



DUFOUR Arthur
B2 EPSI 2018-2019

RAPPORT DE STAGE

Du 26 décembre
au 01 mars 2019
au sein de :



Remerciements

Avant tout développement sur cette expérience professionnelle, il m'apparaît opportun de commencer ce rapport d'activité par des remerciements, à ceux qui m'ont beaucoup appris au cours de ce stage, et même à ceux qui ont eu la gentillesse de faire de ce dernier un moment très profitable.

Ainsi, je remercie Christophe DUC, mon maître de stage, qui m'a formé et accompagné tout au long de cette expérience professionnelle avec une bonne dose de patience, de pédagogie et d'humour. Enfin, je remercie l'ensemble de l'équipe Road-b-Score pour les bons moments passé, et les conseils qu'ils ont pu me prodiguer au cours de ces deux mois que j'ai pu passer avec eux, et tout particulièrement Arnaud VINCENT, sans qui ce stage n'aurait pas été possible. Grâce à eux, je poursuis mes études avec encore plus d'enthousiasme.

Merci.

Table des matières

REMERCIEMENTS	3
TABLE DES MATIÈRES	4
INTRODUCTION	5
I – PRÉSENTATION	6
1.1 Présentation de l’entreprise	6
1.1.1 Raison sociale, secteurs d’activités, historique, effectifs	6
1.1.2 Produits / marques, cible, concurrents	8
1.2 Présentation du projet	9
1.2.1 Contexte professionnel du projet (base de départ, objectifs et enjeux)	10
1.2.2 Contexte technologique (défis, contraintes et limites)	10
II – DÉMARCHE SUIVIE	12
2.1 Organisation du projet	12
2.1.1 Formation et préparation... ..	12
2.1.2 Planning, réunions et comptes-rendus	13
2.2 Méthode agile (Scrum)	14
2.2.1 Présentation de la méthode Scrum	14
III – RÉALISATION DE LA MISSION	19
3.1 Développement de l’application	19
3.1.1 Développement de l’application : partie « front end »	19
3.1.2 Développement de l’application : partie « back end »	22
3.2 L’après développement (mise en place de tests)	24
CONCLUSION	26
GLOSSAIRE	27

Introduction

Dans le cadre de la validation de ma deuxième année de bachelor à l'EPPI (et en parallèle ; de mon BTS SIO SLAM), j'ai effectué mon second stage en entreprise, d'une durée de 9 semaines au sein de la société Road-b-Score, en tant que Développeur Full Stack.

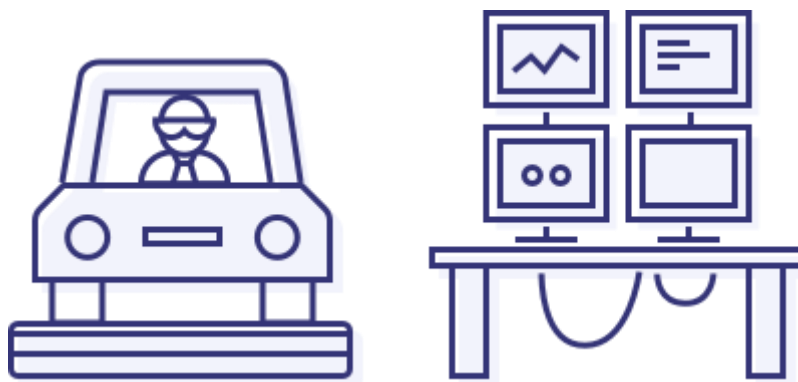
Le but de ce stage était pour moi de confirmer mon appétence pour les start-ups. En effet, par suite d'une première expérience très enrichissante l'année dernière, je souhaitais renouveler cette expérience, tout en changeant de secteur.

De plus, ce stage se présentait aussi comme un « défi technologique ». En effet, j'ai principalement travaillé avec des technologies que je ne maîtrisais que peu, voir même pas du tout... et aillant toujours été de nature curieuse, ainsi qu'aillant le désir de repousser mes limites, j'ai trouvé dans ce stage la magnifique opportunité de monter rapidement, et surtout professionnellement en compétences sur des technologies d'actualité.

Ainsi, lors de ma phase de recherche d'entreprise, je me suis tourné vers Road-b-Score ; petite startup d'une dizaine d'employés, car j'ai pensé pouvoir y trouver une bonne ambiance, la possibilité de monter en compétence, et la possibilité de travailler sur quelque-chose de concret, tout en étant très bien entouré.

J'ai donc travaillé en tant que Développeur Full Stack durant 2 mois, sur la refonte de l'un des produits de l'entreprise.

Ce rapport a pour but de présenter et d'analyser ce que j'ai pu réaliser et apprendre au cours de cette expérience, dont je garderais un excellent souvenir.



I – Présentation

1.1 – Présentation de l'entreprise (historique, produits, cible...)

1.1.1 – Raison sociale, secteurs d'activités, historique, effectifs.

Co-fondée en 2016 par Arnaud Vincent et Christophe Meheut, la startup lyonnaise Road-b-Score est une Société par Actions Simplifiées (SAS) incubée au B612 (Caisse d'Epargne) en plein cœur de Lyon.

Road-b-Score est une entreprise exerçant dans le secteur de l'assurance. Elle s'appuie sur les nouvelles technologies afin d'innover et d'introduire de nouveaux outils à tous les acteurs du marché : soit les assurés, les assureurs, les réassureurs, et autres intermédiaires d'assurance. En clair, c'est une « Insurtech » (aussi appelée « Assurtech » en français).

La société est actuellement incubée au B612, situé au 92 Cours Lafayette en plein cœur du 3^{ème} arrondissement de Lyon. En effet, l'entreprise est l'une des premières à s'être vue incubée dans ce nouvel établissement créé par la Caisse d'Epargne Rhône Alpes fin 2016.



C'est donc au sein de cet incubateur que l'équipe de Road-b-Score a la chance d'évoluer aux côtés d'une petite vingtaine d'autres startups des milieux de la finance et de l'assurance.



En effet, en plus d'apporter une structure et un cadre de travail à ses entreprises incubées, le B612 accompagne également chaque société de manière individuelle, que cela soit financièrement, ou bien en organisant divers événements permettant d'élargir réseaux et compétences. C'est dans ce cadre de travail moderne et où il règne une bonne ambiance que travaillent donc la petite dizaine d'employés composant la société Road-b-Score. Parmi eux on peut compter des chercheurs, commerciaux, chargés de marketing, et l'équipe de développement dans laquelle j'ai travaillé durant ces quelques semaines. On peut également noter que Road-b-Score a recours à l'emploi de prestataires de services, tels qu'un juriste et un comptable, qui ne nécessitent pas d'avoir un poste à temps plein.

Le cœur de métier de Road-b-Score est la prédiction de sinistralités des assurés automobiles à partir de leur profil comportemental. Pour se faire, Road-b-Score se charge de déterminer le comportement d'un conducteur, en utilisant deux sources de données distinctes ; à savoir les voitures connectées (ou équipées de boîtiers télématiques), ainsi que « l'activité » sur les réseaux sociaux d'un prospect.

Autrement dit, en analysant les données de conduite, et /ou les réseaux sociaux d'une personne, les algorithmes de Road-b-Score, boostés par l'intelligence artificielle, sont capables de définir les traits de personnalités d'une personne, et de les mettre en lien avec le comportement que cette dernière aura au volant d'une voiture.

L'entreprise se place toutefois dans une démarche d'apporter de la technologie aux courtiers et assureurs, laissant ainsi ces derniers se poser la question : « *quel est le bon le curseur ou la bonne limite entre la personnalisation et la mutualisation* ».

Ainsi, Road-b-Score repose en partie sur la possibilité de traiter les données, personnelles ou non, des utilisateurs. Ce qui est, entre autres, un pari assez osé étant donné les débats actuels aillant survenus suite aux scandales tels que celui du « Cambridge Analytica ».

Et bien en effet, c'est bien là le très gros enjeu de Road-b-Score ; le souci de la « privacy » ... L'entreprise doit à la fois savoir répondre aux questions de « *qu'est-ce que l'on fait des données personnelles ?* » ainsi que « *qu'est-ce qui est acceptable ou qu'est-ce qui ne l'est pas ?* ».

La protection des données personnelles est un point très important aux yeux de Road-b-Score, qui met un point d'honneur sur le fait d'être, et de rester totalement « RGPD compliant ».

Pour cela, l'entreprise présente une approche « Socratique » de la protection des données personnelles, où l'utilisateur est à la fois l'initiateur, le messager et le bénéficiaire exclusif de l'utilisation de ses données personnelles.

Ainsi, aucune donnée n'est « collectée », car ce sont les utilisateurs qui donnent explicitement l'accord de les transmettre à l'entreprise. De plus, une fois l'entreprise en possession de ces données, ces dernières sont totalement anonymisées, rendant ainsi totalement impossible toute réidentification des utilisateurs...

(Voici un extrait de la Directive 2016/680, paragraphe 21, tiré du site de la CNIL :

« [...] Il n'y a dès lors pas lieu d'appliquer les principes relatifs à la protection des données aux informations anonymes, à savoir les informations ne concernant pas une personne physique identifiée ou identifiable, ni aux données à caractère personnel rendues anonymes de telle manière que la personne concernée ne soit pas ou plus identifiable. ».

On notera également que ces données ne sont en aucun cas stockées, et disparaissent des serveurs de l'entreprise en moins de 30 jours...

1.1.2 – Produits / marques, cible, concurrents.

Comme évoqué précédemment, les clients de Road-b-Score sont tous des acteurs du milieu de l'assurance (bien que certains soient également présent dans le domaine de la banque). En effet, grâce à ces nouvelles technologies, les assureurs sont en mesure de mieux évaluer les risques d'accidents de leurs clients, et ainsi d'adapter leur prix (à la baisse ou à la hausse), en fonction des clients.

De la même manière, les clients peuvent eux-mêmes décider de leur plein gré de faire analyser son profil, et s'il obtient un bon score (par bon score on entend : « un profil comportemental indiquant qu'il ait moins de chance d'avoir un sinistre au volant »), il peut se diriger vers son assureur afin de lui demander une remise. C'est donc un modèle « retourné » qui a été mis en place par l'entreprise, dans l'idée d'également avoir pour clients des particuliers.

Road-b-Score ne possède actuellement aucun concurrent direct. En effet, elle est actuellement la seule entreprise dans le monde connue pour vendre ces produits, ce qui lui offre un champ de possibilité presque infini vu l'importance du marché de l'assurance dans le monde. Cependant, de plus en plus de sociétés s'équipent de boîtier télématiques, et collectent de nombreuses données de conduite, sans ne pouvoir rien y faire. Cela montre donc bien l'intérêt des solutions de Road-b-Score, tout en lui laissant la possibilité de vendre ses solutions, mais cela signifie également qu'un concurrent pourrait très probablement voire un jour le jour.

Road-b-Score a plus récemment lancé une nouvelle marque, du nom de 357coupons, également basée sur le concept de pouvoir établir un profil comportemental depuis le réseau social d'un individu, mais cela en sortant du contexte automobile.

Le principe initial de la première version de la plateforme 357coupons (qui se présente sous la forme d'un site web) était le suivant :

On retrouvait deux types d'utilisateurs, des entreprises que l'on nommera « commercialisateurs », et des particuliers que l'on nommera « indicateurs ».

Un indicateur pouvait devenir partenaire d'un commercialisateur, dans le but de promouvoir les produits de ce dernier à d'autres particuliers, et ainsi leur faire bénéficier de réductions ou autres avantages, moyennant un dédommagement (comme à peu près tout partenariat...).

Cependant, là où les algorithmes de Road-b-Score rentrait en jeu, c'est que pour que l'indicateur puisse faire profiter de « codes » (ou « coupons ») de réduction les autres utilisateurs, il fallait que ces derniers soient éligibles aux offres des commercialisateurs, selon certains critères définis par les entreprises. Et ces critères sont donc d'ordre comportementaux...

Prenons l'exemple d'une société proposant de l'aide au placement d'argent en tant que commercialisateur, et de moi-même en tant qu'indicateur. Je signe un partenariat via la

plateforme 357coupons, indiquant que je gagnerais 20€ si en faisant la promotion de leurs offres, j'arrive à faire inscrire disons 5 personnes sur le site de l'entreprise en utilisant un coupon.

Cependant, afin d'obtenir ce fameux coupon que je pourrais transmettre à un particulier, je suis obligé de faire analyser le profil Facebook de cette personne (en aillant son autorisation), pour voir si son profil correspond aux critères imposés par l'entreprise, et ainsi potentiellement obtenir un coupon de réduction.

Comme vous pouvez vous en rendre compte, le concept n'est pas des plus simples à comprendre, ni même à expliquer... ce qui est sûrement une des causes du fait que la marque 357coupons n'a encore jamais « bien » fonctionné. Cependant, cette jeune plateforme n'est en réalité que le fruit de travaux de recherche et de développement (R&D).

J'ai donc été recruté chez Road-b-Score afin de participer dans le développement de la « v2 » de cette plateforme 357coupons, dont le principe a été retravaillé en espérant mieux fonctionner.

1.2 –Présentation du projet (contexte technologique & professionnel)

Comme évoqué ci-dessus, mon stage était centré autour d'une mission ; le développement de la deuxième version de la plateforme 357coupons (développement aussi bien frontend que backend).

Lors de mon arrivée fin décembre, l'idée de lancer la 2^{ème} version de la plateforme était déjà bien présente, cependant tout n'était pas encore défini : il restait des points à éclaircir tant sur le principe de fonctionnement de la nouvelle version, que sur les petits aspects techniques qu'il fallait ajouter sur telle et telle pages...

Ainsi, j'ai tout d'abord commencé par prendre le temps de me former sur l'existant : la première version de la plateforme.

Cette dernière était séparée en plusieurs « modules », dans un esprit un peu « micro services ». Ainsi, on comptait une base de données, trois « fronts » (partie du site web que l'utilisateur va utiliser) et deux APIs, servant à transmettre les données entre les fronts et la base de données. Cela donne le schéma ci-contre.

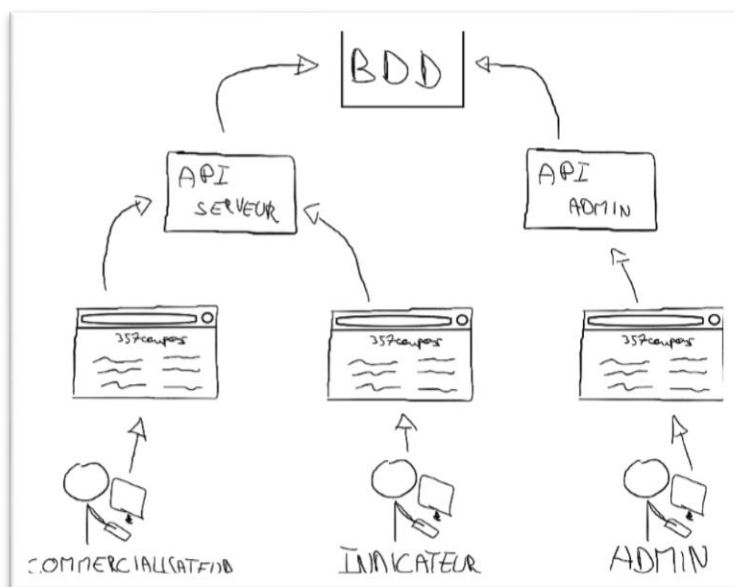


Schéma simplifié de l'architecture de la plateforme 357coupons

L'intérêt de diviser l'application de cette manière est multiple :

- Possibilité de déployer des modifications sur un module sans affecter les autres.
- Plus de facilité pour séparer et repartir le travail.
- Plus de scalabilité (capacité à s'adapter à un changement d'ordre de grandeur de la demande, comme le besoin de plus de performances).

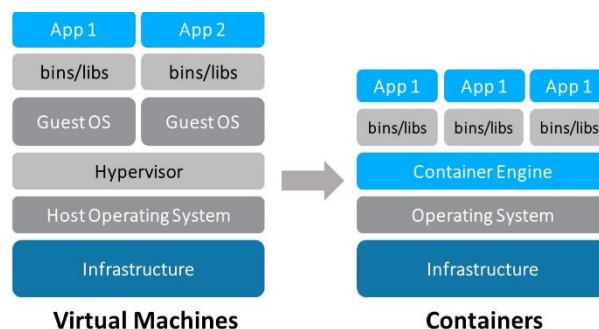
En revanche, cela présente également une forte contrainte : **le temps**. En effet, la multiplication des modules, induit un temps de développement plus long, et donc plus cher. Ce modèle, bien qu'actuellement très à la mode, n'est pas forcément des plus adaptés pour une startup, qui a généralement besoin de développements très flexibles et réalisés dans de brefs délais... C'est pourquoi l'architecture qui a été adoptée pour la plateforme de 357coupons n'est en réalité qu'un semblant de micro-services, revisité à la sauce de l'entreprise afin d'au mieux lui correspondre.

Les technologies utilisées dans les modules décrit sur le schéma, et donc avec lesquelles j'ai pu travailler sont les suivantes :

- Vue.js pour les 3 fronts (Framework Web Javascript open-source).
- Node.js pour les 2 apis (Javascript server runtime environment).
- MongoDB pour la base de données (SGBD NoSQL document-based).



Afin de faciliter la gestion de ces applications, et ainsi d'éviter d'avoir à installer chaque application manuellement sur chaque poste de travail, avec la configuration adéquate etc, les applications tournaient dans des conteneurs Docker. Docker est un logiciel révolutionnaire qui exploite la capacité dite de « conteneurisation » des systèmes d'exploitation. De manière très simplifiée, un conteneur est une sorte de machine virtuelle qui partage le même noyau de système d'exploitation et qui isole les processus de l'application du reste du système, mais plus performante qu'une véritable machine virtuelle (car ils s'affranchissent du système



d'exploitation en utilisant directement celui de l'hôte). Et un conteneur, ça ne prend que quelques secondes à installer !

Grâce à Docker et à sa surcouche docker-compose, il suffisait d'une ligne de commande pour installer chacun des 7 modules présentés dans le schéma, et ce avec les configurations adéquates déjà intégrées etc. C'est un outil moderne, de plus en plus utilisé en entreprise, que j'affectionne tout particulièrement.

Pour résumer, les technologies utilisées pour le développement de la plateforme 357coupons sont toutes des technologies ultra-actuelles, qui sont particulièrement adaptées aux startups du fait de la flexibilité qu'elles apportent, et la productivité qu'elles apportent au développeur : travailler avec ces technologies, c'est une véritable partie de plaisir !

II – Démarche suivie

2.1 – Organisation du projet

2.1.1 – Formation et préparation...

Lors de mon arrivée chez Road-b-Score, l'une des premières choses à laquelle j'ai été confronté (bien qu'aussi évident qu'elle soit), était l'appropriation des outils et de l'environnement applicatif qui composaient la plateforme 357coupons.

En effet, bien que j'eusse déjà connaissance des technologies employées ainsi que de leurs avantages et inconvénients (cf. partie 1.2), je ne les maîtrisais que peu voire pas du tout (je fais ici principalement référence à Vue.js & Docker). J'ai donc eu quelques jours pour me former, et ainsi devenir le plus productif possible.

Pour cela j'ai principalement parcouru les documentations respectives de chaque techno, tout en essayant de faire le parallèle avec le code et les fichiers de configuration de l'application déjà existante. J'ai également recréé quelques mini-applications afin de pratiquer. Toutefois, lorsque je ne comprenais pas un point avec la documentation, Christophe, mon maître de stage était toujours là pour m'expliquer, tout en me donnant des exemples assez clairs.

(Une des choses que j'ai particulièrement aimé durant ce stage, était justement le fait que Christophe a toujours su donner de vraies réponses à mes questions ; c'est-à-dire, en y répondant, en donnant un exemple, et en expliquant l'intérêt (« pourquoi est-ce que l'on fait comme ça et non autrement »), ce qui m'a permis d'acquérir des bases solides, ainsi qu'une plus grande compréhension des technologies utilisées, et ce en développant également mon esprit critique.)

Une fois assez à l'aise avec les différentes technos, j'ai dû m'approprier le code déjà existant, et la logique dans laquelle il avait été conçu. Fort heureusement, ce dernier était plus ou moins commenté, et possédait surtout de la modélisation UML. Ainsi, grâce à un diagramme de classe assez complet, il m'était déjà bien plus facile de comprendre l'architecture de l'application que si j'avais dû directement déchiffrer les dizaines de milliers de lignes de code composant cette dernière.

On notera tout de même que ce diagramme de classe n'était pas totalement à jour, et que l'une de mes premières actions fût de le remettre à jour pour qu'il corresponde à l'existant de la « v1 » (ce qui m'a permis de comprendre en profondeur l'application), avant de le mettre à jour pour qu'il corresponde à la « v2 » que j'allais développer.

En ce qui concernait la préparation de mon poste de travail, ce fût assez rapide. J'étais libre d'utiliser mon propre poste de travail, tant que je travaillais sous Linux, ce qui était déjà le cas.



Caractéristiques de mon poste de travail basé sur Manjaro

Je possédais également la liberté d'utiliser mes propres outils de développement, ce qui présentait pour moi l'avantage d'être plus productif (pas besoin d'installer, ni de configurer, mais surtout d'apprendre à utiliser, de nouvel outil).

Cependant, il ne faut pas négliger le fait de devoir se débrouiller lorsque quelque chose ne fonctionne pas sur son poste (*par exemple, les procédures pour activer le débogage d'une application Node sous Visual Studio Code sont différentes de celle sous IntelliJ IDEA*), ainsi, seule la maîtrise de mes outils pouvait et devait empêcher d'empiéter sur ma productivité.

J'ai donc exclusivement utilisé mon propre laptop tournant sous Linux, ainsi que Visual Studio Code comme principal éditeur de texte / IDE.



2.1.2 – Planning, réunions et comptes-rendus

Une seule deadline m'a été fixée durant mon stage : finir les développements initiaux de la plateforme fin janvier (par plateforme j'entends ici la deuxième version de 357coupons). En effet, cette dernière devait pouvoir être présentée à des clients pour des fins commerciales.

Lors de mon arrivée, il m'a donc suffi de moins d'une semaine pour me former, plus la semaine remplies des réunions de co-écriture évoquées auparavant, j'étais opérationnel dès début Janvier.

Ces réunions consistaient à définir un équivalent de cahier des charges, ainsi qu'une maquette, que j'allais devoir suivre lors de la phase de développement, tout en priorisant les

tâches. On parle souvent de « Lot 1 » pour désigner la première version de l'application à livrer, puis de « Lot 2 » et cætera.

Une fois le cahier des charges écrit, j'ai entamé la phase de développement. Cette phase s'est décomposée en deux grandes parties : le développement applicatif, et le développement de tests.

La première consistait à effectuer tous les développements directement liés à l'application en elle-même (celle que les utilisateurs utilisent), c'est également la partie que je devais terminer avant la fin du mois de Janvier. La seconde partie quant à elle, consistait à développer des tests unitaires ainsi que des tests d'intégration, puis de les intégrer dans une chaîne d'intégration continue. Ce processus permet de vérifier à chaque modification du code, que le résultat de ces dernières ne produit pas de régression dans l'application, augmentent ainsi sa fiabilité, tout en réduisant le temps à allouer aux tests manuels.

Pour en venir aux comptes-rendus, je faisais régulièrement valider mes développements à Christophe (mon maître de stage), sûrement afin de me rassurer de ne pas faire n'importe quoi, mais également pour être sûr de ne pas perdre trop de temps si le développement ne convenait pas.

J'ai également eu une réunion de revue de code à mi-parcours, qui m'a plus permis de consolider des points théoriques que pratiques, mais qui n'en n'était pas moins importants ! *(Je peux citer comme exemple technique, le fait que je ne dupliquais parfois pas mes contrôles de champs entre parties front end et back end, ce qui pouvait causer des comportements non désirés et facilement évitables).* Avec un peu de recul, je me rends compte que ce sont ce genre de micro-informations qui parfois manquent lors de formations purement théoriques, et je suis bien content d'avoir la chance de pouvoir me former dans ce contexte.

Une fois le développement terminé, j'ai fait une présentation orale de tous les aspects de la plateforme à l'ensemble de l'équipe, afin que chacun puisse s'imprégner de la nouvelle forme du produit, et ainsi potentiellement proposer des améliorations ou autres...

J'ai également pu réaliser une vidéo à mi-chemin entre présentation et tutoriel visant à être utilisée par les clients.

2.2 – Méthode agile (Scrum)

2.2.1 – Présentation de la Méthode Scrum

Dans le cadre de la découverte de l'agilité via l'application de la méthode Scrum chez Road-b-Score, je souhaitais réserver une partie de ce rapport d'activité à sa présentation, car cela fait partie des points que j'ai réellement apprécié chez Road-b-Score.

La « Méthode Scrum » (« Scrum » signifiant « Mêlée » en français) une méthode de gestion de projet, qui a pour objectif d'améliorer la productivité de son équipe. Elle est de loin la méthode agile la plus utilisée dans le monde.

Cependant, parler d'une « méthode » concernant Scrum n'est pas ce qu'il y a de plus approprié. Scrum ne se considère pas comme une méthode, mais plutôt comme un cadre méthodologique. En effet, une méthode dit généralement « comment » faire les choses, cependant Scrum ne dit pas « comment réussir son logiciel », ni « comment surmonter les obstacles », mais se contente d'offrir un cadre de gestion de projet Agile, avec des rôles, un rythme itératif, des réunions précises et limitées dans le temps, des artefacts (product backlog, sprint backlog, graphique d'avancement) et des règles du jeu.

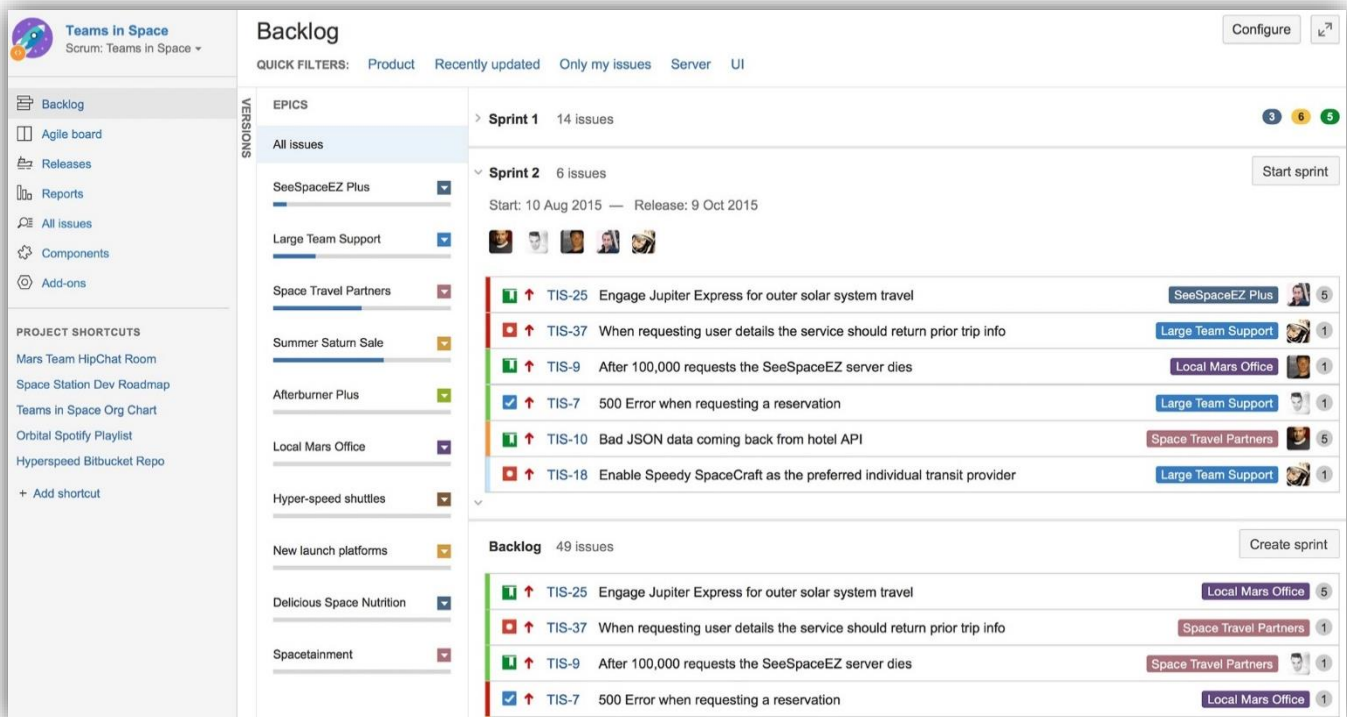
On retrouve en effet une bonne dizaine de termes assez spécifiques, qu'il faut connaître afin de pratiquer.

Parmi ces termes, on compte 3 rôles distincts dans une équipe Scrum :

- Le **Product Owner**, qui est la personne qui porte la vision du produit à réaliser, et qui travaille en interaction avec l'équipe de développement. Il s'agit généralement d'un expert du domaine métier du projet.
- L'**Equipe de développement**, qui est chargée de transformer les besoins exprimés par le Product Owner (aussi appelé « PO ») en fonction utilisables. Elle est pluridisciplinaire et peut donc encapsuler d'autres rôles tels que des développeurs, des architectes logiciels, des analystes fonctionnels, des graphistes ou bien des ingénieurs système...
- Le **Scrum Master**, qui doit maîtriser Scrum et s'assurer que ce dernier est correctement appliqué. Il a donc un rôle de coach à la fois auprès du PO et auprès de l'équipe de développement. Il doit donc faire preuve de pédagogie. Il est également chargé de s'assurer que l'équipe de développement est pleinement productive. Généralement le candidat tout trouvé au rôle de Scrum Master est le chef de projet. Celui-ci devra cependant renoncer au style de management « commander et contrôler », pour adopter un mode de management participatif.

(On notera également qu'une équipe est généralement composée d'une petite dizaine de personnes, et qu'elle doit obligatoirement comprendre un Scrum Master ainsi qu'un Product Owner.)

Le Product Owner a pour rôle de définir le référentiel des exigences initiales avec le client, et constitue ainsi ce que l'on nomme le **product backlog**. Ce backlog va contenir les fonctionnalités attendues dès le début du projet, bien qu'il puisse tout de même évoluer durant, en parallèle des besoins du client (agilité). Il se présente très souvent sous forme de liste de tâches (cf. la capture d'écran suivante).



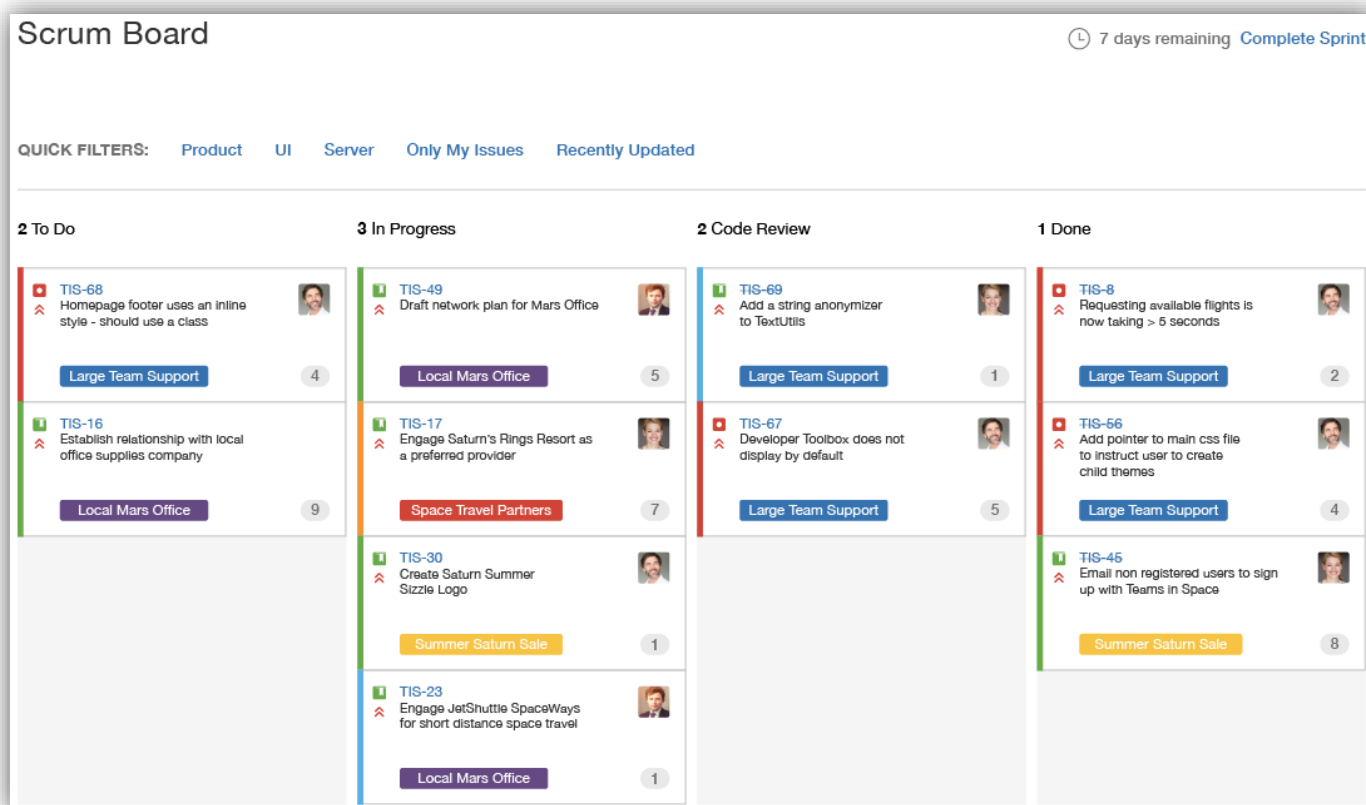
Exemple de backlog sur la plateforme « Jira » (système de gestion de projets développé par Atlassian)

Les fonctionnalités se trouvant dans le product backlog portent le nom d'**user stories**. Elles sont décrites en employant la terminologie du client, et contiennent généralement les informations suivantes :

- Un **nom** : le plus court possible, décrivant la fonctionnalité attendue par le client. Il doit cependant être assez clair pour que les membres de l'équipe et le PO comprennent de quelle fonction il s'agit.
- Niveau d'**importance** : un entier qui fixe la priorité de la story (peut-être amené à changer en cours de réalisation).
- Une **estimation** : la quantité de travail nécessaire pour développer/tester/valider la fonctionnalité. L'unité de mesure peut-être un nombre de jours idéaux, ou bien un nombre de points. Les estimations se font en relatif en comparant les estimations des stories terminées avec la story à estimer.
- Des **notes** : avec tout autre informations (clarifications, documentation...).

Le dernier terme à connaître, et le plus important de tous : le **sprint**. Un sprint est une itération du cycle de vie de Scrum. En effet, au cours de cette période d'une durée de 2 à 4 semaines se répétant, l'équipe se concentre sur l'accomplissement des tâches du sprint backlog. Dans le cadre d'un sprint, les développements s'effectuent de manière verticale : c'est-à-dire que le but est de développer une fonctionnalité de bout en bout (de la conception jusqu'au tests), et ainsi d'éviter un mini cycle en V au sein du sprint, ou bien qu'un développeur se retrouve avec plusieurs tâches en cours simultanément sans pouvoir les achever avant la fin du sprint.

Pour cela, l'utilisation d'un tableau de tâches, qu'il soit physique (avec des post-it), ou bien logiciel (voir capture d'écran ci-dessous), est indispensable. En effet, il permet d'avoir une vision claire du travail à accomplir, en cours, et terminé.



Exemple de tableau de tâches sur la plateforme Jira

Ainsi, une fois l'équipe constituée, le déroulement d'un sprint est le suivant :

- Réunion de **sprint planning** : on sélectionne dans le product backlog les exigences les plus prioritaires pour le client. Elles seront développées, testées, et livrées au client à la fin du sprint, et elles constituent le sprint backlog, qui est donc un sous ensemble du product backlog.
- **Début du sprint** : les développeurs « tirent » les tâches depuis la liste des tâches à faire. Durant toute la période du sprint a également lieu chaque matin une réunion appelée le Scrum (ou encore « mêlée », « daily meeting » ...).

Cette mini-réunion d'une durée de 15 minutes sert autant à but informatif qu'à stimuler l'esprit de travail en équipe, et le niveau d'engagement de chaque membre de l'équipe dans le projet. Durant ces 15 minutes, chaque membre de l'équipe doit prendre la parole et présenter principalement les éléments suivants :

- Ce qu'il a fait hier et les éventuels problèmes rencontrés
- Ce qu'il va faire aujourd'hui
- Est-ce qu'il a des difficultés pour continuer son travail

En faisant cet exercice quotidiennement, chaque membre de l'équipe est donc au courant de ce que font ses collègues, et il peut coordonner son travail ou se faire aider en cas de difficultés.

Attention toutefois, le Scrum Meeting n'est pas une réunion pendant laquelle on cherche à résoudre les problèmes, mais uniquement à les identifier et les exprimer. C'est au Scrum Master que revient le rôle d'apporter des solutions ou de déléguer à un autre membre de l'équipe la résolution des problèmes alors soulevés.

- Le **Sprint Review Meeting** : également appelé « retrospective », il s'agit d'une réunion où l'on fait le bilan du sprint, ainsi qu'une démonstration au client. C'est l'occasion de faire le point sur le fonctionnement de l'équipe, et de trouver des points d'amélioration.

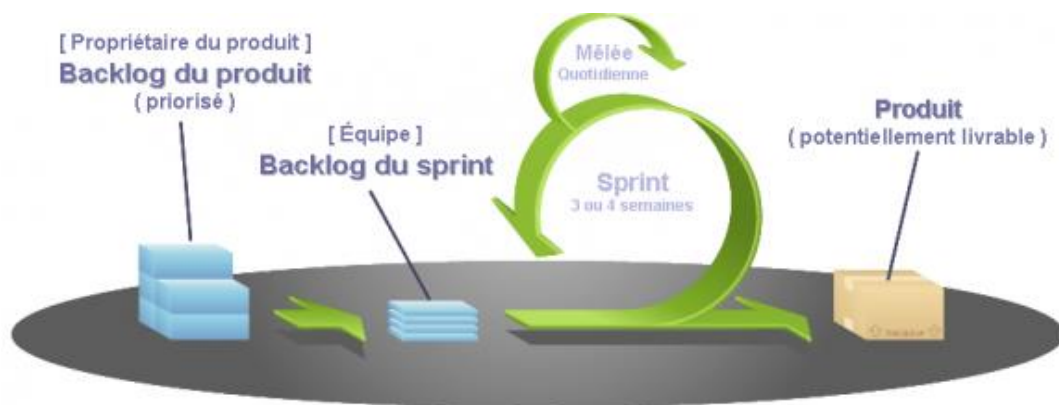


Schéma synthétisant le déroulement d'un sprint

Pour conclure, la méthode Scrum propose donc une approche très itérative de la gestion de projet. Le succès de cette méthode repose sur le strict respect des rôles de chacun, ainsi que sur des cycles de travail courts, à la fois rigoureux et flexibles. Le respect de ces règles octroie dans le même temps une grande autonomie et liberté à l'ensemble de l'équipe. Au regard de la complexité croissante dont les projets innovants font preuve, la méthode Scrum paraît être la meilleure solution pour répondre aux exigences d'exécution de ces derniers, ce qui explique le succès qu'elle rencontre aujourd'hui.

III – Réalisation de la mission

3.1 – Développement de l'application

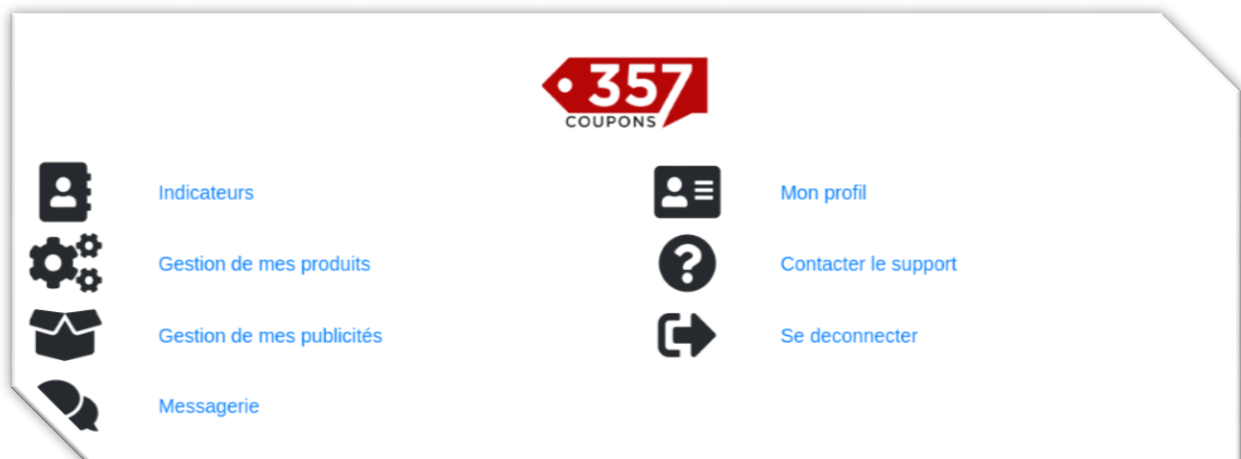
3.1.1 – Développement de l'application : partie front end.

Comme expliqué dans la partie 1.2, le front end de la plateforme 357coupons est découpé en 3 modules distincts : le « front commercialisateur », le « front indicateur », et le « front administrateur ». Chacun de ces fronts est donc une application Vue.js.

(Vue.js qui est rappelons le, un Framework Javascript open-source utilisé pour construire des interfaces utilisateurs, qui fait partie de cet esprit « web moderne » et qui est aujourd'hui très à la mode).

Grâce à Docker, l'entièreté de la plateforme était répliquée sur mon poste de travail, ce qui m'a donc permis de développer localement.

Le développement front end était globalement très simple à réaliser, car les interfaces à réaliser étaient des plus simples. En effet, au vu de la clientèle ciblée (professionnels), et du fait que la plateforme n'en n'était encore qu'au statut d'essai, il était autant économiquement que techniquement préférable de ne pas mettre en place un design trop lourd.



Menu commercialisateur de la plateforme 357coupons v2

J'ai donc dû redévelopper la plus grande partie de ces pages, en m'occupant en même temps de la partie front end que je vais ici illustrer en prenant l'exemple de la page « Indicateurs », et de la tout l'aspect back end qui sera présenté dans la partie suivante.

Voici donc ce à quoi ressemble la page « Indicateurs » une fois terminée.

The screenshot shows a web application interface for '357 COUPONS'. The page is titled 'Mes indicateurs - Accueil'. It features two main sections: 'Indicateurs sollicités' and 'Indicateurs actifs'.

Indicateurs sollicités: This section contains a search bar with a magnifying glass icon, a 'Rechercher' button, and a dropdown menu labeled 'Nom'. Below the search bar is a table with the following columns: Nom, Date de sollicitation, Références URL, Lien d'inscription, Clic sur le lien, Contact, Hashtags, and Supprimer l'invitation. The table lists two entries: Jeanne Delatable and Jean Redier, each with their respective dates, URLs, and contact information. A red 'Supprimer' button is visible next to each entry.

Indicateurs actifs: This section also has a search bar with a magnifying glass icon, a 'Rechercher' button, and a dropdown menu labeled 'Nom'. Below the search bar is a table with the following columns: Nom, Bio, Formation, Pays, and Hashtags. The table lists two entries: Pierre DURAND and Jacques DELAPELLE, each with their respective bios, formations, and countries. A red 'Supprimer' button is visible next to each entry.

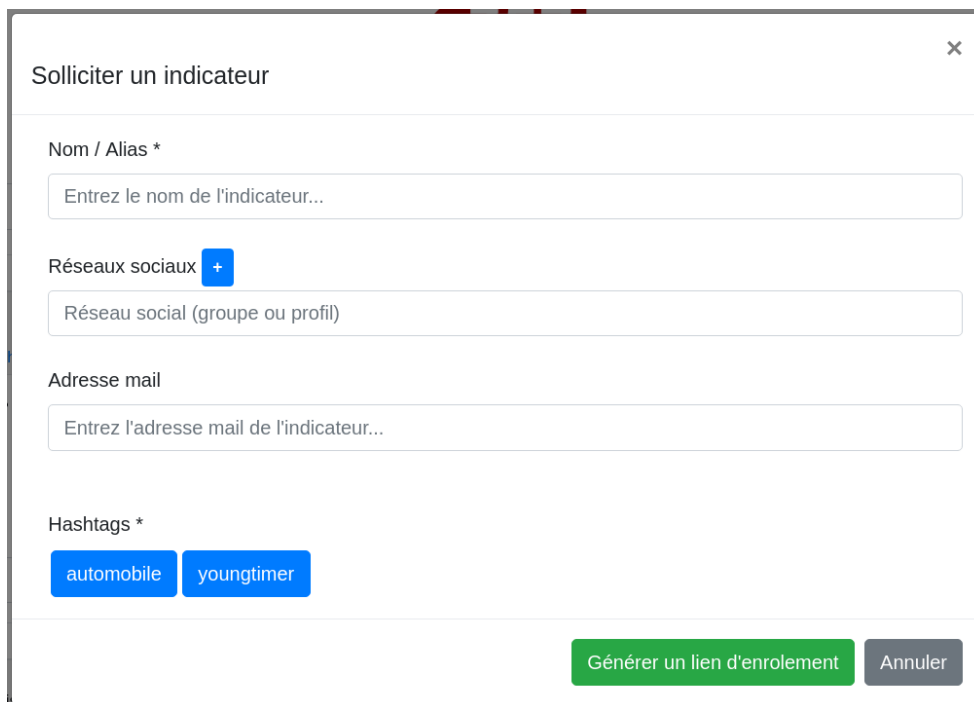
Sans trop rentrer dans les détails, le but de cette page qui se situe sur le front commercialisateur, est de permettre à ces derniers d'afficher, dans un premier temps les indicateurs qu'ils ont sollicités (par sollicitation on entend invitation à s'inscrire sur la plateforme), et dans un second temps, la liste des indicateurs actifs (inscrits) sur la plateforme, et avec qui il partage au moins un « hashtag » en commun.

On compte un total de 8 fonctionnalités sur cette page, dont :

- **L'affichage des indicateurs sollicités et des indicateurs actifs :** ce dernier doit être déclenché lors du chargement de la page, et s'effectue en effectuant une requête vers l'API, qui renvoie une liste d'indicateurs, qu'il suffit alors d'afficher sous forme de tableau.
- **La possibilité de supprimer une invitation en cours :** en effet, dans le cas d'un indicateur sollicité, le commercialisateur l'ayant sollicité peut révoquer l'invitation (rendant ainsi le lien impossible à utiliser).

```
mounted () {
  this.getMyInfluencers()
  this.getHashtags()
},
methods: {
  // get all pending influencers sought by the insurer
  // get all active influencers who are matching with the insurer
  getMyInfluencers () {
    this.loader = true
    this.$http.get(this.serverUrl + '/insurer/secure/myinfluencers')
      .then((response) => {
        this.myInfluencersInvite = response.body.myInfluencersInvite
        this.myInfluencersInvite.sort(this.sortDates)
        this.activeInfluencers = response.body.activeInfluencers
        this.loader = false
      }, (response) => {
        this.infoError = response.body.error
        this.loader = false
      })
  },
}
```

- **La possibilité de solliciter un nouvel indicateur** : les indicateurs ne peuvent rejoindre la plateforme qu'en étant invité par un commercialisateur. Pour ce faire, il suffit au commercialisateur de remplir le formulaire suivant :



Une fois cela fait, si l'adresse mail du futur indicateur a été indiquée, ce dernier reçoit automatiquement un mail contenant un lien d'invitation lui permettant de s'inscrire, sinon il suffit au commercialisateur de faire parvenir le lien à l'indicateur par ses propres moyens.

- **La possibilité de rechercher un indicateur (en fonction de filtres)** : grâce à la puissance de Vue.js et de la syntaxe ES2015, il est extrêmement facile d'effectuer de créer un système de recherche, et ce tout en n'ajoutant aucune nouvelle requête à effectuer vers le serveur, car les pages HTML sont générées par le client, où les données sont stockées. Il suffit donc juste de réordonner ces données, et la nouvelle page est automatiquement rendue, sans avoir à la rafraîchir manuellement. *(On obtient donc de meilleures performances, tout en augmentant l'expérience de l'utilisateur...)*

```
computed: {
  filteredList1() {
    return this.myInfluencersInvite.filter(influencerInvite => {
      if (this.filter1 === 'Nom') return influencerInvite.name.toLowerCase().includes(this.search1.toLowerCase())
      if (this.filter1 === 'Hashtags') return influencerInvite.hashtagRefs.find(hashtag => hashtag.libelle.toLowerCase().includes(this.search1.toLowerCase()))
    })
  },
  filteredList2() {
    return this.activeInfluencers.filter(activeInfluencer => {
      activeInfluencer.name = `${activeInfluencer.firstname} ${activeInfluencer.lastname}`
      if (this.filter2 === 'Nom') return activeInfluencer.name.toLowerCase().includes(this.search2.toLowerCase())
      if (this.filter2 === 'Hashtags') return activeInfluencer.hashtagRefs.find(hashtag => hashtag.libelle.toLowerCase().includes(this.search2.toLowerCase()))
    })
  }
}
```

Code gérant le filtrage des données « stockées » dans le cache du client

```

<tbody>
<tr v-bind:key="influencerInvite._id" v-for="influencerInvite in filteredList">
  <td @click="getInfluencerInviteDetail(influencerInvite)" data-toggle="modal" data-target="#influencerInvite" class="pointerHand">
    <b><u>{{influencerInvite.name}}</u></b>
  </td>
  <td>{{influencerInvite.date | formatDate}}</td>
  <td>
    <li v-bind:key="socialNetwork._id" v-for="socialNetwork in influencerInvite.socialNetworks">
      <a :href="socialNetwork.url" target="_blank">{{ socialNetwork.url }}</a>
    </li>
  </td>
  <td>{{influencerInvite.invitationLink}}</td>
  <td>{{influencerInvite.dateLastClicked ? $options.filters.formatDate(influencerInvite.dateLastClicked) : "Jamais"}}</td>
  <td>{{influencerInvite.email}} <br> {{influencerInvite.phone}}</td>
  <td>
    <li v-bind:key="hashtag._id" v-for="hashtag in influencerInvite.hashtagRefs">
      {{ hashtag.libelle }}
    </li>
  </td>
  <td><button class="btn btn-sm btn-danger" data-toggle="modal" data-target="#confirmInviteCancel" @click="toCancel = influencerInvite">Supprimer</button>
  </td>
</tr>
</tbody>

```

Code gérant l'affichage des données filtrées

- **La possibilité de bloquer un indicateur** : évitant ainsi toute interaction entre le couple commercialisateur / indicateur concerné...
- **La possibilité d'envoyer un mail d'invitation / de relance** : via l'intégration du module « nodemailer » permettant l'envoi de mails...

3.1.2 – Développement de l'application : partie back end.

Comme expliqué auparavant, sans API, la plateforme ne peut exister. Par API on entend une API web, permettant via diverses requêtes http d'exécuter du code sur le serveur tout en interagissant avec les données de la base de données.

Etant donné que l'API utilise des requêtes http de type GET, PUT, POST et DELETE, elle peut être qualifiée d'API RESTful (qui est conforme au style d'architecture REST).

Cette dernière, réalisée en Node, est entièrement basée sur le framework d'application web de loin le plus connu de la techno : Express JS. Ce framework offre une multitude de possibilités d'application, en passant par la création d'API, jusqu'au server-side rendering grâce à l'utilisation de très nombreux langages de templating.

Le fonctionnement de l'API est assez simple : Express écoute toutes les requêtes HTTP sur le port 3000, et son routeur effectue certaines actions en fonction des routes appelées.

Un middleware (élément mettant en relation deux applications (ou parties d'applications) séparées) permet de gérer l'authentification des utilisateurs via la mise en place via le standard JWT (Json Web Token). Le principe est assez simple, l'utilisateur remplit les champs du formulaire de connexion, et lorsqu'il clique sur le bouton « Se connecter », une requête http de type POST contenant le nom d'utilisateur et le mot de passe est envoyée depuis le front, l'API vérifie la validité du couple nom d'utilisateur / mot de passe, et si ce dernier est

valide, l'API renvoie un « token », qui est en fait une chaîne de caractère qui contient certaines informations sur l'utilisateur, encodées à partir d'une sorte de mot de passe maître.

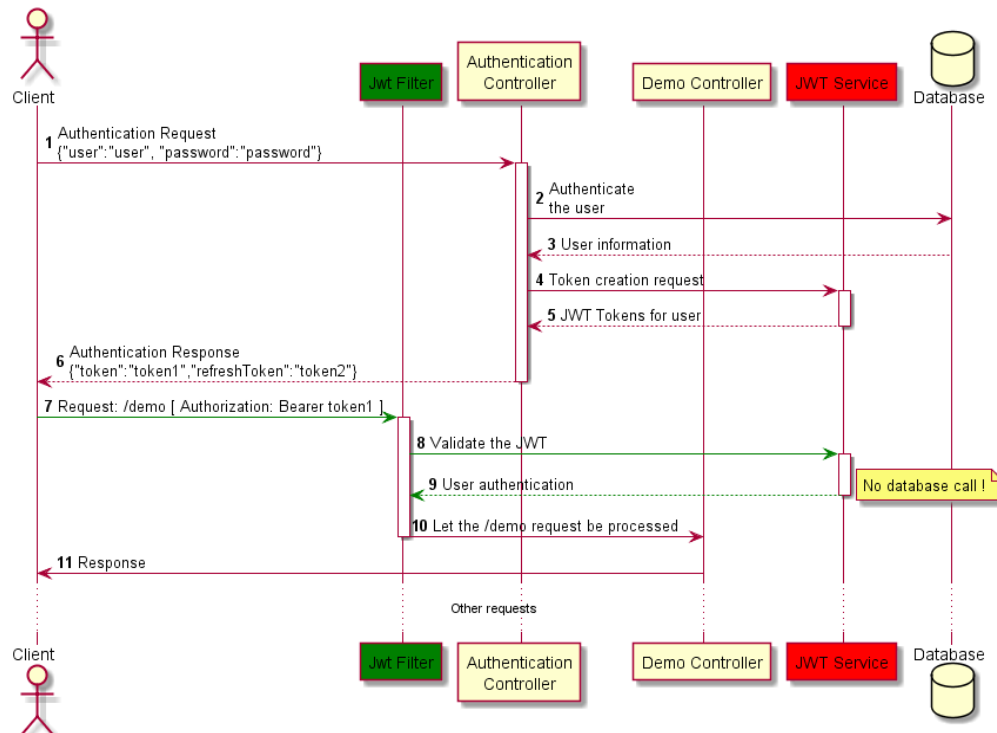


Diagramme de séquence présentant l'authentification via JWT

```

You, 11 days ago | 1 author (You)
const express = require('express')
const UserController = require('../controllers/user.controller')
const PermissionMiddleware = require('../middlewares/auth.permission.middleware')
const ValidationMiddleware = require('../middlewares/auth.validation.middleware')
const UserVerificationMiddleware = require('../middlewares/verify.user.middleware')

const router = express.Router()

router.post('/', [
  UserVerificationMiddleware.hasRegisterValidFields,
  UserVerificationMiddleware.isEmailOrUsernameNotInUse,
  UserController.create
])

router.get('/:userId', [
  ValidationMiddleware.validJWTNeeded,
  UserController.getById
])

router.patch('/:userId', [
  ValidationMiddleware.validJWTNeeded,
  PermissionMiddleware.onlySameUserOrAdminCanDoThisAction,
  UserController.update
])

module.exports = router

```

Exemple de routage avec Express JS

```
/**
 * Allows Insurer to list his Products
 * **/
router.get('/secure/[insurerId]', VerifyToken, async function (req, res, next) {
  try {
    const insurer = await Insurer.findById(req.insurerId)
    if (!insurer) {
      winston.error(`GET [insurerId] - !insurer for ${req.insurerId}`)
      return res.status(500).json({
        title: 'Une erreur est survenue',
        error: 'Une erreur est survenue'
      })
    }

    const products = await Product.find({insurer: insurer._id})

    res.status(200).json({
      products: products
    })
  } catch (err) {
    winston.error(`GET [insurerId] - error : ${err}`)
    return res.status(500).json({
      title: 'Une erreur est survenue',
      error: 'Une erreur est survenue'
    })
  }
})
```

Exemple d'implémentation d'une route de 357coupons, avec connexion à la bse de données MongoDB via l'ORM « mongoose »

3.2 – L'après développement : mise en place de tests

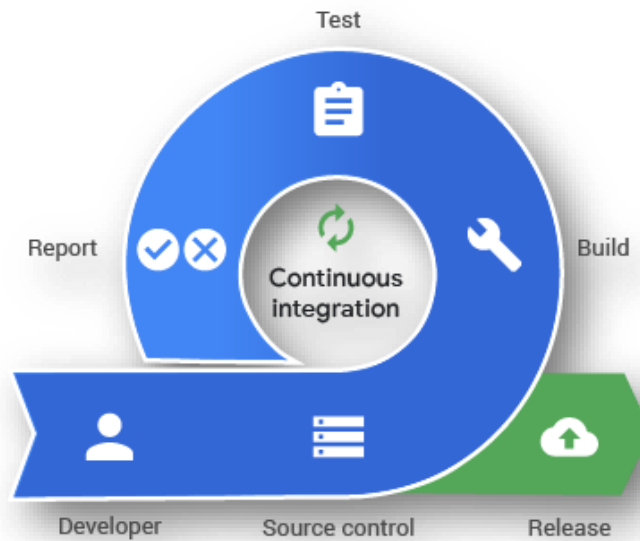
Une fois le développement des diverses fonctionnalités effectué, j'ai développé une centaine de « tests d'intégration ». Ces tests consistent à vérifier que chacun des modules indépendants de l'application fonctionne bien une fois assemblés. Les tests d'intégrations sont utiles pour tester des fonctions non testables via des tests unitaires.

Un test d'intégration peut-être par exemple : insérer une valeur en base de données, puis vérifier que cette dernière y est bien présente. Alors qu'un test unitaire serait par exemple, additionner 1 à 2, et vérifier que le résultat est bien 3.

On peut écrire de tels tests assez facilement grâce à des bibliothèques spécifiques, telles que Jest, Chai ou Mocha. Ces bibliothèques mettent à disposition des commandes, permettant de lancer les chaînes de tests manuellement. Cependant, si l'on souhaite tirer pleinement à profit la puissance des tests unitaires et d'intégration, il faut les embarquer dans un cycle d'intégration continue (aussi connue sous « CI », signifiant « Continuous Integration » en anglais).

L'intégration continue est un service qui permet d'automatiser l'exécution de tests, voir même de gérer des déploiements. En effet, il est possible de la configurer pour qu'à chaque nouvel ajout de code, l'exécution des tests écrit au préalable se déclenche, envoyant un rapport

d'erreur si erreur il y a, ou bien exécutant par la suite le déploiement de l'application dans le cas contraire.



Exemple de workflow avec un système d'intégration continue

Les avantages de l'intégration continue sont nombreux, mais elle permet dans un premier temps de réduire les risques liés à l'intégration. En effet, lorsque de nombreuses personnes travaillent sur des tâches séparées sur un même projet, les risques de bugs sont accrus. Or, grâce à l'intégration continue, une énorme quantité de bugs sont détectés avant que le code soit livré, et ils sont en plus de cela plus facile à corriger du fait qu'il est plus simple de déterminer leur provenance.

En plus de cela, cette pratique permet d'améliorer la qualité du code. En effet, en passant moins de temps sur le débogage, les développeurs ont davantage de temps pour se concentrer sur la qualité du code. La qualité du produit fini s'en trouve donc améliorée.

Par extension, l'intégration continue permet d'augmenter le niveau de confiance des développeurs, qui n'ont plus peur de « casser » le code et qui peuvent ainsi se révéler plus productifs et enthousiastes. Les nouveaux venus pourront aussi se lancer dans le projet plus facilement.

Conclusion

A titre de conclusion, il semble intéressant de se demander quelles sont les limites à fixer, et comment les fixer, lorsque l'on parle de la personnalisation des services ?

Grâce aux technologies de pointe utilisées chez Road-b-Score, de nouveaux secteurs peuvent être équipés d'outils auxquels ils n'avaient auparavant pas accès. L'esprit R&D / apport de technologie est quelque-chose que j'ai vraiment apprécié au sein de l'entreprise, car j'aime pouvoir réfléchir à inventer de nouvelles solutions tout en relevant de nouveaux challenges.

Au-delà du domaine d'activité de Road-b-Score, mon stage a été extrêmement instructif. Au cours de ces 9 semaines j'ai pu confirmer l'engouement que je voue au fonctionnement des startups, que ce soit du point de vue de l'articulation des différents services, des relations humaines entre les différents employés de la société, ou bien de l'esprit de travail en général. Au travers de cette convivialité, j'ai pu comprendre que l'activité d'une société est bien plus performante dans une atmosphère chaleureuse et bienveillante.

Glossaire

API : Acronyme anglais de « Application Programming Interface », il s'agit d'une interface de programmation qui permet de se « brancher » sur une application pour échanger des données.

Backend : Lorsque l'on parle de développement dit : « backend », il s'agit de la partie immergée de l'iceberg. On peut le décomposer en trois parties essentielles : serveur, application et base de données.

Bootstrap : Bootstrap est une collection d'outils utile à la création du design de site et d'applications web. C'est un ensemble qui contient des codes HTML et CSS, des formulaires, boutons, outils de navigation et autres éléments interactifs, ainsi que des extensions JavaScript en option.

Caching : Terme anglais désignant le fait de stocker des données dans un cache. Un cache est une zone servant au stockage temporaire de données dans le but d'accélérer le chargement d'une application.

CSS : Acronyme anglais de « Cascading Style Sheets » : il s'agit d'un langage de programmation qui permet de gérer l'apparence et la mise en page d'une page web (la couleur, les polices, la disposition des éléments etc...).

Docker : Logiciel leader de l'industrie de la virtualisation par conteneurs.

Frontend : Lorsque l'on parle de développement dit : « frontend », il s'agit des éléments qui sont visibles à l'écran et avec lesquels on peut interagir.

FTP : Acronyme anglais de File Transfer Protocol : il désigne un protocole utilisé pour le transfert de fichiers entre deux postes distants.

HTML : Acronyme anglais d'« Hyper Text Markup Language » : il s'agit d'un langage de balisage utilisé pour créer des pages web.

IDE : Acronyme anglais d'« Integrated Development Environment », « Environnement de Développement Intégré » en français) Terme utilisé pour désigner un ensemble d'outils visant à augmenter la productivité des développeurs.

Insurtech : (aussi appelé « assurtech » en français) Terme utilisé pour désigner une entreprise exerçant dans le secteur de l'assurance, et s'appuyant sur les nouvelles technologies pour introduire des innovations qui conduisent inéluctablement à l'éclosion de nouveaux modèles économiques, de nouveaux processus, et de nouveaux produits.

jQuery : jQuery est un Framework (bibliothèque de contenu) Javascript qui permet de faciliter des fonctionnalités communes de Javascript. L'utilisation de cette bibliothèque permet de gagner du temps de développement.

JS : Acronyme de JavaScript : langage de Scripting qui a la particularité d'être exécuté sur le poste client, c'est-à-dire par votre navigateur.

PHP : Acronyme récursif anglais de PHP : HyperText Processor : il s'agit d'un langage de programmation utilisé pour rendre un site web dynamique, notamment en communiquant avec une base de données. Le PHP a pour caractéristique d'être exécuté du côté du serveur, qui renverra une page HTML/XML au poste client.

Plugin : Terme anglais désignant un « paquet » qui vient compléter un logiciel hôte pour lui apporter de nouvelles fonctionnalités. (Aussi nommé « module d'extension » en français).

Server-side : (« côté serveur ») fait référence à des opérations qui sont effectuées par le serveur dans la communication entre client et serveur dans un réseau informatique.

SGBD : Acronyme de « Système de Gestion de Base de Données » : il s'agit d'un logiciel qui permet de stocker des informations dans une base de données.

SQL : Acronyme anglais de « Structured Query Language » : il s'agit d'un langage informatique utilisé pour exploiter des bases de données.

UML : Acronyme anglais de « Unified Modeling Language » : il s'agit d'un langage de modélisation graphique à base de pictogrammes conçu pour fournir une méthode normalisée pour visualiser la conception d'un système.

Workflow : anglicisme pour « flux de travaux » : il s'agit de la représentation d'une suite de tâches ou opérations effectuées par une personne, un groupe de personnes, un organisme, etc.