Université de Toulouse

MASTER 2 GEOMATIQUE

« **Sc**I**e**nces **G**éomatiques en environne**M**ent et **A**ménagement » (**SIGMA**)

http://sigma.univ-toulouse.fr

RAPPORT DE STAGE

Développement d'une application web de cartographie interactive d'un réseau de transports en commun

PARADELLE Nelly

NEOGEO TECHNOLOGIES

Maître de stage : Guillaume SUEUR
Enseignant-référent : Sébastien LE CORRE

Septembre 2020

Remerciements

Je tiens à remercier Guillaume Sueur, gérant de Neogeo Technologies et Maître de conférences associé à l'Université Jean-Jaurès de Toulouse, pour m'avoir proposé ce stage au sein de son entreprise. La mission qu'il m'a confiée, à savoir gérer le projet et participer au développement d'une application web de cartographie dynamique d'un réseau de transport, m'a permis d'acquérir de nombreuses compétences techniques et relationnelles qui me serviront tout au long de ma vie professionnelle.

Je remercie Xavier Vantal, administrateur système, Florent Lavelle ainsi que Maël, développeurs, sans qui le prototype d'application ne serait pas ce qu'il est aujourd'hui. Vous m'avez soutenu tant moralement que techniquement lorsque j'étais en difficulté sur les développements Python. Le fruit de votre travail a permis d'aboutir à une interface ergonomique et intuitive que le client apprécie beaucoup. Je garderai en mémoire vos précieux conseils.

Je présente de nouveau mes remerciements à Xavier et à Kevin Rakotobe, administrateur système junior, pour m'avoir si bien accueillie dans leur bureau. La bonne humeur qui y régnait et les plaisanteries que nous avons pu échanger me permettaient de décrocher de temps en temps mes yeux de l'écran. Ces quelques moments de relâchement ont réellement étaient bénéfiques et nécessaires pour me permettre de réaliser efficacement mes tâches.

Je remercie également mon tuteur Sébastien Le Corre qui s'est toujours montré disponible et qui a suivi mes différentes activités tout au long de mon stage. Mais je dois aussi adresser mes remerciements à l'ensemble du corps enseignant pour la qualité de la formation SIGMA. Je commence seulement à me rendre compte de toutes les compétences que j'ai acquises durant les 6 mois de formation. Mais ce résultat n'aurait pas été possible sans un effort acharné, et le soutien de mes camarades de promotion.

Résumé

Dans la métropole d'Orléans, la gestion du service public de transports en commun est assurée par l'entreprise privée Keolis depuis 2012. En fin d'année 2019, elle a contacté le prestataire Neogeo Technologies pour la réalisation d'un démonstrateur d'application web de cartographie dynamique de son offre de transport. Le but est de vérifier la faisabilité d'un tel produit avec les données disponibles, et de déterminer si un tel outil permettrait aux services internes de mieux connaître l'offre de transport et de pouvoir évaluer sa pertinence. L'objectif du stage a consisté à gérer ce projet de démonstrateur à soumettre au client en Septembre 2020, et d'assurer le développement back-end de l'application. La méthode de gestion de projet mise en place a été la méthode SCRUM qui se caractérise par des rôles et des cycles courts de développement. Le développement back-end a consisté à structurer la base de données stockant les données métiers, et à coder l'API via le framework python Flask, l'API servant d'interface entre le navigateur web et la base de données. Il a également fallu configurer le fichier .map de mapserver qui sert des cartes au front-end. Finalement, le démonstrateur développé répond aux exigences initiales du client qui va présenter le prototype à ses services internes pour que ces derniers puissent le tester/manipuler, et décider s'ils souhaitent réellement disposer d'un tel outil. Le client songe également à présenter l'outil en externe, au service géomatique de la métropole d'Orléans, et lors de manifestations en lien avec la géomatique comme les Geodatadays. (250 mots)

Abstract

In the metropolis of Orleans, the management of public transport service is carried out by the private company Keolis since 2012. At the end of the year 2019, it contacted the service provider Neogeo Technologies to develop a proof of concept of a web mapping application of its public transport offer. The aim of the proof of concept is to check the feasibility of this application given the available data, and to see whether this product would be useful for Keolis' internal services to better understand the public transport offer and to assess its relevance regarding specific criteria. The goal of the internship consisted in managing this project that must end for September 2020, and to develop the back-end of the prototype. The project management method that was used is the SCRUM method, which is based on roles and fast development cycles. The back-end development included database structuring and API programming by using python Flask framework, the API representing the interface between web browser and database. Moreover, it has been necessary to configure the mapfile of mapserver, a software that provides maps to the front-end. At the end, the prototype meet the client requirements. It will be tested by the internal services that will decide whether or not they really need this product for their activities. In addition, the client thinks of using the product outside the company and presenting this tool to the metropolis' GIS office, or in the scope of events related to geographic information such as the Geodatadays.

TABLE DES FIGURES

TABLE DES TABLEAUX

SOMMAIRE

Introduction

En 2018[1], 10.5 % des déplacements étaient réalisés en transports en commun sur la métropole d'Orléans, l'objectif du Plan de Déplacements Urbains[2] de la métropole étant d'atteindre les 20% d'ici 2028. En effet, le trafic routier est le premier secteur émetteur de gaz à effet de serre et concentre ¼ des dépenses énergétiques de la métropole. Les enjeux sociaux (fracture sociale entre les mobiles et les immobiles n'ayant pas le même accès à l'emploi et à divers services), économiques (les ménages dépensent en moyenne 5000€ par an pour leur mobilité[3]) et environnementaux (effets de serre, pollution de l'air) placent les transports en communs au cœur des politiques d'aménagement des villes.

Pour mieux connaître son offre de transport et évaluer la pertinence des services qu'elle propose à ses voyageurs, l'entreprise Keolis exploitant pour le compte d'Orléans Métropole son réseau de transports publics songe à disposer d'un outil de cartographie dynamique de son offre. **Elle souhaite disposer d'un démonstrateur (un prototype minimaliste) début Septembre 2020 pour vérifier la faisabilité et la désirabilité d'un tel produit au sein de ses services. Elle a fait appel à l'entreprise Neogeo Technologies pour développer ce démonstrateur**. Dans ce contexte, le sujet de stage a consisté à gérer ce projet en tant que cheffe de projet junior, et de participer à une partie des développements du démonstrateur d'application web de cartographie dynamique de l'offre de transport en communs à Orléans.

Après une brève présentation de l'entreprise Neogeo Technologies qui permettra également d'introduire le contexte du projet Keolis, la seconde partie visera à expliquer la méthode de gestion de projet ayant été mise en œuvre pour gérer ce projet de démonstrateur. La partie III détaillera le fonctionnement et la structure de l'application, ainsi que la méthode mise en œuvre pour développer la partie back-end (l'ensemble des éléments invisibles pour l'utilisateur utilisant l'application web) du démonstrateur à travers un exemple de fonctionnalité demandée par le client. La partie IV précédant la conclusion et le bilan personnel consistera à évaluer la cohérence entre le démonstrateur développé et les demandes exprimées par le client.

[1] Orléans Métropole (2019) « Les chiffres clés de la mobilité dans la métropole orléanaise 2018 » disponible à l'adresse suivante : https://www.orleans-metropole.fr/deplacements/politique-de-deplacements
[2] Document définissant l'organisation des mobilités sur le territoire sur 10 ans
[3] « Les enjeux de la mobilité durable » consulté le 06/09/2020 et disponible à l'adresse suivante : http://www.centre-val-de-loire.developpement-durable.gouv.fr/les-enjeux-de-la-mobilite-durable-r1086.html

I. Une entreprise spécialisée dans les infrastructures de données géographiques et en pleine croissance

A. Une entreprise tournée vers l'Open Data

Neogeo Technologies est une SARL (société à responsabilité limitée) créée en 2008 par Guillaume Sueur. Elle propose des services informatiques dans le domaine de l'information géographique. Ses prestations reposent sur une gamme de logiciels Open Source comme QGIS, PostGIS, MapServer ou encore GeoServer.

Ses activités d'ingénierie concernent l'accompagnement, le déploiement, l'expertise, la maintenance et le support dans la mise en œuvre de solutions sur mesure dans le domaine de l'information géographique. Neogeo Technologies s'est notamment spécialisée dans la mise en œuvre ou le maintien et l'évolution de plateformes de partage de données en environnement OpenSource :
-des plateformes régionales (DataSud pour la région Sud, OPenIG pour la région Occitanie, IdéoBFC pour la région Bourgogne-Franche-Comté, Géo2France pour les Hauts-de-France),
-pour des métropoles (la métropole du Grand Lyon),
-ou des portails métiers (portail GéoFoncier à destination des géomètres-experts pour centraliser et décrire l'ensemble de leurs interventions foncières depuis 1997 sur l'ensemble du territoire français)

L'entreprise propose aussi un service d'infogérance, c'est-à-dire la prise en charge de tout ou d'une partie de la gestion et/ou de l'exploitation du système d'information de ses clients. Ce service recouvre les tâches de maintien en condition opérationnelle des infrastructures de données, l'intégration, l'optimisation et le traitement des données géographiques, l'administration des serveurs avec un système de monitoring pour suivre en temps réel et corriger les éventuelles défaillances et suivre la disponibilité des services.

L'entreprise est aussi impliquée dans l'utilisation et la diffusion de l'information géographique et des logiciels Open Source en proposant des formations et en participant à des projets collaboratifs. Guillaume Sueur est notamment secrétaire de l'association AFIGEO, (association française pour l'information géographique). Elle regroupe les professionnels du domaine de l'information géographique en France et œuvre pour promouvoir et favoriser le développement de ce secteur en France et à l'international. L'association anime notamment des formations, organise des rencontres telles que les GéoDataDays (l'évènement national de référence de la géographie numérique) pour échanger et débattre autour des thématiques liées à l'information géographique, et diffuse diverses informations sur l'actualité et les opportunités du secteur.

B. Une entreprise en évolution et à la recherche de nouveaux marchés

Entre 2015 et 2020 le chiffre d'affaire a été multiplié par trois, et atteignait 832 000 € en 2019, l'objectif étant de rester sur cette dynamique pour les cinq prochaines années.

En termes d'effectifs, le nombre de salariés a doublé entre 2018 et 2020 (treize employés) et l'objectif est d'atteindre la trentaine de collaborateurs d'ici trois ans. L'équipe comptait deux administrateurs systèmes responsables des serveurs, deux chefs de projet, et quatre développeurs sur le site de Toulouse en début d'année 2020. Une agence a été ouverte à Lyon en Juin 2020, et compte pour l'instant deux développeurs. Un service de support a été mis en place en Juillet 2020 afin de traiter les demandes des utilisateurs des plateformes régionales développées par l'entreprise.

Le recrutement en Mars 2020 d'une directrice commerciale a pour objectif de renforcer les positions actuelles de Neogeo Technologies dans le domaine des plateformes régionales et de diversifier la clientèle vers les acteurs du spatial notamment (intégrer les images satellitaires dans son offre de services).

C. Un exemple de diversification : le projet Keolis de cartographie dynamique de l'offre de transport à Orléans - le contexte

Keolis Métropole Orléans, une entreprise privée filiale du groupe Keolis, assure un service d'exploitation et de gestion des transports en commun dans l'agglomération orléanaise pour le compte d'Orléans métropole. La filiale regroupe environ 750 employés dont 485 conducteurs de bus et tramway transportant chaque jour 130 000 voyageurs sur le réseau des transports de l'agglomération orléanaise (TAO). Ce dernier s'étend sur les 22 communes de la métropole. Le réseau exploité comporte 2 lignes de tramway, une quarantaine de lignes de bus, et 9 zones de transport à la demande (TAD). Contrairement aux lignes de bus et de trams qui suivent des horaires et des itinéraires fixes, les véhicules des TAD ne respectent pas d'itinéraires ni d'horaires précis mais couvrent des zones géographiques. La planification et l'organisation des trajets et des horaires sont définies par les voyageurs lorsqu'ils réservent leurs trajets. La figure 1 de la page suivante permet de visualiser les différentes lignes et les TAD du réseau.

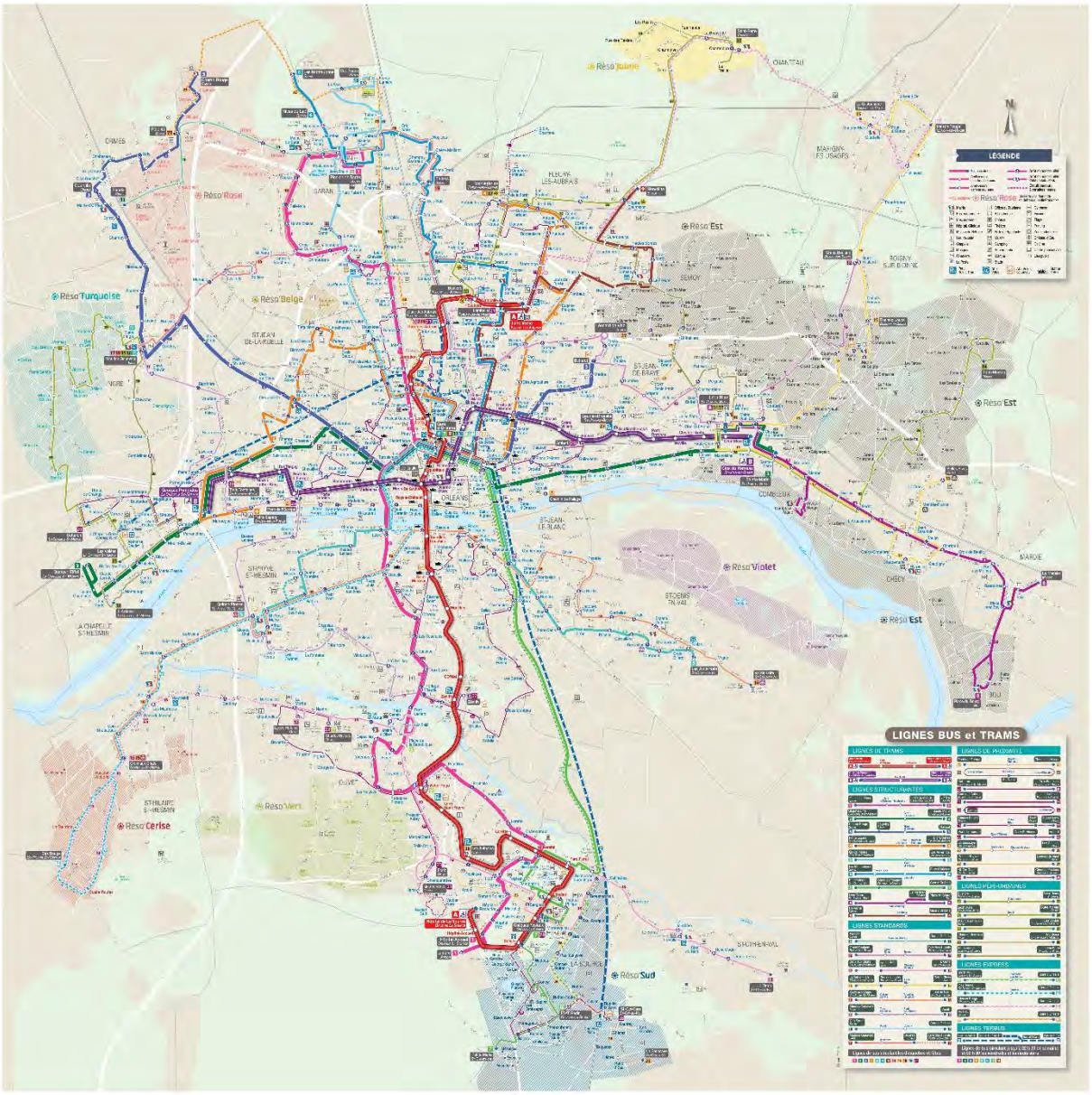


Figure 1: Plan général du réseau TAO
Source : Site du réseau TAO (https://www.reseau-tao.fr/15-Plans-et-fiches-horaires.html)

La direction Innovation, Projets et Systèmes d'Information de Keolis Orléans Métropole souhaite tester une idée innovante d'application web de cartographie dynamique de l'offre de transport à destination de ces services internes majoritairement (le service marketing). Celle-ci a deux finalités potentielles :

-mieux connaître l'offre de transport (savoir théoriquement quelles sont les lignes qui circulent où, à quelle date et à quelle heure) en valorisant des données d'exploitation du réseau, telles que les données horaires fournies par le service d'aide à l'exploitation aux conducteurs de bus, ces données étant régulièrement actualisées en fonction d'événements "prévisibles" comme les manifestations de gilets jaunes, la crise de coronavirus etc.);

-évaluer l'offre de transport et analyser sa pertinence (exemple : la proximité avec les entreprises, les personnes âgées, l'accessibilité du réseau pour les personnes à mobilité réduite etc.) et donc proposer des améliorations (dans tel quartier, il y a de nombreuses entreprises avec

un nombre important de salariés, mais l'arrêt le plus proche est à plus de 300 mètres, il n'est desservi que par une seule ligne de bus, peut-être faudrait-il faire passer plus de lignes différentes, à des fréquences plus élevées à l'heure de la débauche etc.)

Ainsi, ce produit permettrait de valoriser des données de l'offre théorique pour mieux connaître les services proposés aux voyageurs, mais également de les croiser avec d'autres sources d'informations afin d'analyser la pertinence de l'offre et d'aider à la prise de décisions (connaître l'offre dans le temps, étudier la manière dont l'offre s'est adaptée au coronavirus par exemple, anticiper sur les évolutions du réseau, vérifier qu'il n'y ait pas des quartiers mal desservis, identifier de nouvelles opportunités comme par exemple rajouter des arrêts à proximité d'écoles, d'entreprises etc.).

Elle a contacté en fin d'année 2019 le prestataire Neogeo Technologies pour étudier la faisabilité d'un tel projet dont elle ne connaît pas encore précisément les contours. Pour un budget de 30 000 euros, Keolis voudrait tester la possibilité de développer un outil répondant à ces deux finalités avec les données dont elle dispose. Elle souhaite disposer d'un POC (proof of concept) ou démonstrateur à faire tester en interne (au service marketing) pour début Septembre 2020, et ainsi valider ou non la continuation du projet vers un produit complet (voir l'encadré page suivante pour avoir plus de détails sur le POC).

Les représentants de Keolis dans ce projet sont une chargée d'études, le directeur innovation, projets et système d'information, ainsi qu'une chargée d'études en alternance qui a été l'interlocutrice durant le projet. Les représentants côté Neogeo Technologies sont Guillaume Sueur et moi-même.

Le POC dans le développement d'un produit

Différentes étapes se succèdent dans le processus de développement d'un produit. Chacune de ces étapes constitue un sous-projet en lui-même. La figure 2 ci-dessous met en perspective les différentes phases d'un projet (même si selon les types de projet et les sources les termes peuvent varier). Dans le cadre du projet Keolis, la construction du POC a débuté courant Mars 2020, et le client a fixé Septembre 2020 comme échéance.

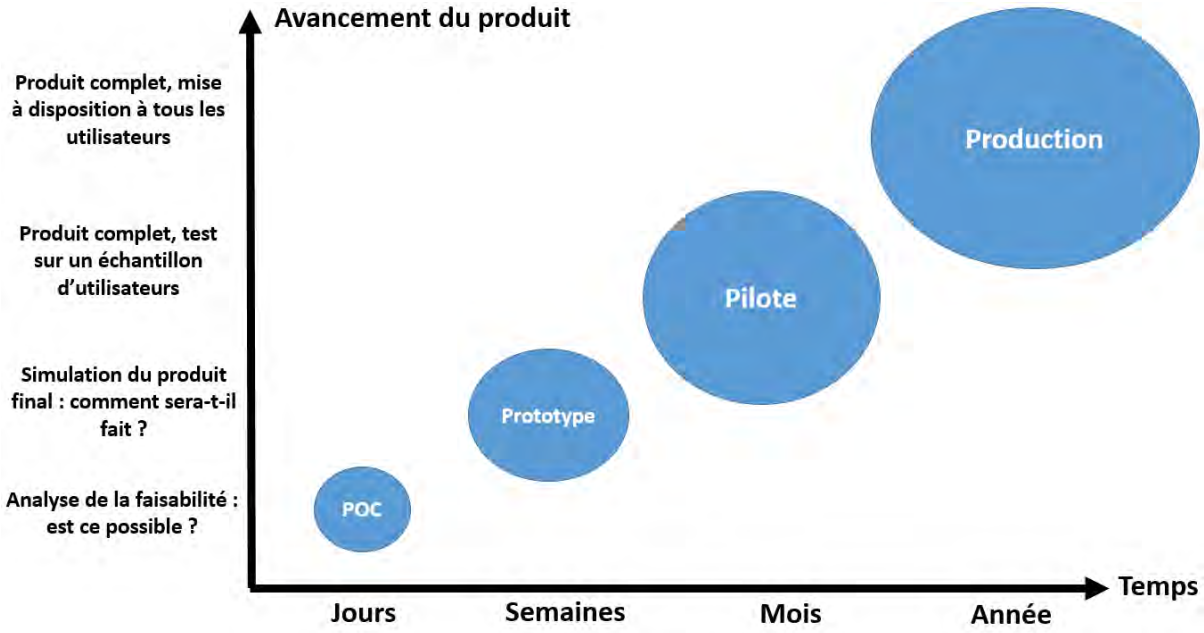


Figure 2: Mise en perspective du POC dans un projet global

Source : Georges (2020) « La méthode 3P pour valider votre solution IoT : Problème, PoC, Prototype » disponible ici : https://www.matooma.com/fr/s-informer/actualites-iot-m2m/comment-realiser-le-prototype-de-mon-objet

Le POC constitue la première étape de la conception d'un produit, et peut se définir comme un prototype minimal visant à donner une idée au client de ce à quoi le produit pourrait ressembler, ou la manière dont l'idée formulée par le client pourrait se concrétiser (ou non). C'est une démarche expérimentale visant à valider des hypothèses : il permet de tester la désirabilité du produit (le client en a-t-il réellement besoin, le désire-t-il vraiment ?) et sa faisabilité (est-il possible de développer ce produit ?). Le POC est réalisé sur une période bien définie et relativement courte avec un investissement minimal. Il permet de prouver la faisabilité d'un projet, d'identifier les potentiels problèmes qui pourraient survenir plus tard, de faciliter le dimensionnement du budget et du produit (mieux identifier les attentes du client et le coût). A l'issue du POC, il doit être décidé de la continuité ou non du projet (GO/NO GO).

S'en suit le développement du prototype, qui permet de répondre à la question "comment sera réalisé le produit ?". Le prototype est une simulation de la façon dont le produit final va fonctionner. Il permet de tester et de valider différentes hypothèses dans un environnement contrôlé.

Durant la phase pilote, le produit est testé en condition réelle sur un panel restreint d'utilisateurs finaux qui feront des suggestions d'améliorations techniques ou ergonomiques. Le but est de comprendre comment le produit sera déployé chez le client.

La production est la mise en service du produit final chez le client, en assurant la livraison continue de nouvelles fonctionnalités, en assurant la maintenance du produit, et en garantissant un support aux utilisateurs. Le produit est alors disponible à tous les utilisateurs finaux.

Neogeo Technologies est une SARL Toulousaine promouvant l'open data et les logiciels open source et qui s'est principalement spécialisée dans la conception de plateformes d'infrastructures de données géographiques. Mais ce marché est limité, et Neogeo Technologies souhaite diversifier son offre. L'entreprise est dans une dynamique d'expansion, c'est pourquoi quatre collaborateurs ont rejoint l'équipe depuis Mars 2020. En fin d'année 2019, une opportunité de diversification des activités s'est présentée avec Keolis, une entreprise privée de gestion des transports en commun ayant contacté Neogeo Technologies pour tester une idée d'application web de cartographie dynamique de l'offre de transport à Orléans. Avec un budget de 30 000 €, Keolis veut disposer d'un POC début Septembre 2020 pour le présenter et le faire tester au service marketing. Si ce dernier ressent la nécessité de disposer d'un tel produit, alors le projet pourra se poursuivre et déboucher sur un produit complet et parfaitement fonctionnel. La mission assignée consiste donc à gérer le projet de réalisation du POC pour Septembre 2020, et de participer à une partie des développements de l'application (la partie backend de l'application, comme cela sera détaillée plus loin dans le rapport).

II. Quelle méthode de gestion de projet mettre en œuvre pour la réalisation d'un POC ?

A. Une méthodologie Agile ou traditionnelle ?

Il existe de nombreuses méthodes de gestion de projet. Les méthodes traditionnelles se caractérisent par une certaine rigidité dans la mesure où le client doit expliciter les fonctionnalités attendues dès le début du projet. Ces méthodes ne prévoient pas d'évolution dans les besoins du client. Une feuille de route pour le déroulement du projet est définie au début du projet. L'objectif est de suivre autant que faire se peut ce plan, et de limiter l'écart entre les résultats attendus et ceux obtenus. Le client a peu de visibilité sur le produit pendant la phase de conception.

A l'inverse, les méthodes Agile prennent en compte les imprévus et l'évolution des besoins du client en privilégiant un modèle de développement itératif (sur plusieurs cycles répétitifs) permettant de construire le produit au fur et à mesure et de manière participative. Ces méthodes se caractérisent par une forte participation des développeurs et des clients, des livraisons fréquentes des nouvelles fonctionnalités, et la prise en compte des modifications dans les besoins du client pendant le projet. Ce sont bien les méthodes Agile qui sont préconisées dans le projet Keolis puisque le client ne sait pas précisément ce qu'il veut, et que ces besoins vont se préciser au fur et à mesure de l'ajout de nouvelles fonctionnalités. Il existe différentes méthodes Agile, la plus utilisée et choisie pour le projet étant la méthode SCRUM. Elle se base sur des rôles, une succession d'itérations appelées sprints, d'évènements (mêlée quotidienne, de revues de sprint, une rétrospective de sprints), et des backlog produits.

B. Une méthode basée sur des rôles

L'approche SCRUM se base sur des rôles comme résumés à la figure 3 ci-dessous.

Figure 3: Les rôles de la méthode SCRUM

Source : Bruno Borghi (2017) « Au-delà de l'équipe Scrum : les rôles dans l'entreprise agile (Partie 1) » dans Les Cahiers Agiles

1. Le product owner : le client, représenté par la chargée d'études en alternance

Le product owner est le représentant du client. Il connaît leurs attentes et donc la manière dont ils imaginent le produit. Il priorise les fonctionnalités à développer ou à corriger et valide/rejette les fonctionnalités terminées. Il répond aux questions de l'équipe de développement et prend les décisions importantes le moment voulu. A l'issue de chaque sprint (expliqué dans le paragraphe suivant), il peut ajouter de nouvelles exigences, en retirer, ou en redéfinir certaines. Ces fonctionnalités définies et priorisées par le product owner sont consignées dans le backlog produit dont il est responsable. Un backlog produit est un document contenant une liste hiérarchisée de "user stories". Une user story se définit par une fonctionnalité (Y), un bénéfice (Z), et un utilisateur bénéficiant de cette fonctionnalité (X) : « En tant que X, je veux Y afin de Z ». Elle peut aussi inclure une description du test attestant de la validité de la fonctionnalité. A chaque sprint sont sélectionnées les fonctionnalités du backlog produit qui devront être réalisées au cours du sprint (dans la bibliographie le terme sprint backlog est parfois utilisé). Ce document est amené à évoluer au fur et à mesure des sprints, de l'évolution des attentes du client et des concertations avec l'équipe de développeurs.

Dans le cadre du projet keolis, le product owner peut être assimilé à la chargée d'études en alternance puisqu'elle a été l'interlocutrice privilégiée durant le projet, la personne qui répondait aux questions (compréhension du jeu de données, sur la manière dont était utilisé le format GTFS détaillé plus loin, des précisions par rapport à certaines fonctionnalités attendues

etc.), de la transmission du backlog produit et des données etc. Elle représente le client final : le service marketing.

2. L'équipe de développement : une équipe de trois personnes essentiellement

L'équipe de développement est chargée de développer les fonctionnalités prévues pour un sprint donné en se basant sur les user stories du sprint backlog. A la fin de chaque sprint elle fournit au product owner les fonctionnalités initialement prévues dans le sprint en cours et développées au cours des sprints antérieurs. Ces fonctionnalités doivent être terminées et utilisables. L'équipe travaille en interaction avec le product owner qui leur donne des précisions éventuelles sur ce qui est attendu.

Dans le cadre du projet Keolis, le développement back-end de l'application (détaillé plus loin dans le rapport) a été réalisé par moi-même, avec l'assistance de Guillaume en ce qui concerne certains choix de traitements des données (exemple : est-il pertinent de prendre tous les arrêts physiques ?) ou des difficultés techniques (exemple : installation d'un environnement virtuel, développement du code python pour l'intégration d'un nouveau flux GTFS). Le développement front-end de l'application a été réalisé par un développeur débutant dans la programmation (Florent) et assisté par un développeur expérimenté (Maël) qui travaillaient tous deux sur d'autres projets, tandis que j'ai pu y consacrer quatre mois complets de mon stage. L'administrateur système a aussi apporté un soutien en installant les logiciels sur le serveur (mapserveur, postgre, Python, diverses bibliothèques etc.). Le développement back-end inclut la gestion et l'exploitation de la base de données postGis, le développement en Python de l'API (application programming interface) via le framework Flask, des points avec le développeur front-end pour coordonner les développements du back-end à ceux du front-end (exemple : le client souhaitant telle fonctionnalité, le front-end peut-il gérer seul la fonctionnalité, ou lui faut-il des données du back-end, et si oui lesquelles ?).

3. Le scrum master : un rôle assez ambigu selon les sources

Le scrum master est chargé de faire appliquer la méthode SCRUM. Auprès du product owner, le scrum master vérifie que les objectifs vis-à-vis du produit soient bien explicites et compris de toute l'équipe en veillant à la clarté des éléments du backlog produit que le product owner rédige. Auprès de l'équipe de développement, le scrum master veille à ce que les développeurs soient productifs, organisés, qu'il n'y ait pas d'obstacles à leur progression (identification, priorisation et résolution des problèmes). Le scrum master planifie et anime les divers sprints et fait le bilan des réunions.

Bien qu'il n'y ait pas eu de scrum master officiellement nommé, les tâches de gestion de projet que j'ai pu mener pourraient éventuellement s'apparenter à celles d'un scrum master, même si dans la plupart des sources le scrum master n'est pas associé à un chef de projet.

Par rapport à l'équipe de développement, je servais d'intermédiaire entre les développeurs front-end et le product owner : j'indiquais aux développeurs quelles étaient les fonctionnalités qu'attendait le product owner pour le prochain sprint, j'évaluais avec eux la faisabilité des fonctionnalités dans les temps impartis, et si la fonctionnalité demandait trop d'efforts par rapport au temps ou au budget alors je devais expliquer au client que sa demande ne pouvait être satisfaite et la raison. Je remontais aussi les questions des développeurs au product owner. Dans la mesure où je développais aussi la partie back-end de l'application, je sollicitais directement le product owner par mail ou échange téléphonique pour des questions relatives aux données fournies (compréhension des données métiers notamment les arrêts de TAD, du format GTFS, de la manière dont le client utilisait le format), ou pour lui faire part d'anomalies constatées dans les données (exemple : des horaires dépassant 24 : 00 : 00).

En termes d'application des méthodes SCRUM, nous planifions avec le product owner les réunions de fin de sprint au cours desquelles je présentais les avancées du POC. J'étais chargée de vérifier le respect des délais de développements et du fonctionnement des fonctionnalités pour s'assurer que le prototype fonctionne lors des démonstrations. Je faisais également part des éventuelles difficultés rencontrées et réfléchissais avec le product owner à des alternatives.

C. Les attentes du client : le backlog produit dans sa version initiale

Le tableau 1 de la page 13 résume les user stories (donc les attentes du client vis-à-vis du POC) telles que transmises par le product owner début Mars 2020. Le client a organisé les fonctionnalités autour de six thématiques :

-l'observation de l'offre de transport qui prend uniquement en compte la composante spatiale, c'est-à-dire l'affichage sur une carte des lignes de bus, de tram et des zones de TAD et des arrêts associés. Elle fait référence à l'affichage de données géographiques, donc à la gestion des erreurs de géométrie (géométrie non valide pour un objet donné) et des erreurs de topologie (donc les relations entre les objets comme les recouvrements entre les objets, la connectivité d'un réseau routier etc.)

-l'observation de l'offre de transport dans le temps qui intègre la composante temporelle, c'est-à-dire l'affichage de l'offre de transport en fonction de paramètres temporels : une date ponctuelle, ou éventuellement plusieurs dates ainsi qu'une plage horaire. Les données n'étant pas toutes disponibles sous forme de tables et donc directement exploitables dans une base de données, le développement de ces fonctionnalités nécessite un travail d'abstraction : réflexion d'un modèle conceptuel de données pour retranscrire une information dans une base de données où elle pourra être exploitée. Un travail d'archivage (l'information doit être saisie dans des tables dans une base de données), d'accès et d'analyse de la donnée (construction de requêtes plus ou moins complexes pour accéder à l'information correspondant aux critères de jour et de plage horaire et répondre aux questions : quelles sont les lignes de bus/tram et les TAD disponibles à telle(s) date(s) entre telle et telle heure ? Quels sont les arrêts desservis ?) est également nécessaire. Le travail d'affichage du résultat a déjà été explicité avec la thématique d'observation de l'offre de transport.

-l'analyse de l'offre qui prend en compte les fonctionnalités qui aideront l'utilisateur à mieux évaluer la pertinence de l'offre en la croisant avec des informations complémentaires. Pour

certaines informations, cela nécessitera uniquement un travail d'affichage (travail sur la sémiologie cartographique pour l'affichage de données de population etc.) mais pour d'autres un travail d'analyse également (pour ce qui concerne les fréquences de passage sur un tronçon)
-**la mise à jour des données** comprend deux types de fonctionnalités : celles mettant à jour les données permettant de mieux connaître l'offre de transport (modification d'un arrêt, rajout des données du dernier mois etc.), et celles permettant d'actualiser les données complémentaires
-**l'ergonomie et la lisibilité** concernent les caractéristiques de l'interface de l'application (appelée le front-end) avec laquelle l'utilisateur interagit pour afficher les informations qu'il recherche, sur n'importe quel type d'écran
-**le service web** : les fonctionnalités doivent se présenter sous la forme de web services c'est-à-dire sous la forme d'un ensemble de composants logiciels hébergés sur un serveur et capables d'échanger des informations (des pages web au format HTML par exemple ou des données organisées sous le format JSON ou XML) avec des applications clientes (exemple : navigateur web, QGIS) par l'intermédiaire du web (protocole http).

En termes de priorisation dans le développement des fonctionnalités :
1-le POC doit d'abord permettre de mieux connaître l'offre de transport actuelle. La première tâche consiste donc à afficher l'offre de transport pour une date donnée et dans une plage horaire choisie.
2-Dans la mesure où les données de tram et de bus sont mises à jours régulièrement (environ tous les mois un nouveau jeu de données est produit pour indiquer les horaires théoriques prévus pour le mois suivant) et que celles-ci se présentent sous un format particulier faisant intervenir plusieurs fichiers (format GTFS : General Transit Feed Specification), la mise en place d'une fonctionnalité permettant d'intégrer facilement ces nouvelles données dans la base de données constitue la deuxième priorité.
3-Une fois que le POC permet de déterminer quelles sont les lignes de bus et de tram et de TAD disponibles avec leurs arrêts pour une plage horaire et une date choisies, il est alors possible de s'intéresser aux fonctionnalités permettant d'analyser/d'évaluer l'offre actuelle (rajout de données complémentaires sur l'interface de l'application).
4-Une fois l'offre de transport connue et analysé, il est alors possible de s'intéresser au réseau futur, à la mise à jour des données qui sont moins fréquemment modifiées, et à améliorer la lisibilité et l'ergonomie.

Les parties précédentes ont permis d'introduire les rôles de la méthode SCRUM et d'associer chaque partie prenante au projet Keolis à un/deux de ces rôles : l'alternante côté Keolis en tant que product owner et représentant le service marketing, l'équipe de développement composée d'un développeur débutant assisté par un développeur expérimenté et moi-même avec le soutien de Guillaume, puis le rôle de scrum master qui n'a pas été officiellement attribué, mais qui pourrait être associé aux tâches de « gestion de projet » que j'ai pu réaliser. Le dernier paragraphe a introduit les attentes du client vis-à-vis du POC au début du projet grâce au backlog produit fourni par le product owner courant Mars 2020. Bien qu'il n'y avait pas de priorisation des tâches dans le backlog, et qu'il était difficile d'anticiper la difficulté ou la durée qu'allait nécessiter le développement des diverses fonctionnalités, un premier ordre de priorité a été imaginé : d'abord connaître l'offre de transport actuellement proposée, puis

développer les fonctionnalités qui vont permettre de l'analyser, puis enfin s'intéresser aux perspectives avec le réseau futur et à la mise à jour des données rarement et facilement actualisées. L'objet du paragraphe suivant et d'expliquer le processus itératif de la méthode SCRUM qui permet de redéfinir régulièrement les objectifs à atteindre, et de construire progressivement le POC en collaboration avec le product owner.

Utilisateur	Bénéfice de la fonctionnalité	Fonctionnalité	Compléments
Service marketing	visualiser l'offre de transport au cours du temps	observer l'offre de transport en fonction de la période, du jour et de l'heure	Une option permet de modifier la période visible à l'écran ex : le 10/03, le 10/03 après 21h, le 10/03 entre 8h et 9h, du 10/03 au 12/03 -toutes les lignes circulant à la période définie sont affichées, les autres cachées -à valider : les arrêts compris -à valider : montrer uniquement la portion de ligne circulant pendant la période
		observer le réseau futur (2021)	une option permet d'accéder au réseau futur comprenant TAD, Tram et Bus sous la forme d'une image fixe
	visualiser l'offre de transport	sélectionner une ou plusieurs lignes pour les visualiser plus facilement	Pouvoir sélectionner une ou plusieurs lignes et voir uniquement ces lignes et les arrêts correspondants -à valider : pouvoir sélectionner un itinéraire spécifique de la ligne -à valider : conserver le filtre sur les lignes lorsque la période varie
		visualiser les zones de TAD, et l'offre de transport associée	pour chacune des zones de TAD, afficher la zone avec la couleur adéquate et les arrêts
	Analyse de l'offre	Observer le nombre de courses par tranche horaire et par tronçon afin d'étudier la fréquence de passage des bus	Avoir un mode de vue permettant de sélectionner des tronçons -un tronçon = une portion de ligne (ou superposition si plusieurs lignes) entre 2 arrêts successifs -à la sélection d'un tronçon, pouvoir voir le nombre de courses sur la période horaire actuelle -lorsque la période horaire est modifiée, le nombre de courses pour le tronçon évolue -à définir : un tronçon par sens / affichage des informations pour les 2 sens séparément à la sélection du tronçon / on additionne les 2 sens dans le calcul ?
		Afficher diverses informations complémentaires permettant de mieux analyser la pertinence de l'offre	Les données complémentaires à afficher : -stations vélo + -parkings à vélos sécurisés -répartition de l'emploi dans la métropole -répartition de la population dans la métropole -plan cadatral -données topographiques (équipements de l'agence d'urbanisme d'Orléans Métropole - 2012) -estimation carroyée de la population à 200 m -orthophotoplan à 5 cm (2017)
Producteur de données	mise à jour des données	pouvoir mettre à jour les cartes disponibles, modifier les lignes ou arrêts existants	Pour la V1, la mise à jour peut se faire directement par la mise à jour de la base de données. Définir des processus simples pour : -mettre à jour une carte (ex : remplacer la carte de répartition de l'emploi de 2017 par 2019) -mettre à jour un élément du réseau (ex : rajouter une nouvelle zone de TAD, modifier des arrêts)
Service marketing	ergonomie, lisibilité	bénéficier d'une interface intuitive et ergonomique	interface simple, ne nécessitant pas de connaissances particulières en SIG, avec un contraste de couleur suffisant pour les textes
		observer de manière claire les tronçons en commun (y compris sur les virages)	Afficher les lignes côte-à-côte (et non superposées) sur les troncs communs quel que soit le zoom
		comprendre l'association entre les étiquettes et les objets	Quel que soit le niveau de zoom, il faut une absence de superposition des étiquettes et comprendre facilement à quelle entité correspond chaque étiquette
		pouvoir utiliser l'interface correctement avec un ordinateur ou une tablette	
	service web	disposer des données sous forme d'un service web pour accéder aux données grâce à un logiciel SIG (comme QGIS etc.)	

Table 1: Synthèse du backlog produit remis par le client en mars 2020

D. Une méthode basée sur une succession de cycles courts

1. Un projet segmenté en sprints : la théorie

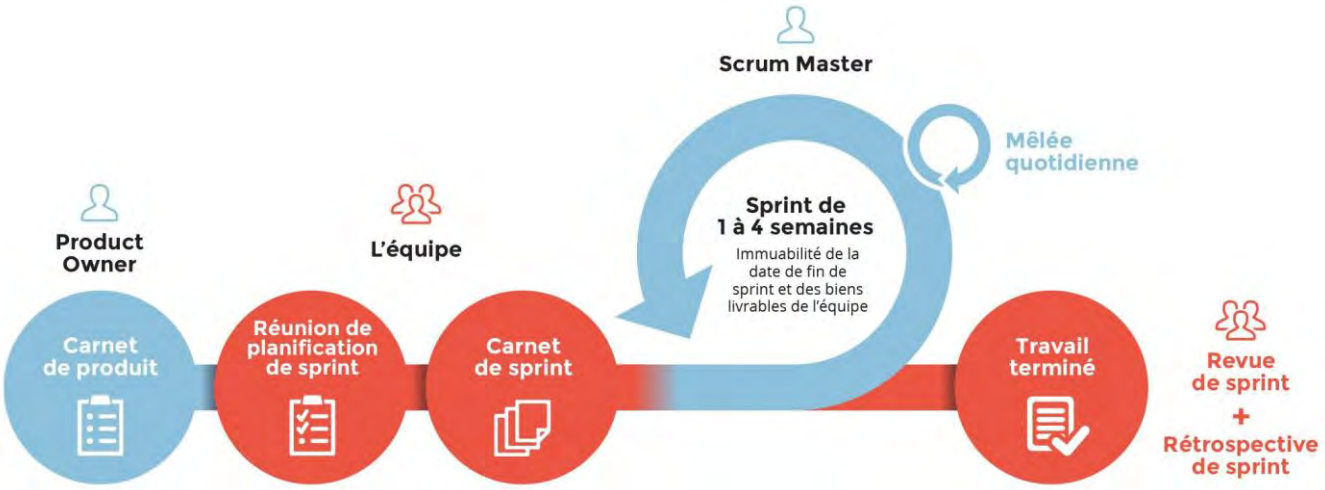


Figure 4: Schéma de l'application de la méthode SCRUM

Source : Grégory (2018) « Agile/Scrum : Ne pas tout mélanger en gestion de projet ! » disponible https://bubbleplan.net/blog/agile-scrum-gestion-projet/ consulté le 27/08/2020

La première étape est la réunion de planification de sprint. Elle permet de sélectionner les tâches prioritaires du backlog produit à réaliser pour le sprint à venir et qui dure tout au plus un mois. Elle fait intervenir toute l'équipe SCRUM et permet de répondre aux deux questions :
-« Que peut-on livrer comme incrément à la fin du sprint ? » : Le product owner sélectionne les tâches du backlog produit qu'il souhaite voir se réaliser en se basant sur le dernier incrément réalisé, sur l'avis de l'équipe de développement qui doit évaluer ses capacités, et sur l'objectif global que le product owner associe au sprint. C'est l'équipe de développement qui fixe le nombre de tâches qu'elle peut réaliser parmi celles sélectionnées par le product owner. Les tâches sont consignées dans un sprint backlog.
-« Comment le travail choisi sera-t-il réalisé ? » : Cela consiste pour l'équipe de développement à planifier le travail nécessaire pour concrétiser les tâches et construire un incrément fonctionnel fini à l'issu du sprint. Si l'effort estimé est trop grand, elle peut réajuster les tâches à réaliser avec le product owner qui fera des compromis. Avant la fin de la réunion, l'équipe de développement doit décomposer le travail prévu pour les prochains jours et continuera à s'auto-organiser tout au long du sprint.

Après cette réunion s'ensuit la phase de développement, le sprint en lui-même. Tous les jours du sprint, la mêlée quotidienne d'une quinzaine de minutes permet à l'équipe de développement de planifier les tâches à réaliser dans la journée, d'évaluer ce qui a été réalisé, d'améliorer la collaboration au sein de leur équipe, d'identifier les obstacles, de favoriser la prise de décisions. C'est le scrum master qui est chargé de la tenue de la réunion mais c'est l'équipe de développement qui est responsable de son déroulement. Tout au long du sprint, l'équipe de développement se concentre sur la réalisation des tâches du sprint backlog. Le but du sprint est de développer chaque fonctionnalité en totalité (de la conception au test) au fil de l'eau, et selon

l'avancement il est possible de rajouter ou de supprimer des tâches en accord avec le product owner. Le sprint s'appuie sur une collaboration permanente entre l'équipe de développement, le scrum master et le product owner. Pour avoir une vision claire de l'avancement du projet, il est conseillé dans la bibliographie de disposer d'un tableau des tâches visible par tous les membres de l'équipe de développement, afin qu'à tout moment il soit possible de voir l'avancement des tâches et du projet. Les tâches sont réparties sur 3 ou 4 colonnes : les tâches à faire, les tâches attribuées et en cours, les développements réalisés et à tester, les développements terminés.

A l'issue du sprint, la revue de sprint permet de présenter l'incrément et d'adapter le product backlog si besoin. Cette réunion fait intervenir l'équipe SCRUM au complet, et tout acteur du projet intéressé (comme les utilisateurs finaux par exemple). L'équipe de développement présente au product owner les fonctionnalités qui ont été développées durant le sprint, les difficultés éventuellement rencontrées et les problèmes résolus. Le product owner valide ou non ces fonctionnalités en fonction de ce qui est attendu dans le backlog produit. L'ensemble de l'équipe SCRUM discute de ce qui sera fait pour le prochain sprint, ces éléments étant explicités lors de la réunion de planification de sprint.

La rétrospective de sprint est une réunion suivant la revue de sprint et faisant intervenir l'équipe SCRUM. Elle permet de faire un point sur la manière dont s'est déroulé le sprint pour identifier les points ayant bien fonctionnés et les pistes d'amélioration dans l'application de la méthode SCRUM pour les prochains sprints (personnes, relations, utilisation des outils, processus).

2. Une mise en œuvre plus ou moins fidèle à la théorie

La revue de sprint en elle-même était suffisante pour faire le point sur l'avancement du POC et définir les tâches à faire pour la suite. Ces réunions faisaient intervenir Guillaume et moi-même côté Neogeo Technologies, et côté Keolis le product owner, le directeur innovation et une chargée d'études essentiellement. Je présentais les diverses fonctionnalités développées, les difficultés rencontrées et les éventuels choix/compromis que devait faire le product owner vis-à-vis de certaines fonctionnalités. Nous discutions de potentielles fonctionnalités qu'il serait pertinent de mettre en place ou non (exemple : inutilité de créer un filtre pour ne sélectionner que les lignes de bus, de tram ou de TAD puisqu'il est possible de les sélectionner manuellement une par une, voire de tout sélectionner/désélectionner en même temps), et des tâches à réaliser pour le sprint suivant. A l'issue de ces réunions en visioconférence, le product owner nous envoyait à moi-même et à Guillaume un mail récapitulatif des points abordés lors de la revue, et les tâches à effectuer pour le prochain sprint telles que convenues lors de la réunion (ce qui peut correspondre au sprint backlog de la bibliographie), avec une date approximative pour la prochaine revue de sprint.

Une fois cette réunion terminée, je faisais un point informel avec Florent du retour du client sur les fonctionnalités développées et des tâches à réaliser pour le prochain sprint. Dans la mesure

où Florent était chargé du développement de l'interface visible par l'utilisateur (le front-end) et moi du back-end de l'application, nous répartissions les tâches en trois catégories :
-les tâches qui concernaient uniquement l'interface et que Florent pouvait développer de son côté de manière indépendante (exemple : élargir le panneau d'affichage des horaires de passage pour les arrêts)
-les tâches qui ne dépendaient que de la base de données ou de l'API (application programming interface, détaillée plus loin) et qui ne concernaient donc que moi (exemple : modifier des données de la base de données, modifier les options d'une liste déroulante dans le code de l'API)
-les tâches dépendant à la fois du back-end et du front-end (donc les tâches impliquant un échange d'informations du back-end vers le front-end)
Dans le troisième cas, Florent devait attendre que je finisse mes développements pour pouvoir faire les siens, c'est la raison pour laquelle j'exécutais ces tâches en priorité par rapport à celles qui ne dépendaient que de moi, afin qu'il puisse avoir le temps de faire ses développements avant la fin du sprint.
Ces points peuvent être assimilés aux réunions de planification de sprint dans le sens où les tâches étaient priorisées et réparties.

Il n'y avait pas de mêlées quotidiennes, mais des points avec Florent lorsque cela était nécessaire, que ce soit pour me transmettre des questions relatives aux souhaits du product owner concernant le fonctionnement de l'interface (exemple : le client souhaite-t-il que les buffers s'affichent automatiquement lorsqu'il a sélectionné une date et une plage horaire, ou préfère-t-il un bouton pour sélectionner/désélectionner les buffers, ou bien que le buffer s'affiche sur un arrêt quand on clique dessus etc. ?), pour vérifier la validité de certains résultats que le back-end lui renvoyait, ou pour faire part de difficultés ou d'impossibilité de répondre à une demande au vu du temps. Je faisais ainsi l'intermédiaire entre les développeurs et le product owner avec lequel j'avais des échanges réguliers, que ce soit par téléphone, visioconférence ou mails pour clarifier certains points sur ses attentes, sur la structure des données fournies etc. Une fois les développements de Florent terminés, les diverses fonctionnalités étaient testées pour s'assurer qu'elles étaient opérationnelles avant la revue de sprint.

Les différentes tâches explicitées dans le mail du product owner étaient reportées sur l'outil Redmine pour que tout le monde (Guillaume, moi-même, Florent, Maël mais aussi le product owner) puisse voir l'avancement des tâches. Redmine est une application web gratuite de gestion de projet qui permet de créer des tickets (de déposer une nouvelle demande, avec un nom, une description, une pièce jointe etc.) et de l'assigner à une personne qui sera chargée de traiter la demande. Il est possible de renseigner l'état d'avancement du traitement du ticket, de renseigner le temps passé sur cette tâche, ou encore de donner une priorité etc. Assez rapidement nous avons transmis l'url du POC au product owner pour que celui-ci puisse le manipuler. Si des anomalies étaient constatées, il pouvait déposer un ticket (une nouvelle tâche) sur la plateforme Redmine. Bien que ce soit un outil proposant de nombreuses fonctionnalités, il a essentiellement été utilisé pour lister les tâches à réaliser pendant le sprint, les assigner, et voir leur état d'avancement. La figure 5 page suivante donne un exemple de ticket déposé par le product owner sur une correction à effectuer.

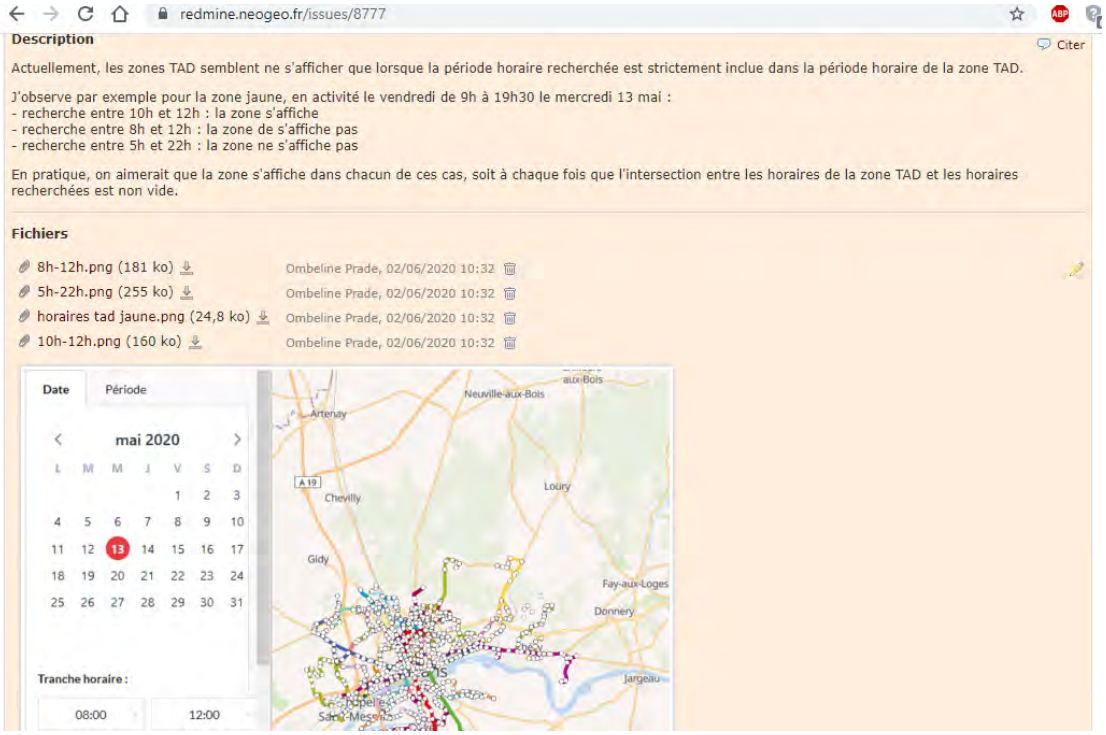


Figure 5: Un exemple de ticket dans l'outil de gestion de projet Redmine

Le tableau 2 des deux pages suivantes permet de lister les différentes tâches qu'il fallait réaliser tout au long des sprints, avec une décomposition en sous-tâches (les sous tâches concernant le back-end sont davantage explicitées puisque cela concernait les développements que j'ai réalisés, les sous-tâches côté front-end ne sont pas détaillés). Le GANTT permet quant à lui de visualiser dans le temps les tâches effectuées durant le stage.

sprint	Tâche exprimée par le product owner
1	affichages lignes de bus GTFS en fonction d'un jour et d'une plage horaire
	affichage TAD en fonction d'un jour et d'une plage horaire
2	Affichage de fonds de cartes supplémentaires
	intégrer le dernier GTFS sorti dans la base de données
	affichage des arrêts de GTFS desservis à la date et dans la plage horaire choisies
	affichage des arrêts de TAD pouvant être desservis à la date et dans la plage horaire choisies
	réfléchir à la fonctionnalité d'import d'un nouveau flux GTFS dans la base de données sans devoir le faire à la main
	coordonner l'affichage des arrêts d'une ligne ou d'une zone TAD à l'affichage ou non de la ligne ou du TAD (ex : quand je sélectionne la ligne 13, les arrêts correspondant s'affichent, et quand je désélectionne la ligne les arrêts disparaissent aussi)
	introduire les services standards : afficher les TAD, les lignes de bus et leurs arrêts associés pour un service standard et une plage horaire choisis
3	rajouter les nouveaux arrêts de TAD dans la base de données
	correction de la requête pour l'affichage des TAD
	rajouter un bouton de sélection/désélection de toutes les lignes simultanément
	échanger l'ordre des onglets en mettant d'abord celui des services standards
	mettre le fond osm pastel par défaut
	permettre de cacher le panneau de sélection pour voir la carte en grand
	revoir la sélection des horaires (un slider, avec les horaires variant de 3h à 3h30 j+1, proposer des horaires au pas de 30 min)
	rajouter une barre de recherche des arrêts en mode full text, une fois l'arrêt tapé zoomer sur celui-ci
	au clic sur un arrêt, afficher les horaires de passage
	fusionner les 2 listes déroulantes avec le choix des jours (du lundi au vendredi, samedi, dimanche), et des vacances (période scolaire, petites vacances, été) en une seule liste déroulante avec 9 choix
	introduire la liste déroulante du choix de la quinzaine
4	pour l'affichage des arrêts : prendre la moyenne des points géographiques et non les coordonnées minimales
	rajouter les derniers GTFS
	corriger l'affichage des horaires pour les arrêts : -pour les arrêts desservis à la fois par les TAD et les lignes de bus, afficher dans un onglet les horaires de passage pour les bus, et avoir un autre onglet pour le TAD en indiquant le message 'transport à la demande. Horaires non disponibles' -pour les arrêts uniquement desservis par les TAD, afficher le message 'transport à la demande. Horaires non disponibles' -pour les arrêts desservis que par les lignes de bus, afficher les horaires de passage

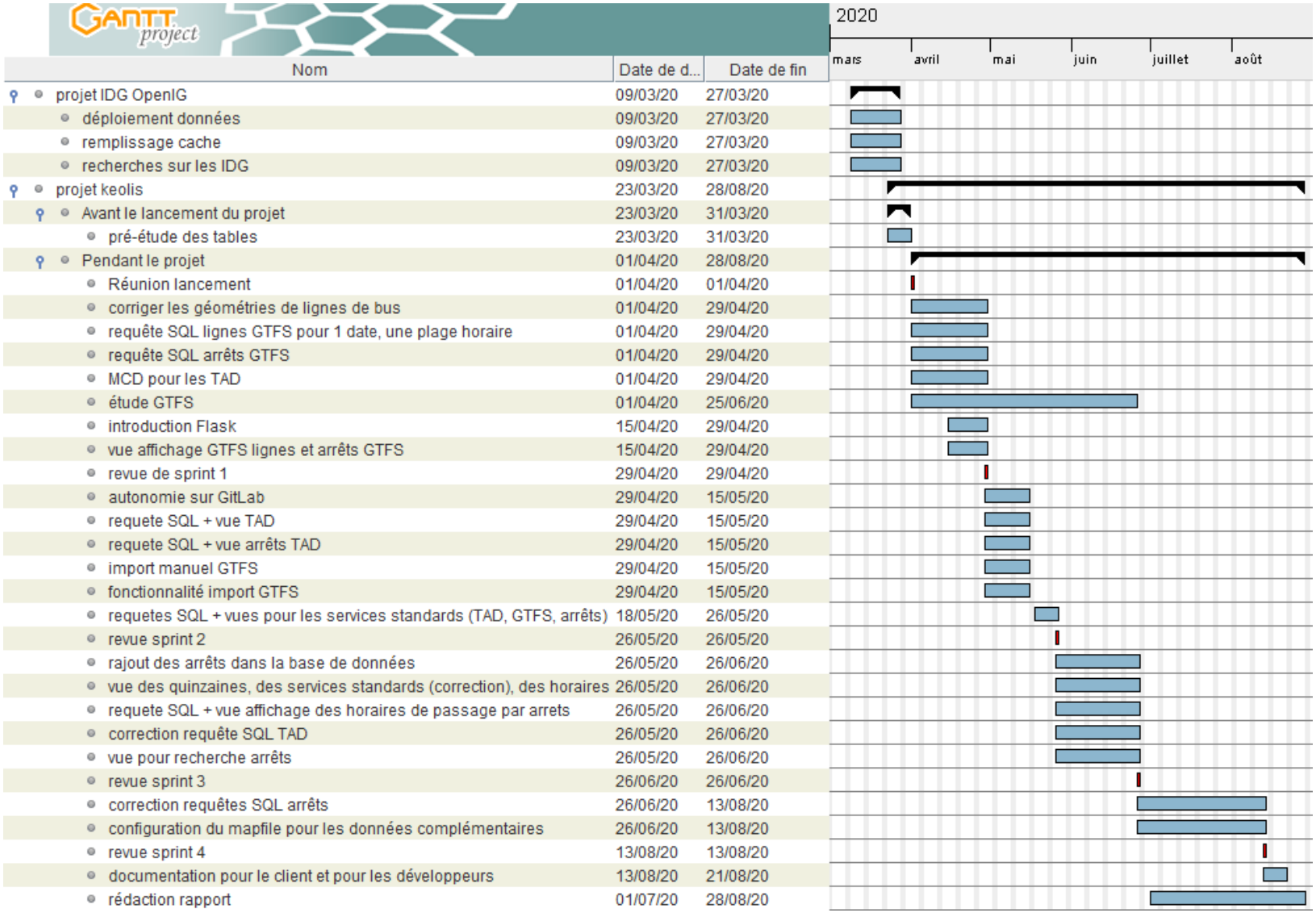
sprint	Tâche exprimée par le product owner
	rajouter un bouton permettant d'afficher des buffers de 300 m autour des arrêts affichés sur la carte
	bouton pour afficher une couche fixe montrant tous les arrêts existants (arrêts physiques et non virtuels) et pouvoir différencier 4 types d'arrêts à travers différents symboles : -arrêt sans abribus et non accessible aux personnes à mobilité réduite (PMR) -arrêt sans abribus et accessible aux PMR -arrêt avec abribus et non accessible aux personnes à mobilité réduite (PMR) -arrêt avec abribus et accessible aux PMR
	bouton pour afficher une couche fixe des arrêts accessibles aux PMR (avec et sans abribus en les différenciant)
	bouton pour afficher une couche fixe de répartition des stations de vélos
	bouton pour afficher une couche fixe de répartition des stations de vélos sécurisées
	bouton pour afficher une couche fixe de la répartition des salariés par entreprise en 2020
	bouton pour afficher une couche fixe de la répartition des habitants de plus de 55 ans en 2015
	bouton pour afficher une couche fixe de données carroyées de population en 2015
5	corriger des intitulés de certaines légendes de couches fixes
	modifier la couleur des zones TAD
	corriger l'affichage des horaires pour les arrêts
	pour l'affichage des données complémentaires (les couches fixes), modifier la présentation pour cacher la légende et au clic pouvoir la dérouler
	pouvoir afficher par défaut la quinzaine correspondant à la date actuelle dans la liste déroulante des quinzaines
	pourvoir afficher par défaut le service standard correspondant au jour actuel
	fournir une documentation de la structure de la base de données
	fournir une documentation sur l'installation des divers modules nécessaires sur un serveur pour faire fonctionner l'application
	pouvoir utiliser l'application sur portable

-------- : tâche faisant intervenir uniquement des développements front-end (Florent)

-------- : tâche faisant intervenir uniquement des développements back-end (moi)

-------- : tâche faisant intervenir des développements front-end et back-end

Table 2: Récapitulatif des différentes tâches à réaliser pour les différents sprints



Nom	Date de d...	Date de fin
projet IDG OpenIG	09/03/20	27/03/20
déploiement données	09/03/20	27/03/20
remplissage cache	09/03/20	27/03/20
recherches sur les IDG	09/03/20	27/03/20
projet keolis	23/03/20	28/08/20
Avant le lancement du projet	23/03/20	31/03/20
pré-étude des tables	23/03/20	31/03/20
Pendant le projet	01/04/20	28/08/20
Réunion lancement	01/04/20	01/04/20
corriger les géométries de lignes de bus	01/04/20	29/04/20
requête SQL lignes GTFS pour 1 date, une plage horaire	01/04/20	29/04/20
requête SQL arrêts GTFS	01/04/20	29/04/20
MCD pour les TAD	01/04/20	29/04/20
étude GTFS	01/04/20	25/06/20
introduction Flask	15/04/20	29/04/20
vue affichage GTFS lignes et arrêts GTFS	15/04/20	29/04/20
revue de sprint 1	29/04/20	29/04/20
autonomie sur GitLab	29/04/20	15/05/20
requete SQL + vue TAD	29/04/20	15/05/20
requete SQL + vue arrêts TAD	29/04/20	15/05/20
import manuel GTFS	29/04/20	15/05/20
fonctionnalité import GTFS	29/04/20	15/05/20
requetes SQL + vues pour les services standards (TAD, GTFS, arrêts)	18/05/20	26/05/20
revue sprint 2	26/05/20	26/05/20
rajout des arrêts dans la base de données	26/05/20	26/06/20
vue des quinzaines, des services standards (correction), des horaires	26/05/20	26/06/20
requete SQL + vue affichage des horaires de passage par arrets	26/05/20	26/06/20
correction requête SQL TAD	26/05/20	26/06/20
vue pour recherche arrêts	26/05/20	26/06/20
revue sprint 3	26/06/20	26/06/20
correction requêtes SQL arrêts	26/06/20	13/08/20
configuration du mapfile pour les données complémentaires	26/06/20	13/08/20
revue sprint 4	13/08/20	13/08/20
documentation pour le client et pour les développeurs	13/08/20	21/08/20
rédaction rapport	01/07/20	28/08/20

Figure 6: GANTT

3. Regard critique sur la gestion de projet et points d'amélioration

Les deux principales difficultés rencontrées ont été la communication au sein de l'équipe et la gestion du temps / l'anticipation. Cela s'est manifesté la veille d'une revue de sprint, lorsque Florent avait terminé ses développements, transféré son code sur le serveur et que le prototype cessa de fonctionner. Dans la précipitation il a été possible d'identifier l'erreur et de la corriger/masquer le lendemain matin de sorte que l'incrément présenté l'après-midi soit fonctionnel et réponde aux objectifs initialement prévus pour le sprint. Cela a généré du stress, tant pour moi qui m'imaginais devoir expliquer au product owner et ses deux supérieurs que rien ne fonctionnait et qu'il était impossible de leur présenter quoi que ce soit, que pour Florent qui a dû revoir son code en urgence (et moi aussi revoir mon code dans la précipitation). Mais que s'était-il passé pour en arriver à cette situation ?

La première raison est un manque de communication entre moi-même et Florent : dans le développement back-end de la fonctionnalité je n'avais pas pris en compte certaines situations (en l'occurrence le cas où l'utilisateur ne sélectionne aucune valeur dans une liste déroulante), alors que les développements front-end de Florent permettaient ces situations que je n'avais pas gérées (par défaut sur l'interface, la liste déroulante ne propose aucune valeur, donc si l'utilisateur clique directement sur le bouton de chargement, alors le prototype plante). Il a fallu un certain temps pour comprendre cette erreur. Si dès le début il y avait eu une meilleure communication avec Florent, que je lui avais explicité le raisonnement que j'avais utilisé pour mon code, alors nous aurions géré le cas plus en amont dans le développement, et pas remarqué le disfonctionnement au dernier moment.

Mais cela a permis de mettre en évidence une deuxième erreur : la gestion du temps. En effet, nous avions négligé la phase de test et pas prévu suffisamment de temps pour les faire et surtout corriger les éventuelles erreurs. Il aurait été nécessaire :
-de prévoir une semaine avant chaque revue de sprint pour faire les tests : est-ce que ma fonctionnalité est opérationnelle quand je la teste toute seule ? Est-ce que lorsque je l'intègre au prototype, toutes les fonctionnalités restent opérationnelles ou bien est-ce que certaines se mettent à planter ? Si oui alors identifier la ligne dans le code générant cette erreur, essayer de la corriger. S'il n'y a pas le temps alors expliquer au product owner que telle ou telle fonctionnalité n'est plus opérationnelle mais que le problème a été identifié et est en cours de traitement
-pour faire ces tests, considérer que l'utilisateur ne soit pas logique, et donc ne pas hésiter à cliquer sur tous les boutons de manière répétée, tester des choses improbables pour déceler les cas particuliers où le prototype se met à ne plus fonctionner, et gérer ces cas
-garder une trace de ces phases de test dans un tableau excel en indiquant pour chaque fonctionnalité la date de test, si le code back-end fonctionne, si le code front-end fonctionne, si non quels sont les bugs rencontrés, à quelles dates ils ont été corrigés
-compléter/actualiser ce tableau au fur et à mesure que des tests sont réalisés tout au long du projet

Une autre difficulté a été le contexte dans lequel le projet a démarré. En effet le stage a commencé le 9 mars, soit une semaine avant le confinement qui a été décrété avant que je n'ai eu le temps de me familiariser avec le fonctionnement de l'entreprise, du matériel (travailler avec un mac demande un petit temps d'adaptation) ou de faire connaissance avec les collègues de travail. La première réunion de lancement du projet s'est déroulée le 9 Avril par visioconférence sans que Florent et Maël n'y assistent, si bien que le premier sprint s'est déroulé chacun de notre côté sans qu'il n'y ait de véritable communication puisque je ne savais pas qui ils étaient, et la manière dont nous devions travailler ensemble, d'autant plus que Florent était lui aussi arrivé récemment dans l'entreprise. Mon interlocuteur privilégié était Guillaume à qui je transmettais mes avancées dans le code, et qui indiquait à Maël ce qui devait être fait du côté de l'interface. C'est la raison pour laquelle j'ignorais si nous disposions tous des mêmes informations de base concernant ce projet de POC. Lors du déconfinement un embryon de prototype avait déjà été mis en place, et j'ai pu plus facilement communiquer avec les développeurs. Après le retour en présentiel en entreprise, il n'y a pas réellement eu de mise au point pour veiller à ce que tout le monde dispose de toutes les informations nécessaires, ce qui aurait aidé Florent et Maël à comprendre la trajectoire du projet.

Néanmoins, un bon esprit d'équipe a pu se mettre en place par la suite, que ce soit avec l'équipe de développement où le product owner avec qui j'ai pu créer de bonnes relations professionnelles dès les premiers échanges durant le confinement, que ce soit pour des questions relatives à la structure des données, à la manière dont elles sont utilisées, exposer les difficultés et problèmes identifiés dans les données etc.

La deuxième partie de ce rapport a permis d'expliquer la gestion de projet qui a été mise en œuvre dans la réalisation de ce POC. Dans la mesure où le client ne connaît pas de manière explicite ce qu'il attend du produit (pas de cahier des charges précis et définitif), la méthode SCRUM était la plus appropriée car elle permet une réelle participation du client dans le processus de conception du produit. Cette méthode se base sur une succession de cycles courts de développements débouchant chacun sur une réunion au cours de laquelle le client peut voir l'évolution du produit, et en fonction de ces progressions d'affiner progressivement son produit en redéfinissant les objectifs. La prise en compte de ces imprévus nécessite une bonne cohésion et une bonne communication au sein de l'équipe SCRUM, ce qui devrait être garanti à travers des rôles, divers évènements et outils. Néanmoins chaque projet est différent et selon le contexte, l'expérience et la disponibilité de chacun, la méthode est plus ou moins facilement mise en œuvre. Dans le cadre du projet Keolis, les principales difficultés ont été la communication et la gestion du temps/l'anticipation à certains moments du projet. Néanmoins au fur et à mesure un bon esprit d'équipe a pu se mettre en place. Cela a conduit à la réalisation d'un POC assez avancé comme nous allons le voir à la partie suivante.

III. Le fruit des cinq sprints : un démonstrateur opérationnel et suffisamment étoffé pour l'associer à un prototype

A. Comment fonctionne l'application ?

1. En tant qu'utilisateur

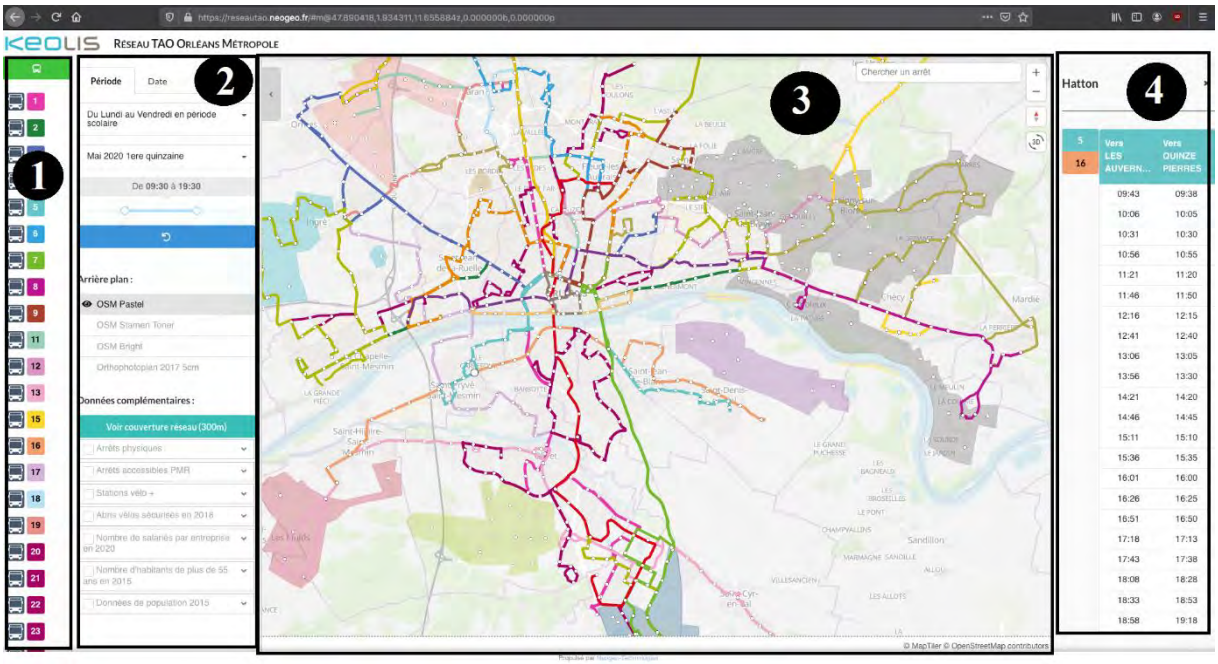


Figure 7: L'interface (front-end) de l'application
Source : https://reseautao.neogeo.fr/

L'interface utilisateur comporte 4 zones principales. Le panneau 2 permet à l'utilisateur de choisir le moment où il veut afficher l'offre de transport. Il présente 2 onglets différents : l'onglet période et l'onglet date.

L'onglet période permet d'afficher l'offre de transport en fonction des services standards. La première liste déroulante permet de sélectionner le service parmi les 9 choix possibles (du lundi au vendredi, samedi, ou dimanche en période scolaire, petites vacances ou en été). La deuxième liste déroulante permet de sélectionner la quinzaine pour laquelle s'applique le service standard. Le slider permet à l'utilisateur de sélectionner la plage horaire (heure de début et heure de fin) pour laquelle il souhaite voir l'offre de transport. Le bouton bleu juste en dessous permet de lancer la recherche d'offre de transport pour les options précédemment choisies.

L'onglet date permet d'afficher l'offre de transport à une date donnée. Un calendrier permet de choisir la date, et le slider la plage horaire pour laquelle l'utilisateur veut connaître l'offre de transport. Un bouton bleu identique à celui de l'onglet période permet de lancer la recherche. Les deux onglets présentent la rubrique "arrière plan" permettant de choisir un fond de carte parmi les 4 proposés et la rubrique "données complémentaires". Cette rubrique propose différentes couches que l'on peut activer ou non :

-une couche d'arrêts physiques (donc tous les arrêts) classés en 4 catégories : arrêts avec abribus et accessibles aux personnes à mobilité réduite (PMR), arrêts sans abribus et accessibles aux PMR, arrêts avec abribus et non accessibles aux PMR, arrêts sans abribus et non accessibles aux PMR
-une couche avec uniquement les arrêts accessibles aux PMR (avec et sans abribus)
-une couche avec les stations de vélos +
-une couche avec les stations de vélos sécurisées
-une couche de la répartition du nombre de salariés par entreprises en 2020
-une couche de la répartition du nombre d'habitants de plus de 55 ans par immeuble en 2015
-une couche carroyée de la population en 2015

Cette rubrique contient aussi un bouton permettant de voir la couverture du réseau (des buffers de 300 mètres autour des arrêts). Les figures page suivante permettent de visualiser ces différentes couches lorsqu'elles sont activées. Ces données complémentaires permettent d'analyser la pertinence de l'offre selon certains critères, et de proposer des améliorations. Par exemple, il peut être intéressant de croiser l'information relative à la répartition des personnes de plus de 55 ans et des arrêts accessibles aux personnes à mobilité réduite, ou bien de vérifier que les entreprises soient bien desservies (des lignes diversifiées, des arrêts proches et desservies très régulièrement à l'heure de la débauche entre 16h30 et 19h par exemple).

Le panneau 3 permet de visualiser l'offre de transport résultant de la recherche : les zones de TAD et ses arrêts, ainsi que les lignes de bus et de trams et les arrêts associés disponibles selon les critères choisis par l'utilisateur à l'onglet 2. Au clic sur un tracé de ligne, celui-ci devient plus épais et passe au premier plan pour mieux le visualiser, et le panneau 4 apparaît pour indiquer le numéro de ligne et son nom. S'il s'agit d'une zone de TAD, celle-ci devient plus foncée au clic et le panneau 4 affiche le nom du TAD. Lorsque l'utilisateur clique sur un arrêt desservi par des lignes de bus ou de tram, les horaires de passage dans la fourchette horaire choisie par l'utilisateur au panneau 2 sont affichées, et ceci pour chaque ligne desservant l'arrêt et pour chaque sens de circulation. La figure 7 page 23 montre par exemple les horaires de passage pour l'arrêt Hatton, entre 09h30 et 19h30.

Une barre de recherche en haut à droite dans la zone 3 permet de chercher un arrêt parmi tous ceux du réseau (et non pas les arrêts uniquement affichés sur la carte). Des arrêts sont suggérés au fur et à mesure que l'utilisateur tape le nom d'un arrêt. La recherche d'un arrêt permet de centrer la carte dessus au zoom 17.

Enfin, le panneau 1 permet de sélectionner ou désélectionner les lignes et les TAD. Le bouton vert permet de sélectionner/désélectionner toutes les lignes/TAD en même temps (et les arrêts aussi). Sinon il est possible d'activer/désactiver individuellement les lignes et les TAD (et les arrêts correspondant) en cliquant sur les différents icônes correspondants dans le panneau 1.

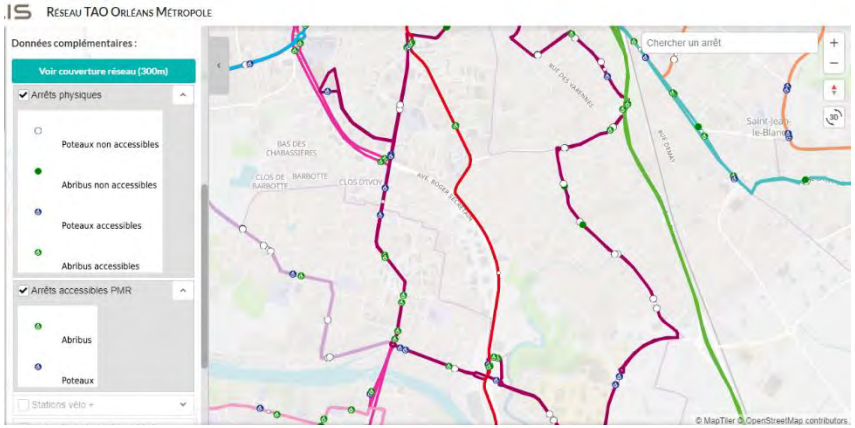


Figure 9: Affichage des deux couches « Arrêts physiques » et « Arrêts accessibles par les PMR »

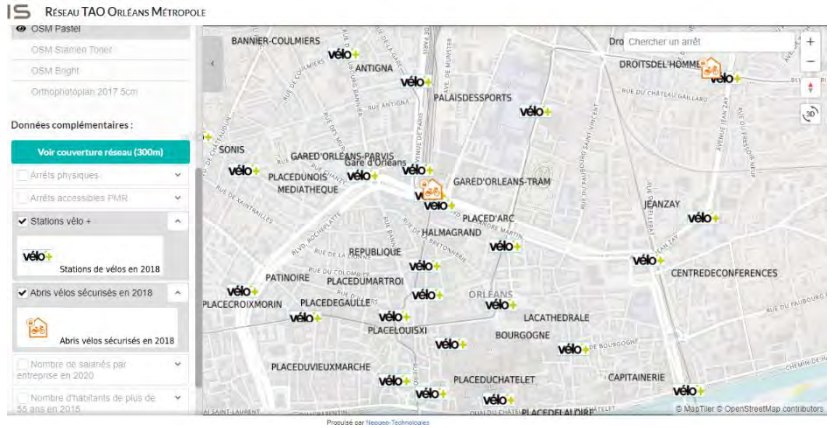


Figure 8: Affichage des couches liées au mobilier vélo

Figure 11: Affichage de la couche du nombre de salariés par entreprise, croisée avec la couverture des arrêts, un vendredi entre 16h30 et 19h30

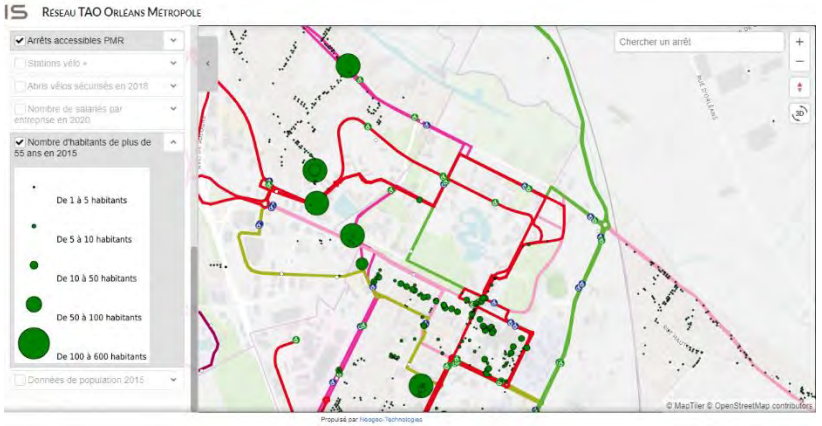


Figure 10: Affichage de la couche du nombre d'habitants de plus de 55 ans par immeuble, croisée avec la répartition des arrêts accessibles aux PMR

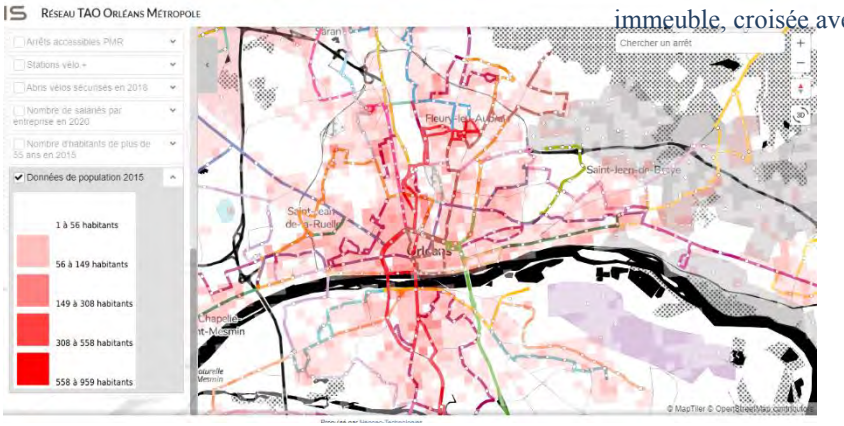


Figure 12: Densité de population par carroyage en 2015 (carreaux de 200m)

Une fonctionnalité permet également d'importer un nouveau flux GTFS automatiquement dans la base de données (figure 13). Pour que l'import fonctionne, l'utilisateur doit choisir un fichier zippé d'un flux GTFS contenant les fichiers texte des différentes tables exploitées dans la base de données (fichiers calendar_dates.txt, routes.txt, stop_times.txt, stops.txt, trips.txt).

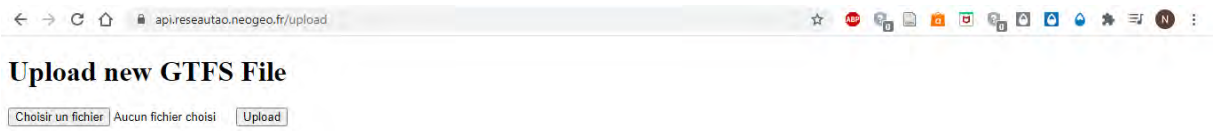


Figure 13: Capture de la page web permettant d'importer un nouveau flux GTFS dans la base de données
Source : https://api.reseautao.neogeo.fr/upload

2. En tant que développeur

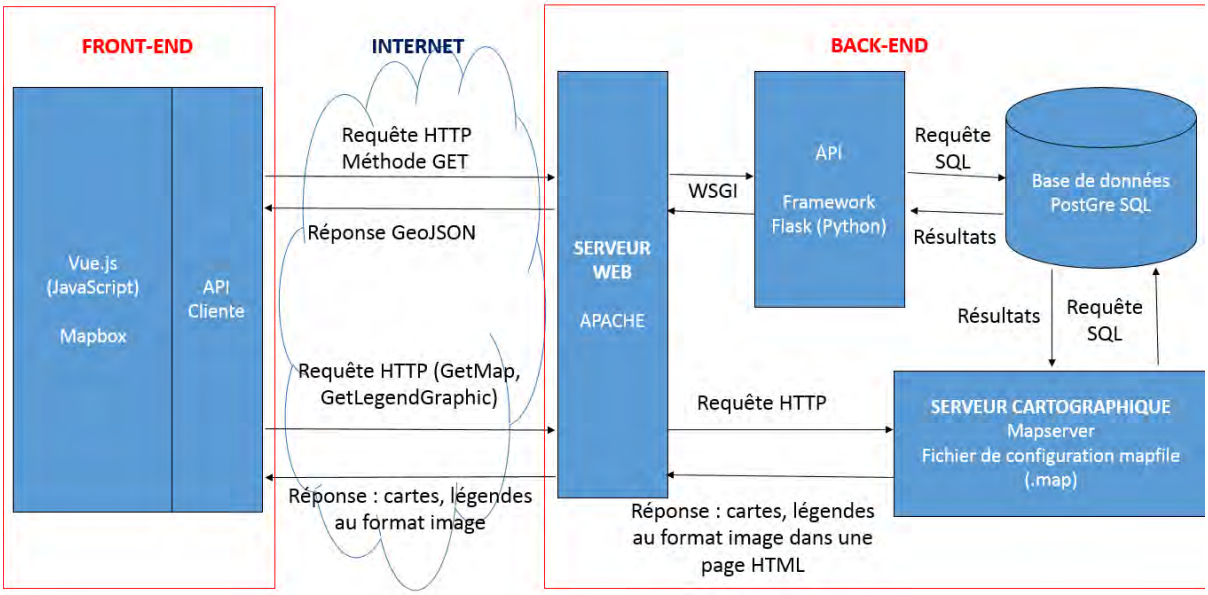


Figure 14: Architecture de l'application

L'application peut être décomposée en deux parties principales :
-le front-end correspond au code de l'interface, c'est-à-dire à ce qui est visible à l'écran de l'utilisateur. Le navigateur web interprète le code front-end pour afficher un « visuel », à savoir l'interface de la figure 7 page 23 avec laquelle interagit l'utilisateur. Cette interface est codée avec le framework[4] Vue.js, tandis que la carte dynamique est fournie par Mapbox. L'interface permet d'afficher à l'utilisateur les informations que ce dernier demande lorsqu'il sélectionne des choix dans une liste déroulante, clique sur un bouton, tape un nom de station dans la barre

[4] un framework est un cadre de travail proposant un ensemble de composants (bibliothèques, classes etc.) que le développeur va pouvoir réutiliser pour faciliter son travail de développement (d'application web dans le cadre du projet) (site internet Formation Django "Qu'est-ce qu'un framework")

de recherche etc. Ces requêtes sont envoyées au back-end qui est chargé de trouver les informations demandées et de les renvoyer au front-end pour que l'utilisateur puisse les visualiser. Le front-end constitue le client qui va envoyer des requêtes, tandis que le back-end constitue le serveur : il reçoit la requête du client, la traite et lui renvoie une réponse.
-le back-end est la partie invisible pour l'utilisateur. Elle va traiter les demandes que le front-end (son client, c'est à dire le navigateur web) lui envoie au travers de requêtes HTTP contenant des paramètres qui spécifient la demande. La figure 15 ci-dessous montre l'URL envoyée par le client (le navigateur web) lorsqu'il demande les zones de TAD disponibles le 09/03/2020 (paramètre date) entre 8h (paramètre from) et 12h (paramètre until).

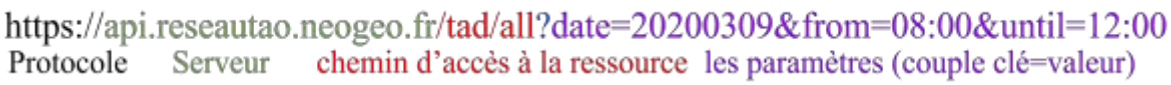


Figure 15: Exemple de requête

Le serveur web (apache) réceptionne la requête http. Il la transmet à l'API (Application Programming Interface)[5] via le protocole WSGI qui permet la communication entre un serveur web et une application Python. En effet cette API est codée en python et utilise le framework Flask, un framework de développement d'application web. L'API va interpréter la requête et envoyer la requête SQL correspondante à la base de données pour récupérer les données nécessaires. Le système de gestion de base de données utilisé est PostgreSQL, avec son extension spatiale PostGis.
L'API reçoit les données de la base de données et les met en forme (format Json ou GeoJSON) avant de les transmettre au serveur web Apache via le protocole WSGI. Celui-ci renvoie cette réponse au client (le front-end) pour affichage sur le navigateur web.

La figure 17 page suivante montre la réponse GeoJson de la requête de la figure 15 renvoyée au front-end. Celui-ci va traiter cette réponse GeoJson pour qu'elle puisse être affichée correctement sur l'interface, à l'image de la figure 16 (nous remarquons que les arrêts sont affichés également, mais ils correspondent au résultat d'une autre requête qui est exécutée simultanément avec celle des zones de TAD, c'est pourquoi ils sont aussi affichés).

Les cartes des données complémentaires sont quant à elles servies par mapserver, un logiciel permettant de servir des cartes à des clients par l'intermédiaire du web. En fonction des paramètres de la requête transmise à mapserver et de son fichier de configuration (le mapfile), mapserver va interroger la base de données pour récupérer les données géographiques avant de les mettre en forme dans une carte qui sera renvoyée au client. Dans le cadre de l'application, mapserver diffuse uniquement des web map services, et les requêtes utilisées sont le getMap pour retourner une image de carte, et le GetLegendGraphic pour la légende des cartes.

[5] une interface applicative de programmation permet "d'établir des connexions entre plusieurs logiciels pour échanger des données" (site internet Mobizel (2015) "[Définition] C'est quoi… une API" publié par Emilie le 27/07/2015, consulté le 08/08/2020 et disponible à https://www.mobizel.com/definition-cest-quoi-une-api/)

Figure 17: Réponse GeoJson pour les zones de TAD le 09/03 entre 8h et 12h

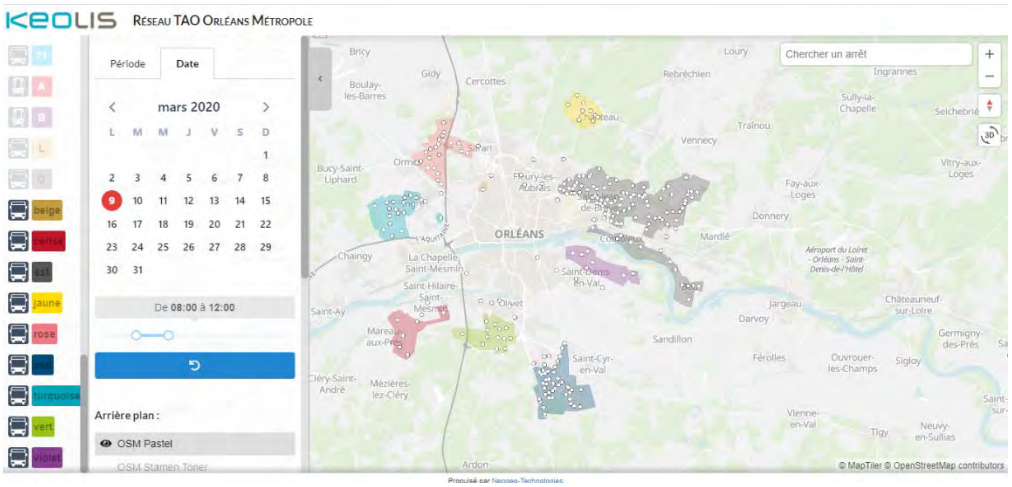


Figure 16: Affichage des TAD le 09/03 entre 8h et 12h

B. Les données utilisées

1. Pour les lignes de bus et de tram : un format à maîtriser

Keolis utilise le format standard GTFS statique pour gérer les horaires théoriques de trams/bus. Le General Transit Feed Specification (GTFS) n'est pas une norme officielle mais un format mis en place par Google pour échanger des horaires de transports en commun et les informations géographiques associées. C'est le format standard utilisé aux Etats-Unis et pour la publication de la plupart des données de transports en commun en open data.

Les informations se présentent sous la forme d'un dossier compressé contenant des fichiers (certains obligatoires, d'autres optionnels) texte séparés par des virgules. Chaque fichier décrit un aspect particulier du réseau, et contient des champs obligatoires et optionnels. Sauf évènement particulier, un nouveau flux GTFS (dossier zippé) est produit environ tous les mois (mais pendant le coronavirus cela pouvait être plus régulier du fait d'adaptation de l'offre) pour la métropole d'Orléans. Ce ne sont pas des données en temps réelles, mais des données prévisionnelles de l'offre. Le GTFS sert de référence puisque ces données (les horaires) sont transmises directement aux conducteurs de bus. Ainsi, l'exploitation de ce type de données dans l'application permet d'avoir une information assez réaliste de ce qu'il est sensé se passer sur le réseau à tel jour, telle heure. Comme de nouvelles données sont produites régulièrement, le

développement d'une fonctionnalité permettant d'intégrer facilement de nouveaux flux GTFS était nécessaire.
La figure 18 page suivante montre les relations entre les différentes tables tel que cela est expliqué dans la documentation de google. Seules les tables utilisées par keolis sont représentées ici. Pour plus d'informations, consulter la documentation Google accessible au lien suivant : https://developers.google.com/transit/gtfs/reference?hl=fr

Dans le flux GTFS initialement fourni par Keolis il n'y avait pas la table calendar_dates : il n'y avait que la table calendar, et chaque enregistrement correspondait à une seule date (le start_date et le end_date étaient les mêmes). Par la suite, la table calendar a été abandonnée au profit de la table calendar_dates. La figure 19 page 31 montre les tables actuellement exploitées dans la base de données pour ce qui concerne l'affichage de l'offre de bus et de tram.

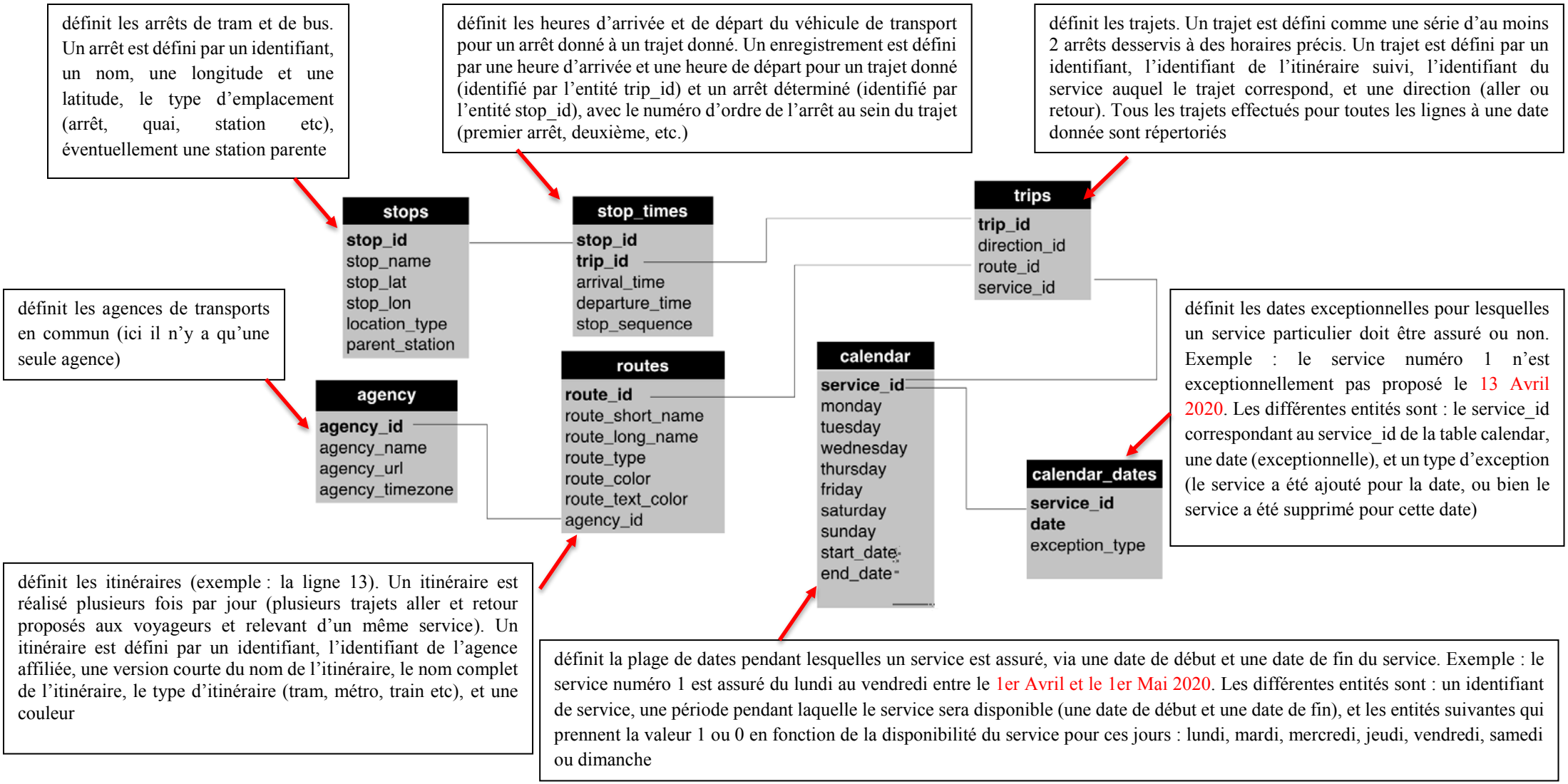


Figure 18: Modèle conceptuel du GTFS (uniquement les tables des fichiers de keolis)

Note : les champs en gras représentent les clés primaires

Source : https://developers.google.com/transit/gtfs/reference?hl=fr

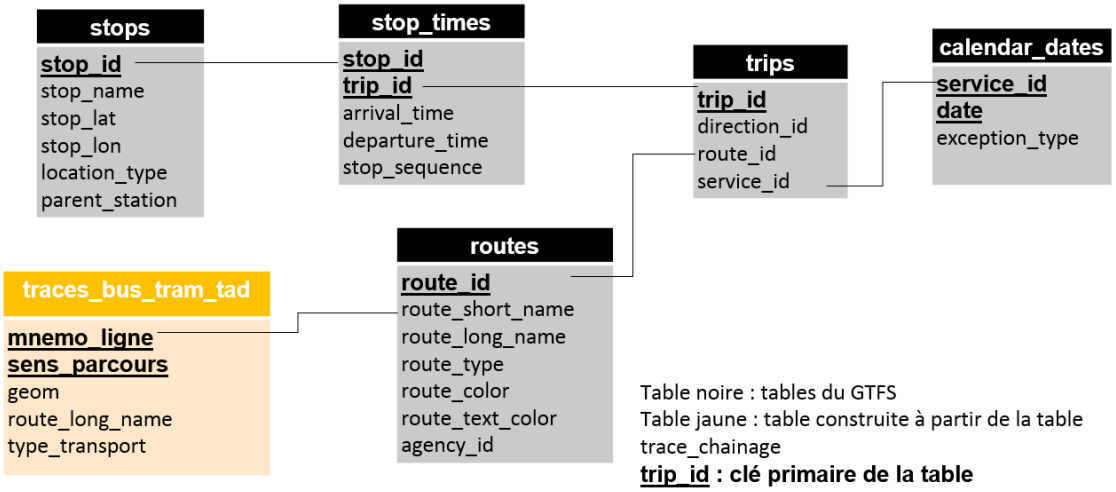


Figure 19: Modèle de données actuellement exploité dans l'application pour l'affichage des bus et des trams

Les tracés des lignes de bus et de trams ne font pas partis du GTFS. Deux sources de données étaient disponibles :
-le service marketing : une table par ligne de transport dans un sens donné contenant tous les tronçons de voirie concernés par la ligne
-le service d'aide à l'exploitation : une seule table contenant tous les points de forme (points géo localisant les bus) pour les différentes lignes, et une table contenant le tracé des chainages (les lignes reliant les points de forme pour redessiner l'itinéraire réalisé par le bus).

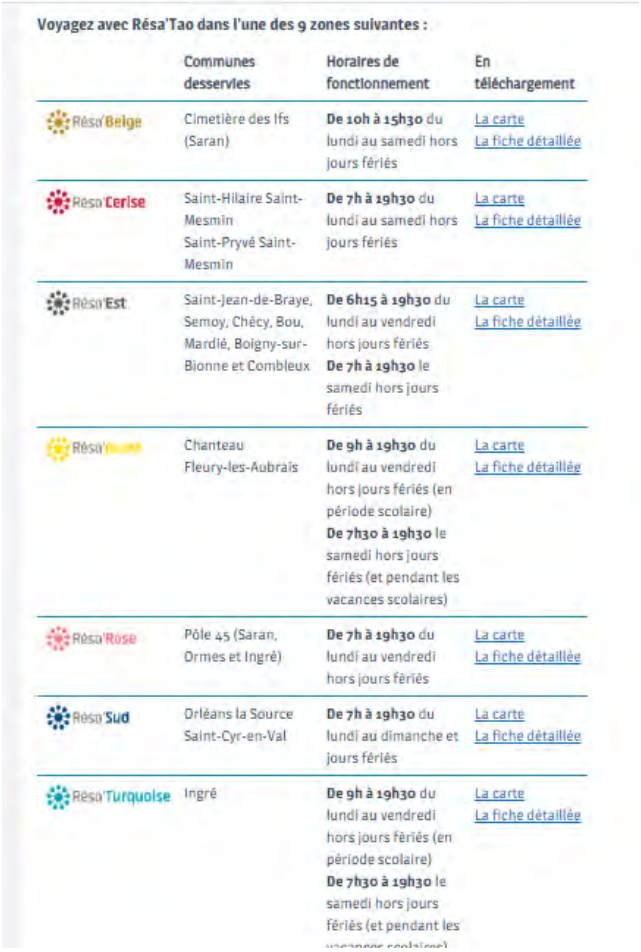
Les données du service marketing étant déjà réunies dans une seule table, et dans la mesure où il y avait un champ permettant de faire une jointure avec la table routes du GTFS, ce sont ces données qui ont été sélectionnées. Un enregistrement correspond au tracé d'une ligne dans un sens donné. Le champ type_transport a été rajouté afin de distinguer les tracés correspondant à des bus et les tracés correspondant à un tram, cette information étant nécessaire au front-end pour attribuer un logo à chaque ligne (les logos du panneau de gauche permettant l'affichage ou non des lignes).

En termes de bases de données, la difficulté consistait ici à faire le lien entre les tracés de lignes et les tables du GTFS. Le travail le plus complexe consiste à comprendre comment ce format est utilisé par Keolis (notamment les cardinalités qui ne sont déjà pas très bien explicitées dans la documentation Google), et comment l'exploiter au mieux dans des requêtes SQL.

2. Les données pour les TAD : un modèle conceptuel de données à imaginer

Concernant les tracés des 9 zones de TAD, le service marketing a pu fournir des fichiers .shape qui ont été intégrés dans la base de données, avec les tracés réunis dans une seule table. Les arrêts desservis par les TAD étaient disponibles dans une table issue du service d'aide à

l'exploitation contenant tous les arrêts du réseau. Les données horaires étaient disponibles sur le site internet des transports en commun à Orléans, comme le montre la figure 20.



Voyagez avec Résa'Tao dans l'une des 9 zones suivantes :

	Communes desservies	Horaires de fonctionnement	En téléchargement
Résa'Beige	Cimetière des Ifs (Saran)	**De 10h à 15h30** du lundi au samedi hors jours fériés	La carte La fiche détaillée
Résa'Cerise	Saint-Hilaire Saint-Mesmin Saint-Pryvé Saint-Mesmin	**De 7h à 19h30** du lundi au samedi hors jours fériés	La carte La fiche détaillée
Résa'Est	Saint-Jean-de-Braye, Semoy, Chécy, Bou, Mardié, Boigny-sur-Bionne et Combleux	**De 6h15 à 19h30** du lundi au vendredi hors jours fériés **De 7h à 19h30** le samedi hors jours fériés	La carte La fiche détaillée
Résa'Jaune	Chanteau Fleury-les-Aubrais	**De 9h à 19h30** du lundi au vendredi hors jours fériés (en période scolaire) **De 7h30 à 19h30** le samedi hors jours fériés (et pendant les vacances scolaires)	La carte La fiche détaillée
Résa'Rose	Pôle 45 (Saran, Ormes et Ingré)	**De 7h à 19h30** du lundi au vendredi hors jours fériés	La carte La fiche détaillée
Résa'Sud	Orléans la Source Saint-Cyr-en-Val	**De 7h à 19h30** du lundi au dimanche et jours fériés	La carte La fiche détaillée
Résa'Turquoise	Ingré	**De 9h à 19h30** du lundi au vendredi hors jours fériés (en période scolaire) **De 7h30 à 19h30** le samedi hors jours fériés (et pendant les	La carte La fiche détaillée

Figure 20: L'information brute relative aux horaires de disponibilité des TAD
Source : https://www.reseau-tao.fr/205-resa-tao2C-une-mobilite-souple-et-dynamique.html

Dans le cas des TAD, il n'y a pas d'organisation entre les données : nous disposons de tables apportant des informations d'intérêt (des tracés de zones, des arrêts) mais il n'y a pas de champ commun permettant une jointure directe, et nous disposons d'informations brutes (des horaires) qu'il faut intégrer dans la base de données. Il est donc nécessaire de réfléchir à un modèle conceptuel de données (MCD). Celui-ci a été réfléchi pour répondre à la question : « quelles sont les TAD disponibles à telle date, entre telle et telle heure, et les arrêts associés ? ». La figure 21 ci-dessous montre le MCD imaginé, et la figure 22 de la page suivante montre le modèle logique correspondant (MLD).

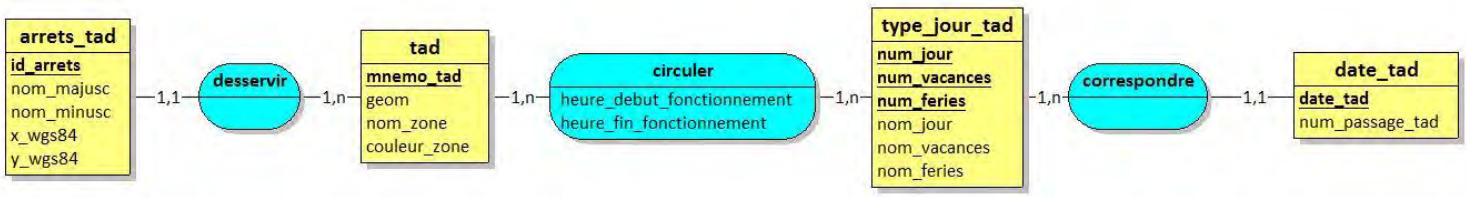


Figure 21: MCD pour les TAD

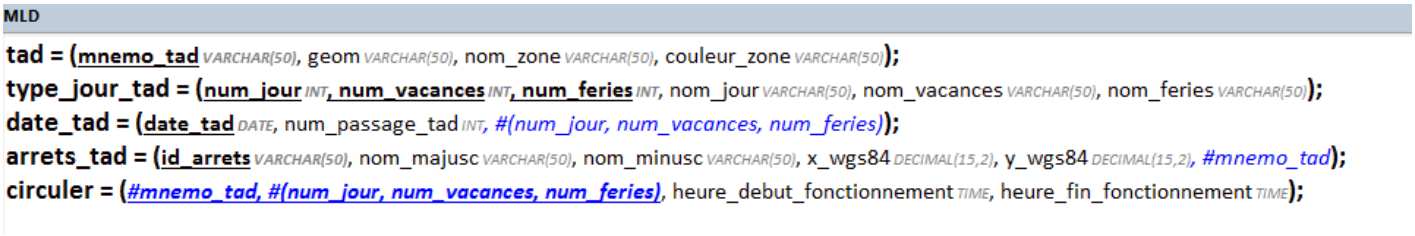


Figure 22: MLD pour les TAD

Au vu des horaires de fonctionnement des TAD, une table type_jour a été créée pour répertorier les types de jours possibles dans une année en fonction de 3 paramètres :
-le numéro de jour de la semaine (lundi, mardi etc.)
-les vacances (période scolaire, petites vacances scolaires, grandes vacances)
-si c'est un jour férié ou non (sachant qu'un jour férié est toujours considéré comme un Dimanche)
La table ci-dessous montre les différents enregistrements possibles.

num_jour	num_vacances	num_feries	nom_jour	nom_vacances	nom_feries
1	1	0	Lundi	Période scolaire	Non férié
2	1	0	Mardi	Période scolaire	Non férié
3	1	0	Mercredi	Période scolaire	Non férié
4	1	0	Jeudi	Période scolaire	Non férié
5	1	0	Vendredi	Période scolaire	Non férié
6	1	0	Samedi	Période scolaire	Non férié
7	1	0	Dimanche	Période scolaire	Non férié
1	2	0	Lundi	Petites vacances scolaires	Non férié
2	2	0	Mardi	Petites vacances scolaires	Non férié
3	2	0	Mercredi	Petites vacances scolaires	Non férié
4	2	0	Jeudi	Petites vacances scolaires	Non férié
5	2	0	Vendredi	Petites vacances scolaires	Non férié
6	2	0	Samedi	Petites vacances scolaires	Non férié
7	2	0	Dimanche	Petites vacances scolaires	Non férié
1	3	0	Lundi	Grandes vacances	Non férié
2	3	0	Mardi	Grandes vacances	Non férié
3	3	0	Mercredi	Grandes vacances	Non férié
4	3	0	Jeudi	Grandes vacances	Non férié
5	3	0	Vendredi	Grandes vacances	Non férié
6	3	0	Samedi	Grandes vacances	Non férié
7	3	0	Dimanche	Grandes vacances	Non férié
7	1	1	Dimanche	Période scolaire	Férié
7	2	1	Dimanche	Petites vacances scolaires	Férié
7	3	1	Dimanche	Grandes vacances	Férié

Table 3: Table type_jour_tad

Chaque date de la table date_tad correspond à un et un seul type de jour (les jours fériés sont considérés comme des dimanche), et un type de jour donné correspond à plusieurs dates. Pour un type de jour donné, il circule de 1 à n TAD. Un TAD peut circuler à différents types de jour. Pour un type de jour donné, un TAD est disponible pendant une plage horaire définie par une heure de début et une heure de fin. En prenant l'exemple du TAD beige, la table « circuler » (nommée plage_horaire_tad dans la base de données) contient les enregistrements que l'on peut voir dans la table de la page suivante).

mnemo_tad	num_jour	num_vacances	num_feries	heure_debut_fonctionnement	heure_fin_fonctionnement
beige	1	1	0	10:00:00	15:30:00
beige	2	1	0	10:00:00	15:30:00
beige	3	1	0	10:00:00	15:30:00
beige	4	1	0	10:00:00	15:30:00
beige	5	1	0	10:00:00	15:30:00
beige	6	1	0	10:00:00	15:30:00
beige	1	2	0	10:00:00	15:30:00
beige	2	2	0	10:00:00	15:30:00
beige	3	2	0	10:00:00	15:30:00
beige	4	2	0	10:00:00	15:30:00
beige	5	2	0	10:00:00	15:30:00
beige	6	2	0	10:00:00	15:30:00
beige	1	3	0	10:00:00	15:30:00
beige	2	3	0	10:00:00	15:30:00
beige	3	3	0	10:00:00	15:30:00
beige	4	3	0	10:00:00	15:30:00
beige	5	3	0	10:00:00	15:30:00
beige	6	3	0	10:00:00	15:30:00

Table 4: Extrait de la table plage_horaire_tad

Dans la mesure où les zones de desserte des TAD sont distinctes et ne se chevauchent pas, il est logique de penser qu'un arrêt ne puisse être desservi que par un seul TAD, et qu'un TAD puisse desservir plusieurs arrêts. Mais plus tard, lorsqu'il a été possible de visualiser les données sur l'interface, il s'est avéré que les TAD peuvent aussi desservir des arrêts qui sont en dehors de leur zone de desserte (à l'affichage, on peut voit une zone de TAD, et des arrêts à l'extérieur de cette zone). Ce sont des arrêts de « rabattement », une particularité des données métiers de Keolis. Cela signifie donc qu'un arrêt TAD peut potentiellement être desservi par plusieurs TAD, donc il faudrait corriger le MCD dans le cas où le POC se poursuit sur un vrai produit. Pour l'heure, si un arrêt est desservi par plusieurs TAD il faut répéter l'enregistrement en changeant la valeur de TAD : par conséquent, la clé primaire de la table arrets_tad est composée de 2 champs : id_arrets et mnemo_tad. Plus tard, le product owner a indiqué qu'à cause du coronavirus, les TAD n'ont pas circulé du 16 mars au 10 mai. Cette éventualité n'avait pas été imaginé dans le MCD, c'est pourquoi le champ num_passage_tad a été rajouté dans la table du calendrier : si la valeur vaut 0 pour une date, aucun TAD ne circule. Quand la valeur vaut 1, les TAD circulent.

La partie III a permis d'expliquer le fonctionnement du POC. Du point de vue de l'utilisateur, le POC offre des fonctionnalités permettant de connaître l'offre de transport dans le temps, mais propose également d'afficher des informations complémentaires permettant d'analyser la pertinence de l'offre. D'un point de vue développeur, l'application est composée de deux parties majeures : la partie front-end qui gère l'interface visuelle et développée avec le framework vue.js, puis la partie back-end invisible pour l'utilisateur. Elle comprend le serveur web qui reçoit les requêtes du front-end, de la base de données stockant les données métiers, et une API assurant l'interface entre le serveur web et la base de données : elle interprète la requête http que lui transmet le serveur web, et lui renvoie une réponse adaptée en allant chercher les données dans la base PostGre. Le serveur web envoie cette réponse au navigateur,

c'est-à-dire le client. Le back-end comprend aussi mapserver qui permet de servir des cartes en fonction de la requête http qu'il reçoit. Les paragraphes liés aux données ont permis de mettre en évidence la distinction entre les lignes de bus/tram et les TAD. Les données du premier se présentent sous un format standardisé et très répandu : le GTFS. Il n'y avait donc pas de traitements particulier à réaliser sur les tables, la difficulté se présente plutôt dans la compréhension des tables et des relations entres elles, et sur la manière dont le client Keolis utilise ce format. Pour le deuxième il n'y avait pas de modèle, mais des tables indépendantes, et des informations brutes qu'il fallait intégrer dans la base de données. Il a donc fallu réfléchir à un modèle conceptuel de données, et créer certaines tables intégralement. Mais dans la mesure où il était difficile d'anticiper les demandes futures du client et que les données étaient mal connues lors de la conception du MCD, celui-ci n'est pas optimal. La partie suivante vise à expliquer le raisonnement et les étapes mises en œuvre pour coder l'API, c'est-à-dire pour passer d'une demande de fonctionnalité à une page web avec les données demandées au format Json ou GeoJson (donc arriver à la figure 17 de la page 28). Nous prendrons comme exemple la fonctionnalité suivante : « afficher pour une date donnée et une plage horaire donnée les TAD disponibles ». Le travail réalisé concernant la configuration du mapfile pour créer les services de cartes pour les données complémentaires ne sera pas détaillé.

C. L'exploitation des données : développement de l'API en prenant un exemple de fonctionnalité demandée

« afficher pour une date donnée et une plage horaire donnée les TAD disponibles »

1. Traduire la demande en requête SQL renvoyant un objet GeoJson avec les données correspondantes

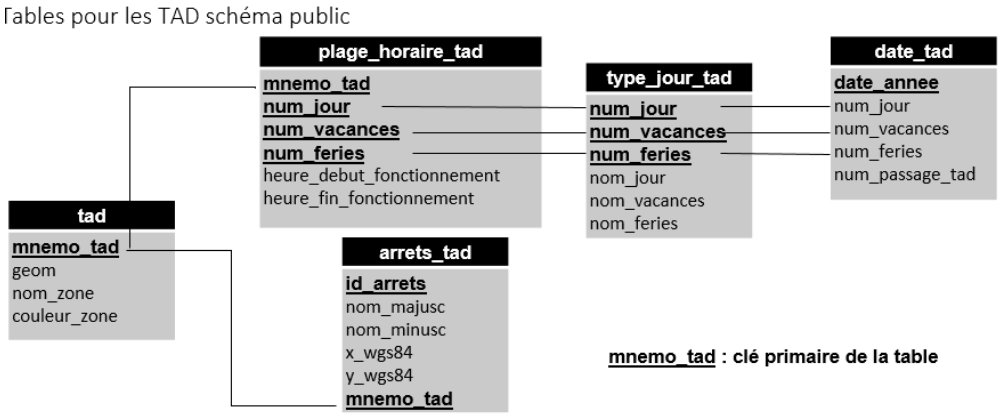


Figure 23: Modèle de données utilisé dans l'application pour la gestion des TAD

Pour répondre à la question « afficher pour une date donnée et une plage horaire donnée les TAD disponibles », il faut d'abord réfléchir aux informations à renvoyer au front-end. Il faut naturellement renvoyer la géométrie des zones et la couleur pour afficher les polygones sur la carte, mais également le nom complet du TAD (exemple : Résa'Sud) pour que lorsque l'utilisateur clique sur une zone il puisse accéder au nom, et enfin le code associé (mnemo_tad

exemple : sud) qui est utilisé dans le panneau 1 de l'application. Donc il faut renvoyer tous les champs de la table tad.

Pour mieux comprendre la requête SQL à créer, on peut reformuler la question en prenant un exemple de date et d'heures : « afficher les champs mnemo_tad, geom, nom_zone et couleur_zone des TAD disponibles le 9 mars entre 8h et 12h ». Nous comprenons alors que le filtre sur les TAD dépend de 3 champs qui interviendront dans la clause WHERE : date_annee doit être égale à 20200309, et les valeurs de heure_debut_fonctionnement et heure_fin_fonctionnement. Concernant les bornes des heures, plusieurs cas sont à considérer pour déterminer si un TAD doit être sélectionné ou non comme le montre la figure 24 ci-dessous. Le choix des cas à prendre en compte a été réalisé avec le product owner :

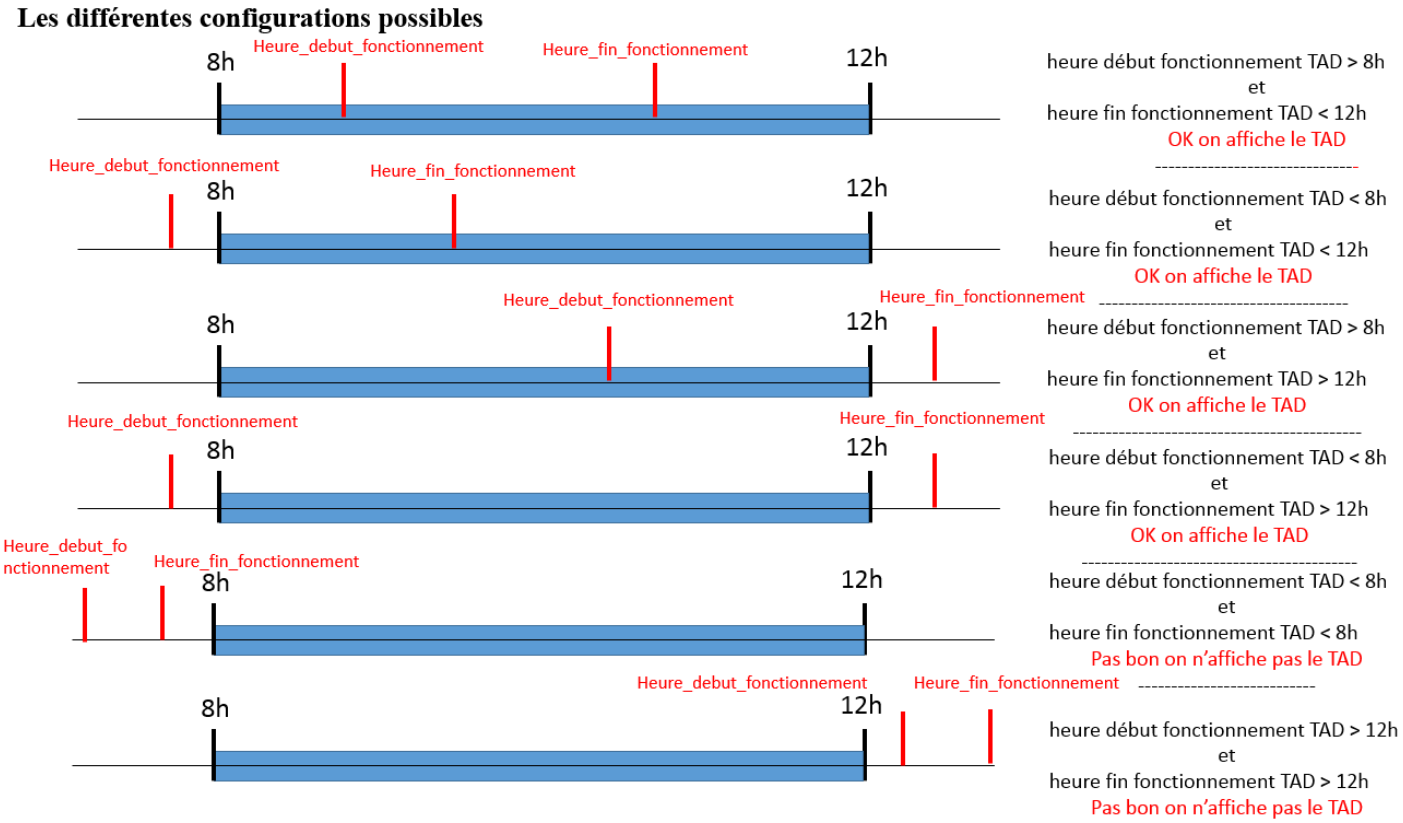


Figure 24: Un schéma pour mieux comprendre comment structurer une requête SQL

Nous pouvons finalement reformuler la question comme ceci : « afficher les champs mnemo_tad, geom, nom_zone et couleur_zone des TAD pour lequel la valeur date_annee vaut 20200309, et pour lequel la valeur heure_fin_fonctionnement n'est pas inférieure ou égale à 8h ou la valeur heure_debut_fonctionnement n'est pas supérieure ou égale à 12h». Après avoir clarifié les données à envoyer au front-end, et décortiqué les champs intervenant dans la clause WHERE il est possible de réfléchir à une requête SQL et de la tester sur Pgadmin. Une fois que cette requête fonctionne, il faut la compléter pour obtenir en résultat un objet json (ou geojson), à l'image de la figure 25 ci-dessous.

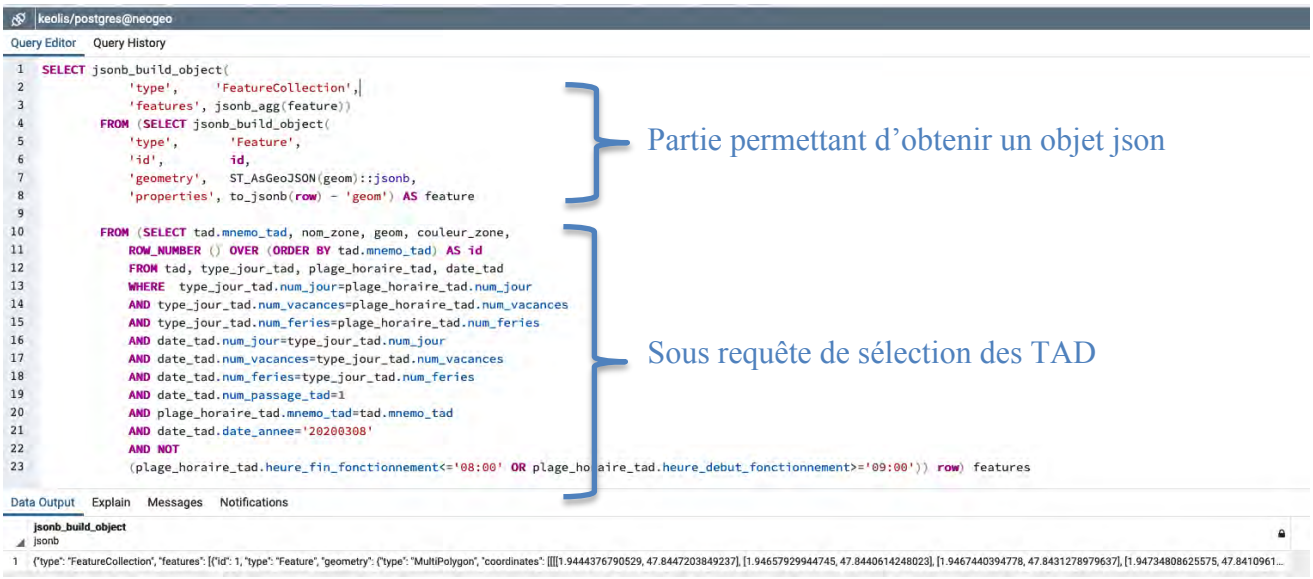


Figure 25: Requête SQL pour la sélection des TAD

2. Construire la vue dans le code Python de l'API et test en local après avoir installé un environnement virtuel

Après avoir trouvé la requête SQL que l'API devra envoyer à PostGre pour récupérer les données, il faut l'intégrer dans le code python.
Comme énoncé précédemment, le rôle de l'API est de recevoir une requête (une url) d'un client (le navigateur web). A la réception de cet url l'API doit réaliser des tâches spécifiques pour retourner une réponse au client. On appelle « route » l'url à laquelle une fonction particulière va être réalisée et elle est créée grâce au décorateur @app.route(). On appelle « vue » la fonction qui sera utilisée pour une route définie. Elle est définie à la ligne suivant le décorateur @app.route(). La figure 26 suivante présente un code minimaliste d'application permettant de définir la vue accueil() qui sera utilisée pour le chemin d'accès '/exemple'.

```python
from flask import Flask
app = Flask(__name__) #création d'une instance flask (de l'application)

@app.route("/exemple") #définition de l'URL pour laquelle la vue accueil() doit être utilisée
def accueil(): # définition de la vue accueil() : elle retourne au navigateur la chaine
                #de caractères "hello world"
    return "Hello world"

if __name__ == "__main__": #permet de lancer l'application à l'execution du programme
     app.run()

#réponse de la console Python
#runfile('C:/Users/nel/Desktop/exemple.py', wdir='C:/Users/nel/Desktop')
# * Serving Flask app "exemple" (lazy loading)
# * Environment: production
#   WARNING: This is a development server. Do not use it in a production deployment.
#   Use a production WSGI server instead.
# * Debug mode: off
# * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
#127.0.0.1 - - [02/Sep/2020 16:27:21] "GET /exemple HTTP/1.1" 200 -

```

Figure 26: Code d'une application minimale

La réponse de la console python indique que l'application est active au lien : http://127.0.0.1:5000/ , ce qui signifie que l'application tourne en local (le code de l'application est localisé sur la machine locale). En tapant dans le navigateur web l'url http://127.0.0.1:5000/exemple nous obtenons la réponse 'hello world' de la vue accueil() affichée à l'écran comme le montre la figure 27 ci-dessous.

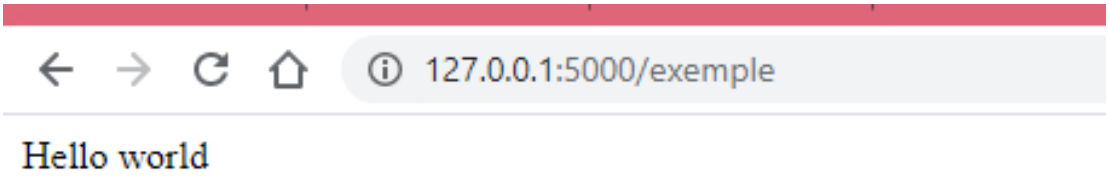


Figure 27: Réponse à la requête http://127.0.0.1:5000/exemple

Pour le POC, l'API est constitué de plusieurs fichiers :
-le fichier app.py contenant les différentes routes et vues associées
-le fichier database.py avec la classe Database contenant des méthodes permettant de se connecter à la base de données PostgreSQL et d'interroger les tables avec des requêtes SQL
-le fichier import_gtfs.py contenant les méthodes permettant d'importer un nouveau flux GTFS dans la base de données et de mettre à jours les tables
-le fichier keolis.py contenant des méthodes pour des fonctionnalités diverses et ne nécessitant pas d'interroger la base de données (exemple : la liste des quinzaines, la liste des services standards etc.)
-le fichier config.py de configuration de l'application contenant les noms, mot de passe etc. de la base de données
La vue pour l'affichage des TAD est un peu plus complexe que celle vue précédemment. En effet elle fait intervenir des paramètres dans l'url, comme le montre la figure 28 ci-dessous. En effet l'utilisateur doit pouvoir choisir une date et une plage horaire (dans notre exemple la date du 20200309 et une plage horaire allant de 8h à midi). La méthode request.args.get de Flask

permet justement de récupérer ces valeurs de paramètres qui seront injectées dans la fonction

https://127.0.0.1:5000/tad/all?date=20200309&from=08:00&until=12:00

protocole **serveur : localhost** **route** **paramètres**

```python
@app.route('/tad/all')
def get_tad():
    tad_date = request.args.get('date', '20200224')
    #request.args.get('date', '20190101') permet de récupérer la valeur du paramètre date
    #dans l'url, et s'il n'y a pas de valeur, alors prendre par défaut la valeur 20200224
    tad_from = request.args.get('from', '09:00')
    #récupérer la valeur du paramètre from
    tad_until = request.args.get('until', '10:00')
    #récupérer la valeur du paramètre until
    res = db.select_tad(tad_date, tad_from, tad_until)
    #appeler la méthode select_tad du fichier database.py avec les paramètres de l'url
    response = app.response_class(
        response=json.dumps(res), #conversion de l'objet python res en un objet json
        status=200,
        mimetype='application/json' )# indique le type de données à renvoyer au navigateur
    return response #réponse retournée par l'application flask vers le client (navigateur)
```

```python
def select_tad(self, date, h_from, h_until):
    sql = """
    SELECT jsonb_build_object(
        'type',     'FeatureCollection',
        'features', jsonb_agg(feature)
    )
    FROM (
      SELECT jsonb_build_object(
        'type',       'Feature',
        'id',         id,
        'geometry',   ST_AsGeoJSON(geom)::jsonb,
        'properties', to_jsonb(row) - 'geom'
      ) AS feature
    FROM (SELECT tad.mnemo_tad, nom_zone, geom, couleur_zone,
        ROW_NUMBER () OVER (ORDER BY tad.mnemo_tad) AS id
        FROM tad, type_jour_tad, plage_horaire_tad, date_tad
        WHERE  type_jour_tad.num_jour=plage_horaire_tad.num_jour
        AND type_jour_tad.num_vacances=plage_horaire_tad.num_vacances
        AND type_jour_tad.num_feries=plage_horaire_tad.num_feries
        AND date_tad.num_jour=type_jour_tad.num_jour
        AND date_tad.num_vacances=type_jour_tad.num_vacances
        AND date_tad.num_feries=type_jour_tad.num_feries
        AND date_tad.num_passage_tad=1
        AND plage_horaire_tad.mnemo_tad=tad.mnemo_tad

        AND date_tad.date_annee=%s
        AND NOT
        (plage_horaire_tad.heure_fin_fonctionnement<=%s OR plage_horaire_tad.heure_debut_fonctionnement>=%s)) row) features;
    """

    with self.conn.cursor() as cur:
        cur.execute(sql, (date, h_from, h_until)) #éxécution de la requête en remplaçant le premier %s par la variable date,
                                                  #le second %s par la variable h_from,
                                                  #le troisième %s par la variable h_until
        records = [row for row in cur.fetchall()] #stockage des enregistrements dans une variable, on obtient une liste de tuples
    return records[0][0]
```

Dans la requête SQL, les valeurs de date_annee, heure_debut_fonctionnement et heure_fin_fonctionnement sont remplacées par %S (quand dans la requête testée sur pgadmin les valeurs valaient 20200309, 08 :00 et 12 :00). Ces trois %S prennent les valeurs des paramètres date, h_from et h_until de la requête http du client

Figure 28: Vue get_tad() du fichier app.py et la fonction select_tad(self, date, h_from, h_until) du fichier database.py

Pour vérifier que les développements sont justes, l'url est testée. La réponse affichée sur le navigateur se présente sous le format GeoJson (comme le montre la figure 17 page 28). Après que le code ait été testé en local et qu'il fonctionne, et que ces nouvelles lignes ne perturbent pas le reste du code, il est possible de déposer ces développements sur le GitLab.

3. Publier le code sur le GitLab et le déposer sur le serveur hébergeant le POC

GitLab est une plateforme permettant d'héberger et de gérer des projets. Il se base sur l'outil Git, un logiciel de gestion de versions.

En effet, lorsqu'un développeur produit du code il doit l'enregistrer sur un outil sécurisé où il pourra être partagé avec les autres collaborateurs du projet qui pourraient en avoir besoin pour leurs développements ou pour directement travailler dessus. Ces types d'outils sont des logiciels de gestion de versions qui vont pouvoir stocker le code, le partager avec les autres développeurs et également conserver les différentes versions. Cela signifie que lorsqu'un code est ajouté à une version antérieure, les modifications entre les deux versions sont enregistrées dans un historique qui permet de retracer toutes les évolutions du code, et éventuellement de revenir à une version antérieure lorsque des disfonctionnements apparaissent. L'espace dédié à un projet et contenant tous les fichiers de code est appelé dépôt.

Il est possible au sein d'un projet de travailler simultanément sur plusieurs versions différentes pour diverses raisons (exemple : la validation d'une nouvelle version peut prendre du temps, donc les développeurs vont continuer leurs travaux sur d'autres versions). Chacune de ces versions s'appelle une branche, et celle qui est en production, la version courante, s'appelle généralement la branche master. Un développeur ne travaille pas directement sur la branche en production, il créée une nouvelle branche à partir du master pour rajouter le code de sa/ses nouvelle(s) fonctionnalité(s), ses corrections etc. Une fois les développements terminés sur la branche, celle-ci peut être fusionnée à la branche master, et le nouveau code de la branche master est testé pour vérifier qu'il fonctionne correctement et qu'il n'engendre pas de régression dans l'application développée. La figure 29 ci-dessous illustre ce processus à travers un exemple, avec la création de plusieurs branches qui sont modifiées à plusieurs reprises (les modifications sont sauvegardées dans un historique afin de pouvoir éventuellement y revenir si besoin), une fusionnée au master et l'autre non. Le processus d'assemblage de deux versions de code et de test du code résultant s'appelle l'intégration. Lorsque les tests sont réalisés automatiquement par un autre logiciel (et non pas à la main par le développeur), on parle d'intégration continue.

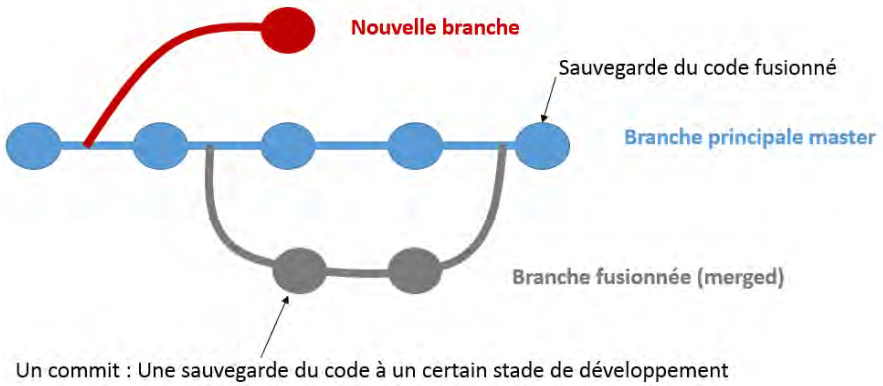


Figure 29: Le système de branches dans Git

Ainsi, lorsque le code testé en local fonctionne, je crée une nouvelle branche dans le projet Keolis (dossier back-end) à partir de la branche master et j'y actualise le code avec les nouveaux développements. Puis je fusionne cette branche à la branche principale grâce à une merge request. La figure 30 ci-dessous montre l'interface du GitLab et le code à actualiser.

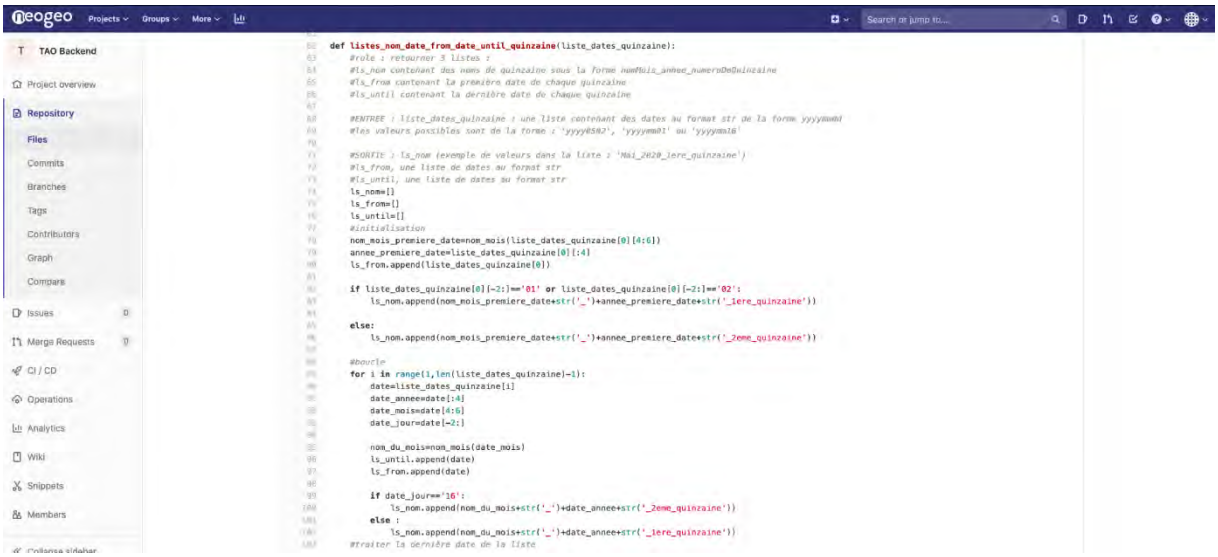


Figure 30: Visualisation du code dans GitLab

Enfin, la sauvegarde la plus récente de la branche principale est déposée sur le serveur de l'application. Il faut utiliser le terminal afin de se connecter au serveur (commande ssh) et utiliser la commande $git\ pull$ dans le répertoire dédié au back-end. Cette commande permet de télécharger sur le serveur le contenu d'un dépôt (en l'occurrence celui du back-end) présent sur la plateforme GitLab, et de mettre à jour le code du serveur correspondant à ce dépôt[6]. Pour vérifier que la commande ait bien fonctionné est que le back-end de l'application soit fonctionnel pour la nouvelle fonctionnalité développée (donc pouvoir afficher les TAD disponibles à une date et dans une plage horaire donnée pour notre exemple) il suffit d'accéder à l'url suivant et de changer les valeurs des paramètres date (format AAAAMMJJ), from et until : https://api.reseautao.neogeo.fr/tad/all?date=20200309&from=08:00&until=12:00. Le navigateur web affiche bien la réponse attendue au format GeoJson.

Le tableau 5 suivant récapitule toutes les vues ayant été développées selon la méthode décrite précédemment, et qui résume toutes les fonctionnalités apportées par le back-end.

[6] Pour plus d'informations sur le commande git pull, consulter https://www.atlassian.com/fr/git/tutorials/syncing/git-pull

Rôle de la vue	url d'accès	Paramètres de l'url
Affichage des lignes de bus et de tram circulant à une date et dans une plage horaire données	https://api.reseautao.neogeo.fr/routes/all?date=20200309&from=08:00&until=12:00	date : AAAAMMJJ from : HH :MM (heure début) until : HH :MM (heure fin)
Affichage des TAD disponibles à une date et dans une plage horaire données	https://api.reseautao.neogeo.fr/tad/all?date=20200309&from=08:00&until=12:00	date : AAAAMMJJ from : HH :MM (heure début) until : HH :MM (heure fin)
Affichage des arrêts desservis par des lignes de bus et de tram circulant à une date et dans une plage horaire données	https://api.reseautao.neogeo.fr/arrets/all?date=20200309&from=08:00&until=12:00	date : AAAAMMJJ from : HH :MM (heure début) until : HH :MM (heure fin)
Affichage des arrêts desservis par les TAD disponibles à une date et dans une plage horaire données	https://api.reseautao.neogeo.fr/arrets_tad/all?date=20200309&from=08:00&until=12:00	date : AAAAMMJJ from : HH :MM (heure début) until : HH :MM (heure fin)
Importer un nouveau flux GTFS dans la base de données	https://api.reseautao.neogeo.fr/upload	
Affichage des TAD disponibles pour un service standard donné, pour une période et une plage horaire choisies	https://api.reseautao.neogeo.fr/service_standard_tad/all?service=1&from=09:00&until=12:00&date_from=20200315&date_until=20200401	Service : entre 1 et 9 inclus en fonction du service standard choisi (1 = du lundi au vendredi en période scolaire, 2= samedi en période scolaire, 3=dimanche en période scolaire, 4, 5 et 6 la même chose mais en petites vacances scolaires, 7, 8 et 9 la même chose mais en grandes vacances from : HH :MM (heure début) until : HH :MM (heure fin) date_from : AAAAMMJJ (date début) date_until : AAAAMMJJ (date fin)
Affichage des arrêts desservis par les TAD disponibles pour un service standard donné, pour une période et une plage horaire choisies	https://api.reseautao.neogeo.fr/service_standard_tad_arrets/all?service=1&from=09:00&until=12:00&date_from=20200315&date_until=20200401	Service : entre 1 et 9 inclus en fonction du service standard from : HH :MM (heure début) until : HH :MM (heure fin) date_from : AAAAMMJJ (date début) date_until : AAAAMMJJ (date fin)
Affichage des lignes de bus et de tram circulant pour un service standard donné, pour une période et une plage horaire choisies	https://api.reseautao.neogeo.fr/service_standard_routes/all?service=1&from=09:00&until=12:00&date_from=20200315&date_until=20200401	Service : entre 1 et 9 inclus en fonction du service standard from : HH :MM (heure début) until : HH :MM (heure fin) date_from : AAAAMMJJ (date début) date_until : AAAAMMJJ (date fin)
Affichage des arrêts desservis par les lignes de bus et de tram circulant pour un service standard donné, pour une période et une plage horaire choisies	https://api.reseautao.neogeo.fr/service_standard_routes_arrets/all?service=1&from=09:00&until=12:00&date_from=20200315&date_until=20200401	Service : entre 1 et 9 inclus en fonction du service standard from : HH :MM (heure début) until : HH :MM (heure fin) date_from : AAAAMMJJ (date début) date_until : AAAAMMJJ (date fin)

Rôle de la vue	url d'accès	Paramètres de l'url
Pour la liste déroulante des quinzaines : vue renvoyant au format json un nom de quinzaine, une date de début et une date de fin de quinzaine	https://api.reseautao.neogeo.fr/periodes_services_standards	
Liste des heures pour le slider : le nom de l'heure à afficher sur l'interface et la référence de l'heure correspondante dans la base de données (ex : 2h j+1 correspond à 26 :00 :00 dans la table)	https://api.reseautao.neogeo.fr/heures	
Pour la liste déroulante des services standards : vue renvoyant un nom de service standard à afficher et un numéro correspondant (ex : du lundi au vendredi en période scolaire correspond à 1)	https://api.reseautao.neogeo.fr/services_standards	
Afficher les horaires de passage pour un arrêt donné en ayant le nom des lignes qui le desservent avec les sens de circulation, pour une date et une plage horaire donnée	https://api.reseautao.neogeo.fr/heures_passages_arrets/all?date=20200309&from=08:00&until=12:00&stop_name=8%20mai	date : AAAAMMJJ from : HH :MM (heure début) until : HH :MM (heure fin) stop_name : nom d'un arrêt
Afficher les horaires de passage pour un arrêt donné en ayant le nom des lignes qui le desservent avec les sens de circulation, pour un service standard donné, pour une période et une plage horaire choisies	https://api.reseautao.neogeo.fr/service_standard_heures_passages_arrets/all?service=1&from=09:00&until=12:00&date_from=20200301&date_until=20200315&stop_name=8%20mai	Service : entre 1 et 9 inclus en fonction du service standard from : HH :MM (heure début) until : HH :MM (heure fin) date_from : AAAAMMJJ (date début) date_until : AAAAMMJJ (date fin) stop_name : nom d'un arrêt
Pour la barre de recherche d'arrêts : le nom des arrêts de TAD et des lignes de bus et de tram avec leurs coordonnées (longitude, latitude)	https://api.reseautao.neogeo.fr/recherche_arrets	

Table 5: Les vues du back-end

D. Les difficultés rencontrées dans le développement back-end de l'application

Outre la difficulté technique liée au développement Python ou à la construction de requêtes SQL, le principal obstacle a été de comprendre la manière dont était utilisé le format GTFS par Keolis. A la visualisation des premiers résultats sur l'interface, on observait parfois des incohérences entre l'affichage des lignes et des arrêts. Après confirmation auprès du product owner, il s'est avéré que lors de la production d'un nouveau flux GTFS, l'incrémentation de la clé primaire pour la table des trajets était réinitialisée : cela a pour conséquence d'avoir des doublons de clés primaires entre deux flux GTFS alors que les enregistrements sont différents et font référence à des trajets distincts. La jointure entre la table trips et stop_times ne se fait donc pas correctement ce qui explique pourquoi certains arrêts étaient associés à des lignes de bus qui ne les desservaient pas comme le montre la figure 31 ci-dessous.

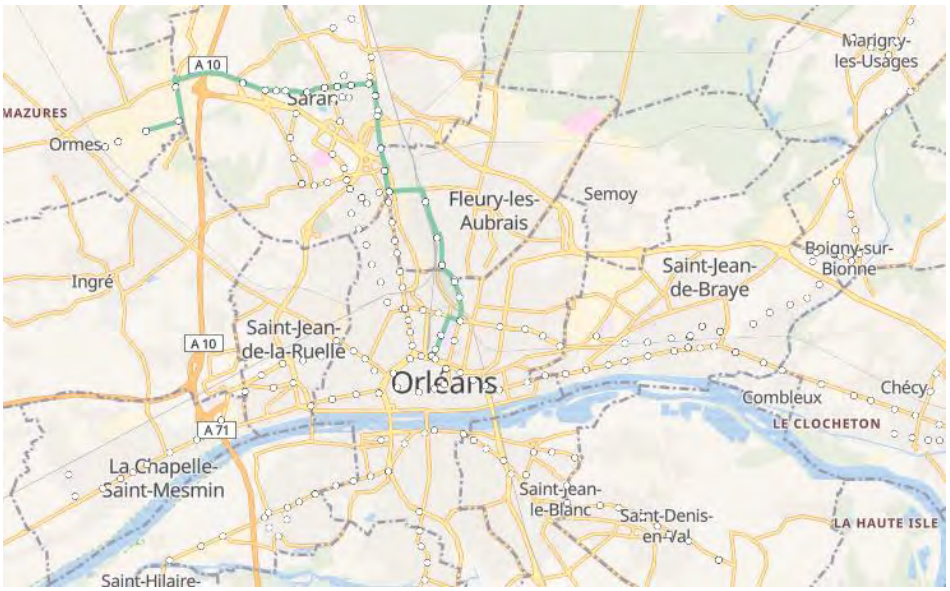


Figure 31: Incohérence entre les lignes de bus/tram et les arrêts

Ce problème a été résolu en modifiant les valeurs de trip_id des enregistrements en concaténant la valeur initiale au nom du fichier et à la date d'import du flux dans la base de données. De cette manière tous les trajets sont uniques et il n'y a plus de confusion dans la jointure entre la table trips et la table stop_times.

Une autre difficulté a été de concilier données et demande du client. En effet, les données horaires pour le GTFS sont définies par date. Elles sont actualisées en fonction des divers évènements « prévisibles » (exemple : manifestation de gilets jaunes, crise du coronavirus, festivals etc.). Nous comprenons donc que l'affichage de l'offre de transport en fonction des services standards (du lundi au vendredi, samedi, dimanche en période scolaire, petites vacances scolaires ou été) n'est pas cohérent vis-à-vis des données. L'offre de transport d'un lundi scolaire en période de coronavirus ne sera pas la même qu'un lundi scolaire habituel, d'autant plus qu'entre deux années scolaires il peut y avoir une restructuration des horaires. Par ailleurs entre le lundi et le vendredi les horaires ne sont pas nécessairement les mêmes. Lorsque l'utilisateur veut afficher l'offre de transport un dimanche en petites vacances scolaires, que faut-il prendre comme date dans ce cas ? La date la plus récente de la table correspondant à un dimanche ? Mais cette date est-elle représentative de l'offre disponible un dimanche en période

scolaire ? Faut-il prendre les 5 dernières dates ? Dans ce cas cela fait remonter à des vacances scolaires de l'année scolaire précédente. En prenant toutes les dates correspondant à un dimanche scolaire, la requête SQL devient plus longue à s'exécuter, et quand il s'agit du service 'du lundi au vendredi en période scolaire' le temps de réalisation dépasse les 5 secondes malgré la mise en place d'index. Après avoir abordé ce sujet avec le client lors d'une revue de sprint, celui-ci a indiqué que la quinzaine était une bonne échelle de temps, d'où la mise en place de la liste déroulante du choix d'une quinzaine dans l'onglet des services standards.

IV. Un POC qui répond à la majorité des demandes formulées dans le backlog product

Le tableau 6 suivant résume le backlog produit mis à jour en juin et l'état d'avancement de ces demandes à la fin du stage :

demande	détail	Le POC le permet-il ?	commentaire
Observer l'offre de transport en fonction de la période, du jour et de l'heure		oui	
Je peux sélectionner précisément la période à afficher	-choisir un service standard -choisir un jour de semaine voir plusieurs (ex : lundi, mardi et samedi)	-oui -non	Concernant le jour de la semaine, la demande n'était pas prioritaire dans les sprints restant
Observer le réseau futur	Définir un cahier des charges du format de données	oui	Le format est le GTFS
Sélectionner une ou plusieurs lignes pour les visualiser plus facilement	-sélectionner une ou plusieurs lignes et afficher les horaires correspondants -bouton pour activer/désactiver toutes les lignes d'un coup -pourvoir filtrer par bus, tram et TAD -pouvoir conserver le filtre	Oui sauf les filtres	Le filtre est un détail qui n'apporte pas une réelle plus-value au produit en termes de fonctionnalités
Visualiser les zones de TAD et l'offre associée		oui	
Visualiser les informations d'un arrêt	-les lignes -l'accessibilité de l'arrêt -présence ou non d'abribus	oui	Accessibilité et abribus visibles avec les données complémentaires
Visualiser les informations concernant une ligne	-horaires de fonctionnement en fonction des critères choisis -terminus alternatifs en fonction du jour, de la période, de l'horaire NON PRIORITAIRE	non	
Accéder à un maximum d'informations		oui	Toutes les données fournies ont été traitées, le reste des données n'a pas été fourni par le client
Visualiser la couverture de l'offre	Cercle de 300m autour des arrêts	oui	
Pouvoir importer l'offre théorique facilement (GTFS)		oui	
Mettre à jour les données géographiques facilement	Produire une documentation permettant de mettre à jour les cartes, les arrêts, ajouter ou supprimer un tracé de ligne	oui	Documentation présentant la base de données, le client sait manipuler des tables
Interface intuitive et ergonomique		oui	
Comprendre l'association entre les étiquettes et les objets		oui	
Utiliser l'interface correctement sur ordinateur ou tablette		En cours	La visualisation n'est pas optimale sur portable, mais dans la mesure où le projet est un POC cela ne semble pas prioritaire

demande	détail	Le POC le permet-il ?	commentaire
Les données se présentent sous la forme d'un web service, pouvoir accéder aux données grâce à un logiciel SIG		oui	Sur QGIS il est possible de visualiser les cartes générées par mapserver, et de se connecter à la base de données pour faire des requêtes et afficher les résultats

Table 6: Tableau récapitulant les fonctionnalités à mettre en œuvre dans le POC

Les fonctionnalités essentielles apportant une réelle plus-value au POC ont toutes été développées. Les éventuelles demandes restantes n'étaient pas essentielles pour le client. Lors de la dernière revue de sprint, les trois représentants de Keolis (la chargée d'études, le directeur innovation, projets et système d'information, ainsi que le product owner, à savoir l'alternante) ont exprimé une réelle satisfaction vis-à-vis du POC qui apporte une plus-value supérieure par rapport à ce qu'ils espéraient au lancement du projet. C'est un outil qui valorise des données métiers, à savoir les horaires théoriques mises en application sur le réseau, et qui permet de connaître l'offre de transport théorique à toute date et toute heure. Cet outil permet donc de mieux connaître l'offre de transport. Il permet aussi d'analyser sa pertinence grâce à des données complémentaires notamment l'accessibilité aux personnes à mobilité réduite, la répartition des personnes âgées et des entreprises, et donc à justifier les décisions prises par Keolis en matière d'offre de transport en commun et à proposer des améliorations (exemple : vérifier qu'à proximité des bâtiments à forte concentration de personnes âgées comme les maisons de retraites il y ait des arrêts accessibles aux personnes à mobilité réduite, ou augmenter la fréquence de passage d'une certaine ligne à l'heure de la débauche car elle passe à proximité d'un pôle où se concentrent plusieurs entreprises etc.). Mais c'est également un outil de communication grâce à son interface très intuitive. Il permettrait à Keolis de montrer aux élus d'Orléans la pertinence de l'offre proposée par Keolis et ainsi assurer le renouvellement du contrat de délégation de service public en 2024. Cet objectif est d'autant plus important qu'en Juillet 2020 le maire sortant Olivier Carré n'a pas été réélu et a laissé sa place à Serge Grouard. Dans la mesure où le réseau va être restructuré en 2022, cet outil de visualisation de l'offre de transport pourrait aider les élus à prendre des décisions quant au futur réseau.

Le POC va être présenté au service marketing chargé de la communication et de l'analyse de la pertinence de l'offre, mais également au service d'aide à l'exploitation chargé d'enregistrer les données d'exploitation et de les analyser pour mieux décrire l'offre. En fonction du retour de ces services vis-à-vis de l'utilité du POC pour leurs activités, il sera décidé de la continuité du projet vers un outil complet et industrialisable. Mais outre une exploitation en interne, le directeur du pôle système d'information songe à utiliser le POC pour faire de la communication externe auprès des services de la métropole d'Orléans. Il envisage aussi de présenter le POC lors d'évènements autour de la géomatique (exemple : les Géodatadays) pour promouvoir l'idée de construire un outil polyvalent permettant :

-d'exploiter des données métiers

-d'analyser l'offre par rapport à divers critères (répartition des personnes âgées, des salariés, des stations de vélos, accessibilité des arrêts, couverture de l'offre)

-et de faire de la communication.

V. Conclusion

Afin de mieux connaître son offre de transport et de la perfectionner pour répondre aux besoins des voyageurs, la société Keolis qui exploite pour le compte d'Orléans métropole le réseau de transport urbain orléanais a fait appel au prestataire Neogeo Technologies, une entreprise toulousaine développant des solutions sur mesure dans le domaine de l'information géographique. Le but est de disposer d'un outil innovant de cartographie dynamique de l'offre de transport. Cette idée initiée par le directeur innovation, projets et système d'information de l'entreprise consistait en une application permettant de visualiser à tout moment l'offre de transport théorique, et d'afficher des données complémentaires permettant d'étudier la pertinence de l'offre par le service marketing essentiellement. Ignorant si cet outil serait réellement utile pour améliorer l'offre de Keolis et le cahier des charges n'étant pas bien défini, Keolis a commandé à Neogeo Technologies un démonstrateur, sorte de prototype minimaliste permettant de savoir si un tel produit est réalisable, et d'évaluer sa désirabilité auprès du service marketing. Le projet débutant officiellement le 1er Avril 2020 avec la réunion de lancement, devait s'étendre sur 5 mois et déboucher sur un démonstrateur fonctionnel à livrer pour Septembre 2020, dans la limite d'un budget de 30 000 €. Dans ce contexte, l'objet du stage a consisté à gérer ce projet en tant que « cheffe de projet junior », et à participer au développement du démonstrateur en prenant en charge la partie back-end du POC.

Cela s'est concrétisé par la mise en œuvre d'une méthode de gestion de projet appelée SCRUM, basée sur des rôles et des cycles de développements courts permettant au client de voir progressivement les avancements dans le développement du POC, et d'affiner au fur et à mesure sa demande en fournissant une liste de tâches à faire à chaque cycle de développement qui s'étend sur un mois environ. J'ai été la principale interlocutrice du client et été chargée de faire le lien entre les développeurs et le client, donc de rapporter les questions des développeurs au client, de transmettre aux développeurs les demandes du client, de veiller à ce que les fonctionnalités soient développées dans les temps et soient opérationnelles pour les réunions de présentation du POC. Pendant ces réunions je présentais les avancées, explicitais les éventuelles difficultés et réfléchissais avec le client à des alternatives potentielles. Je sollicitais le client pour qu'il me transmette les données, et lui communiquais les éventuelles anomalies détectées ou interrogations vis-à-vis des données.

En termes de développement back-end, le travail a consisté à intégrer les données dans une base de données PostgreSQL, à faire des choix concernant les données à utiliser, à réfléchir sur un modèle conceptuel de données, et à s'approprier un format de données standardisé pour communiquer des données de transport en commun : le GTFS. Une fois la base de données structurée, il a fallu développer l'API servant d'intermédiaire entre la base de données stockant les données et le front-end, c'est-à-dire l'interface qu'affiche le navigateur web et sur laquelle un utilisateur clique via des boutons, et où il tape des mots, sélectionne des choix pour afficher l'offre de transport selon divers critères. Cette API a été développée en python avec le framework Flask. Les pratiques de développements utilisées consistant à tester le code en local à chaque modification/ajout de lignes, puis à le stocker sur la plateforme GitLab après avoir créé une branche, puis à le transférer sur le serveur de l'application et à tester l'application

s’appelle l’intégration. Ces méthodes visent à identifier au plus tôt les disfonctionnements dans le code et à les corriger le plus rapidement afin d’éviter toute régression dans l’application développée. La partie back-end comporte aussi mapserver, un logiciel permettant de servir des cartes et nécessitant la configuration d’un fichier particulier : le mapfile.

A l’issue des cinq mois de développement, le client Keolis a exprimé sa satisfaction quant aux fonctionnalités fournies par le démonstrateur. Cet outil permet à la fois de connaître l’offre théorique à tout moment sur la base de données d’exploitation, et d’évaluer la pertinence des choix de Keolis quant aux itinéraires des lignes, la position des arrêts ou encore le choix des horaires par rapport à des critères d’intérêt pour la métropole (exemple : accessibilité aux personnes à mobilité réduite, répartition des entreprises, couverture des arrêts etc.). Le client songe également à utiliser ce démonstrateur comme outil de communication auprès des élus de la métropole pour légitimer les choix de Keolis et éventuellement de les aider dans la prise de décisions quant au réseau futur de 2022. Cet outil est donc polyvalent, néanmoins il ne propose qu’un nombre limité de fonctionnalités qui ne permettent pas de décrire en profondeur les caractéristiques de l’offre (exemple : graphe présentant les pics de fréquentation, le nombre de trajets dans une journée etc.) ou de l’analyser de manière approfondie (les données complémentaires sont peu nombreuses), c’est pourquoi le démonstrateur va être présenté au service marketing et au service d’aide à l’exploitation qui pilote en temps réel l’offre de transport. En fonction de leur avis concernant le démonstrateur, il sera décidé de la continuité ou non du projet pour aboutir à un produit complètement opérationnel.

VI. Bilan personnel

A travers ce stage, je souhaitais sortir de ma zone de confort (l'agronomie) pour découvrir des domaines que je n'avais pas encore eu l'occasion d'explorer. J'ai ainsi pu approfondir mes connaissances sur le fonctionnement des infrastructures de données géographiques, et sur l'organisation de l'information géographique en France. Le projet Keolis m'a quant à lui permis de découvrir un vocabulaire spécifique au domaine des transports en commun. J'ai aussi pu me familiariser avec un format de données couramment utilisé dans le développement d'applications, et dans le partage de données liées à la thématique des transports en communs : le GTFS. A travers ce projet j'ai également consolidé mes connaissances sur le fonctionnement théorique et technique de l'architecture client/serveur.

En termes de savoir-être, la gestion de projet m'a aidé à développer mes compétences relationnelles à travers les échanges réguliers avec le client, mais aussi avec les développeurs. Elle m'a aussi permis d'améliorer mon organisation en m'obligeant à prioriser les tâches et à respecter des délais. Cela a également été un bon exercice pour la gestion de stress. J'ai également découvert l'outil de gestion de projet Redmine et compris globalement la manière dont était géré un projet d'informatique. J'ai également amélioré ma capacité d'adaptation à diverses situations, à de nouveaux outils et développé mes aptitudes de prise d'initiatives.

En termes de savoir-faire, j'ai pu apprendre à utiliser un interpréteur de lignes de commandes, développer mes compétences en gestion et exploitation de bases de données mais également dans la programmation d'application web Python, à utiliser mapserver mais aussi GitLab en mettant en application des pratiques d'intégration semi-continue (car les tests ne sont pas automatisés par des logiciels).

Les compliments que m'a adressés le client lors de la dernière réunion en ce qui concerne la gestion de projet mise en œuvre avec lui, et sa satisfaction évidente quant à ce que l'équipe de développement a pu produire, m'ont fait prendre conscience de mes capacités et de l'exigence excessive que j'avais envers moi-même.

Cette expérience riche d'enseignements ne pourra m'être que bénéfique pour mon parcours professionnel. Elle m'a permis de me rendre compte que le développement d'outils pour autrui ne me suffisait pas, et que j'étais davantage intéressée par l'exploitation même des outils que des entreprises comme Neogeo Technologies développaient : pourvoir analyser les données, mettre en relation certaines informations pour expliquer certains phénomènes, et permettre ainsi de faire des recommandations et d'aider à la prise de décisions sur des thèmes variés.

Bibliographie

-Ange Miezan (2019) « Un POC ? Mais pour quoi faire ? » disponible ici : https://islean-consulting.fr/fr/organisation-dsi/un-poc-mais-pour-quoi-faire/

-AURÉLIE LE GUILLOU (2018) « Bien rédiger ses User Stories : Les Bonnes Pratiques » disponible ici : https://www.turn-on.fr/blog/bien-rediger-ses-user-stories

-Bruno Borghi (2017) « Au-delà de l'équipe Scrum : les rôles dans l'entreprise agile (Partie 1) » dans Les Cahiers Agiles disponible ici : https://medium.com/les-cahiers-agiles/au-del%C3%A0-de-l%C3%A9quipe-scrum-les-r%C3%B4les-dans-l-entreprise-agile-961d737a9e28

-Corentin Risselin (2017) « Git : comprendre la gestion de versions » disponible ici : https://blog.axopen.com/2017/02/git-comprendre-la-gestion-de-versions/

-DG Transformation digitale SPF Stratégie et appui « 0.3 GÉNÉRALITÉS : Quelle est la différence entre un service web et une application web ? » disponible ici : https://dtservices.bosa.be/fr/faq/03-generalites-quelle-est-la-difference-entre-un-service-web-et-une-application-web

-Documentation Google sur le format GTFS : https://developers.google.com/transit/gtfs/reference?hl=fr

-Emilie (2015) « [Définition] C'est quoi… une API » consulté le 08/08/2020 et disponible ici : https://www.mobizel.com/definition-cest-quoi-une-api/

-Florent TETART (2010) « Qu'est-ce qu'un Webservice ? » disponible ici : https://doc.sigb.net/doc_webservices_pmb/co/01_quesqu_un_webservice.html

-Frédéric HARDY (2016) « POC, Prototype et Pilote, quelles différences ? » disponible ici : https://fhconsult.wordpress.com/2016/11/07/poc/

-Georges (2020) « La méthode 3P pour valider votre solution IoT : Problème, PoC, Prototype » disponible ici : https://www.matooma.com/fr/s-informer/actualites-iot-m2m/comment-realiser-le-prototype-de-mon-objet

-Graven – Développement (2020) « LES BASES DE GIT (tuto débutant) » vidéo youtube disponible ici : https://www.youtube.com/watch?v=gp_k0UVOYMw

-Grégory (2017) « Méthodes de gestion de projet: comment faire son choix ? » disponible ici : https://bubbleplan.net/blog/methodes-gestion-projet-comment-choisir/

-Grégory (2018) « Agile/Scrum : Ne pas tout mélanger en gestion de projet ! » disponible ici : https://bubbleplan.net/blog/agile-scrum-gestion-projet/

-Jean-Pierre Lambert (2017) « GIT, GitFlow et l'intégration continue pour les nuls » disponible ici : https://jp-lambert.me/git-gitflow-et-lint%C3%A9gration-continue-pour-les-nuls-a0b2f0b7c788

-John Wiley & Sons, Inc. (2015) *Le déploiement d'applications pour les Nuls* (partie 2-1. Suivre le cycle de vie du développement logiciel) disponible ici : https://ibmcloud.developpez.com/tutoriels/developpement-applications/deploiements-applications-pour-les-nuls/#L1

-Julien Domecq (2019) « SCRUM, KANBAN: quelle est la méthode la plus adaptée pour le pilotage de vos projets digitaux ? » disponible ici : https://kanbios.fr/scrum-kanban-quelle-est-la-methode-la-plus-adaptee-pour-le-pilotage-de-vos-projets-digitaux/

-Ken Schwaber et Jeff Sutherland (2017) *Le Guide Scrum™ Le Guide de Référence de Scrum: Les Règles de Jeu* disponible ici : https://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-French.pdf

-Laurent GRANGER (mis à jour le 03/06/2020) « Proof of concept : l'essentiel à connaître pour l'utiliser à bon escient » disponible ici : https://www.manager-go.com/gestion-de-projet/poc.htm

-Le médiateur des entreprises (2019) *De l'idée à l'industrialisation : réussissez votre preuve de concept - 30 questions à se poser pour innover ensemble disponible ici* disponible ici : https://www.economie.gouv.fr/files/files/directions_services/mediateur-des-entreprises/PDF/4_INNOVER_ENSEMBLE/guide-poc.pdf

-Mickaël Andrieu (mis à jour le 12/12/2019) « Créez une application web en PHP de qualité professionnelle » disponible ici : https://openclassrooms.com/fr/courses/6031956-creez-une-application-web-en-php-de-qualite-professionnelle/6107751-quest-ce-quune-application-web

-Nicolas Bouliteau (2017) « Méthode traditionnelle VS. Méthodes agiles » disponible ici : https://fr.linkedin.com/pulse/m%C3%A9thode-traditionnelle-vs-m%C3%A9thodes-agiles-nicolas-bouliteau-pmp

-Orléans Métropole (2019) « Les chiffres clés de la mobilité dans la métropole orléanaise 2018 » disponible ici : https://www.orleans-metropole.fr/deplacements/politique-de-deplacements

-Raphael Yende. COURS DE METHODES DE CONDUITE DES PROJETS INFORMATIQUES. Licence. Congo-Kinshasa. 2019, 90p. ffcel-02004689f disponible ici : https://hal.archives-ouvertes.fr/cel-02004689/document

-Site internet de l'agiliste « Guide de démarrage Scrum » disponible ici : https://agiliste.fr/guide-de-demarrage-scrum/#Les-R-les-en-bref

-Site internet de l'agiliste « Introduction aux méthodes agiles et Scrum » disponible ici : https://agiliste.fr/introduction-methodes-agiles/

-Site internet de la DREAL « Les enjeux de la mobilité durable » consulté le 06/09/2020 et disponible ici : http://www.centre-val-de-loire.developpement-durable.gouv.fr/les-enjeux-de-la-mobilite-durable-r1086.html

-site internet Formation Django « Qu'est-ce qu'un framework » disponible ici : http://www.formation-django.fr/generalites/framework.html

-Site internet Nemesis-Studio (2020) « COMMENT METTRE EN PLACE UN WEB SERVICE/API ? » disponible ici : https://www.nemesis-studio.com/mettre-place-web-service-api/

-Site internet scient.fr (2020), « L'industrialisation des POC (Proof-of-Concept) » disponible ici : https://scient.fr/lindustrialisation-des-poc-proof-of-concept/

TABLE DES MATIERES