TABLE DES MATIÈRES

Illustrations

Tableaux

Liste des abréviations

URL : Uniform Resource Locator, localisateur uniforme de ressource en français

API : application programming interface

INTRODUCTION

La collecte d'informations par les utilisateurs afin de remplir une base de données peut révéler de réels problèmes. Premièrement, il faut savoir motiver ses utilisateurs afin qu'ils collectent des informations, ce qui est le cœur de la réussite ou non de l'application. Puis, il faut pouvoir avoir un contrôle sur les données entrées afin d'éviter les robots ou les personnes mal intentionnées qui détruiraient la crédibilité de notre projet.

Beaucoup d'applications demandent à leurs clients de renseigner des données afin d'améliorer les propositions faites aux autres utilisateurs. Cependant, chacune a sa technique, car cela dépend du domaine que nous cherchons à toucher. Des personnes cherchant un restaurant n'auront pas les mêmes attentes vis-à-vis de l'application que des personnes cherchant un itinéraire au sein d'une ville. Notre domaine consiste en une application pour les personnes à mobilité réduite visant à faciliter leur trajet au sein de l'espace urbain.

L'institut de recherche de la Haute Ecole Spécialisée de Sierre a développé une carte interactive permettant de collecter différents dangers dans une ville. David Mack, un ancien étudiant a réalisé son travail de Bachelor dans le but d'ajouter le calcul d'itinéraire sur le site web. L'application actuelle est fonctionnelle, mais le nombre d'utilisateur et de données est actuellement très faible.

Ce travail de Bachelor a été principalement consacré au développement d'une application mobile facilitant la collecte de données et l'interaction avec l'utilisateur. Il nous a aussi fallu nous pencher sur les différents moyens de validation afin de fournir des informations correctes et développer des algorithmes.

Nous avons travaillé en quatre étapes. Tout d'abord, nous avons réalisé un état de l'art théorique en commençant par définir le Crowdsourcing. Puis, nous avons parcouru des documents scientifiques afin de ressortir les meilleurs moyens de motivations et de validations connus. Nous avons trouvé des documents se regroupant en deux types : calcul mathématique et test concret. Dans le premier cas, il s'agit de jeux de données existants qui sont analysés avec des méthodes de calcul brut. Le second cas ne va partir de rien et mettre

en confrontation deux ou plusieurs hypothèses. Nous avons donc synthétisé tous ces documents afin de voir les aspects les plus conseillés à utiliser.

La seconde phase consistait à étudier les applications qui utilisent la foule comme source d'informations principale. Nous avons choisi deux applications très connues par le public, deux applications semblables à la nôtre et les services de Google afin d'avoir un point de vue assez général. Puis nous avons comparé les points faibles et les points forts de chaque programme afin de ne retenir que le meilleur pour notre projet. À la fin de cette étape, nous avons également comparé la différence entre les conseils théoriques trouvés dans l'étape une et les conseils pratiques de cette phase. Cela a mis en évidence les meilleurs choix à prendre pour notre application mobile.

Dans la troisième phase, nous avons commencé à développer. Nous avons débuté par la construction d'un schéma de base de données qui pouvait subvenir à toutes les futures fonctionnalités. Puis, nous avons mis en place notre environnement de travail et nous avons intégré les différentes fonctionnalités déjà présentes sur le site web. Nous avons rajouté la notion d'utilisateur avec une notation afin de pouvoir valider nos points collectés. Cet utilisateur va aussi permettre la liaison entre lui et les rapports collectés automatiquement par l'application. Nous avons fourni la possibilité de désactiver ces rapports automatiques, car certaines personnes n'aiment pas être suivies tout au long de la journée. Nous avons également ajouté un système de notification dans le cas où l'utilisateur se trouverait à proximité d'un point pour qu'il nous aide à collecter des données. Nous avons limité le nombre de notifications à une par jour afin de ne pas être trop invasifs. Finalement, nous avons ajouté trois algorithmes : le premier afin de valider un point, le second afin de correctement noter les utilisateurs par exemple lors de l'inscription et le dernier afin d'avoir les points qui se trouvent réellement à proximité du parcours que nous voulons effectuer.

Dans la quatrième phase, nous avons testé notre application afin d'éviter tous potentiels bugs qui pourraient grandement nuire à l'attrait des utilisateurs pour notre application. Nous avons également essayé d'importer les anciennes données dans la nouvelle base grâce à un robot qui automatise cette tâche. Nous avons également modifié quelques points dans notre projet

pour qu'ils soient facilement modifiables. Par exemple, une simple variable pour l'utilisation ou non de l'algorithme de validation de point. Cela afin de rejoindre notre hypothèse de base qui est que les algorithmes pour la collecte de données par les utilisateurs ne sont bien que si le nombre de données collectées par jour est assez important. Dans le cas contraire, cela aurait l'effet inverse et nuirait à l'attrait des gens pour l'application.

1. État de l'art

1.1. Problématique

L'Institut d'informatique de gestion de la HES-SO de Sierre a déjà développé une application permettant d'enregistrer des points d'intérêts et de trouver un itinéraire pour les personnes à mobilité réduite. Lors d'un précédent travail de Bachelor, un élève s'est concentré sur le traçage du chemin le plus court. Il n'y a cependant pour le moment aucun moyen de valider les données entrantes et le nombre de personnes utilisant l'application reste faible.

Ces lacunes pourraient apporter le problème suivant. Une personne en chaise roulante souhaite se rendre dans un restaurant en utilisant notre application. Il rentre son point de départ et son point d'arrivée et entame la route proposée. Cependant, il se heurte rapidement à des escaliers qui n'étaient pas sur la carte dû au manque de danger renseigné. Il contourne cet obstacle pour, au final, se retrouver devant le restaurant qui avait été indiqué comme utilisable par les handicapés. Cependant, il s'avéra que le patron de l'établissement avait lui-même rempli les informations afin d'attirer plus de personnes et finalement les toilettes ne sont pas accessibles.

Le problème majeur dans ce cas de figure est que l'utilisateur va rapidement blâmer et se désintéresser de l'application dû au nombre d'erreurs élevé.

Il sera donc nécessaire d'analyser les différentes solutions existantes sur le marché pour l'attrait des gens et la validation des données pour palier à ce problème.

1.2. Crowdsourcing

Afin que notre application soit plus populaire et garantisse une meilleure fiabilité, nous devons compter sur le soutien de la foule. Les utilisateurs devront en effet saisir des points dans le but de pouvoir avoir des données exploitables. Ce processus est appelé Crowdsourcing : la foule (crowd) va participer à la récolte d'informations (David Mack, p. 57).

Le souci majeur est d'obtenir une solution simple et motivante afin que les utilisateurs ne voient pas cela comme une contrainte, mais plutôt comme un apport pour la société. Nous allons donc détailler ci-dessous ce processus et les différentes solutions que propose le marché.

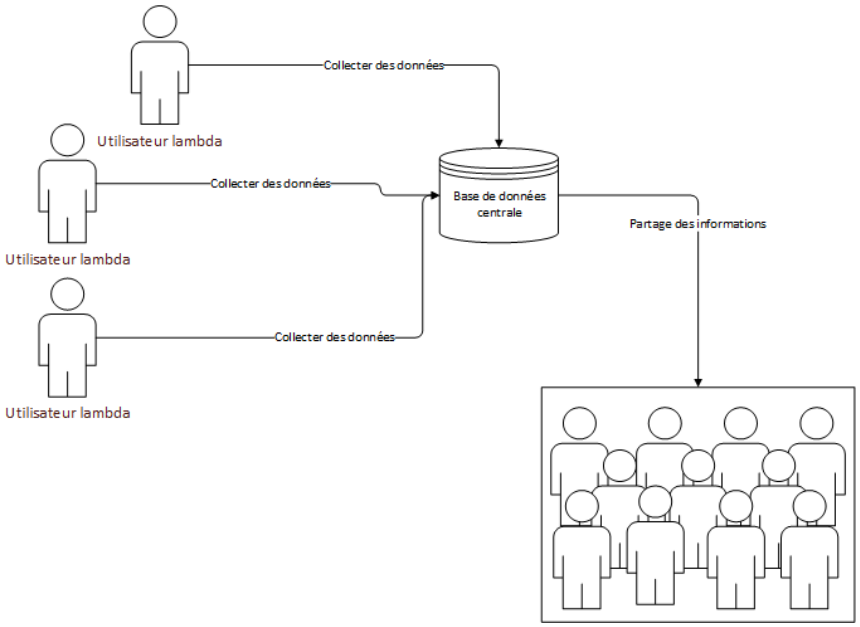


Figure 1 : Exemple simplifié du Crowdsourcing
Source personnelle

1.2.1. Généralités

Nous n'allons pas ici faire un cours d'histoire sur l'origine du Crowdsourcing, mais il faut savoir que ce terme se compose de 2 mots anglais : Crowd qui signifie la foule et de outsourcing qui est la sous-traitance des données. Il faut soulever le fait que nous pouvons traduire ce terme par « externalisation ouverte » (Jean-Fabrice Lebraty, p.11).

Selon Yuxiang Zhao et Qinghua Zhu (p. 4), il existe plusieurs types de motivation afin d'attirer les gens à collecter des données. Nous allons utiliser les termes anglais afin de ne pas déformer le vrai sens par une traduction douteuse. Premièrement, nous avons l'« external motivation » qui permet de motiver l'utilisateur par des moyens externes à sa personne. L'illustration la plus simple est le fait de donner une récompense monétaire pour chaque donnée ajoutée. Dans certains cas, cela pourra apporter des perspectives d'emploi. Imaginons

que la collecte de données est organisée par une grande entreprise et qu'elle décide d'engager les personnes ayant collecté le plus de données de bonne qualité.

En second, nous avons l'« introjected motivation ». Il s'agit d'un moyen de motivation où l'utilisateur va suivre un but désigné par une tierce personne cela afin de gagner la reconnaissance de ses pairs et que son utilité soit perçue. Cela a pour but d'améliorer l'ego de la personne sans avoir de culpabilité ou d'anxiété tout en se basant sur une confiance générale ou une norme subjective.

La troisième source de motivation est l'« identified motivation ». Ce moyen est semblable au précédent à la différence que les objectifs à atteindre sont plus en accord avec la personne qui va les accepter comme étant personnellement importants. Cela va faire que nous allons plus tendre vers de la gloire plutôt que la reconnaissance. Cette source se doit de jouer clairement sur identification sociale et sur un lien de confiance spécifique.

Quatrièmement, nous avons l'« integrated motivation ». Il s'agit de la forme la plus autonome de motivation. Part autonome, nous voulons dire que l'utilisateur va remplir naturellement des données. Cela vient du fait que l'objectif commun de la collecte est profondément ancré au sein de l'individu et fait partie de ses comportements.

Finalement, il y a toutes les sources de motivation qui n'apportent rien aux utilisateurs si ce n'est l'intérêt et le plaisir. Nous appelons ces moyens « intrinsic motivation ». Lorsque le collecteur va ajouter des données, il n'attend rien en retour de l'application. Il peut être motivé par plusieurs sources :

- Un simple amusement
- Une curiosité
- Un développement d'une compétence spécifique
- De l'altruisme

La figure 2 est un récapitulatif des moyens vus précédemment.

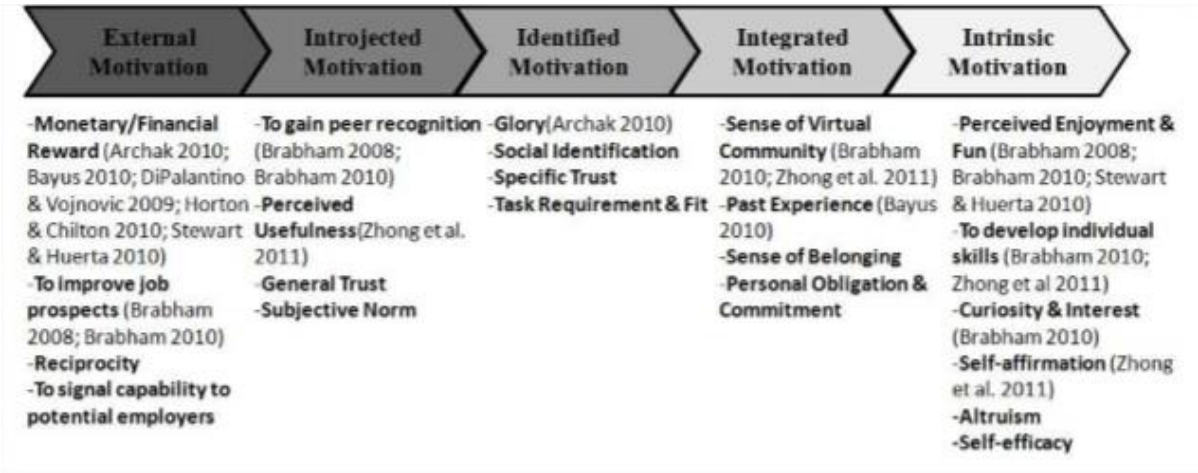


Figure 2 : Récapitulatif des sources de motivation
Zhao, Y., & Zhu, Q. p. 4

1.3. Attrait de la population

1.3.1. Calcul des facteurs de motivation

Dans le document « Task Design, Motivation, and Participation in Crowdsourcing Contests », il a été mis en évidence les différents moyens de motivation et leur influence sur la population. Préliminairement à l'étude, neuf hypothèses ont été formulées.

1. L'intention de participer est étroitement liée à la réelle participation. (H1)
2. La motivation extrinsèque afin d'obtenir une récompense monétaire est associée positivement au fait de vouloir participer. (H2a)
3. La motivation extrinsèque afin d'obtenir une reconnaissance est associée positivement au fait de vouloir participer. (H2b)
4. La motivation intrinsèque est liée positivement au fait de vouloir participer. (H3)
5. L'autonomie (liberté des horaires pour la collecte d'informations) est associée positivement à la motivation intrinsèque. (H4)
6. La variété (tâche non répétitive) est associée positivement à la motivation intrinsèque. (H5)
7. Partager et récolter des informations tacites a un effet négatif sur la motivation intrinsèque. (H6)

8. La capacité d'analyse (disponibilité des connaissances concrète afin de réaliser la tâche) a un effet positif sur la motivation intrinsèque. (H7a)
9. La variabilité (fréquence d'événement inattendu lors de la collecte) a un effet positif sur la motivation intrinsèque. (H7b)

En plus de ces hypothèses, une analyse portant sur la confiance et sur plusieurs facteurs démographiques a été réalisée.

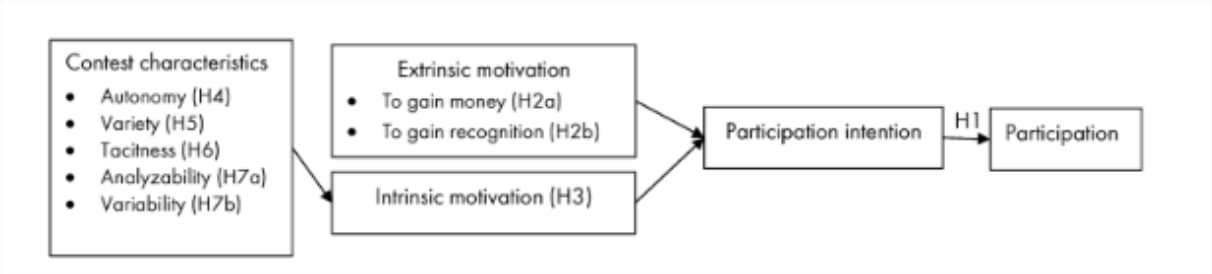


Figure 3 : Liste des hypothèses
Zheng, H., Li, D., & Hou, W. p.61

Après quelques calculs à la suite d'une expérience, les hypothèses ont pu être validées ou réfutées. La synthèse se trouve dans la figure ci-dessous.

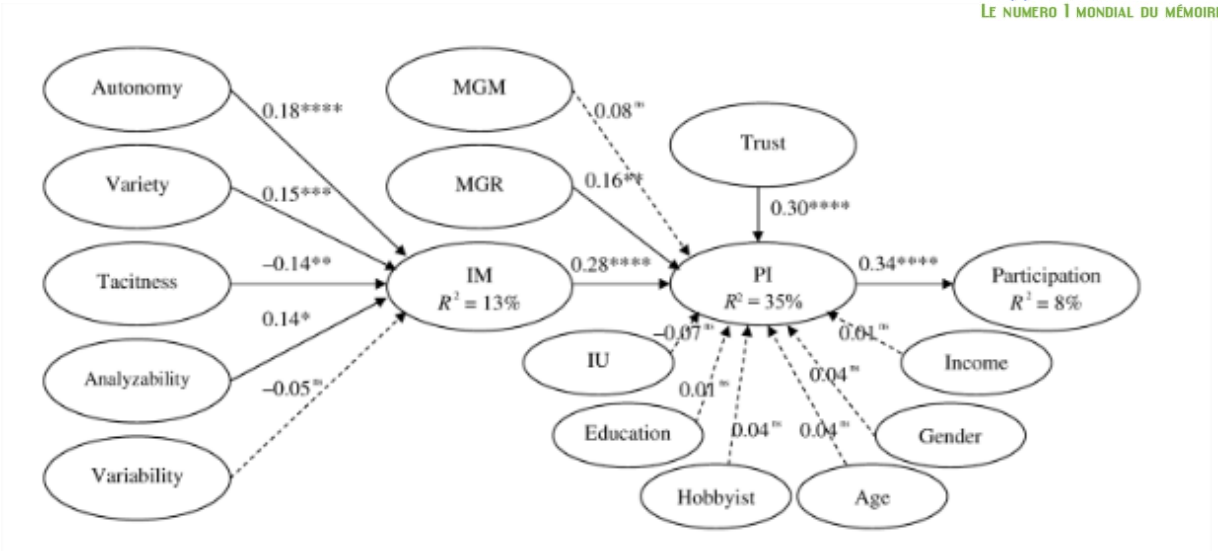


Figure 4 : Résultat graphique des hypothèses
Zheng, H., Li, D., & Hou, W. p.75

Nom de l'hypothèse	Coefficient	Est validée
H1	0.34, $p < 0.001$	Oui

H2a	0.08, p >0.1	Non
H2b	0.16, p < 0.05	Oui
H3	0.28, p < 0.001	Oui
H4	0.18, p < 0.001	Oui
H5	0.15, p < 0.01	Oui
H6	-0.14, p < 0.05	Oui
H7a	0.14, p < 0.1	Oui
H7b	-0.05, p > 0.1	Non

Tableau 1 : Résultat des hypothèses sous forme de tableau

Il en ressort donc que la motivation pécuniaire n'a que très peu d'importance dans l'attrait pour le Crowdsourcing face à la motivation intrinsèque ou la réputation. De plus, il n'y a qu'une très faible influence si l'utilisateur se retrouve face à un événement inattendu. Ces deux facteurs seront donc enlevés dans la suite de l'étude.

1.3.2. Étude pratique de la motivation

Concernant cette partie, nous allons nous baser sur une étude faite dans le document Crowdsourcing Urban Accessibility : Some Preliminary Experiences with Results qui met en évidence les différentes formes d'application qui peuvent motiver les utilisateurs à collecter des données en se basant sur un test de 50 personnes.

L'étude va porter sur les trois principales sortes d'application pour collecter des données :

- Une application de base
- Une application qui fonctionne sur un système de récompense
- Une application qui fonctionne sous forme de jeu

1.3.3. Application de base

Il s'agit ici d'une application qui permet de collecter des points où il y a des barrières architecturales. Ces données sont envoyées sur un serveur distant depuis le smartphone des utilisateurs. Avec ces données, une carte digitale est mise à jour et affiche les différents points. Nous pouvons voir que cette application ressemble à la première version de We-Map, la

version qui ne contenait pas encore le calcul d'itinéraire. Pour ce cas, l'étude ne compte que sur la bonne volonté et la compassion des gens afin de recenser les points.

Figure 5 : Application de base faite par l'université de Bologne
Salomoni, P., Prandi, C., Roccetti, M., Nisi, V., & Nunes, N. J. p.133

1.3.4. Application récompense

Cette application va mettre en place un système de récompense. Vu que cette expérience a eu lieu dans l'université de Bologne, la récompense était la réussite d'un cours enseigné dans cette même école. Pour obtenir la récompense, il fallait remplir un minimum de 5 points dans l'application.

Il est clair que pour le cas de WE-MAP, il faudra trouver une récompense pouvant être utile à l'utilisateur et vu qu'il n'y a pas de budget, il n'est pas possible de rémunérer les utilisateurs.

1.3.5. Application jeu

Pour ce cas, l'application va engager les gens en transformant une application en jeu afin d'augmenter l'attrait des gens. L'université de Bologne a décidé de faire une application ou des zombies apparaissent et pourchassent le joueur. Le seul moyen que possède le joueur pour se défendre est de renseigner des obstacles architecturaux afin de pouvoir acheter des munitions pour son arme.

Figure 6 : Application de jeu réalisé par l'université de Bologne
Salomoni, P., Prandi, C., Roccetti, M., Nisi, V., & Nunes, N. J. p.131

1.3.6. Résultat de l'expérience

Il était évident que l'application de base aurait moins de points recensés que les deux autres. Les chiffres sont les suivants : 95 points pour l'application basique, 311 points avec récompense et 286 points avec la gamification. Nous pouvons expliquer cette différence par le fait que la motivation est plus concrète sur les deux dernières applications. Sur celle de base, il n'y a aucun moyen d'encourager l'utilisateur à continuer de renseigner des points sur le long terme.

Une constatation importante à soulever est que le nombre de points renseignés par les étudiants ayant utilisé l'application avec récompense avoisine toujours autour les 6, alors que ce nombre varie beaucoup plus avec l'application de jeu. Ci-dessous, vous pouvez voir les résultats sous forme de tableau.

Applications	Nombre de points	Nombre d'utilisateurs	Nombre de points par utilisateurs (moyenne)	Nombre de points par utilisateurs (écart type)
Basique	95	48	1.97	1.28
Récompense	311	47	6.60	3.56
Jeu	286	48	5.95	16.60

Tableau 2 : Comparatif des applications

Avec ce tableau, nous voyons que l'écart type est assez élevé lors de l'utilisation de l'application sous forme de jeu. Cela signifie que les utilisateurs qui n'ont pas eu d'intérêt dans ce concept n'ont renseigné que très peu de points alors que les utilisateurs pris au jeu ont renseigné beaucoup plus de barrières que l'application avec récompense. Nous pourrions donc supposer que le nombre d'utilisateurs pourrait être grandement réduit si le concept plaît à beaucoup de monde. De plus, lorsque nous utilisons la ludification, cela ne nous coûte rien alors que les récompenses doivent être budgétées.

Jusqu'à maintenant, nous avons parlé du nombre de rapports, mais nous avons oublié de tenir compte d'un facteur important : le temps. En effet, si les points sont récoltés le premier jour et plus du tout par la suite, nous aurons un problème de motivation évident. Ci-dessous, nous pouvons voir un graphique qui montre la récolte sur 7 jours.

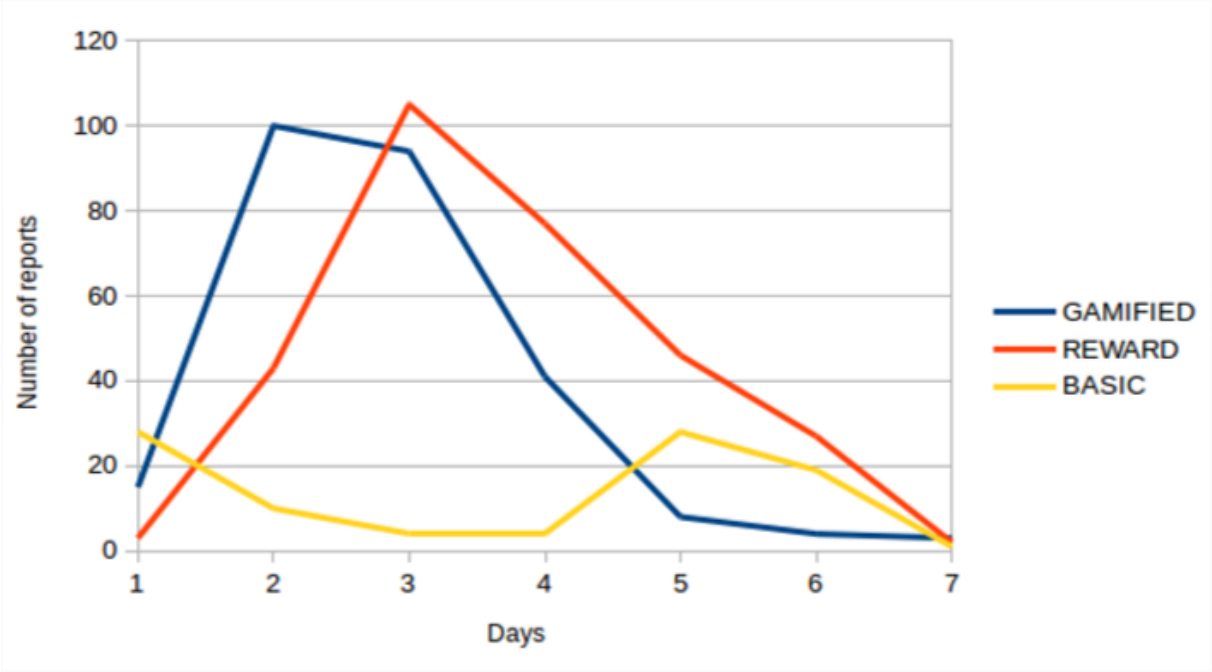


Figure 7 : Nombre de rapports par jour
Salomoni, P., Prandi, C., Roccetti, M., Nisi, V., & Nunes, N. J. p.132

Cependant, ces données doivent être groupées avec d'autres afin d'obtenir une bonne signification. Lors de cette étude, il a été relevé que les élèves utilisant l'application basique et de récompense prirent leur chemin habituel pour aller de l'université à la gare. Cela a eu pour effet d'avoir une récolte ciblée des points. Lorsqu'il s'agit de l'application de jeu, nous ferons face à une marche sans destination orientée, c'est-à-dire que notre utilisateur va se

balader dans la ville tout en évitant les blocages de l'application. Pour une application future, il serait possible de faire apparaître des barrières virtuelles afin d'orienter les personnes vers des zones encore inexplorées. Dans le cas présent, cela a permis de récolter une plus grande quantité de points dans la zone prédéfinie. Les figures 6 et 7 montrent la comparaison des rapports relevés entre l'application récompense et celle de jeu.

Figure 8 : Rapports avec récompense
Salomoni, P., Prandi, C., Roccetti, M., Nisi, V., & Nunes, N. J. p.133

Figure 9 : Rapports avec jeu
Salomoni, P., Prandi, C., Roccetti, M., Nisi, V., & Nunes, N. J. p.133

Maintenant que nous avons ces données supplémentaires, nous pouvons analyser le facteur temps. Finalement au sein de cette étude, il a été relevé que la décroissance de la motivation était due à un manque de point, car ces derniers avaient tous été récoltés lors des premiers jours. Les cercles de point rouge restant au centre sont des endroits inaccessibles par les étudiants, c'est pourquoi ils n'ont pas été récoltés.

En conclusion, nous pouvons dire que la manière la plus exploitable afin de toucher des gens qui pourraient récolter sur un large spectre serait d'implémenter une ludification au sein de l'application. Selon l'entreprise IBM, il existe quatre facteurs à mettre en place pour avoir un attrait maximum (Carlos Cardonha, Diego Gallo, Priscilla Avegliano, Ricardo Herrmann, Fernando Koch et Sergio Borger ; p.2). À savoir :

- Être légers, intuitifs et faciles à utiliser
- Implémentable pour tous les environnements (iOS, Android, etc)
- Avoir des rétroactions avec les utilisateurs pour garder l'engagement
- Une gamification ou ludification

Nous n'avons développé que ce dernier point, car il s'agit de la plus grosse lacune au sein de l'application. De plus, les rétroactions seront amenées avec le processus de validation des données.

1.3.7. Autres méthodes proposées

Lors du travail de Bachelor réalisé par David Mack en 2017, il a été proposé de fonctionner avec un système de récompense afin de motiver les gens à utiliser l'application. En effet, la collecte de points rapporterait des jetons à l'utilisateur qu'il pourrait ensuite dépenser afin d'utiliser le système de navigation. Cependant, ce système risque de freiner l'utilisation de l'application par les personnes à mobilité réduite, car ils ne pourraient pas collecter beaucoup de points afin d'utiliser la navigation. De plus, une personne valide n'obtiendrait rien d'utile en ajoutant des points.

Il est également proposé de délivrer des certificats virtuels afin de remercier les entreprises qui ont récolté des données pour l'application. Cela serait une forme de preuve de générosité que les compagnies pourraient afficher sur leur site, ce qui nous ferait également de la publicité gratuitement.

1.3.8. Synthèse des méthodes de motivation

Lors de ce chapitre, nous avons effectué une analyse théorique des facteurs de motivation avec comme point clé le fait que la motivation pécuniaire est faible. Nous avons également découvert les aspects favorisant la motivation intrinsèque, à savoir l'autonomie, la variété, l'échange d'informations tacites, la capacité d'analyse et la variabilité. Tous ces facteurs sont déjà bien présents au sein de notre application : l'autonomie, car un point peut être collecté à n'importe quelle heure ; la variété, car nous pouvons ajouter ou valider des points. Il n'y a pas d'échanges d'informations tacites, car il est facile de s'imaginer un point et de concevoir les dangers qu'il peut représenter. Les informations utiles à la collecte de point ne sont pas trop difficiles à comprendre. Notre application remplit donc pour le moment toutes les approches théoriques afin de pleinement attirer l'utilisateur.

Par la suite, nous avons analysé une étude sur une application similaire à la nôtre. Cependant, cette dernière s'est vue décliner en trois applications différentes : une basique, une de jeu et une de récompense. Puis elle a été testée par des élèves afin d'obtenir un classement sur le type d'application le plus rentable. Il en découle les résultats suivants : l'application de jeu peut générer beaucoup de points à condition que l'utilisateur accroche au concept sur le long terme. L'avantage de ce modèle face à celui de récompense est que ce dernier ne génère aucuns frais pour l'entreprise. Il faudra donc envisager la possibilité de créer une seconde application plus ludique afin de collecter les points pour notre système.

Finalement, les idées proposées dans l'ancien travail de Bachelor se basent les deux sur un système de récompense qui est, comme nous l'avons vu lors de l'analyse théorique, beaucoup moins attirant qu'une motivation intrinsèque.

1.4. Validation des données

Le second point important dans notre étude est de pouvoir prouver que les données insérées sont valides grâce à plusieurs méthodes de test. Nous commencerons par citer Craig Silverman: "*These days, forget about having 100 percent verified information — but you can have trusted sources or things with high probability*.". Cette phrase veut simplement dire qu'il est clairement impossible avec du Crowdsourcing d'obtenir cent pourcent de données valides, mais qu'il est possible d'avoir des sources sures afin d'avoir un haut potentiel de fiabilité. Afin d'obtenir cette sureté, il convient de diversifier ses sources.

1.4.1. Citizen Sensing et Breadcrumb

L'institut de recherche d'IBM du Brésil a travaillé sur une plateforme de Crowdsourcing afin de créer une carte d'accessibilité. Afin d'obtenir le plus de données possible, ils sont partis sur le principe suivant : créer 2 applications, une qui ne demande aucune interaction humaine pour collecter des données (breadcrumb) et l'autre qui envoie des rapports après que l'utilisateur ait complété un rapport (citizen sensing).

Figure 10 : Croisemenet des données entre Breadcrumb et Citizen Sensing
Cardonha, C., Gallo, D., Avegliano, P., Herrmann, R., Koch, F., & Borger, S p.3

Grâce aux rapports générés automatiquement, nous avons un parcours avec la longitude, la latitude et l'altitude. La collecte de données a lieu chaque 10 secondes. Avec ces informations, il est possible de calculer la vélocité. Nous pouvons donc déduire si un passage pourrait poser des problèmes aux personnes à mobilité réduite, dus à une pente trop raide ou des escaliers. Cependant pour avoir des calculs corrects, il faut réduire le bruit généré par le manque de précision du GPS des téléphones portables qui est environ d'une vingtaine de mètres. Pour réduire ces erreurs, le calcul de vitesse s'effectue sur les 10 derniers points et non pas sur le dernier uniquement. De plus, une vitesse maximum a été fixée à 6 km/h. Cela a permis de corriger la vitesse de l'utilisateur, illustré par la figure ci-dessous.

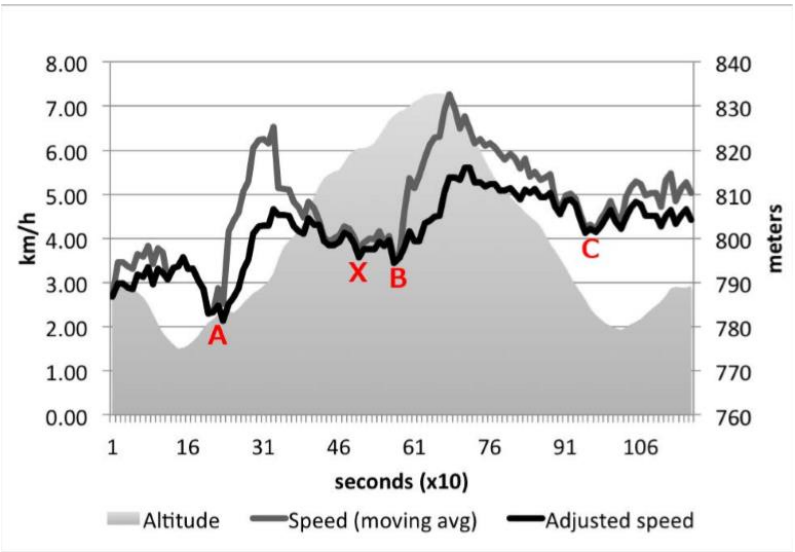


Figure 11 : Correction de la vitesse
Cardonha, C., Gallo, D., Avegliano, P., Herrmann, R., Koch, F., & Borger, S p.3

Plus le nombre de personnes participantes est élevé, meilleur pourra être le data mining afin de ressortir les barrières architecturales. De plus, l'article propose 3 moyens d'améliorer encore la validité des données.

Le premier est d'utiliser la corrélation. Cela peut s'avérer très utile afin de corriger les rapports des utilisateurs dans le cas où ces derniers classifieraient des barrières avec un autre label, mais qui reste cohérent. Grâce à une corrélation naïve (Naive Bayes Classifier), il sera possible de redéfinir les rapports de l'application Citizen Sensing. Pour le moment, seule cette classification peut s'appliquer, mais dans le cas où la technologie atteindrait un bon niveau, il serait possible de créer des modèles de causalité. Cette technique est également proposée dans le Framework KATARA afin de nettoyer les données émergentes du Crowdsourcing (Xu Chu, John Morcos, Ihab F. Ilyas, Mourad Ouzzani, Paolo Papotti, Nan Tang, Yin Ye ; p. 2).

Le second est l'analyse de sentiment. Grâce à cette analyse, il sera possible de définir l'état émotionnel du récolteur d'informations. Cela va viser à faire ressortir les problèmes cruciaux de la ville. Elle pourra se baser sur l'analyse d'un commentaire textuel ou audio afin de ressortir des signes distinctifs, comme une opinion positive, neutre ou négative.

Le dernier est l'établissement des priorités. En se basant sur le nombre de personnes ayant été affectées par un danger, il devient possible de mettre des priorités sur les obstacles

urbains. Cela va apporter une vision au responsable de ville afin de corriger les problèmes les plus urgents, car affectant le plus grand nombre d'usagers au quotidien. Nous pouvons utiliser un processus de hiérarchie analytique afin d'attribuer des pondérations aux différents critères utilisés.

En conclusion, nous pouvons voir que la solution d'IBM permet de croiser des données brutes afin d'obtenir des jeux de données avec une plus grande fiabilité. Actuellement, leurs deux produits se sont retrouvés combinés au sein de l'application « ibm cognos analytic ». Cependant, il n'exploite pas à 100 % les rapports utilisateurs qui ne sont que très peu contrôlés.

1.4.2. mPASS

mPass (mobile Pervasive Accessibility Social Sensing) est un système de collecte de données de l'environnement urbain. Il va se baser sur deux moyens de collecte.

Le premier est la récolte de données lorsque l'utilisateur se déplace dans l'environnement urbain. Elle ne demande aucune interaction. Cette méthode correspond donc à l'application breadcrumb de IBM, c'est pourquoi elle ne sera pas plus détaillée. Cette méthode va générer des S-reports (sensing report).

La seconde est la récolte de données à l'aide d'un formulaire rempli par l'utilisateur. Elle est semblable à l'application Citizen Sensing d'IBM cependant elle sera divisée en deux groupes d'utilisateurs. Des utilisateurs lambdas qui vont générer des U-reports (user report) et des administrateurs qui vont créer des A-reports (administrator report). Ces administrateurs sont des personnes travaillant dans des organisations qui visent à contrôler l'architecture urbaine. Ils seront plus fiables que des novices et pourront donc corriger leur rapport. Il y aura 2 moyens de collecter les données. Soit l'utilisateur ou l'administrateur va prendre l'initiative de lui-même afin de collecter un point, soit l'application va notifier l'utilisateur afin que ce dernier confirme ou corrige un point.

La correction va s'effectuer de la manière suivante. Un rapport généré automatiquement sera corrigé par un utilisateur du groupe U-reporter et un rapport d'utilisateur sera corrigé

par un administrateur. Il n'y a cependant aucun mélange de données entre les groupes. En effet, lors du calcul du chemin le plus court, l'application va simplement prendre le rapport dans la base de données le plus élevée. Dans le tableau, nous pouvons voir un récapitulatif des moyens utiliser afin d'obtenir des données.

Nom du collecteur	Senseur (rapport automatique)	Utilisateur lambda (correction et rapport)	Administrateur (correction et rapport)
Illustration des moyens			
Importance pour le système	+	++	+++

Tableau 3 : Récapitulatif de la logique mPASS

En conclusion, mPASS va se focaliser sur l'interaction avec les utilisateurs en leur demandant de corriger des rapports. Il s'agit donc ici de rétroaction dont nous avons parlé plus tôt dans ce document. Cependant, il n'y a aucun croisement des données afin d'améliorer la logique du système. Le seul point positif lors de leur synthèse est d'avoir fait une façade afin de se greffer à l'application Foursquare qui est plus largement répandue.

1.4.3. Recherche préliminaire de la HES-SO

Lors d'une précédente recherche sur le Crowdsourcing, la Haute Ecole Spécialisé de Suisse Occidentale (HES-SO) a publié un article sur la motivation et la validation des données. Le point qui nous intéresse est la proposition de validation de données. Dans cet article, il n'y a aucune trace de rapport automatique, car tous se basent sur l'interaction humaine.

En effet, chaque utilisateur doit avoir un compte au sein de l'application afin de pouvoir l'utiliser. Lors de l'inscription, ce dernier va devoir passer un test avec des données déjà validées au préalable. Cela va permettre de lui attribuer une note de base. Cette phase se nomme le test de qualification. Il permet d'écarter dès l'inscription les mauvais collecteurs qui

viendraient polluer la base de données. Si l'utilisateur n'est pas qualifié, il peut retenter le test jusqu'à un maximum de trois fois, à la suite duquel il se fera définitivement exclure de l'application.

Une fois inscrit, il devra ajouter des points ou valider des points tout comme dans l'application mPASS. Lorsque le système juge que le rapport est bon et prêt à être utilisé, il va recalculer la note de chaque personne ayant participé à la collecte de ce point. En fonction des réponses de l'utilisateur et de la valeur acceptée par le système, la note donnée au collecteur va augmenter ou diminuer. Cette phase s'appelle le système de réputation.

Afin de valider un point, le système va tenir compte de la note courante de chaque participant. Si nous fixons la limite à 5 utilisateurs afin de valider le point, l'application va additionner la note des personnes pour et la note des personnes contre. Le total le plus haut l'emportera et sera donc jugé comme accepté par le logiciel. Cela s'appelle la phase d'agrégation.

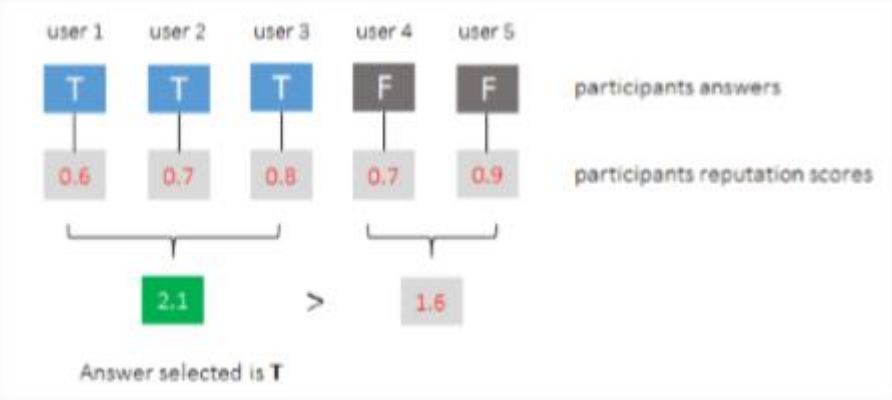


Figure 12 : Système d'agrégation
Liu, Z., Shabani, S., Balet, N. G., Sokhn, M., & Cretton, F. p.5

En conclusion, le système proposé corrige les données grâce à un système de votation de masse. Il n'y a cependant pas d'analyse des données brutes afin de sortir des paternes et ainsi effectuer une classification.

1.4.4. Synthèse des moyens de validation

Après cette explication de trois moyens théorique de validation, il nous est possible de faire un tableau récapitulatif afin de clairement comparer les aspects positifs et négatifs.

	IBM	mPASS	HES-SO
Senseur automatique	Oui	Oui	Non
Exploitation des rapports utilisateurs	Oui	Oui	Oui
Séparation des utilisateurs	Non	Oui	Oui (grâce au test d'entrée)
Notation des utilisateurs	Non	Non	Oui
Correction des données brutes	Oui	Oui	Non (la majorité l'emporte)
Croisement des données (système d'apprentissage)	Oui	Non	Non
Rétroaction avec les utilisateurs	Non	Oui	Oui

Tableau 4 : Comparatif validation donnée

Nous pouvons voir que IBM a essayé d'automatiser au maximum le processus en utilisant de la classification et en proposant de nouvelles méthodes d'analyse afin d'enrichir son système. C'est à cause de ce fait que l'analyse portée sur l'utilisateur est assez faible.

Le système mPass fait exactement l'inverse. Il se concentre sur les rapports collectés par les utilisateurs et compte sur eux pour améliorer l'application. Il n'analyse pas les données et son système parfait serait le cas où tous les rapports sont des rapports d'administrateur qui restent incontestables au sein de l'application. Pour cette application, la création d'une application mobile est intéressante, car elle permet un contact facilité avec l'utilisateur lorsqu'il se situe à proximité d'un point.

Le rapport de la HES-SO propose quant à lui un moyen de valider les données entre utilisateurs, sans que le système analyse quoi que ce soit. Dès que le nombre de participants est atteint, il va simplement calculer la note et enregistrer le point. Il est donc possible que

des erreurs viennent polluer la base de données. Cela dépendra de la limite du nombre d'utilisateurs fixée. Après une petite recherche, nous avons vu que ce système avait été proposé pour l'application We-Map. Cela explique pourquoi il n'y pas de notion de senseur automatique, car une application web ne peut pas collecter ce genre de données. De plus, la rétroaction avec l'utilisateur se complique, car il n'y pas de possibilités d'envoyer des notifications à l'utilisateur. De plus, nous ne pouvons pas savoir où il se situe actuellement.

La meilleure approche théorique serait donc un système indépendant des utilisateurs ce qui éviterait le problème de la motivation chez les participants. Il est donc important de trouver un moyen de classifier les points collectés par des senseurs.

2. Applications utilisant le Crowdsourcing

2.1. TripAdvisor

2.1.1. Descriptif de l'application

TripAdvisor est une application web basée sur l'idée que les voyageurs planifient leur voyage en fonction d'autres commentaires de voyageurs. Elle a été fondée en 2000. Actuellement, TripAdvisor contient 10 millions de rapports de voyage et 5 millions de membres enregistrés. Elle compte 25 millions de visiteurs par mois (J. Miguéns, R. Baggio, and C. Costa ; p.2). Cette plateforme est déclinée sous forme de site web et d'application mobile. L'évaluation d'un lieu par un utilisateur se fait sous forme de note, en ajoutant du contenu multimédia et en écrivant une critique en texte.

2.1.2. Moyen de motivation

TripAdvisor ayant une grande influence sur le marché touristique, tous les restaurateurs/hôteliers voulant avoir une visibilité se doivent de référencer leur établissement. En ce qui concerne la motivation afin que les utilisateurs valident les données, il s'agit de l'« integrated motivation ». En effet, le but commun des voyageurs est d'avoir un maximum d'informations sur un lieu avant de s'y rendre. Ils vont donc tout simplement collecter des informations gratuitement après chaque repas dans un restaurant pour TripAdvisor. Un aspect social intéressant est que l'être humain va préférer partager ces mauvaises expériences plutôt que ces bonnes expériences (marketing charts).

32%	of consumers give feedback directly to companies following a very bad experience, compared to 23% who do so following a very good experience
60%	of 25-34-year-olds tell a friend directly following a bad experience, while 31% share it on Facebook and 20% write a review
57%	of 25-34-year-olds tell a friend directly following a good experience, while 28% share it on Facebook and 18% write a review

Figure 13 : comparatif des expériences partagées
https://www.marketingcharts.com/wp-content/uploads/2014/02/TemkinGroup-Consumer-Reactions-to-Service-Experiences-Feb2014.png

La collecte des rapports négatifs au sein de TripAdvisor est donc plus élevée que la collecte des rapports positifs. La motivation est donc créée ici par le vecteur de la déception procurée par un établissement.

Afin de garder sa communauté active, l'application va envoyer des notifications push basées sur la géolocalisation de l'utilisateur. Le fait que l'application rappelle sa présence va favoriser l'engagement de l'utilisateur afin d'obtenir de plus en plus d'avis, cela dans le but de créer une habitude chez ce dernier dès lors qu'il fréquentera un restaurant ou un hôtel.

2.1.3. Validation des données

Afin de classer les lieux référencés au sein de TripAdvisor, le système va se baser sur 3 critères clés : les notes fournies par les utilisateurs, la date de publication et la quantité de notes collectées. Grâce à ces 3 facteurs, le système peut facilement mettre en avant les meilleurs hôtels. De plus, l'algorithme va également calculer le classement en se basant sur les notes des lieux de la même région. Cette mise à jour de l'algorithme de classement a eu lieu entre 2016 et 2018 et elle apporte un comparatif entre la performance réelle des hôtels. Il n'y a cependant aucun endroit où il est notion d'utiliser la note que l'utilisateur a afin de lui donner plus de poids lors du calcul de l'algorithme.

2.2. AirBnB

2.2.1. Descriptif de l'application

AirBnB est un marché en ligne populaire pour la location à court terme. Il a été fondé en 2008 et a rapidement gagné en popularité pour arriver en 2013 à 535'000 chambres dans le monde entier (Benjamin Edelman Michael Luca ; p 4). Il se décline également sous forme de site web et d'application mobile.

2.2.2. Moyen de motivation

AirBnB offre aux propriétaires un large public pour louer leur bien. Il propose aux voyageurs un comparatif efficace avant de louer une maison/chambre, cela grâce à l'avis des anciens

voyageurs. Il s'agit ici de la même source de motivation que TripAdvisor à savoir l'« integrated motivation ». La technique est exactement la même que TripAdvisor : les personnes vont collecter des données gratuitement, surtout les mauvaises expériences.

Un système de rétroaction a été mis en place grâce à des notifications push. En effet, dès que votre séjour est terminé, l'application va vous envoyer un rappel afin d'évaluer le bien loué. Cela va permettre de prendre directement l'avis de l'utilisateur et donc d'améliorer leur base de données.

De plus, afin de tenir leur communauté active, AirBnB a lancé en 2013 une campagne publicitaire dans laquelle les utilisateurs devaient en 15 secondes présenter leur ville. Cela dans le but de motiver les futurs voyageurs à venir dans cette région (https://www.mainstreethost.com/blog/four-examples-of-clever-crowdsourcing-campaigns/).

2.2.3. Validation des données

Le système d'évaluation de AirBnB se compose de plusieurs catégories. Premièrement, nous avons une liste de critères prédéfinis à noter sur 5 étoiles. Puis l'utilisateur va pouvoir ajouter un commentaire libre afin d'être plus clair sur les aspects positifs et négatifs de sa location. La figure ci-contre représente un exemple de synthèse des notes et des commentaires utilisateurs.

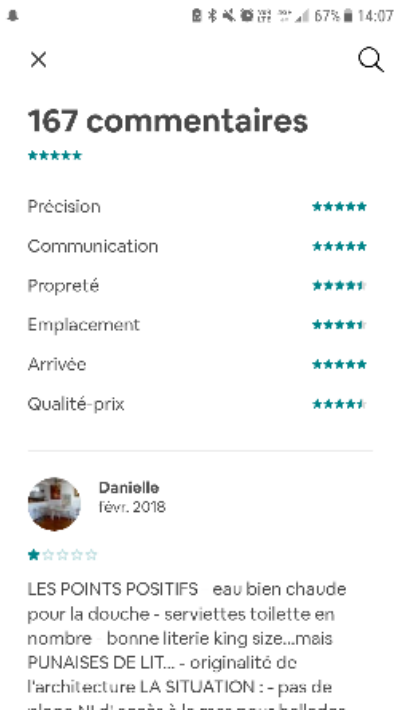


Figure 14 : Exemple de commentaire sur airbnb Application AirBnB

Le système va également utiliser un algorithme tenu secret afin de classer les différentes locations. Nous n'avons à dispositions que quelques bribes d'informations. Nous savons que le classement n'est déjà pas le même pour chaque utilisateur, car ce dernier s'effectue pour chacun en fonction de ses préférences et de ses anciens voyages. Puis, les détails de l'annonce ont également une influence. Nous savons que le prix, le nombre d'étoiles, si la réservation instantanée est activée ou non et la rapidité des réponses auront une influence sur l'ordre. Cette partie correspond à l'implication de l'utilisateur dans l'application. Finalement, les détails du voyage, comme la durée, l'imminence, rentrent également dans l'algorithme de classification (https://fr.airbnb.ch/help/article/39/what-factors-determine-how-my-listing-appears-in-search-results).

2.3. Wheelmap

2.3.1. Descriptif de l'application

Wheelmap est une application pour des personnes en fauteuils roulants ou à déambulateur. Elle permet de répondre aux questions suivantes : « Quelles stations de métro sont équipées d'un ascenseur ? Est-ce qu'il y a des restaurants, cafés ou clubs qui sont accessibles pour vous ? Quels musées ou bibliothèques sont accessibles pour des gens limités dans leur mobilité et quels endroits ne sont pas accessibles ? » (https://play.google.com/store/apps/details?id=org.wheelmap.android.online&hl=fr_CH).

Elle permet de rajouter des points sur une carte virtuelle et d'obtenir des informations sur des localisations existantes. Afin d'ajouter des informations, les utilisateurs doivent être connectés. Le processus de connexion est assez complexe, car il faut un compte Wheelmap et un compte OpenStreetMap et lier ces deux comptes.

Au 29 mai 2018, l'application comptait 877'978 points recensés, principalement situés en Europe. Sur leur site officiel, nous pouvons également voir qu'ils ont un partenariat officiel avec *Immoblien Scout24* (https://wheelmap.org/fr/map#/).

2.3.2. Moyen de motivation

Afin de rendre l'application le plus disponible possible, elle a été développée sous forme d'application web et d'application Android/iOS. Leur application se base sur OpenStreetMap ce qui leur permet d'avoir une fonction qui va lister les points à proximité. Cette technique est utile, car un utilisateur qui connait bien les points environnants pourra les renseigner sans même s'y rendre. Cela permettra aussi d'orienter les personnes perdues sur l'application. Cependant, cette méthode a un coût, car lorsque le point n'existe pas sur OpenStreetMap, il va falloir le renseigner manuellement sur les deux sites, ce qui demande un effort plus important. Si nous prenons le nombre de points sur Sierre, nous pouvons voir qu'il est assez élevé.

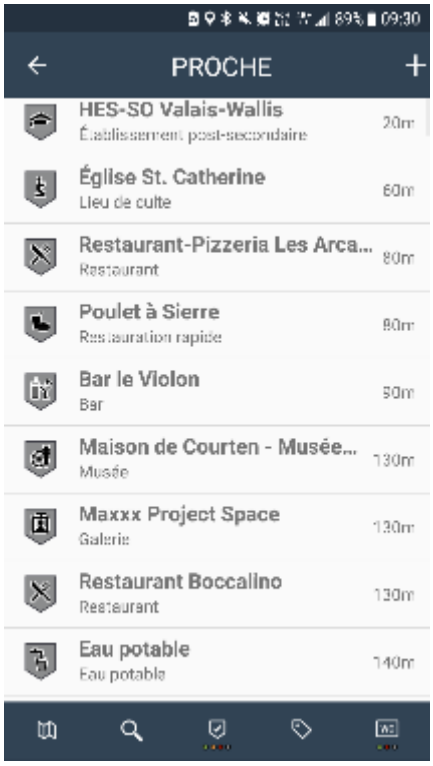


Figure 15 : Wheelmap points à proximité
Application Wheelmap

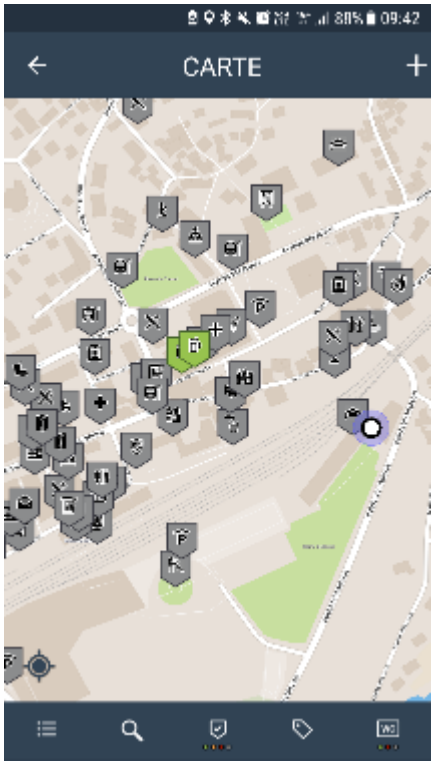


Figure 16 : Wheelmap points présents sur Sierre
Application Wheelmap

D'après nos recherches, il n'y a pas eu de moyen marketing mis en place pour motiver les gens à utiliser l'application. La majorité des gens seront motivés par l'altruisme. Il s'agit donc d'une « intrinsic motivation ».

2.3.3. Validation des données

En ce qui concerne la validation des données au sein de l'application Wheelmap, nous n'avons trouvé aucune forme d'algorithmes de validation. En effet, nous avons donné des informations sur un point (Le soho lounge bar) sur le site, puis nous avons utilisé l'application mobile. Nous avons pu constater que le point est apparu directement en vert. Il convient de dire ici que nous n'avons pas de compte sur ce site ; il en résulte qu'aucun lien n'existe donc entre l'application web et la mobile.

Il nous est également possible de modifier les données déjà collectées, mais les données sont aussi mises à jour directement au sein des différentes applications. Nous sommes donc sûrs qu'il n'y a absolument aucun moyen de validation des données.

2.4. Route4u

2.4.1. Descriptif de l'application

Route4u est une initiative hongroise visant à améliorer la mobilité des personnes invalides au sein des villes. Elle se compose de 3 parties. Premièrement, nous pouvons donner des informations sur des lieux se trouvant dans l'application tels que la police municipale de Sierre. Nous pouvons renseigner 3 champs : l'accessibilité, l'accessibilité des toilettes et un champ pour fournir des détails. Deuxièmement, en gardant le doigt appuyé sur un endroit sur la carte, il est possible de signaler un obstacle urbain ou un chemin accessible ainsi que de prendre une photo. Ici, nous pouvons rappeler qu'un utilisateur va signaler majoritairement les problèmes et non pas les chemins accessibles. Finalement, il est possible d'activer une fonction qui va vous localiser et analyser vos déplacements. Nous pouvons comparer cela à l'application Breadcrumb d'IBM.

2.4.2. Moyen de motivation

Cette initiative est liée à un plan de développement soutenu par l'Union européenne qui a pour but de restaurer la croissance économique malgré le vieillissement de la société. Cela se

base donc sur un sentiment nationalisme hongrois pour motiver la population locale à collecter des données. L'altruisme est aussi un facteur clé de motivation.

Un système de rétroaction sous forme de notifications push existe dans l'application. Elles ne viennent que lorsque nous nous sommes connectés grâce à Facebook. Une fois connecté, le nombre de notifications est en revanche très élevé. Cela a un effet clairement négatif, car l'application s'immisce dans votre vie sans vous avertir. Elle vous remercie pour chaque action effectuée.

Route4U bénéficie aussi d'un fond afin de développer leur système avec une vraie équipe qui travaille sur le projet. De plus, afin de se faire connaitre et prouver leur sérieux, ils affichent leurs partenaires sur leur site. Ces derniers sont nombreux et d'une certaine notoriété. Nous pouvons citer Heineken ou Sony comme exemple.

Sur cette page, afin de susciter l'intérêt des entreprises à rejoindre le projet, il leur est demandé d'apporter une contribution comme des coupons, des rabais, etc. Cela voudrait dire que Route4U devrait récompenser les meilleurs collecteurs. En effet, lors de votre première connexion, vous aurez un score égal à zéro et qui augmentera en collectant des points. Par exemple, lorsque vous atteignez 2980 points, vous allez recevoir un décapsuleur Heineken. Il y a donc une motivation pécuniaire. Cependant, sur leur site, il est question de ludification en se basant sur un système de points et de badges. Cela représente plutôt un système de récompense.

Un point faible de l'application est sa prise en main qui n'est pas évidente lors de la première approche. Afin de signaler un obstacle, il faut tenir le doigt appuyé sur la carte afin d'ouvrir une fenêtre de dialogue.

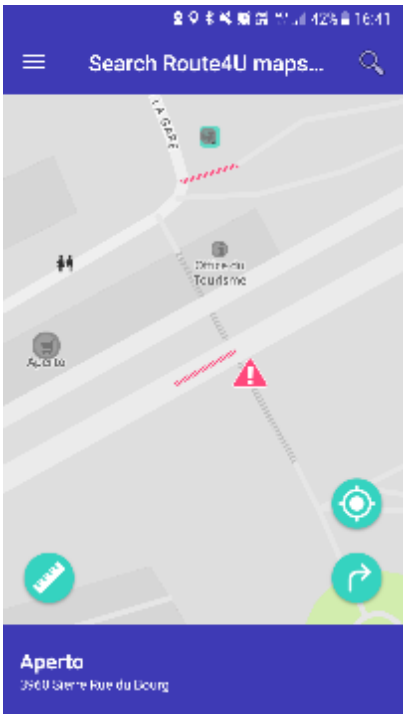


Figure 17 : Interface de Route4U
Application Route4U

2.4.3. Validation des données

Concernant la fonction de signalisation d'obstacle d'urbain, il n'y a aucun système de validation des données, car lorsque nous rajoutons un point sensible dans l'application, elle se met à jour directement.

Lorsque nous changeons des informations sur un point déjà défini sur la carte, une pop-up nous informe que notre changement sera visible dans la prochaine heure. Il est possible que cela montre la présence d'un système de vérification, mais nous pensons qu'il s'agit plutôt du temps qu'il faut pour que la synchronisation entre tous les systèmes se fasse. Cette déduction vient du fait que l'application n'a pas de données avec lesquelles croiser notre rapport et que, de ce fait, il faudrait plus de temps pour vérifier les nouvelles données.

Pour la dernière fonctionnalité, le tracking automatique, nous n'avons aucun retour sur comment le système va utiliser les rapports collectés et le code source de l'application n'est pas disponible. Nous nous retrouvons donc face à une impasse pour l'analyse automatique.

Finalement, nous voyons que des points existent déjà à Sierre. Nous savons donc que l'application va chercher des données sur un autre système, mais une fois de plus, nous ne savons toujours pas de quel outil il s'agit, car il n'y a aucune référence sur l'algorithme ou sur le fonctionnement interne du programme.

2.5. Produits Google

2.5.1. Descriptif de l'application

En mars 2018, Google a publié une annonce sur son blog pour annoncer leur nouvelle fonctionnalité sur leur outil Google Maps. Il s'agit d'une option disponible dans l'application mobile afin de montrer l'accessibilité lors d'un itinéraire à parcourir. Il revendique l'envie de rendre le monde plus accessible à tout le monde (https://www.blog.google/products/maps/introducing-wheelchair-accessible-routes-transit-navigation/). Il est également important de souligner le fait que, même si cette fonctionnalité est disponible dans l'application, elle n'est pas encore disponible sur l'api officiel de google maps.

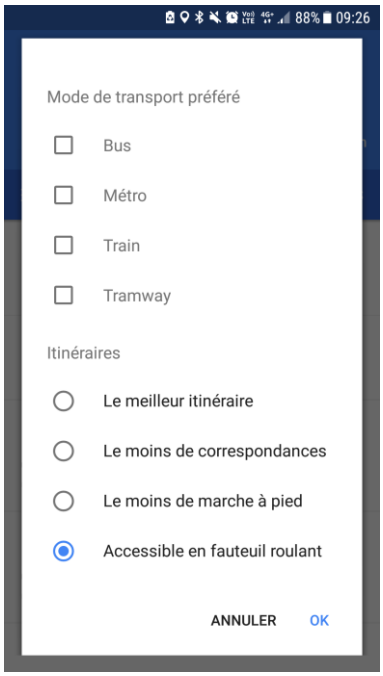


Figure 18 : Nouvelle fonctionnalité Google
Application Google Crowdsource

De plus, Google utilise une autre application appelée Crowdsource afin d'aider leur système à progresser. Elle vous propose de répondre à des questions simples afin d'améliorer leurs base de données.

2.5.2. Moyen de motivation

Il n'y a pas de tâches sur l'application Crowdsource qui permettrait d'ajouter des données sur l'accessibilité d'un lieu, donc nous allons voir comment Google motive les gens afin de remplir des tâches autres. Leur système se base sur un système de niveau et de badges collectés. Ces badges n'apportent rien de concret si ce n'est une simple satisfaction personnelle. Leur application n'a pas de rétroaction avec l'utilisateur et il n'y pas de publicité sur leurs autres applications pour améliorer leur jeu de données.

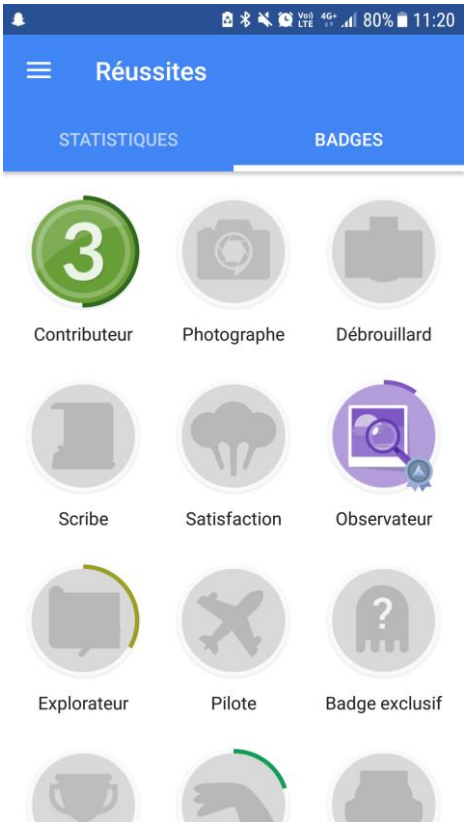


Figure 19 : Système de récompense Google Application Google Crowdsource

2.5.3. Validation des données

Le code source n'étant pas disponible, il est difficile de savoir comment Google va calculer un chemin pour une accessibilité en chaise roulante. Nous n'avons trouvé aucune personne expliquant le backend de cette fonctionnalité. Le problème se pose également avec leur application de Crowdsourcing, car chaque donnée récoltée sera exploitée de manière unique dans les différents services Google.

2.6. Synthèse des applications

Nous avons pris 2 applications très connues sur le marché qui utilisent le Crowdsourcing comme vecteur clé puis 3 applications similaires à la nôtre afin de ressortir les 2 facteurs clés : la popularité et les cartes interactives. Nous allons commencer par synthétiser les moyens de motiver les gens à utiliser l'application.

	TripAdvisor	AirBnB	Wheelmap	Route4U	Produits Google
Spot publicitaire	Oui	Oui	Non	Non	Non
Classement des utilisateurs	Oui	Non	Non	Oui	Oui
Récompense matérielle	Non	Non	Non	Oui	Non
Ludification	Non	Non	Non	Non	Non
Interface facile à prendre en main	Oui	Oui	Oui	Non	Oui
Rétroaction	Oui	Oui	Non	Oui	Non

Tableau 5 : Comparatif de la motivation dans les applications

Nous pouvons voir que même si le processus de ludification est le plus rentable théoriquement, dans la pratique il n'est jamais implémenté. Nous pensons que cela vient du fait que les applications veulent paraître professionnelles ce qui n'est pas forcément le cas

lorsque notre produit est un jeu. La récompense matérielle n'est que très peu utilisée sur le marché du Crowdsourcing. Cependant le système de classement est une bonne alternative afin de motiver les gens à collecter des données. Puis, il faut relever que la rétroaction, afin de garder la communauté active, est un point clé de la réussite. Il faut cependant faire attention à ne pas en abuser comme à tendance à faire Route4U. La publicité est réservée aux grandes entreprises qui ont déjà réussi dans le marché. L'élément le moins surprenant après cette analyse est l'interface qui doit être intuitive à l'utilisation. En effet, si l'utilisateur ne comprend pas comment marche l'interface, il va rapidement se désintéresser ou alors ne pas exploiter l'application à cent pour cent.

Passons maintenant au récapitulatif pour la validation des données lors du processus de Crowdsourcing.

	TripAdvisor	**AirBnB**	**Wheelmap**	**Route4U**	**Produits Google**
Utilise un algorithme	Oui	Oui	Non	Non	Sûrement
Correction des données	Non	Non	Oui	Oui	Non
Voir l'avis des autres utilisateurs	Oui	Oui	Oui	Oui	Non
Donne un avis global (fait une synthèse)	Oui	Oui	Non	Non	Oui

Tableau 6 : Comparatif de la validation dans les applications

Nous pouvons sortir 2 catégories d'application. La première va prendre les données collectées et les passer dans un algorithme afin d'obtenir un résultat final. La seconde va simplement montrer les données sous leur forme brute. TripAdvisor et AirBnB font partie des 2 catégories, car elles utilisent un algorithme pour classer leur donnée, mais elles affichent quand même certains renseignements bruts comme les commentaires utilisateurs. Wheelmap et Route4U qui sont les applications les plus similaires à la nôtre, proposent simplement l'affichage des points sans aucun traitement. Nous pensons que cela vient du fait

que le nombre de points renseignés est trop faible pour appliquer un quelconque algorithme. Finalement, les produits Google vont faire partie de la première catégorie, car ils n'affichent que des résultats dans leurs applications.

La correction en direct peut poser un réel problème, car si nous lançons un robot qui va modifier les points automatiquement avec des renseignements n'ayant aucune valeur, l'application va les assimiler et mettre à jour la carte. La correction va simplement écraser les anciens points et les remplacer par les nôtres.

3. Synthèse de la théorie et de la pratique

3.1. Résultats

La première conclusion que nous pouvons tirer est qu'il n'est clairement pas rentable de récompenser les utilisateurs par des biens matériels, sauf si un soutien par une grande institution est reçu. Il a été également prouvé théoriquement que cette approche n'apporte pas un grand taux de motivation. Cependant, un classement au sein de l'application peut amener un petit taux de participation supplémentaire. Ce classement peut être vu comme une sorte de ludification, car elle transforme l'application en jeu pour être le premier au classement. Nous voyons cela plutôt comme un système de récompense, car les utilisateurs pourront gagner des points et obtenir des badges lorsqu'ils arriveront à certains paliers.

La ludification, qui est un des processus les plus rentables théoriquement, n'est jamais implémentée dans les applications pratiques. Il convient donc de ne pas tenir compte de cet aspect pour motiver les gens.

Un des points le plus importants afin d'attirer les utilisateurs est la rétroaction sous forme de notification push sur les smartphones. En effet, nous recevons tellement d'emails en 2018 qu'une rétroaction sous cette forme n'aurait pas un bon retour. Une bonne utilisation des notifications permet de rappeler à l'utilisateur que notre application est toujours sur son smartphone et que cela lui prendrait que quelques secondes afin de collecter un point.

Finalement, il n'existe pas de recette miracle afin d'attirer diverses personnes à utiliser une application. Il faut simplement du bon sens lors du développement, car un utilisateur sera feignant par nature. Si ces derniers ne se sentent pas concernés par la cause que vise le système, il sera très dur de les attirer.

Pour contrer ce souci, il faudrait créer un système qui ne demande que très peu d'interaction avec l'humain. Une application de type Breadcrumb d'IBM est l'exemple parfait, car il suffit de se déplacer avec son portable dans sa poche et la collecte se fait automatiquement. Cependant, cela a un coût lors de développement, car il faut trouver un algorithme qui s'adapte parfaitement à nos données pour résoudre notre problème. Ici, nous rentrons dans un processus appelé machine learning où l'algorithme s'améliore au fur et à mesure qu'il obtient des données.

Utiliser un simple algorithme de vérification des données ajoutées par les utilisateurs comme proposé par l'institut de recherche de la HES-SO permet d'améliorer la validité de la base de données ainsi que la sécurité de notre application. Cependant, cette méthode a un contre coup si le nombre de points est assez faible comme c'est le cas dans We-Map. En effet, si le nombre de points collectés n'est pas assez élevé, il faudra beaucoup de temps avant qu'un seul renseignement soit validé. C'est pourquoi Wheelmap et Route4U n'utilisent pas d'algorithme de vérification pour le moment.

Se concentrer sur une forme de forum où les gens peuvent voter pour la meilleure réponse peut être une bonne alternative, car cela montre à l'utilisateur ce que pense le plus grand nombre. Malheureusement, ce système ne peut pas vraiment s'appliquer à une carte interactive où les informations doivent être condensées.

3.2. Choix de l'implémentation

L'application We-Map est pour l'instant un simple site web réalisé en PHP. Cela va poser plusieurs problèmes majeurs. Premièrement, il nous est impossible de savoir où se situe l'utilisateur, car la géolocalisation n'est accessible que lorsque l'utilisateur a son mobile déverrouillé et ouvert sur son navigateur. Il faudrait donc développer une application Android

afin d'avoir accès à la géolocalisation à tout instant pour obtenir des rapports automatiques à la manière de Breadcrumb. Puis, il faudrait implémenter un algorithme de type colonie de fourmis et faire des recherches dessus. Il est clairement impossible de réaliser toutes ces étapes en 4 semaines de développement prévu initialement.

Deuxièmement, il est impossible avec le système actuel d'envoyer des notifications dans le but de motiver l'utilisateur. En effet, les applications web n'ont pas accès au système du téléphone.

Il nous serait possible d'ajouter un système d'algorithme simple pour valider les données, mais comme expliquer plus haut, cela risque de nuire à l'application, car elle a un rendement beaucoup trop faible pour le moment.

Nous pouvons proposer de créer l'application Android We-Map et de modifier la base de données afin de préparer l'algorithme de validation des données. De plus, la collecte de rapports automatiques pourra être mis en place. Cela permettrait d'être présent sur différents aspects intéressants : ajouter une source de collecte, motiver l'utilisateur par un système de rétroaction et préparer une vérification des données.

4. Choix technologiques

4.1. Schéma de l'architecture

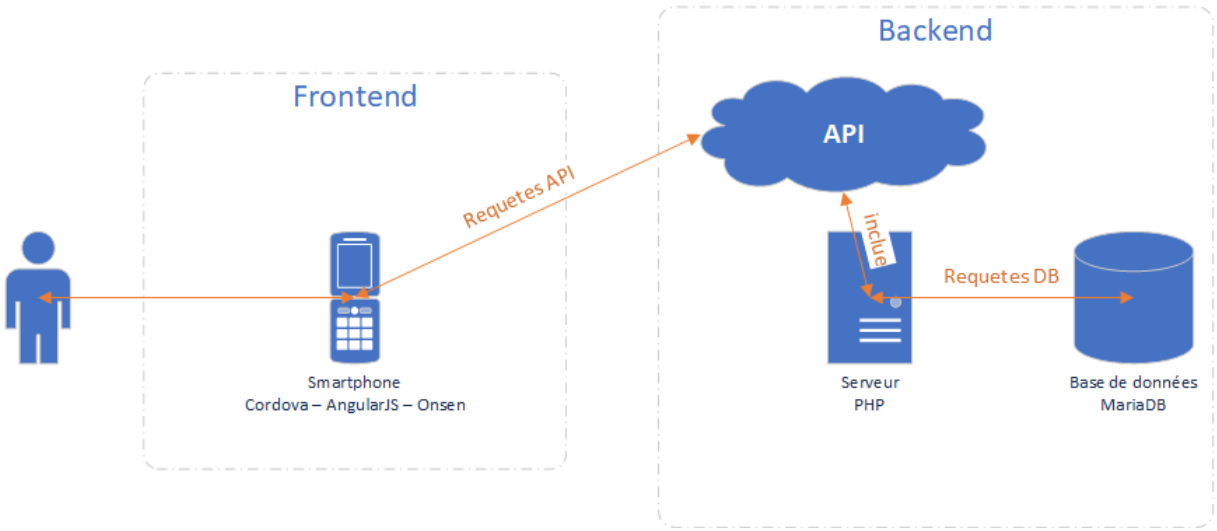


Figure 20 : Illustration architecture
Source personnelle

4.2. Backend

Afin de centraliser nos informations, nous avons une base de données sur un serveur commun. Vu que l'application précédente a été développée sur un serveur PHP avec une database mariadb, nous avons choisi de garder ces deux technologies pour faciliter l'intégration. Le serveur devra posséder une API afin que le frontend puisse effectuer des requêtes sécurisées. Lors de notre première approche, nous avons pu constater que les tables actuelles n'étaient pas suffisantes pour répondre aux besoins de l'application, nous aurons donc besoin de refaire un nouveau schéma.

Afin de ne pas perdre de temps pour développer l'api, nous allons utiliser le logiciel Thirion créé par la société Bétrisey Cobwebsite. Ce dernier va nous permettre, en se basant sur un schéma de base de données, de générer le code de base pour notre api. Son fonctionnement sera expliqué plus tard.

4.3. Logique Frontend

Pour réaliser notre application mobile, plusieurs choix s'offrent à nous. Tout d'abord, il existe deux types d'applications : les hybrides et les natives. L'avantage de l'hybride est que nous allons écrire le code une seule fois puis nous pourrons le compiler pour les différents supports comme Android ou iOS alors qu'en natif, il faudrait développer deux fois l'application. Du fait de notre besoin rapide d'une application fonctionnant sur tous les systèmes, il est normal que nous ayons choisi l'hybride.

Dans les technologies hybrides, nous avons un panel assez divers.

4.3.1. Cordova

Cordova permet aux développeurs de créer des applications mobiles en utilisant CSS3, HTML5 et du code JavaScript. CSS3 est utilisé pour l'interface utilisateur et JavaScript pour la logique de l'application (Stefan Bosnic, Ištvan Papp and Sebastian Novak ; p1).

Son principe est simple, il ouvre simplement un navigateur web qui va afficher les fichiers HTML. Cependant, il n'existe pas de serveur c'est pourquoi, il est impossible d'avoir des fichiers PHP.

4.3.2. Ionic

Ionic est un framework basé sur Cordova et sur AngularJS. Il permet une simplification du code JavaScript et offre des composants déjà prédéfinis.

4.3.3. React Native

React Native permet aux développeurs de créer des applications mobiles en JavaScript seulement. Il se base sur la même logique que React avec un système de composants. L'avantage de React est qu'il va compiler le code pour le faire ressortir en natif. En effet, lorsque vous lancez une application Android réalisée avec React Native, cette dernière aura de vrais composants Android et non pas une simple webview.

4.3.4. Xamarin

Xamarin a les mêmes avantages que React Native. Il crée aussi des applications totalement natives. La seule différence est qu'au lieu d'écrire en JavaScript, il faut utiliser du .NET.

4.3.5. Conclusion

L'application à créer étant assez conséquente, nous avons choisi un langage que nous connaissons bien afin de pouvoir pousser le projet au plus loin. Nous avons donc fait le choix de simplement installer Cordova puis de le coupler nous même à AngularJS 1 sans passer par Ionic. Cela va nous permettre d'être plus libres qu'avec le framework Ionic. De plus, grâce à certains précédents mandats, nous avons déjà plusieurs morceaux de code réutilisables.

L'utilisation de AngularJS 1 n'est pas obligatoire, mais cela permettra d'avoir une structure de code plus facile à relire pour de futures retouches.

4.4. User Interface

Pour la partie que l'utilisateur va voir, il nous faudra faire un design simple et comparable à du natif. Lors de nos précédents projets au sein de la HES-SO, nous avons souvent utilisé la bibliothèque CSS materialize qui respecte les normes material design et qui fournit un joli rendu mobile. Cependant, nous avons choisi de prendre une bibliothèque que nous ne connaissions pas afin d'en tirer de nouvelles connaissances pour le futur.

Notre choix s'est arrêté sur la librairie Onsen UI qui a été créée spécialement pour les applications hybrides HTML5. Elle a pour avantage de changer son design en fonction de si nous utilisons un iPhone ou un Android comme le montre la figure 21. Il est important de préciser qu'il s'agit du même code pour les deux images.

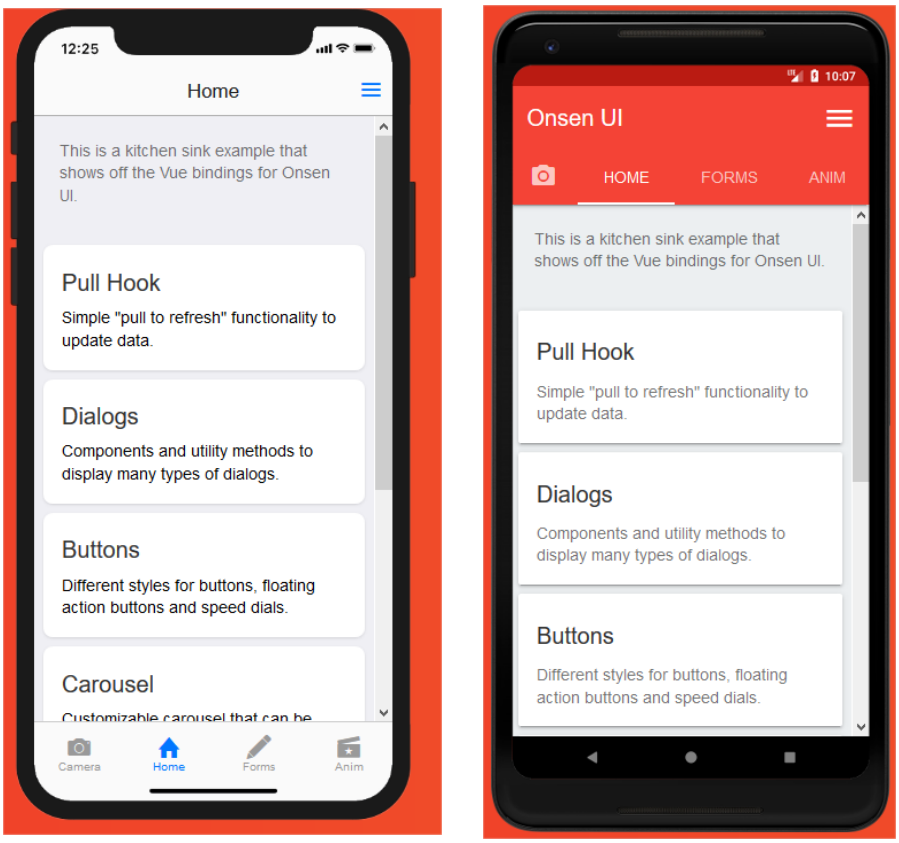


Figure 21 : Exemple Onsen UI
https://onsen.io/

5. Étape de développement

5.1. Mise en place de l'espace de travail

Afin de correctement mener à terme ce projet, nous avons dû installer certains programmes. Premièrement, nous avons dû installer un IDE. Nous avons choisi visual studio code qui est un IDE gratuit sur lequel nous pouvons greffer des plugins pour nous aider.

Ensuite, il nous a fallu installer nodeJS afin de faire tourner certains scripts utiles comme un compteur de ligne. Cela nous a aussi permis d'installer cordova afin d'avoir accès à la ligne de commande. Cela se fait en lançant dans le terminal la ligne suivante :

Npm i cordova -g

Pour les applications Android, il nous faut un jdk que nous devons rajouter au PATH et un SDK Android. Pour le skd, nous avons simplement téléchargé Android Studio puis rajouté les dossiers *tools* et *platform-tools* au PATH. Finalement, afin de compiler notre application, il nous faut avoir accès à apache Ant. Nous avons donc téléchargé ce programme et nous l'avons ajouté au PATH. Avec cela, nous avons tous les outils nécessaires à la réalisation de notre application hybride.

Il faut cependant encore faire tourner un serveur sur notre ordinateur afin d'exécuter PHP. Nous avons donc installé xampp afin d'avoir un serveur web tournant sur le port 80. Xampp installe également MariaDB et fournit une gestion de la database grâce à phpMyadmin. Nous avons donc tous les outils pour notre backend.

Il reste encore à créer un tunnel entre notre ordinateur et notre application mobile afin de réaliser des requêtes api. Nous avons téléchargé le logiciel ngrok afin de pallier à ce problème.

En ce qui concerne la génération de l'api, nous avons installé le programme Thirion directement depuis Github en suivant le ReadMe.

Il nous manque un outil afin de débugger notre application mobile. En effet, il nous faut installer chrome, car il permet d'inspecter des pages web sur un périphérique relié par câble USB en se rendant sur chrome://inspect.

5.2. Design de la base de données

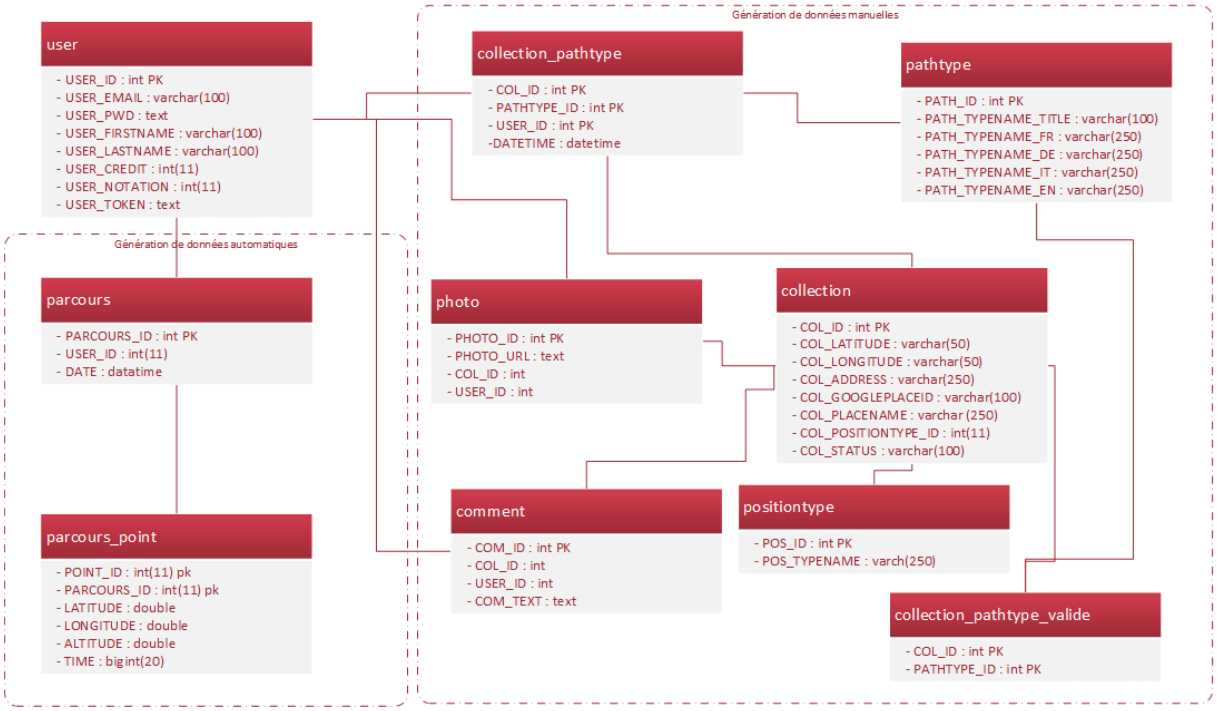


Figure 22 : Schéma de la base de données
Source personnelle

Nous avons dû modifier la base de données afin qu'elle corresponde au mieux aux besoins de l'application. Nous allons décrire ci-dessous les changements majeurs apportés. Premièrement, nous avons rajouté une table user qui va servir à l'authentification. Les champs importants sont *USER_TOKEN,* qui permet de sécuriser les requêtes API et qui doit être unique, *USER_NOTATION,* qui donne une notion de classification entre les bons et les mauvais utilisateurs et *USER_CREDIT,* qui permettra un système de crédit dans le cas où sera implémenté dans le futur.

Nous avons rajouté toute la partie de la base de données gérant la collecte des données automatiques. Nous avons des parcours qui contiennent des points avec la *LATITUDE, LONGITUDE, ALTITUDE et TIME* afin de pouvoir effectuer des calculs.

Nous avons transformé la table collection en sortant, tout d'abord, les photos et les commentaires dans des tables séparées qui se basent sur l'utilisateur. Ensuite, nous avons rajouté un champ *COL_STATUS* afin de savoir si le point a eu assez de données collectées pour être considéré comme valide. Lorsque le point n'est pas encore valide, les dangers (table pathtype) sont stockés dans la table intermédiaire puis, lorsque l'algorithme juge le point valide, il insère les bons dangers dans la table *collection_pathtype_valide.*

5.3. Réalisation de la base de l'api

Vous pouvez retrouver la structure des projets dans l'annexe 3.

5.3.1. Création des fichiers

Pour réaliser une API rapide et efficace en PHP, nous avons choisi d'utiliser le programme Thirion de l'entreprise Bétrisey Cobwebsite. Pour illustrer son fonctionnement, nous allons prendre un exemple d'utilisateur et de photos. Il nous suffit de créer le schéma de la base de données que vous pouvez voir ci-dessous.

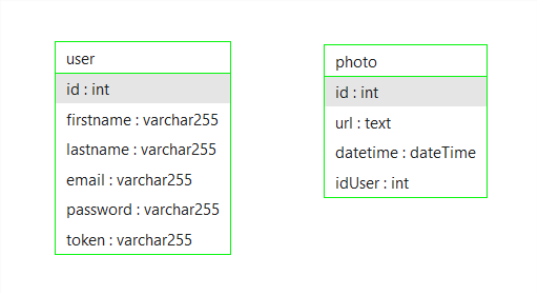


Figure 23 : Thirion schéma de la base de données
Application Thirion

Cela va créer les fichiers *PHP user.class.php* et *photo.class.php* avec les méthodes add, update, delete, getAll et getById comme nous pouvons l'observer avec la figure 24.

user.class.php photo.class.php

```php
<?php
    class photo{
        var $conn;
        public function __construct(){
            $this->conn=Load::load('Connection');
        }
        public function add($url,$datetime,$idUser){
            $conn = $this->conn;
            $sql="INSERT INTO photo(url,datetime,idUser) VALUES (:url,:datetime,:idUser)";
            $stat = $conn->prepare($sql);
            $stat->bindParam(":url",$url);
            $stat->bindParam(":datetime",$datetime);
            $stat->bindParam(":idUser",$idUser);

            $stat->execute();
        }
        public function update($id,$url,$datetime,$idUser){
            $conn = $this->conn;
            $sql="UPDATE photo SET url=:url,datetime=:datetime,idUser=:idUser WHERE id=:id";
            $stat = $conn->prepare($sql);
            $stat->bindParam(":id",$id);
            $stat->bindParam(":url",$url);
            $stat->bindParam(":datetime",$datetime);
            $stat->bindParam(":idUser",$idUser);

            $stat->execute();
        }
        public function delete($id){
            $conn = $this->conn;
            $sql="DELETE FROM photo WHERE id=:id";
            $stat = $conn->prepare($sql);
            $stat->bindParam(":id",$id);

            $stat->execute();
        }
        public function getAll(){
            $conn = $this->conn;
            $sql="SELECT * FROM photo";
            return $conn->query($sql)->fetchAll(PDO::FETCH_ASSOC);
        }
        public function getById($id){
            $conn = $this->conn;
            $sql="SELECT * FROM photo WHERE id=:id";
            $stat = $conn->prepare($sql);
            $stat->bindParam(":id",$id);

            $stat->execute();
            return $stat->fetch(PDO::FETCH_LAZY);
        }
```

Figure 24 : Thirion classe générée
Source personnelle

Maintenant que nos objets existent, nous pouvons configurer notre API. Il suffit de remplir les différents paramètres.

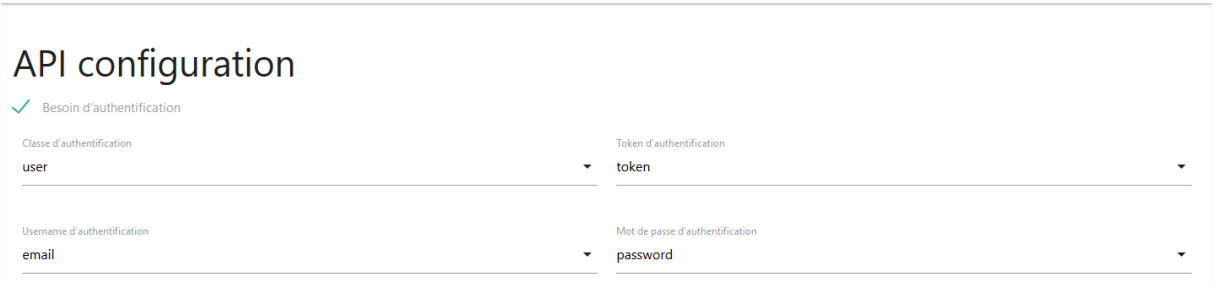


Figure 25 : Thirion API configuration
Application Thirion

Cela va ajouter la couche de sécurité à notre application. Un fichier de login va être généré à la racine du projet afin d'obtenir notre token d'authentification.

apiLogin.php

```php
<?php

    header("Access-Control-Allow-Origin: *");
    header('Access-Control-Allow-Methods: GET, PUT, POST, DELETE, OPTIONS');
    header('Access-Control-Allow-Credentials: true');
    header('Access-Control-Allow-Headers: Content-Type, Content-Range, Content-Disposition, Content-Description');

    require('./php/loader.php');
    $user = Load::load('user');

    echo $user->connect($_POST['email'],$_POST['password']);


?>
```

Figure 26 : Thirion login
Source personnelle

Nous avons donc tous les fichiers utiles pour expliquer le fonctionnement et l'architecture de l'api.

5.3.2. Fonctionnement

La première étape consiste à obtenir son token d'authentification pour pouvoir utiliser l'api. Nous allons utiliser {url} afin de signaler qu'il s'agit de l'url du site. Nous appelons donc l'url {url}/apiLogin.php en POST avec un body contenant email et password. Nous recevons notre token d'authentification. Maintenant, nous pouvons, par exemple, lister toutes les photos.

La seconde étape sera donc de faire un appel POST sur {url}/api/photo/?token ={monToken}. En fait, nous pouvons appeler toutes les méthodes se trouvant dans nos classes avec un appel POST de la manière suivante : {url}/api/{nomClasse}/{nomFonction}?token={monToken} et en donnant le nom des paramètres exactes de la fonction dans le body de notre requête. Le fichier .htaccess va faire une redirection sur index.php en mettant la suite de l'url dans la variable GET request. Grâce à l'autoloader, la classe sera chargée automatiquement.

```php
if($paths[0] == 'user'){
    $temp = $user;
}else{
    $temp = Load::load($paths[0]);
}
```

Figure 27 : Thirion Chargement dynamique de la classe
Source personnelle

Puis, le fichier va analyser la méthode dans la classe et chercher les différents paramètres dans le body du POST.


```php
function loadData($className,$functionName){
    $r = new ReflectionMethod($className, $functionName);
    $params = $r->getParameters();
    $ret = [];
    foreach($params as $param){
        $name = $param->getName();
        $ret[count($ret)] = $_POST[$name];
    }
    return $ret;

}
```

Figure 28 : Thirion chargement des paramètres
Source personnelle

Une fois les paramètres trouvés, le programme va appeler la fonction dans la classe et afficher les résultats au format JSON.

```php
$params = loadData($paths[0],$paths[1]);
$res = call_user_func_array(array($temp,$paths[1]), $params);
```

Figure 29 : Thirion appelle de la méthode
Source personnelle

5.4. Création des fonctionnalités

L'API est en place, nous pouvons commencer à développer l'application mobile.

5.4.1. Connexion

La première étape est la connexion afin de récupérer le token utile pour des informations futures.

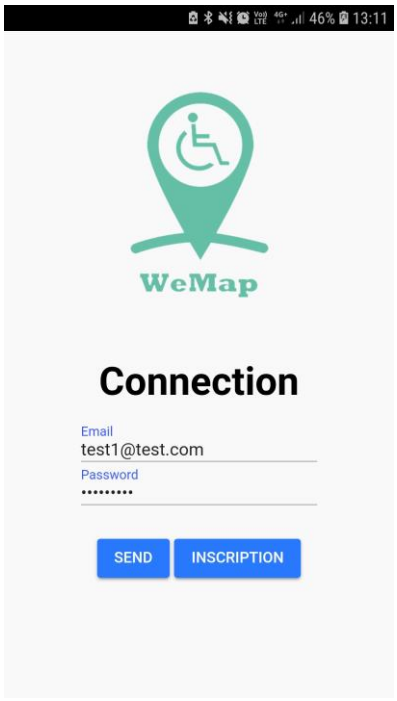


Figure 30 : Ecran de connection
Source personnelle

Nous avons un écran de connexion que nous allons remplir avec notre email et notre mot de passe.

Cette vue est liée au ***connectionCtrl*** et lorsque nous cliquons sur « send » la fonction login sera appelée dans le contrôleur.

```
$scope.login = function () {
    document.querySelector("#ajaxLoader").style.visibility = 'visibl
    document.querySelector("#ajaxLoader").style.opacity = 1;
    $http({
        method: 'POST',
        url: urlSiteToJoin + '/apiLogin.php',
        headers: {
            'Content-Type': 'application/x-www-form-urlencoded'
        },
        transformRequest: ajaxInfo.transform,
        data: $scope.form
    }).then(function (response) {
        ajaxInfo.messageRetour(response, $scope.connectionRes, true)
    });


}
$scope.connectionRes = function (res) {
    var splits = res.split(",");
    tokenConn = splits[0];
    userID = splits[1];
    var userNotation = splits[2];
    parameters["tokenConn"] = tokenConn;
    parameters["userID"] = userID;
    parameters["notation"] = userNotation;
    $scope.loadDefaultData();
    if (userNotation == -1) {
        $scope.launchTestProcess();
    } else {
        var isConn = ajaxInfo.testConn();
        writeParams();
    }
}
```

Figure 31 : Fonction de login
Source personnelle

La méthode appelle notre fichier de Login sur l'api. La factory ***ajaxInfo*** est un middleware qui va gérer l'asynchrone et les erreurs.

Si la fonction réussit, nous sauvegardons le retour dans les ***parameters*** et nous écrivons cette variable dans un fichier sur le téléphone afin d'éviter de devoir nous connecter à chaque fois. Cela est réalisé grâce à la méthode ***writeParams()***.

```
function writeParams() {
    if (logOb == undefined) return;
    logOb.createWriter(function (fileWriter) {
        var temp = JSON.stringify(parameters);
        var blob = new Blob([temp], {
            type: 'text/plain'
        });
        fileWriter.write(blob);
    }, fail);
}
```

Figure 32 : Sauvegarde du token
Source personnelle

Nous écrivons cet objet ***parameters***, transformé en string, dans le fichier ***params.json*** sur le portable.

5.4.2. Navigation

L'étape la plus importante afin d'avoir une bonne expérience utilisateur est d'avoir une navigation facile à prendre en main.

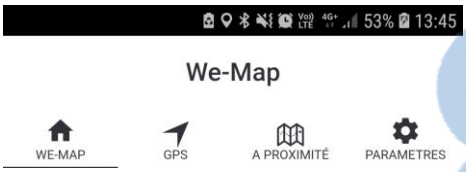


Figure 33 : Navigation
Source personnelle

Nous avons opté pour une navigation par onglets afin de mettre en évidence les fonctionnalités principales de l'application à savoir l'accueil avec la carte, le GPS, les points à proximité et les paramètres.

Cette partie est gérée par le contrôleur appelé ***pageCtrl***. Le code est fourni par la bibliothèque Onsen UI. Il s'agit d'une balise appelée ***ons-navigator*** qui va s'occuper de charger les bonnes pages lors du clic sur l'icône.

```
<ons-navigator id="myNavigator" var="myNavigator">
  <ons-page ng-controller="PageController" id="pageController">
    <ons-toolbar>
      <div class="center">We-Map</div>
    </ons-toolbar>
    <ons-tabbar swipeable position="auto" ons-postchange="launchInit($event)">
      <ons-tab page="partials/carte.html" label="We-Map" icon="ion-home, material:md-home" active></ons-tab>
      <ons-tab page="partials/gps.html" label="GPS" icon="ion-navigate"></ons-tab>
      <ons-tab page="partials/nearby.html" label="a Proximité" icon="ion-map"></ons-tab>
      <ons-tab page="partials/settings.html" label="Parametres" icon="md-settings"></ons-tab>
    </ons-tabbar>
  </ons-page>
</ons-navigator>
```

Figure 34 : Code de navigation
Source personnelle

Ce contrôleur va également se charger d'envoyer les notifications à l'utilisateur tout en vérifiant que ce dernier n'en reçoive pas trop. En effet, le programme va envoyer une

notification si l'utilisateur se trouve à moins de 100 mètres d'un point en attente et s'il n'a pas encore reçu de notification durant les 24 dernières heures.

Du plus, du fait que ce contrôleur soit toujours disponible, car la navigation ne disparait jamais, il possède la fonction qui envoie les parcours collectés automatiquement au serveur.

5.4.3. Carte

Lors du précédent Travail de Bachelor, la navigation avait été implémentée grâce à l'api de Google. Nous avons donc cherché un plugin cordova afin d'intégrer une maps en natif. Nous avons trouvé le plugin appelé *cordova-plugin-googlemaps* sur Github. L'installation d'un plugin est très facile, car il suffit de taper la commande :

cordova plugin add {nomPlugin}

Une fois le plugin installé, il est disponible dans le code sous la variable ***plugin***. Nous avons donc mis la carte en page d'accueil. Elle dépend du contrôleur ***carteCtrl*** et des scripts se trouvant dans le fichier ***map.js***. Lorsque notre smartphone est prêt, nous initialisons la map avec le code ci-dessous.

```
document.addEventListener("deviceready", function () {

    var div = document.getElementById("map_canvas");
    map = plugin.google.maps.Map.getMap(div);
    map.setOptions({
        styles: [{
            "featureType": "poi",
            "stylers": [{
                "visibility": "off"
            }]
        }]
    });
```

Figure 35 : Initialisation de la map
Source personnelle

Lorsque la carte est prête, nous lançons un script de localisation afin de centrer la carte sur la position de l'utilisateur. Dans le cas où l'utilisateur a activé le tracking automatique, nous commençons aussi à le géolocaliser toutes les 10 secondes. Pour que la localisation se fasse

même si l'utilisateur verrouille son portable, nous avons utilisé un autre plugin appelé *cordova-plugin-mauron85-background-geolocation.* Nous travaillons de la manière suivante : dès qu'un point est collecté, il est ajouté à un tableau et nous redessinons la carte afin d'afficher le tracé. Lorsque le plugin notifie que l'utilisateur se trouve en stationnaire, c'est-à-dire qu'il ne bouge plus, l'application regarde le nombre de points enregistrés. S'il est supérieur à 8, le parcours est considéré comme valide et envoyé au serveur ; sinon, on supprime les points. Ensuite, nous mettons le dernier point comme le nouveau premier point et nous redessinons la carte afin d'effacer le précédent tracé.

Sur cette carte, il est également possible d'ajouter des données manuellement. Il suffit de cliquer sur la carte pour qu'un marker apparaisse. Ce marker ouvre une fenêtre d'information dans laquelle il y a un bouton pour rajouter un danger.

Figure 36 : Ajouter un danger
Source personnelle

Lorsque nous cliquons sur ce bouton, une nouvelle page apparaît afin de remplir les données pour le POI.

5.4.4. Ajout d'un POI

La première étape de cette phase est de trouver l'adresse sur laquelle l'utilisateur a cliqué. Pour ce faire, nous prenons la latitude et la longitude et nous utilisons le Geocoder fourni par Google Map qui va nous retourner l'identifiant Google pour ce lieu. Nous demandons ensuite à Google les informations supplémentaires sur le lieu afin d'en obtenir l'adresse et le nom.

Nous les affichons au sommet du formulaire pour informer l'utilisateur du point qu'il renseigne, mais il n'est pas modifiable afin d'éviter des erreurs de frappe.

Ensuite, l'utilisateur remplit les autres champs demandés et peut prendre plusieurs photos afin de mieux expliquer ce point. Les photos sont également un plugin qui donne l'accès à la caméra en natif. Ces dernières sont retournées en format base64 afin de pouvoir les envoyer correctement sur le serveur. Après l'envoi, le code sur le serveur va créer un dossier pour ce POI et écrire l'image dans ce dossier.

Une fois les informations complétées, l'application va les envoyer au serveur PHP. Ici, nous avons implémenté un algorithme afin d'insérer et valider le point qui est expliqué en détail dans l'annexe 1. En résumé, cela permet d'ajouter le point, d'ajouter ou modifier les données utilisateur par rapport à ce point et de valider le point et donc de changer la notation des utilisateurs ayant participé à la collecte du point. Cela va permettre de fournir aux personnes utilisant la navigation des points fiables. De plus, cela va empêcher les propriétaires d'établissement d'entrer des données erronées afin d'attirer plus de clients. Cependant, comme expliqué dans le chapitre *3.1. Résultats*, le fait d'utiliser cet algorithme lors du lancement de l'application aura un gros désavantage : il faudra un certain temps afin d'obtenir un nombre significatif de points valables. Nous pouvons rapidement choisir d'utiliser ou pas cet algorithme en changeant la valeur de la variable ***$statusApp*** par ***validateAll*** afin d'automatiquement rendre les points collectés valides. Si, à l'inverse, nous souhaitons utiliser l'algorithme, il faut utiliser la valeur ***useAlgo.***

```
1  <?php
2      class collection{
3          var $conn;
4          var $statusApp = "validateAll";
5          public function __construct(){
```

Figure 37 : Validation automatique de tous les points collectés
Source personnelle

5.4.5. Points à proximité

Une autre manière d'ajouter des points dans la base de données est d'utilisé l'onglet ***à proximité***. Lorsque nous cliquons dessus, notre code va utiliser la partie ***PlaceServices*** de la

bibliothèque Google maps. Nous cherchons les points aux alentours de la dernière position connue de l'utilisateur et, si nous n'en avons pas, nous en cherchons une. Cet appel AJAX va nous retourner une liste de POI que nous allons envoyer au serveur afin de vérifier si l'utilisateur a déjà collecté ce point. Cela aura pour but de transformer l'ajout en édition et ainsi récupérer les données déjà existantes ainsi que les points valides.

Figure 38 : Point à proximité
Source personnelle

Lorsque nous cliquons sur l'item, trois cas peuvent arriver :

- Le POI est valide (le vu)
- Le POI est en attente et l'utilisateur a déjà fourni des informations sur ce point (le crayon)
- Le POI est en attente et l'utilisateur n'a pas encore fourni d'informations (le plus)

Le schéma ci-dessous explique en détail les différentes possibilités.

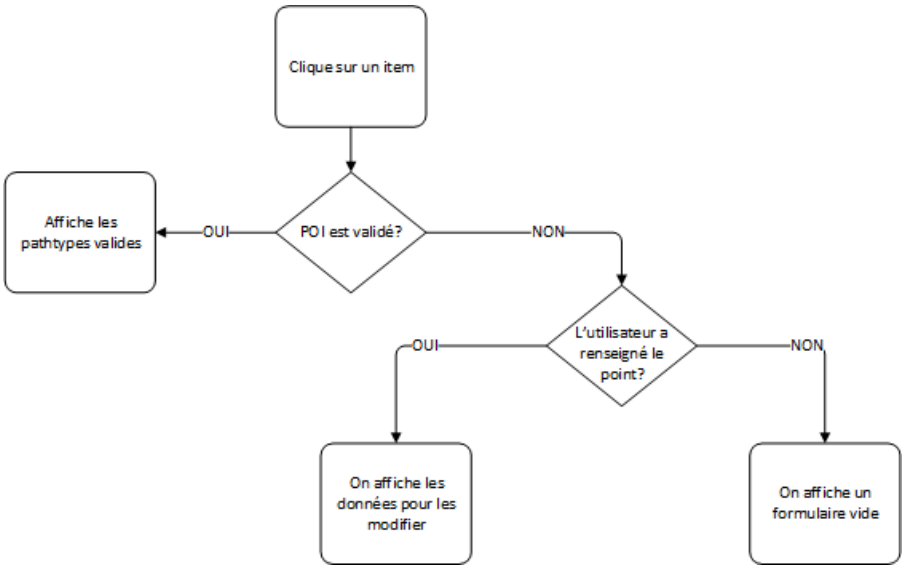


Figure 39 : Schéma point à proximité
Source personnelle

5.4.6. GPS

La fonctionnalité qui apporte un réel plus à notre application comparée à la concurrence est le fait de pouvoir avoir une carte qui affiche les dangers lors d'un parcours planifié.

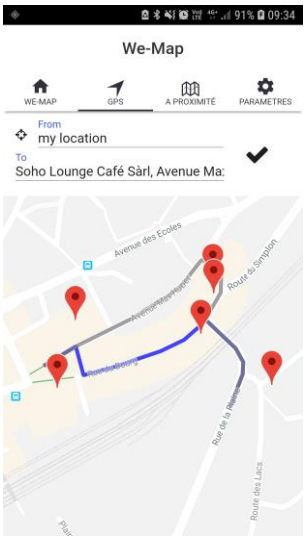


Figure 40 : GPS
Source personnelle

Il suffit de rentrer notre point de départ et notre point d'arrivée et l'application va utiliser la bibliothèque DirectionsService de Google maps. Comparés au GPS utilisé sur le site actuel, nous ne donnons pas la possibilité à l'utilisateur de choisir son type de transport. Il devra se contenter d'un trajet GPS basé sur la marche à pieds, car le but est de se déplacer dans une

ville et non pas d'une ville à l'autre. Nous demandons également plusieurs parcours afin que l'utilisateur puisse choisir son itinéraire préféré.

Nous avons changé la façon d'afficher les POI sur le parcours. Lors du précédent travail de Bachelor, les points étaient tous chargés puis un script JavaScript les traitait pour savoir s'ils étaient dans l'ellipse définie par le point de départ et d'arrivée en ajoutant la tolérance, comme nous pouvons le voir sur la figure ci-dessous.

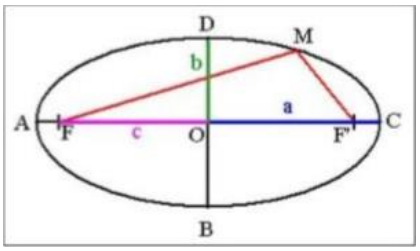


Figure 41 : Ellipse de sélection des données
Source personnelle

Pour notre cas, nous pensons que cette méthode n'est pas assez précise. Nous avons donc repensé le problème en nous basant sur le principe mathématique suivant : une ligne est une suite de points. Nous avons donc commencé par analyser comment trouver les POI à proximité d'un seul point représenté par sa latitude et sa longitude. Grâce à ces coordonnées et une distance en kilomètre, nous pouvons obtenir la latitude et la longitude maximum et minimum afin de dessiner un carré autour de ce point.

```php
private function getBoundingBox($latLng, $distanceKm){
    $lat = $this->deg2rad($latLng["lat"]);
    $lon = $this->deg2rad($latLng["lng"]);
    $halfSide = $distanceKm * 1000;

    $radius = $this->WGS84EarthRadius($lat);
    $pradius = $radius * cos($lat);

    $latMin = $lat - $halfSide / $radius;
    $latMax = $lat + $halfSide / $radius;
    $lonMin = $lon - $halfSide / $pradius;
    $lonMax = $lon + $halfSide / $pradius;
    return [
        "minLat" => $this->rad2deg($latMin),
        "maxLat" => $this->rad2deg($latMax),
        "minLng" => $this->rad2deg($lonMin),
        "maxLng" => $this->rad2deg($lonMax)
    ];
}
```

Figure 42 : Carré autour d'un point
Source personnelle

Maintenant que nous avons les minimums et les maximums, nous pouvons trouver les POI dans notre base de données compris entre ces quatre points. Nous pouvons donc sortir la liste des dangers autour d'un point du parcours. Un dernier souci se pose à nous : comment déterminer la distance en kilomètre qu'il faut utiliser ? Pour résoudre ce problème, nous avons calculé la distance entre chaque point de notre parcours et nous utiliserons cette valeur pour trouver les carrés qui contiennent les points à proximité. Il nous serait encore possible d'améliorer cet algorithme en calculant la distance la plus courte entre le POI et le parcours et de voir si elle est inférieure à 100 mètres. Cela pourrait être un sujet pour un prochain travail de Bachelor.

5.4.7. Inscription

Afin de pouvoir utiliser l'application, nous forçons l'utilisateur à se connecter. Le processus d'inscription va également permettre de définir si l'utilisateur est un bon évaluateur ou non. Lorsqu'il aura rempli le formulaire d'inscription, nous proposons au futur utilisateur de réaliser un test qui se présente sous la forme suivante.

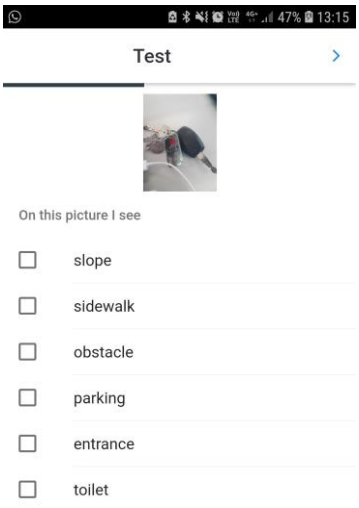


Figure 43 : Test d'entrée
Source personnelle

Nous récupérons 5 points validés, ayant au minimum une photo, dans la base de données et nous les affichons sous forme d'étape à valider. Vu que les points sont considérés comme

corrects, nous pouvons les utiliser pour comparer les réponses proposées par l'utilisateur et ainsi grâce au même algorithme que lors de la validation d'un point, nous calculons la note de ce nouvel usager. Il part avec une note de 5 et la note finale obtenue après l'examen sera comprise entre 0 et 10.

6. Évolution possible

Il est évident qu'en ayant que 360 heures à disposition afin d'écrire un rapport et de développer une application mobile, il existe encore plein de lacunes à combler pour rendre ce projet optimal.

Premièrement, il faudrait implémenter un algorithme qui permettrait d'obtenir des informations sur les parcours collectés automatiquement. En effet, nous avons des données sur des parcours avec la latitude, longitude et altitude, mais elles ne sont pas encore utilisées. Nous proposons, par exemple, d'utiliser un algorithme de type colonies de fourmis qui fournirait le chemin le plus court en évitant des obstacles. Il serait également possible d'obtenir des dangers supplémentaires sur la route. Par exemple, si un utilisateur a monté 15 mètres en 10 secondes, nous pouvons en déduire qu'il y a des escaliers ou une pente assez raide.

Ensuite, il faudrait tester, avec un groupe prédéfini, l'application afin d'avoir des avis réels. Cette partie n'a pas pu voir le jour, car nous avons préféré nous concentrer sur la correction des bugs au sein de notre application afin de ne pas avoir de résultats faussés sur l'ergonomie en cas de crash.

La dernière étape avant la mise en production officielle de cette application est de faire une façade afin de pouvoir utiliser l'ancien site web avec la nouvelle base de données. Enfin nous pourrons déployer notre application en la signant puis en l'ajoutant aux différents magasins d'applications mobiles.

7. Gestion du projet

7.1. Préparation et planification

Afin de bien organiser notre projet, nous avons fait une planification lors des 2 premières semaines. Elle se trouve dans l'annexe 2 : cahier des charges. Nous avons fait un tableau récapitulatif :

No de Phase	Titre de la phase	Heures prévues	Pourcentage
Phase 1	Compréhension du sujet	30	7.69 %
Phase 2	État de l'art et analyse du marché	96	24.61 %
Phase 3	Implémentation	104	26.66 %
Phase 4	Test	80	20.51 %
Phase 5	Rédaction	80	20.51 %
Total		**390**	**100 %**

Tableau 7 : Récapitulatif des horaires

Afin de voir l'état d'avancement de notre travail de Bachelor, nous avons utilisé une feuille Excel annexe dans laquelle nous notions nos heures par jour ; puis, le total de ces heures était calculé sur une autre page sous forme de récapitulatif comme le montre la figure ci-dessous.

	lundi	total heures
Semaine du	14.05.2018	22
Semaine du	21.05.2018	31
Semaine du	28.05.2018	32
Semaine du	04.06.2018	40
Semaine du	11.06.2018	40
Semaine du	18.06.2018	14
Semaine du	25.06.2018	40
Semaine du	02.07.2018	36
Semaine du	09.07.2018	40
Semaine du	16.07.2018	40
Semaine du	23.07.2018	40
Semaine du	30.07.2018	0
Total		375

Figure 44 : Récapitulatif des heures
Source personnelle

Pour valider notre avancé, nous avions rendez-vous une fois chaque une à deux semaines avec Madame Glassey Balet. Cela permettait de valider le travail effectué et d'expliquer sur quels points nous allions travailler. Il s'agit ici de la gestion externe dans laquelle nous présentons au client le projet. Il nous a aussi fallu mettre en place une gestion interne, c'est-à-dire un outil afin de voir ce qui a été fait ou non. Notre choix s'est porté sur l'utilisation de Trello plutôt que sur un vrai logiciel de gestion scrum, car il nous fallait quelque chose de simple et facilement utilisable. De plus, il nous fallait un endroit pour stocker notre code afin de ne pas tout perdre si notre ordinateur venait à avoir un souci. BitBucket permet d'avoir des projets privés, c'est donc pour cette raison que nous l'avons choisi. De plus, il intègre directement Trello dans son site web.

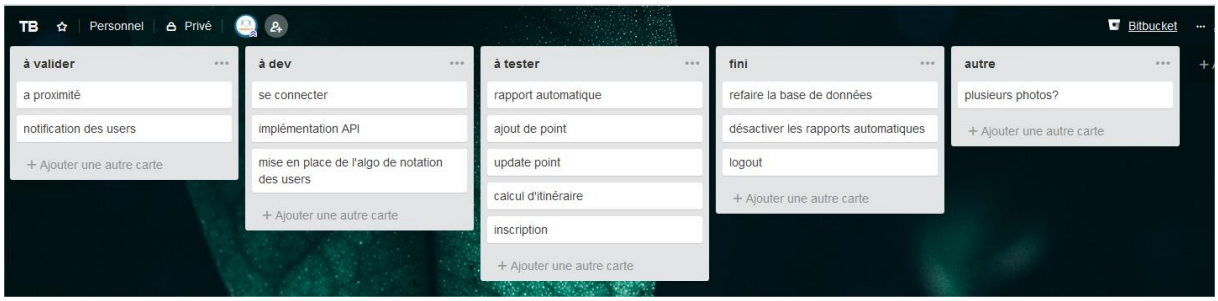


Figure 45 : Trello dans BitBucket
https://bitbucket.org/

7.2. Résultat et comparaison

Nous avons finalement pu comparer les heures planifiées et les heures réellement effectuées. Nous avions prévu 390 heures de travail et nous avons finalement terminé avec 405 heures. Cela représente une augmentation de 3.84 %, ce que nous considérons comme correct vu la taille du travail et le flou encore présent lors de la planification initiale. Ci-dessous, nous pouvons voir la comparaison des différentes phases.

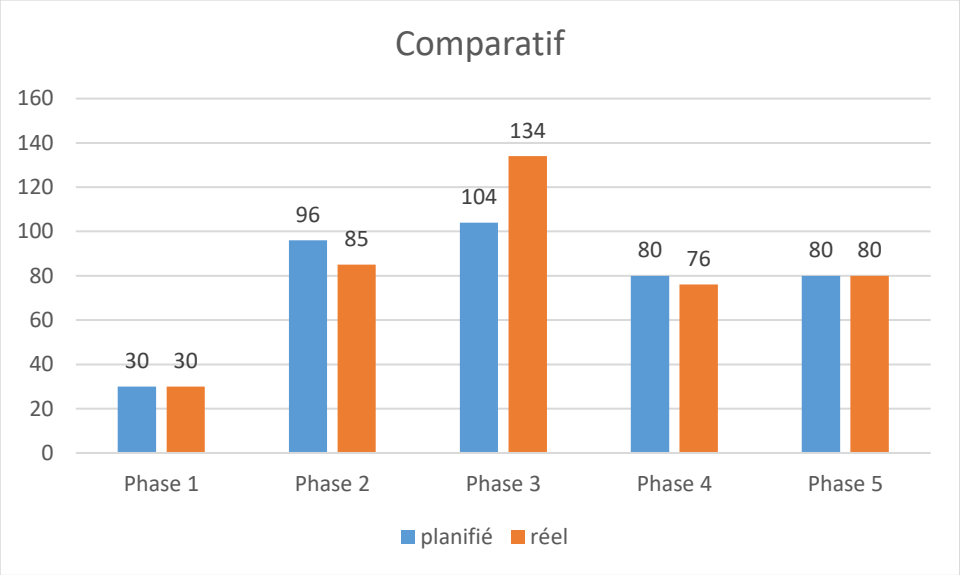


Figure 46 : Comparatif des heures
Source personnelle

Nous avons été un tout petit peu plus rapide sur les phases d'état de l'art et de test, mais il nous aura fallu plus de temps pour le développement. Sur le récapitulatif, le pourcentage d'écart est faible, mais si nous prenons phase par phase, nous voyons que le développement a été mal planifié, car il y a une augmentation de 28.85 %, ce qui n'est pas négligeable.

8. Bilan

Lors de ce travail de Bachelor, nous avons appris à correctement faire des recherches sur des sujets précis. Nous avions déjà l'habitude d'en faire sur les technologies, mais dans le cas présent, il nous a fallu faire des recherches sur des sujets liés plus à la psychologie des personnes. Cela sort du parcours scolaire et nous a montré que pour réaliser correctement un projet informatique, il faut également connaitre et comprendre le fonctionnement du domaine dans lequel ce projet va devoir être réalisé.

La planification réalisée au départ du travail de Bachelor fut un réel défi, car il a fallu prévoir les différentes phases sans vraiment avoir toutes les informations sur le projet et sur les technologies à utiliser.

Lors du développement, nous avons appris à utiliser une nouvelle bibliothèque qui permettra de rendre nos futures applications hybrides beaucoup plus proches d'une application native en termes de design.

Le point qui nous a posé le plus de soucis est la gestion des anciennes données afin de pouvoir les intégrer correctement dans la nouvelle application. De plus, nous avons été confrontés à du code écrit par autrui avec une logique différente. Nous avons donc dû consacrer du temps à la lecture de ce dernier afin de correctement comprendre la direction que les anciens développeurs avaient prise afin de ne pas dénaturer le projet.

Dans sa globalité, le projet WE-MAP est un réel défi, car il faut motiver des gens qui ne sont pas forcément sensibles aux problèmes de la mobilité au sein de l'espace urbain pour les personnes à mobilité réduite. Nous espérons que notre développement apportera un véritable atout afin de toucher un maximum de personnes, même si nous savons que ce travail de Bachelor ne représente que la base d'une application mobile qui pourrait être grandement améliorée avec plus de temps.

Finalement, nous avons apprécié travailler sur ce projet, car il permet de faire une bonne transition entre le monde théorique que représente l'école et le futur monde professionnel que nous serons amenés à côtoyer sous peu.

9. Conclusion

Dans la première partie de ce travail, nous avons commencé par définir le Crowdsourcing et les moyens de motivations psychologiques existants. Puis, nous avons analysé les différentes approches afin de motiver et valider les données au sein d'une méthodologie de Crowdsourcing, cela d'un point de vue théorique. Tout d'abord nous sommes passés par des calculs de statistiques et mathématiques afin d'éliminer les facteurs les moins influents, à savoir la motivation pécuniaire et la rencontre d'événements inattendus. Puis, nous nous sommes basés sur une étude qui comparait pratiquement les trois systèmes les plus connus de Crowdsourcing : basique, avec récompense et ludification. Il en ressort que la ludification est le moyen qui permet une plus grande collecte sur le long terme. Finalement nous avons également tenu compte des conseils de David Mack lors de son précédent travail de Bachelor.

La partie sur la motivation étant finie, nous sommes passés à la partie sur la validation des données collectées. Nous avons regardé comment IBM et une application du nom de mPass faisaient pour nettoyer leurs données et nous avons confronté cela aux recherches préliminaires qu'avait effectuées l'institut de recherche. Il en est ressorti qu'il est bénéfique d'avoir un système de collecte automatique, qu'un classement des utilisateurs permet également d'améliorer la validité des informations et que le meilleur système serait un algorithme de machine learning qui pourrait apprendre automatiquement des données qu'il reçoit en se basant sur une classification simple.

Dans la seconde partie, nous avons regardé le marché. Nous avons sorti cinq applications qui utilisent le Crowdsourcing et nous les avons comparées sur les points de vue de la motivation et de la validation. Le point clé de cette analyse n'est qu'aucune d'entre elles n'utilisent un système de ludification, mais qu'il est important de garder un contact régulier

avec l'utilisateur. Nous avons donc choisi de développer une application mobile afin d'avoir les rapports automatiques ainsi que le contact avec l'utilisateur grâce à des notifications.

Dans la troisième phase, nous avons développé notre application de telle sorte à ce qu'elle soit facile à prendre en main. Nous avions trouvé un peu déplacé que l'utilisateur doive suivre un tutoriel afin de comprendre comment fonctionne le programme. Nous avons commencé par refaire une nouvelle base de données puis nous avons ajouté toutes les anciennes fonctionnalités. Ensuite, nous avons ajouté notre touche avec la notion d'utilisateur, le système de notification et la collecte automatique. Pour centraliser les données, nous avons créé une API sécurisée par un token sur notre serveur. Il a également fallu penser aux différents algorithmes, leur logique interne et comment ils allaient s'influencer l'un l'autre.

Dans la dernière partie, nous avons testé l'application et corrigé les différents bugs présents pour éviter les crashs et ainsi ne pas nuire au confort de la personne lors de l'utilisation de l'application. Nous savons que certaines erreurs nous ont échappé et qu'il faudra encore travailler sur l'application afin de la mettre en production. Il s'agit ici simplement d'un prototype fonctionnel et non pas d'une application entièrement finie.

10. Références

Zheng, H., Li, D., & Hou, W. (2011). Task design, motivation, and participation in crowdsourcing contests. International Journal of Electronic Commerce, 15 (4), 57-88.

Chu, X., Morcos, J., Ilyas, I. F., Ouzzani, M., Papotti, P., Tang, N., & Ye, Y. (2015, May). Katara: A data cleaning system powered by knowledge bases and crowdsourcing. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data (pp. 1247-1261). ACM.

Salomoni, P., Prandi, C., Roccetti, M., Nisi, V., & Nunes, N. J. (2015, September). Crowdsourcing urban accessibility: Some preliminary experiences with results. In Proceedings of the 11th Biannual Conference on Italian SIGCHI Chapter (pp. 130-133). ACM.

Bonter, D. N., & Cooper, C. B. (2012). Data validation in citizen science: a case study from Project FeederWatch. Frontiers in Ecology and the Environment, 10 (6), 305-307.

Cardonha, C., Gallo, D., Avegliano, P., Herrmann, R., Koch, F., & Borger, S. (2013, May). A crowdsourcing platform for the construction of accessibility maps. In Proceedings of the 10th international cross-disciplinary conference on web accessibility (p. 26). ACM.

Prandi, C., Salomoni, P., & Mirri, S. (2014, January). mPASS: integrating people sensing and crowdsourcing to map urban accessibility. In Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th (pp. 591-595). IEEE.

Lebraty, J. F. (2007, June). Vers un nouveau mode d'externalisation : le Crowdsourcing. In 12ème conférence de l'AIM.

Zhao, Y., & Zhu, Q. (2012). Exploring the motivation of participants in crowdsourcing contest.

Liu, Z., Balet, N. G., Sokhn, M., & De Gaspari, E. (2017). Crowdsourcing-Based Mobile Application for Wheelchair Accessibility.

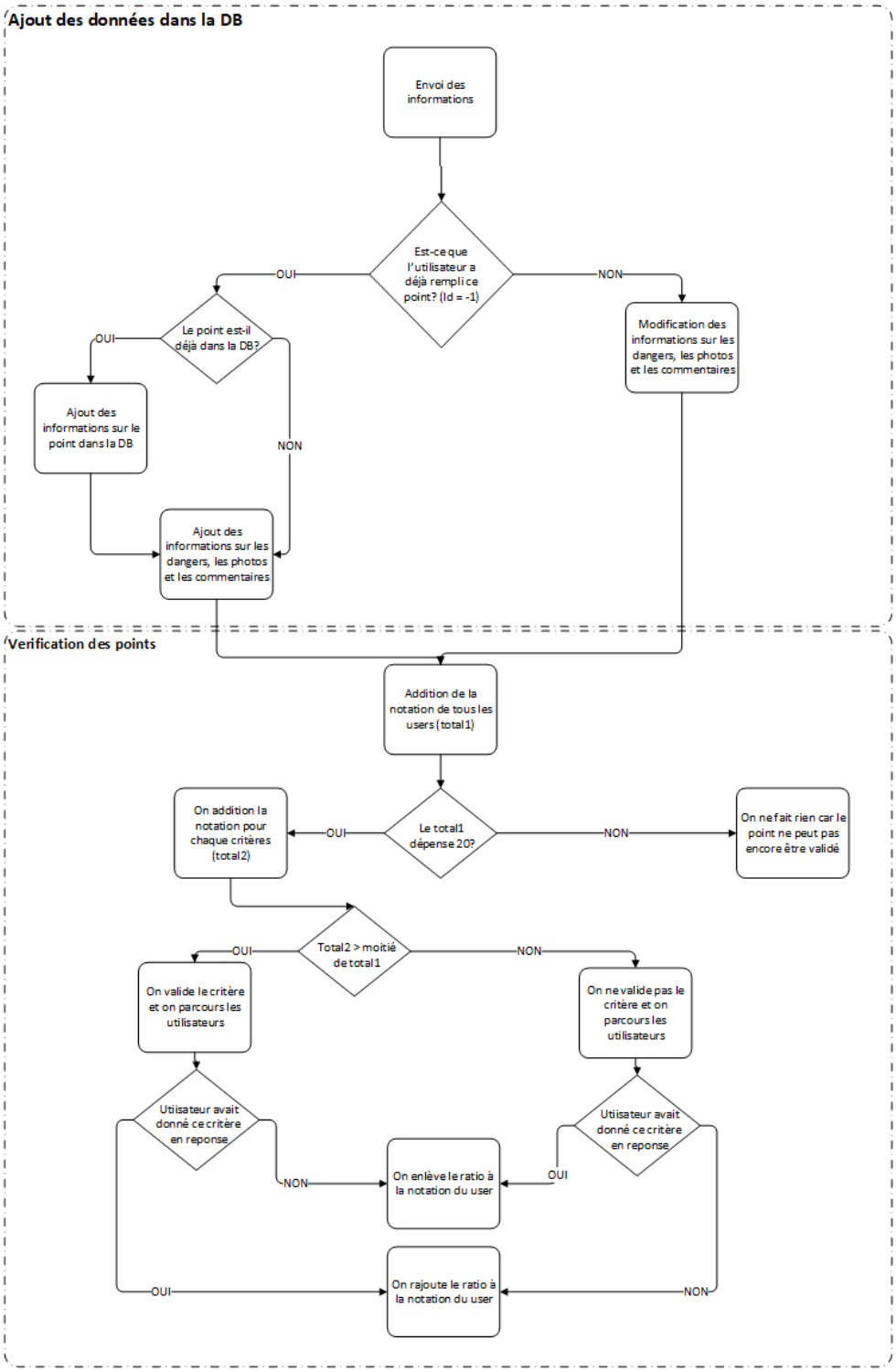
Liu, Z., Shabani, S., Balet, N. G., Sokhn, M., & Cretton, F. (2018, January). How to motivate participation and improve quality of crowdsourcing when building accessibility maps. In Consumer Communications & Networking Conference (CCNC), 2018 15th IEEE Annual (pp. 1-6). IEEE.

Miguéns, J., Baggio, R., & Costa, C. (2008). Social media and tourism destinations: TripAdvisor case study. Advances in tourism research, 26 (28), 1-6.

Edelman, B. G., & Luca, M. (2014). Digital discrimination: The case of airbnb. com.

11. Annexes

11.1. Annexe 1 : Algorithme de validation des données



11.2. Annexe 2 : Cahier des charges

Hes·so

1. INTRODUCTION

Ce cahier des charges s'inscrit dans la phase de planning du travail de Bachelor réalisé en collaboration avec l'institut de recherche de la Haut Ecole Spécialisée de Suisse Occidentale (HES-SO).

2. Contexte du travail

L'utilisation du crowdsourcing afin de remplir sa base de données peut révéler de réels problèmes. De nombreuses applications utilisent ce moyen de récolte, en ayant chacun leur méthodologie afin d'attirer les clients / récolteurs.

L'institut de recherche de la HES-SO a développé, en collaboration avec David Mack ancien étudiant, une application permettant aux personnes à mobilité réduite de rechercher des trajets tout en évitant des obstacles. Cette recherche s'effectue grâce à des points d'intérêt, point of interest en anglais (POI), collectés au préalable par d'autres utilisateurs. Cependant, il n'y a pour le moment encore aucune vérification des données entrantes et le nombre d'utilisateur est assez faible.

3. Problématique

Le projet WE-MAP se pose face à un problème vis-à-vis de crowdsourcing. En effet, son public cible n'est pas le même que le public récolteur. Les personnes à mobilité réduites ne pourront pas enregistrer autant de POI que les autres, qui eux ne vont pas voir une réelle utilité à l'application si ce n'est par compassion. Il faudra donc trouver un moyen de motiver les personnes valides à renseigner des points pour améliorer notre application.

Le second souci de l'application sera le contrôle des informations entrantes afin d'assurer aux utilisateurs réguliers une application fiable. Il s'agit ici d'un problème plutôt mathématique qui pourra se résoudre à l'aide d'algorithmes de machine learning et de classification.

1

Hes·so

4. Travail à effectuer

Les buts de ce travail de Bachelor sont :

- Réaliser l'état de l'art concernant l'attrait pour le crowdsourcing et l'amélioration des données entrantes afin de sélectionner la meilleure solution à implémenter pour notre application.
- Implémenter les solutions trouvées au sein de l'application.
- Effectuer des tests afin d'évaluer les changements apportés

5. Planification

Pour ce travail, nous n'utiliserons pas la méthode agile dans sa totalité. En effet, certain aspect de cette méthode n'apporte rien de plus lorsque le groupe se compose d'une personne comme c'est le cas ici. Le fait d'être libre pour la direction du Bachelor crée aussi une distant par rapport à la méthode agile car nous n'avons pas de client à qui présenter des résultats.

Cependant nous allons garder la méthode itérative avec une présentation du travail effectué chaque 7 / 15 jours afin que les responsables du projet voient l'avancement du projet et puisse rediriger en cas de problème face à leurs attentes.

Afin de bien réaliser cette tâche, nous allons mettre des deadlines.

5.1. Phase 1 : Compréhension du sujet

Cette phase va permettre de réaliser toutes les tâches administratives et de bien comprendre le sujet afin de poser la direction que le travail de Bachelor devra suivre. Elle devra être finie pour le 15 mai 2018 avec le rendu du cahier des charges.

Hes·so

5.2. Phase 2 : Etat de l'art et analyse du marché

Cette phase va permettre d'analyser le marché et de ressortir la meilleure / les meilleures solutions existantes sur le marché. Elle va se découper en deux parties. La première sera un état de l'art basé sur des documents scientifiques afin de définir le crowdsourcing, ses sources de motivation et ses moyens de contrôle des données.

La seconde partie consistera à analyser les applications faisant appel au crowdsourcing afin d'analyser les méthodes qui marchent. Lorsque nous arriverons à la fin de l'analyse de chaque application, il va falloir définir comment implémenter cette solution dans le projet We-Map. Cela dans le but d'avoir une base commune pour la comparaison finale. Cette phase devra être finie pour le 1 juin 2018 avec des tableaux comparatifs.

5.3. Phase 3 : Implémentation

Cette phase va permettre de mettre en place les différentes solutions trouvées au sein de notre application. Il va falloir commencer par redesigner la base de données pour remplir les différents besoins. Puis, nous allons modifier la logique de l'application pour enfin finir par une petite couche de design. Selon les solutions choisies, l'implémentation de la logique pourra prendre plus au moins de temps Nous estimons que cela va nous prendre environ 4 semaines de développement ce qui fixe la deadline au 29 juin 2018.

5.4. Phase 4 : Test

Cette phase va permettre de voir si les changements dans l'application apportent un réel plus. Ces tests seront effectués afin de corriger les différents bugs au sein de notre application afin de la préparer à la mise en production. Il leur sera demandé d'entrer de mauvaises données afin de voir si notre système le détecte. Il sera également question de tester si les changements apportent une source de motivation et ainsi voir si notre travail aura été utile. La deadline sera fixé au 13 juillet 2018.

Hes·so

5.5. Phase 5 : Rédaction

Cette phase va permettre de corriger et d'améliorer le document final afin de le rendre pour le 30 juillet 2018.

5.6. Planification des horaires

No de semaine	Nombre d'heures planifiées	Phase concernée
20 (lundi 14.05.2018)	24	2
21 (lundi 21.05.2018)	36	2
22 (lundi 28.05.2018)	36	2
23 (lundi 04.06.2018)	26	3
24 (lundi 11.06.2018)	26	3
25 (lundi 18.06.2018)	26	3
26 (lundi 25.06.2018)	26	3
27 (lundi 02.07.2018)	40	4
28 (lundi 09.07.2018)	40	4
29 (lundi 16.07.2018)	40	5
30 (lundi 23.07.2018)	40	5
31 (lundi 30.07.2018)	Rendu	5

Hes·so

5.7. Résumé des phases

Phase 2	**96 heures**
Phase 3	**104 heures**
Phase 4	**80 heures**
Phase 5	**80 heures**

6. Délivrables

Il faudra rendre un document appelé rapport qui devra contenir la description des analyses portées, les résultats et la méthodologie de gestion utilisée.

L'application améliorée devra aussi être rendu.

7. Conclusion

Ce document a pour but de clairement poser les attentes et la planification du travail afin d'arriver à un résultat. Il est cependant difficile d'effectuer une planification correcte sans avoir analysé le marché courant. Il est donc possible que dans le futur les deadlines soient légèrement décalés et que les attentes peuvent changer.

11.3. Annexe 3 : Structure des solutions

Pour ce projet, nous avons deux solutions : le serveur PHP et l'application mobile.

11.3.1. Le serveur PHP

Nous avons repris l'ancien serveur et nous avons simplement ajouté des fichiers afin de garder l'ancienne application fonctionnelle. L'architecture proposée pour l'API ne porte pas de nom à proprement parler, car il s'agit d'un dérivé d'une architecture REST.

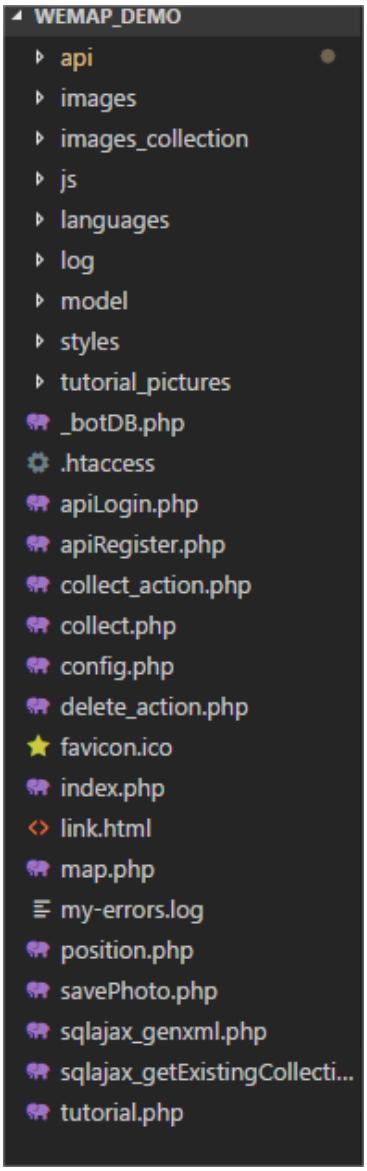


Figure 47 : Organisation WeMap

Le dossier ***api*** va contenir toute la logique pour la requête à la base de données et la protection grâce à un token.

Le dossier ***images_collection*** possède toutes les images collectées par les utilisateurs.

Le fichier ***_botDB.PHP*** est un script exportant les anciennes données de l'application vers la nouvelle base.

Le fichier ***.htaccess*** permet de rediriger correctement les requêtes api.

Le fichier ***apiLogin.php*** permet d'obtenir son token de connexion en fournissant son email et son mot de passe.

Le fichier ***apiRegister.php*** permet de s'enregistrer au sein de l'application.

Les autres fichiers étaient déjà présents dans la version précédente.

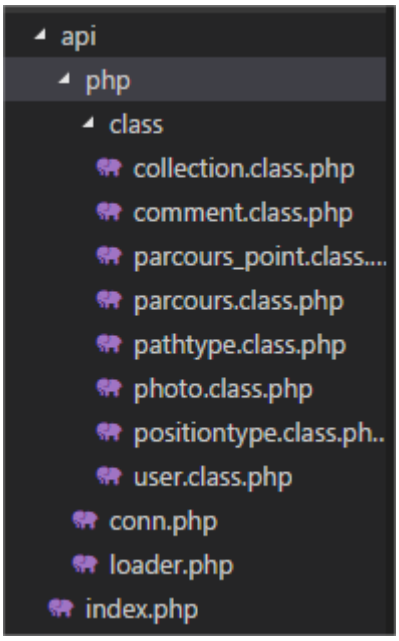


Figure 48 : Dossier API

L'api se compose d'un fichier ***index.php*** qui est le point d'entrée de toutes les requêtes API.

D'un fichier ***loader.php*** qui est un autoloader afin de charger les classes se trouvant dans le dossier class.

Un fichier ***conn.php*** qui contient simplement les informations de connexion pour la base de données.

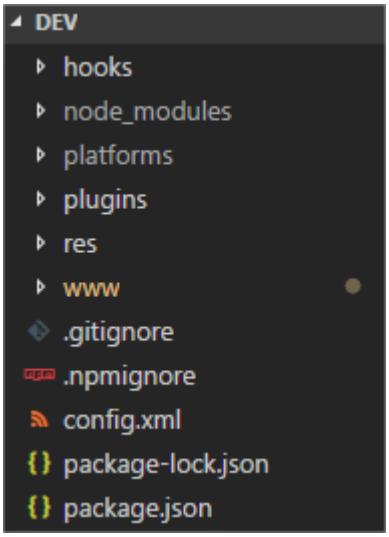
Un dossier ***class*** contenant nos objets (*.class.php) que nous pouvons appeler par API.

11.3.2. L'application mobile

Cordova fournit une ligne de commande afin de pouvoir créer un projet. Nous avons donc utilisé la commande suivante.

cordova create MyApp

Cela va créer cette architecture.

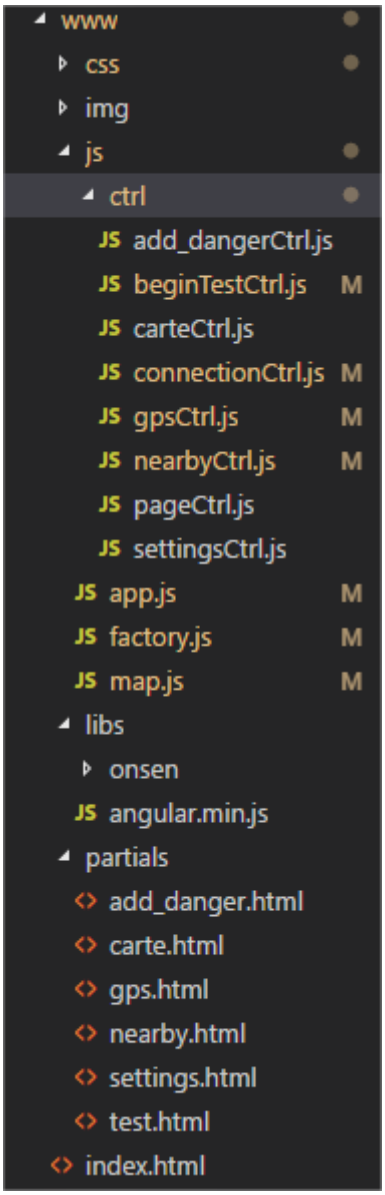


Le dossier ***platforms*** contient les applications compilées Android et iOS.

Le dossier ***plugins*** contient les plugins cordova téléchargés du web.

Le dossier ***res*** contient les ressources pour les différentes applications comme les icônes ou les splashscreens.

Le dossier ***www*** est le contenu de notre application, c'est le seul que nous allons modifier par la suite.



La partie web s'organise comme cela.

Le dossier ***css*** va contenir le style de notre application.

Le dossier ***img*** les images utilisées dans l'application.

Le dossier ***js*** sera la logique de notre application. Il contient un dossier ***ctrl*** qui va permettre d'avoir les contrôleurs pour chaque partials.

Le dossier ***libs*** contient la liste des librairies utilisées dans ce projet.

Le dossier ***partials*** contient les différentes vues de notre application.

Le fichier ***index.html*** est le point d'entrée de notre application web.

12. Déclaration de l'auteur

Je déclare, par ce document, que j'ai effectué le travail de Bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du RF et du professeur chargé du suivi du travail de Bachelor, y compris au partenaire de recherche appliquée avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après : Nicole Glassey Balet, Zhan Liu.