Windows avec TeXStudio par Bastien Jorge et a été compilé grâce à MiKTeX.
Le style s'appuie sur le package open-source `koma-script`.