Cahier des spécifications v0.3

Projet PFA - 2015/2016

Application mobile pour rediriger les usagers lors des pannes de transport sur le réseau de Bordeaux-Métropole

Responsable Pédagogique	Toufik AHMED	ENSEIRB-MATMECA
Contact Client	Nicolas BONNOTE	Qucit

Réalisateurs

Mehdi BOUNAKHLA
Romain RAMBAUD
Nathan REAVAILLE
Mohammed RIHANI
Pierre ROUX
Victor SAINT GUILHEM

Introduction

Contexte

Description initiale du projet par le client

Le projet consiste à développer une application mobile multi-plateforme permettant de rediriger les usagers en cas de panne de transport sur le réseau de Bordeaux Métropole.

Une première étape consiste en la collection de données à partir de sources différents (infotbc, twitter, qucit, …) afin d'avoir une vision globale de l'état du réseau de la ville. Il faudra ensuite détecter les pannes dans le réseau de transport de bordeaux métropole.

La deuxième étape consiste à développer une application mobile simple multi-plateforme (basée sur la plateforme polymer de Google) afin de permettre aux utilisateurs d'enregistrer leurs trajets habituels et de recevoir des notifications si ce trajet est affecté par une panne du réseau. Les notifications permettent non seulement d'informer l'utilisateur de l'existence d'un problème, mais aussi de lui proposer un itinéraire alternatif pour son déplacement.

État d'avancement

Cette deuxième version du cahier des spécifications est le résultat d'une maturation de l'interprétation des besoins du client. Cela a été rendu possible par un premier retour du client pendant une réunion lors de laquelle la première version de ce document avait été apportée pour y être présentée. Une réflexion plus poussée au sein de l'équipe a permis d'apporter de nombreuses précisions et corrections au document initial.

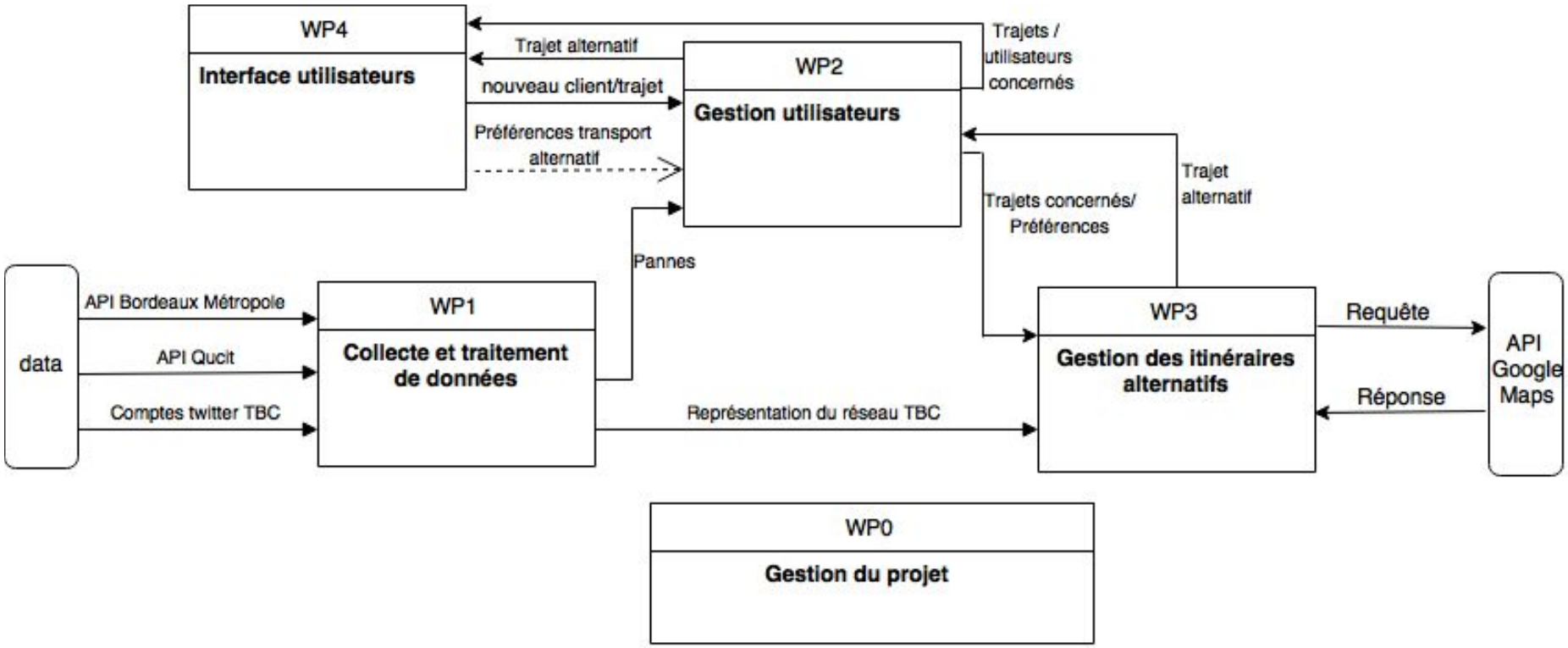
Présentation du document

Objet du document

L'objectif de ce dossier est de décrire l'aspect fonctionnel de l'application à produire, et ainsi de permettre aux réalisateurs de mieux cerner le besoin. Il décrit donc l'aspect visible de l'application, mais aussi après découpage logique, des fonctionnalités internes et des flux de données reliant les modules principaux identifiés.

Guide de lecture

Il a été déterminé avec le responsable pédagogique que le travail à réaliser serait divisé en Working Package (WP) où chaque WP rassemble les fonctionnalités voisines.
Le premier chapitre présente le WP relatif à la gestion du projet en lui-même. Il contient les diagrammes des dépendances faisant référence aux autres WP. Chacun des chapitres suivants présente donc un de ces WP, dont les relations sont détaillées dans le digramme intitulé "Découpage du projet en Working Package" en page suivante.



Découpage du projet en Working Packages

WP0 - Gestion de projet

Ce Working Package a pour objectif de décrire la méthode de suivi de projet suivi durant ce PFA, ainsi que notre planning prévisionnel.

Méthode Agile - SCRUM

Il a été décidé en début de projet que nous allions utiliser la méthode SCRUM pour notre suivi de projet. Les raisons principales d'un tel choix sont les suivantes :

- ce type de méthode permet de **s'adapter plus facilement aux changements** des besoins clients en cours de projet. L'objectif principal étant toujours de proposer à la fin du projet un produit qui soit le plus proches des attentes client ;
- les **rendus de livrables intermédiaires** réguliers, très intéressant vis à vis du client qui souhaite particulièrement le module de collecte de données ;
- la **communication régulière** avec le client avant et pendant la réalisation du projet, permettant la collaboration entre les deux équipes ;
- le **souhait du client** d'utiliser les méthodes Agiles, et le fait que SCRUM soit aujourd'hui la plus utilisée dans le monde professionnel pour ce type de gestion de projet.

Afin de mieux appréhender la suite de cahier des spécifications, la méthode Scrum et ses mécanismes seront introduits à la suite de ce document. Nous présenterons aussi comment nous l'avons adapté afin de coller au contexte particulier des PFA.

Les rôles dans SCRUM

Premièrement, la méthode SCRUM introduit trois rôles importants :

- **Le Product-Owner** : le client et son équipe. C'est cette entité qui va apporter les besoins à développer pour le produit ;
- **Le Scrum-Master**, qui garantit le suivi de la méthodologie SCRUM. Dans notre cas, si il n'a pas été défini précisément, nous savons d'ors et déjà qu'il s'agira d'un membre de l'équipe PFA ;
- **L'équipe de développement**, représentée par l'équipe PFA.

Les réunions dans SCRUM

Tout au long du projet, nous suivrons un rythme précis de réunion, afin d'assurer le bon fonctionnement de notre méthode :

- **Planification du sprint** : cette réunion a lieu avant le début de chaque sprint du projet. Elle a pour objectif de déterminer l'ensemble des tâches qui seront réalisées durant le sprint à venir. Ces tâches sont sélectionnées dans le "Product Backlog", selon leur priorité (définie par le client). Les tâches sélectionnées forment le "Sprint Backlog". Elles sont sélectionnées en fonction des capacités de développement de l'équipe, et en accord avec le client. Dans notre cas, ce sera le client PFA ;

- **Revue de sprint** : cette réunion qui a lieu à la fin de chaque sprint, permet de présenter au Product-Owner les fonctionnalités terminées au cours du sprint. L'équipe et le Scrum-Master doivent y recueillir ses feedbacks, et ajuster si besoin la planification des prochains livrables intermédiaires et donc le nombre de sprints. Dans notre cas, il sera intéressant de faire cette réunion à la fois avec le client PFA, mais aussi avec le responsable pédagogique, en deux fois si nécessaire. Durant ces réunions, le client pourra accepter ou refuser les fonctionnalités que l'on jugera terminées. Enfin, à chaque revue de sprint, il conviendra de calculer la "Vélocité" de l'équipe. Pour cela, il faut additionner les points d'estimation associées aux fonctionnalités acceptées. Une fonctionnalité partiellement terminée ne rapportera aucun point car une telle fonctionnalité n'est pas utilisable. Cela permet de vérifier que le projet a suffisamment de sprints prévus pour atteindre la fin du projet ;

- **Rétrospective du sprint** : réunion avec l'équipe de développement, se déroulant après la revue de sprint. L'objectif de cette réunion est l'amélioration continue lors du projet, on en profite donc pour faire un feedback en interne sur le sprint qui vient de s'écouler, afin de voir si l'on peut s'améliorer sur l'efficacité, la qualité du travail etc… de l'équipe.

- **La mêlée** : l'objectif de cette réunion est de faire un point régulier par l'ensemble de l'équipe de développement. Elle est assez courte et doit répondre aux questions suivantes : "Qu'est ce que j'ai terminé depuis la dernière mêlée ? Qu'est ce que je compte terminer d'ici la prochaine mêlée ? Quels sont les obstacles que je rencontre sur les taches qui m'ont été assignées ?". Dans notre cas, nos sprints devraient durer deux semaines en moyenne, l'organisation de mêlée bihebdomadaire semblent donc le plus approprié pour avoir un bon suivi.

Création du Product Backlog

Le Product Backlog comprend l'ensemble des fonctionnalités souhaitées par le client pour le projet. Il est écrit en début de projet. Chaque fonctionnalités est écrite sous forme de User-Story : "En tant que <*qui*>, je veux <*quoi*> afin de <*pourquoi*>", la partie <*pourquoi*> étant facultative.

Pour chaque User-Story, une estimation va être réalisée. Celle ci est entièrement arbitraire. On part de la tâche la plus simple, que l'on estime à 1 point. Et c'est à partir de cette unité de mesure que l'on estime les autres fonctionnalités. Ces points ne représentent donc en aucun cas des JH, ce qui permet de faciliter la planification des sprints. A partir de cette liste, le Product-Owner choisit ces éléments par ordre de priorité de réalisation.

L'objectif en théorie étant de proposer un cahier des charges très rapidement, qui soit peu détaillé, mais qui sera affiné au fur et à mesure du projet. A cause de nos contraintes d'évaluation sur le PFA, nous avons passé plus de temps sur la rédaction de ce cahier des charges, et l'avons déjà détaillé. Néanmoins, il restera modifiable en cours de projet en ce qui concerne les éléments de précision qui le constitue.

Durant le projet en SCRUM

Le Product Backlog étant fixé avec ce document, et la durée de sprint étant fixé à deux semaines, le projet peut commencer. Le client pourra affiner ses besoins à chaque réunion de début de sprint, et reprioriser les éléments de son backlogs.

Durant chaque sprint, se déroulera les mêlées bihebdomadaires, permettant de synchroniser l'ensemble de l'équipe de développement. Le bon suivi du projet sera effectué par le Scrum-Master, qui devra gérer d'éventuels changement dans le Product Backlog (ajout/modification/suppression de nouvelles fonctionnalités, changement de priorités) et les éventuels retards dans la production.

A la fin de chaque sprint, des tests Agiles sont effectués, afin de vérifier que les fonctionnalités proposées au client ne sont pas buggées.

Chaque sprint se termine par les réunions de revue et de rétrospective du sprint. Puis le cycle se répète jusqu'à atteindre la fin du projet. Du fait de la durée du projet, six sprints sont prévus pour la réalisation. Il sera possible d'en rajouter une septième en fonction de l'avancement du projet et la date de rendu au client.

Aussi, des graphiques seront actualisés à chaque fin de sprint, afin de s'assurer du bon déroulement du projet. Ces derniers représenteront le nombre de points réalisés en fonction du temps, avec en limite haute le nombre de point estimés pour l'ensemble du projet. De même, un autre graphe représentera le nombre de points restants en fonction du temps, avec une courbe représentant la trajectoire idéale. Ces deux graphiques sont bien sûr dans l'optique d'un contrôle de notre avancement au fur et à mesure.

Enfin, cette méthode de suivi de projet va nous permettre de mieux gérer les possibles risques que nous pourrions rencontrer durant le projet. Les risques identifiés à ce jour sont les suivants :

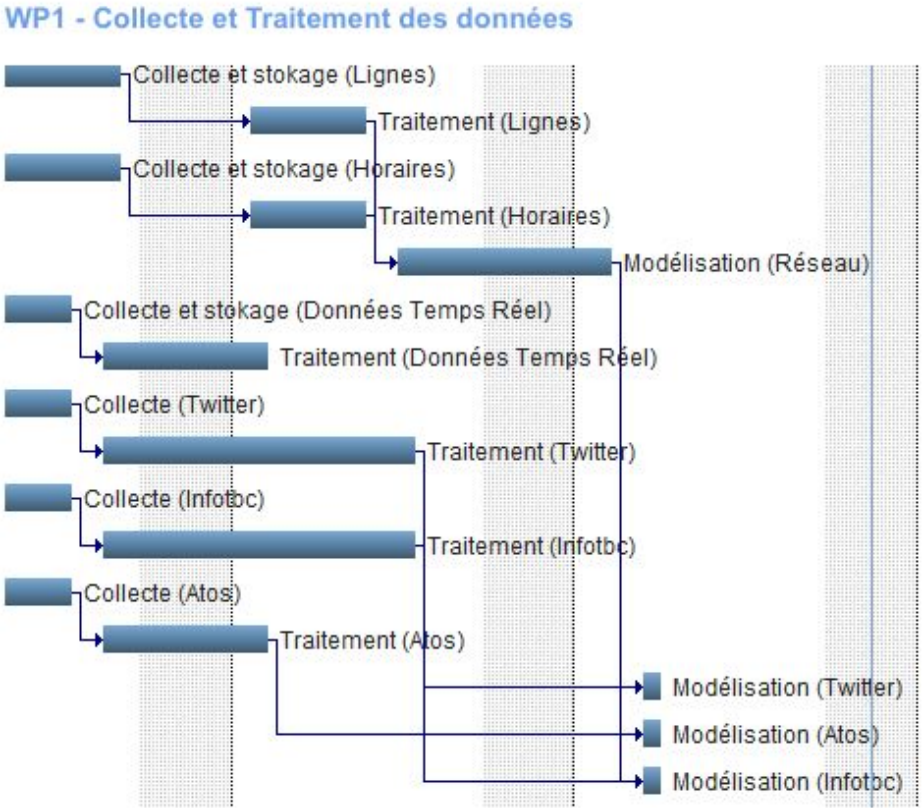
- Le fonctionnement de l'API de Google Maps pour la gestion des itinéraires alternatifs ;
- Le fonctionnement de Polymere pour l'interface utilisateur.

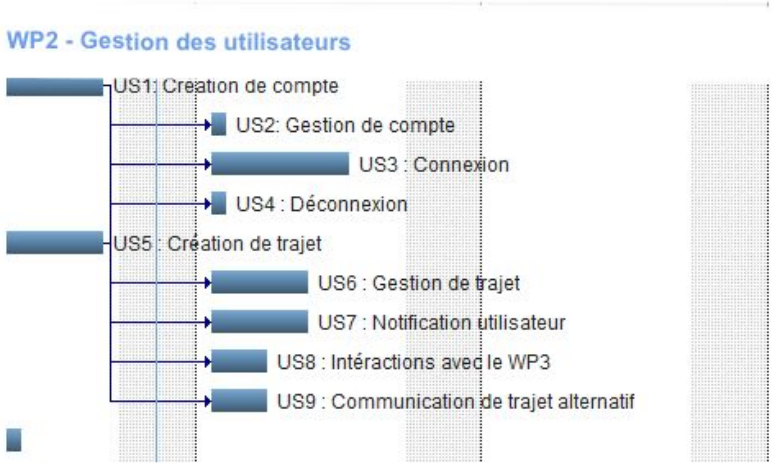
Si l'on rencontre des difficultés face à ces risques durant le projet, nous avons plusieurs possibilités pour ne pas prendre trop de retard :

- L'utilisation d'une autre technologie pour la réalisation du projet (par exemple une autre API) ;
- Changer les priorités de réalisation au début du sprint suivant ;
- Affecter plus de personnes sur la réalisation d'un point difficile.

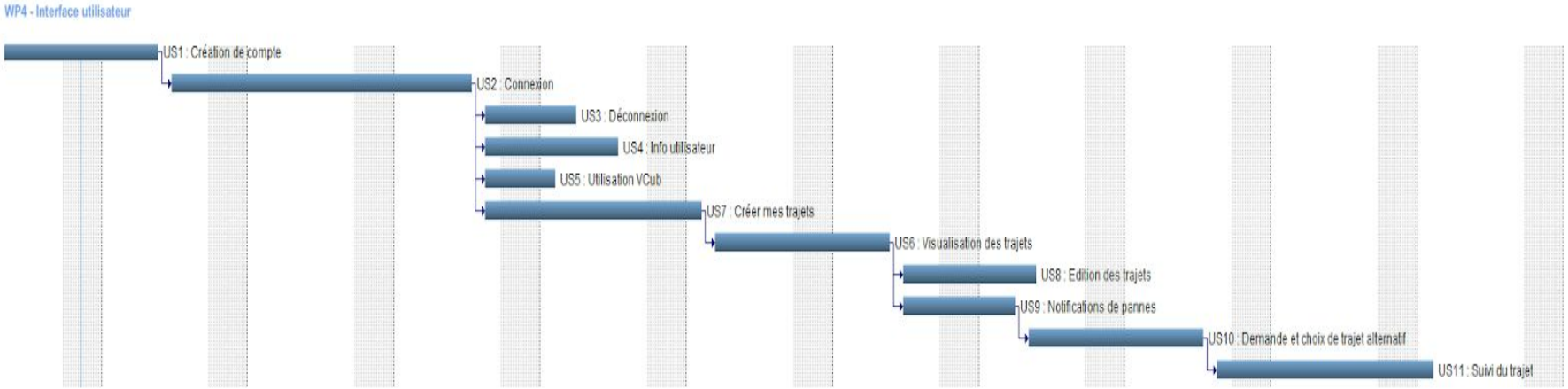
Diagrammes de dépendances

Nous avons découpé notre Product Backlog en plusieurs sous parties, appelées Working Package. Pour chacun d'entre eux, nous avons effectué une évaluation de leurs fonctionnalités. En fonction de celles ci, nous avons pu réaliser des premiers diagrammes de dépendants, qui seront bien sûr amené à évoluer en fonction du déroulement de nos premiers sprints.


WP1 - Collecte et Traitement des données
Collecte et stokage (Lignes)
Traitement (Lignes)
Collecte et stokage (Horaires)
Traitement (Horaires)
Modélisation (Réseau)
Collecte et stokage (Données Temps Réel)
Traitement (Données Temps Réel)
Collecte (Twitter)
Traitement (Twitter)
Collecte (Infotbc)
Traitement (Infotbc)
Collecte (Atos)
Traitement (Atos)
Modélisation (Twitter)
Modélisation (Atos)
Modélisation (Infotbc)

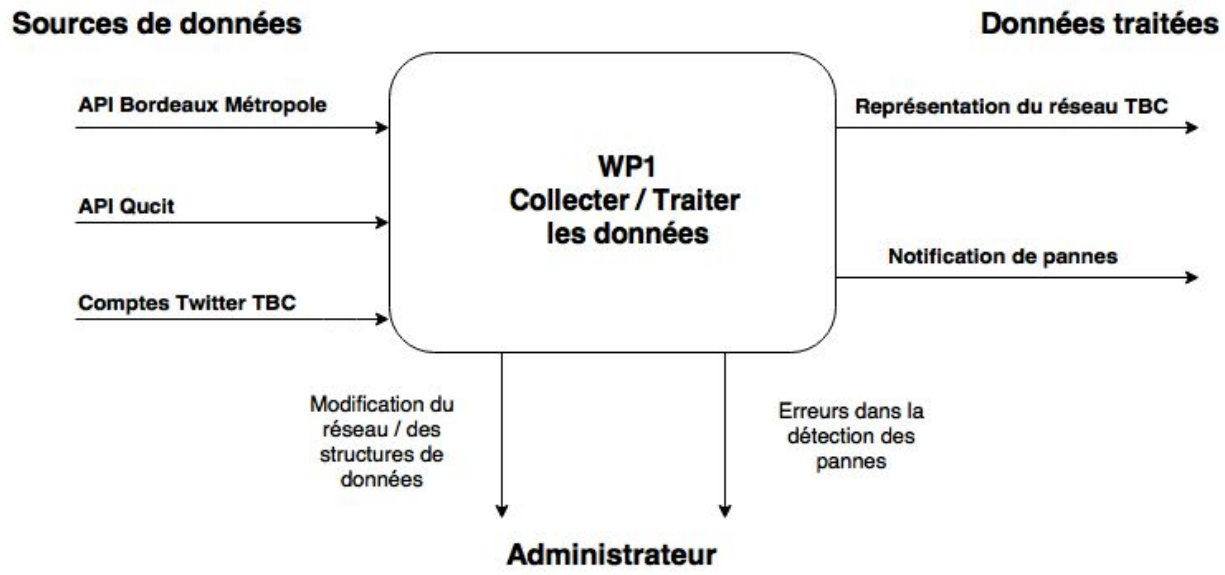

WP2 - Gestion des utilisateurs
US1: Creation de compte
US2: Gestion de compte
US3 : Connexion
US4 : Déconnexion
US5 : Création de trajet
US6 : Gestion de trajet
US7 : Notification utilisateur
US8 : Intéractions avec le WP3
US9 : Communication de trajet alternatif


WP3 - Gestion des itinéraires alternatifs
US1 : Récupération des trajets affectés
US2 : Création des itinéraires alternatifs
US3 : Communication des itinéraires alternatifs


WP4 - Interface utilisateur
US1 : Création de compte
US2 : Connexion
US3 : Déconnexion
US4 : Info utilisateur
US5 : Utilisation VCub
US7 : Créer mes trajets
US6 : Visualisation des trajets
US8 : Edition des trajets
US9 : Notifications de pannes
US10 : Demande et choix de trajet alternatif
US11 : Suivi du trajet

WP1 - Collecte et Traitement des données

Ce WP a pour but de fournir l'ensemble des modules permettant de collecter, traiter et stocker les données du réseau de transport TBC et des pannes. Ce WP, ainsi que les autres WP "backend" seront développés avec le micro-frameword Flask.



WP1 - Vue externe des flux

Cas d'utilisation du WP : flux entrants

Collecte des données : Réseau de transport (API Data Metropole)

➡ En tant que serveur, je dois fournir une représentation à jour de l'ensemble du réseau de transport TBC.

Critères

- Afin de modéliser le réseau TBC, il est nécessaire d'avoir accès aux données suivantes :
  - les arrêts : les types (TRAM/BUS/BATEAU), l'emplacement géographique, les lignes qui y passent ;
  - la constitution des lignes : les types (TRAM/BUS/BATEAU), leurs arrêts, les directions et sens possibles, les boucles, l'existence d'arrêts partiels, et différentes variations ;
  - les messages qui s'affichent sur les écrans du réseaux TBC (stations/tram/bus) ;
  - les horaires en temps réel de chaque ligne, en différenciant les périodes spécifiques : journée, soir, nuit, weekend/jour férié, période scolaire ;
  - les horaires théorique de chaque ligne, en différenciant les périodes spécifiques : journée, soir, nuit, weekend/jour férié, période scolaire ;
- Les requêtes permettant d'obtenir ces données devront être faites automatiquement et périodiquement afin de toujours avoir une représentation la plus à jour du réseau de transport.

Collecte des données : Pannes (Twitter)

➡ En tant que serveur, je dois détecter, traiter et stocker les pannes du réseau TBC.

Critères

- Afin de modéliser les pannes du réseau, il est nécessaire d'avoir accès aux données suivantes :
  - les types de pannes possibles : modification programmée d'une ligne
  - la raison de chaque panne : pour travaux, évènements
- Le traitement d'un message de panne se fait en reconnaissant la partie de réseau affectée (arrêt unique ou tronçons) et la durée de la panne si elle est donnée. Dans le meilleur des cas, cela permet donc d'obtenir des données spatio-temporelles qui vont affecter le graphe de modélisation du réseau..

Collecte des données : Vélos (API Qucit)

➡ En tant que serveur, je dois fournir une représentation des stations de Vcub et de la disponibilité des vélos aux stations.

Critères

- Afin de modéliser les stations de Vcub et la disponibilité des vélos aux stations, il est nécessaire d'avoir accès aux données suivantes :
  - les stations : l'emplacement géographique, le nombre de vélos disponibles et le nombre de places.

Détection de la non-conformité des données

- ➡ En tant que serveur, je dois détecter qu'un changement dans les structures de données des sources de données rend la modélisation du réseau et des pannes non conformes.
- ➡ En tant que serveur, je dois détecter que le traitement des messages de pannes est défectueux.

Critères

- Lorsque le nombre d'erreurs de l'algorithme traitant les données du réseau TBC dépasse une valeur prédéfinie pendant un intervalle de temps, le serveur doit en avertir l'administrateur pour que celui-ci effectue une correction des algorithmes de traitement
- Lorsqu'un nombre de message de pannes non détectés dépasse une valeur prédéfinie pendant un intervalle de temps, le serveur doit en avertir l'administrateur pour que celui-ci effectue une correction des algorithmes de traitement des messages de pannes.

Cas d'utilisation du WP : flux sortants

Fournir les données complètes au module d'itinéraires alternatifs (WP2)

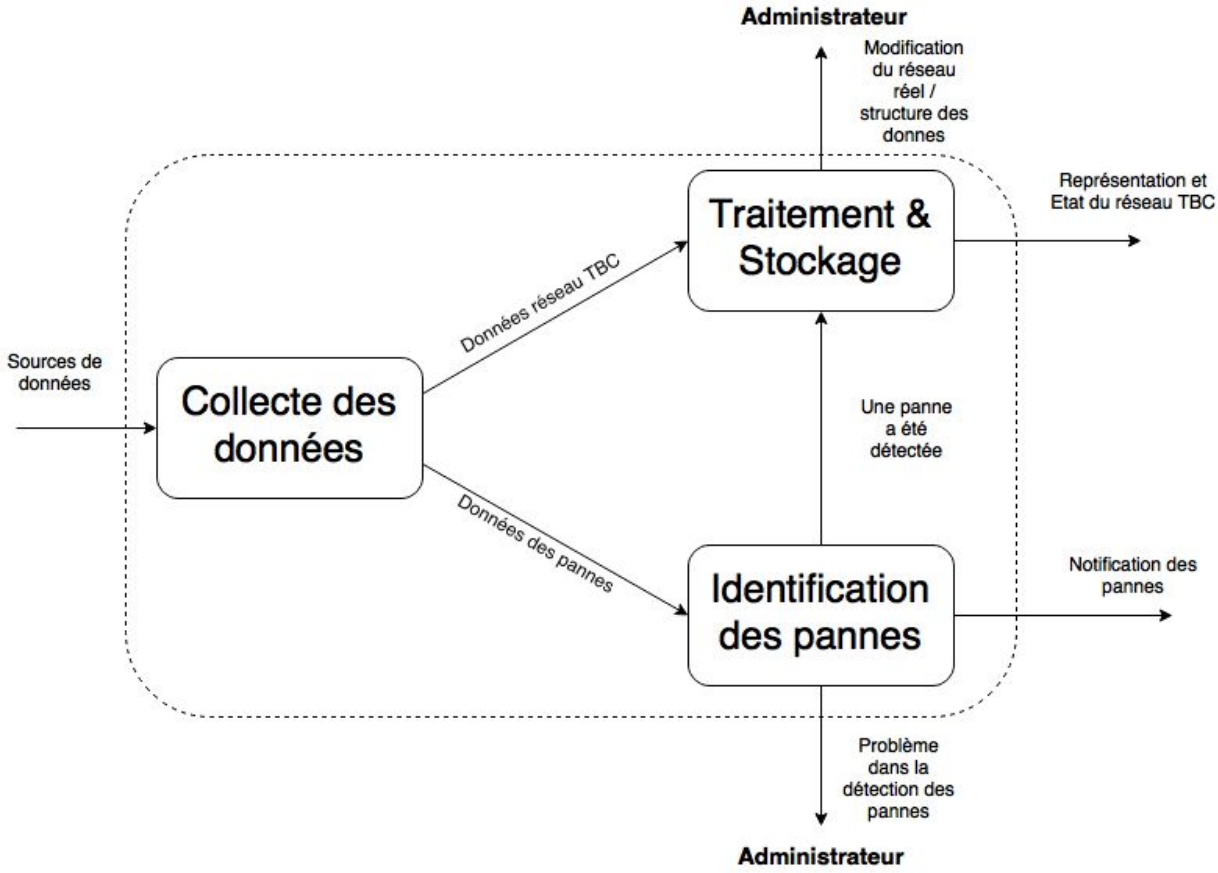
- ➡ En tant que serveur, je doit fournir, au module de recherche d'itinéraires alternatifs, toutes les données nécessaires à son algorithme.

Critères

- Afin de réaliser l'application complète, les autres WP ont besoin d'une d'accéder à un modèle de représentation du réseau TBC. Ce Working Package fait donc office d'API interne pour les autres modules de l'application.

Notification des pannes à la Gestion des utilisateurs (WP3)

- ➡ En tant que serveur, je doit notifier le module de Gestion utilisateur de chaque panne détectée sur le réseau TBC.



WP1 - Vue interne des flux

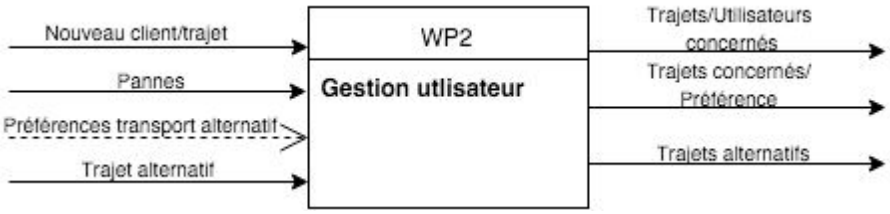
Le WP1 est donc composé de 3 modules principaux :

- **Collecte des données** à partir de plusieurs sources : API Data Metropole, API QUCIT, et API Twitter. Cette collecte devra être périodique
- **Traitement et stockages**
- **Identification des pannes**

Nous avons volontairement fusionné la description des fonctionnalités et des critères des modules de Collecte des données avec celui de Traitement & Stockage car ce WP se traduira par un module unifié qui présentera une interface aux autres modules de l'application.

WP2 - Gestion des utilisateurs

Ce Working Package a pour but de permettre la gestion des utilisateurs. Il fait l'intermédiaire entre l'interface utilisateur et les autres modules de l'application. Les communications entre les clients et le serveur respecteront les standards HTTP, en particulier en ce qui concerne la gestion d'erreur.



WP2 - Vue de flux

Connexion/Déconnexion et inscription

- ➡ En tant qu'administrateur, je souhaite que lorsqu'un utilisateur s'inscrit sur l'application, ses informations soient stockées sur le serveur.
- ➡ En tant qu'administrateur, je souhaite que l'utilisateur puisse modifier/supprimer son compte.
- ➡ En tant qu'administrateur, je souhaite qu'un utilisateur puisse se connecter à l'application s'il est inscrit.
- ➡ En tant qu'administrateur, je souhaite pouvoir prendre en compte l'état d'un utilisateur (actif ou inactif).

Critères

- Un utilisateur qui s'enregistre sur l'application doit avoir ses données stockées sur la base de données "User" de l'application.
- La base de données doit contenir une table "User" qui doit avoir les champs nécessaires pour stocker les informations demandées lors de l'inscription sur l'UI.
- Le mot de passe ne devra pas être stocké en clair dans cette table.
- Un utilisateur a la possibilité de modifier/supprimer son compte.
- Dans le cas d'une tentative de connexion, le serveur devra vérifier que le login renseigné est présent dans la base de données, et que le mot de passe associé est correct. Si c'est le cas, la connexion est validée. Dans le cas contraire, le serveur renvoie un code d'erreur à l'UI.
- Dans le cas d'une demande de suppression de compte depuis l'UI, les informations relatives à l'utilisateur seront supprimées dans la base de données.
- Si, lors d'une nouvelle inscription, le mail renseigné par l'utilisateur est déjà présent dans la base de données, l'inscription n'est pas enregistrée en BDD et un message d'erreur est renvoyé à l'UI.
- Si une requête de nouveau mot de passe est envoyée, le système vérifier que l'adresse mail concernée est bien présente dans la table "User". Si c'est le cas, un mail sera envoyé contenant un lien de réinitialisation du mot de passe.

- Une fois que l'utilisateur s'est connecté sur l'application, les identifiants seront automatiquement enregistrés sur le téléphone pour éviter à l'utilisateur de les taper à chaque lancement.

Gestion des trajets/trajets alternatifs et des pannes

- ➡ En tant qu'administrateur, je souhaite que l'ensemble des trajets de chaque utilisateur soient enregistrés sur le serveur.
- ➡ En tant qu'administrateur, je souhaite que l'utilisateur puisse modifier/supprimer ses trajets et les rendre actifs/inactifs.
- ➡ En tant qu'administrateur, je souhaite qu'une notification soit envoyée à l'utilisateur afin de le prévenir d'une panne.
- ➡ En tant qu'administrateur, je souhaite relayer les trajets concernés par la panne et les préférences utilisateur à l'algorithme de calcul d'itinéraires alternatifs.
- ➡ En tant qu'adminstrateur, je souhaite communiquer les trajets alternatifs aux utilisateurs concernés.

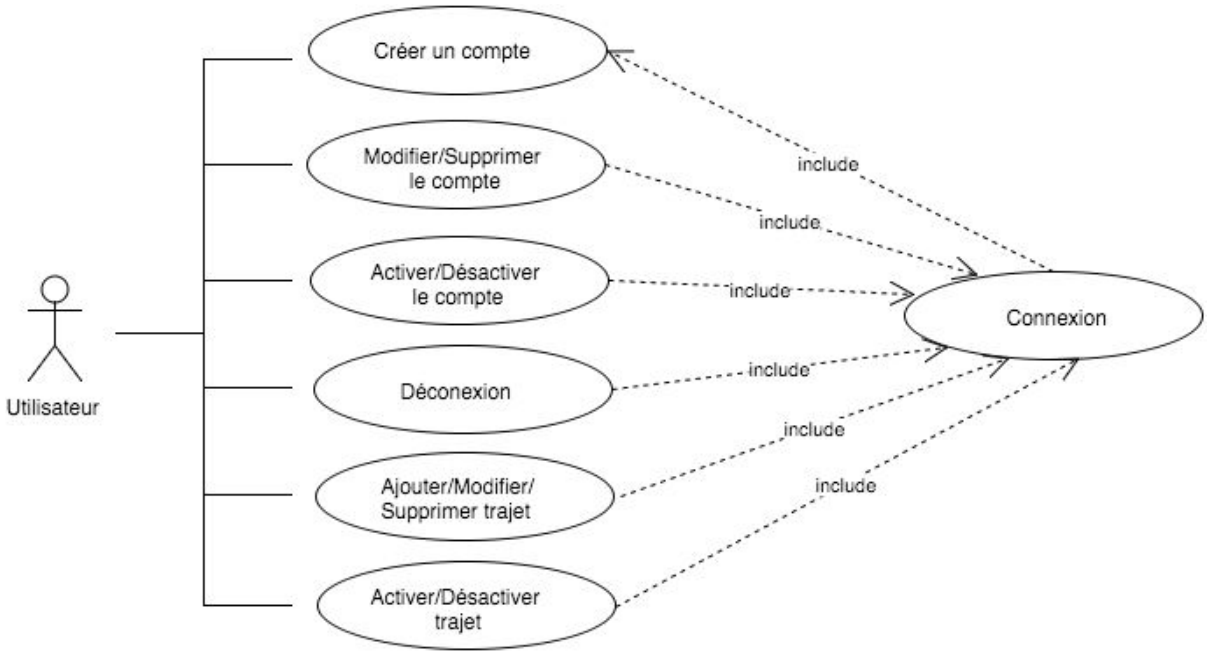
Critères

- Un utilisateur enregistré sur l'application doit avoir ses trajets stockées sur la base de données "Path" de l'application et a la possibilité de modifier/supprimer ces informations.
- La base de données doit contenir une table "Trajet" avec les champs nécessaires à l'enregistrement d'un nouveau trajet : heure de départ, heure d'arrivée, transport empruntés, arrêt de départ, arrêt d'arrivée. Il pourra indiqué des correspondances, et pour chacune d'entre elles, il sera demandé l'arrêt de départ, l'arrêt d'arrivée et le transport emprunté.
- Lorsque le module de collecte et de traitement de données fait émerger une panne, le module de gestion utilisateur doit vérifier si un trajet est concerné ; dans ce cas une notification doit être envoyée à l'utilisateur.
- Si un utilisateur demande un trajet alternatif, la requête sera relayée vers le WP3.
- Lorsque le calcul des itinéraires alternatifs est terminé, le serveur doit pouvoir identifier les destinataires et leur envoyer les différents itinéraires.

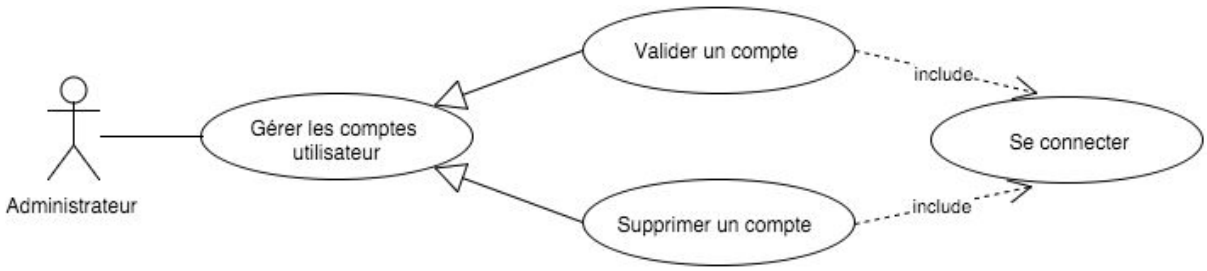
Diagramme des cas d'utilisation

Ci-dessous les diagrammes de cas d'utilisation de ce WP Gestion des utilisateurs :

Utilisateur

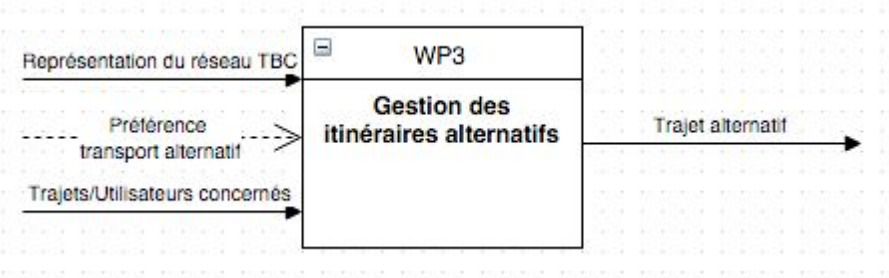


Administrateur



WP3 - Gestion des itinéraires alternatifs

Ce WP a pour but d'offrir le service de gestion des itinéraires alternatifs.



WP3 - Vue de flux

- ➡ En tant qu'algorithme, je souhaite récupérer les trajets/utilisateurs concernés par une panne.
- ➡ En tant qu'algorithme, je souhaite avoir une représentation en temps réel du réseau de transports en commun de Bordeaux.
- ➡ En tant qu'algorithme, je souhaite créer au moins un itinéraire alternatif pour ces trajets, en tenant compte des préférences utilisateur.
- ➡ En tant qu'algorithme, je souhaite communiquer les nouveaux itinéraires au WP2.

Critères

- L'algorithme prendra au moins en entrée un trajet impacté par une panne, les préférences de l'utilisateur concerné (avec ou sans vCub) et l'horaire de départ.
- L'algorithme devra disposer d'une représentation du réseau en temps réel. Il pourra se baser sur la représentation du réseau réalisé par le WP1.
- Pour un trajet, le module devra calculer un nouvel itinéraire. Cet itinéraire devra utiliser essentiellement l'API Google Maps. Dans le cas où l'API Google Maps n'offre les fonctionnalités nécessaires, le module pourra utiliser GraphHopper.
- L'algorithme devra retourner au moins un itinéraire alternatif. Dans un second temps, l'algorithme pourra retourner plusieurs itinéraires afin de laisser le choix à l'utilisateur, notamment sur les types de transport.
- Une fois le nouvel itinéraire calculé, celui ci sera renvoyé au WP Gestion utilisateur.

WP4 - Interface utilisateur

L'ensemble des pages de l'application seront basées sur la charte graphique du client, que ce soit pour la police ou les couleurs.

On retrouvera en haut de l'application une barre de menu, permettant d'accéder aux différentes parties de l'application, ainsi qu'une redirection vers la page principale depuis le logo.

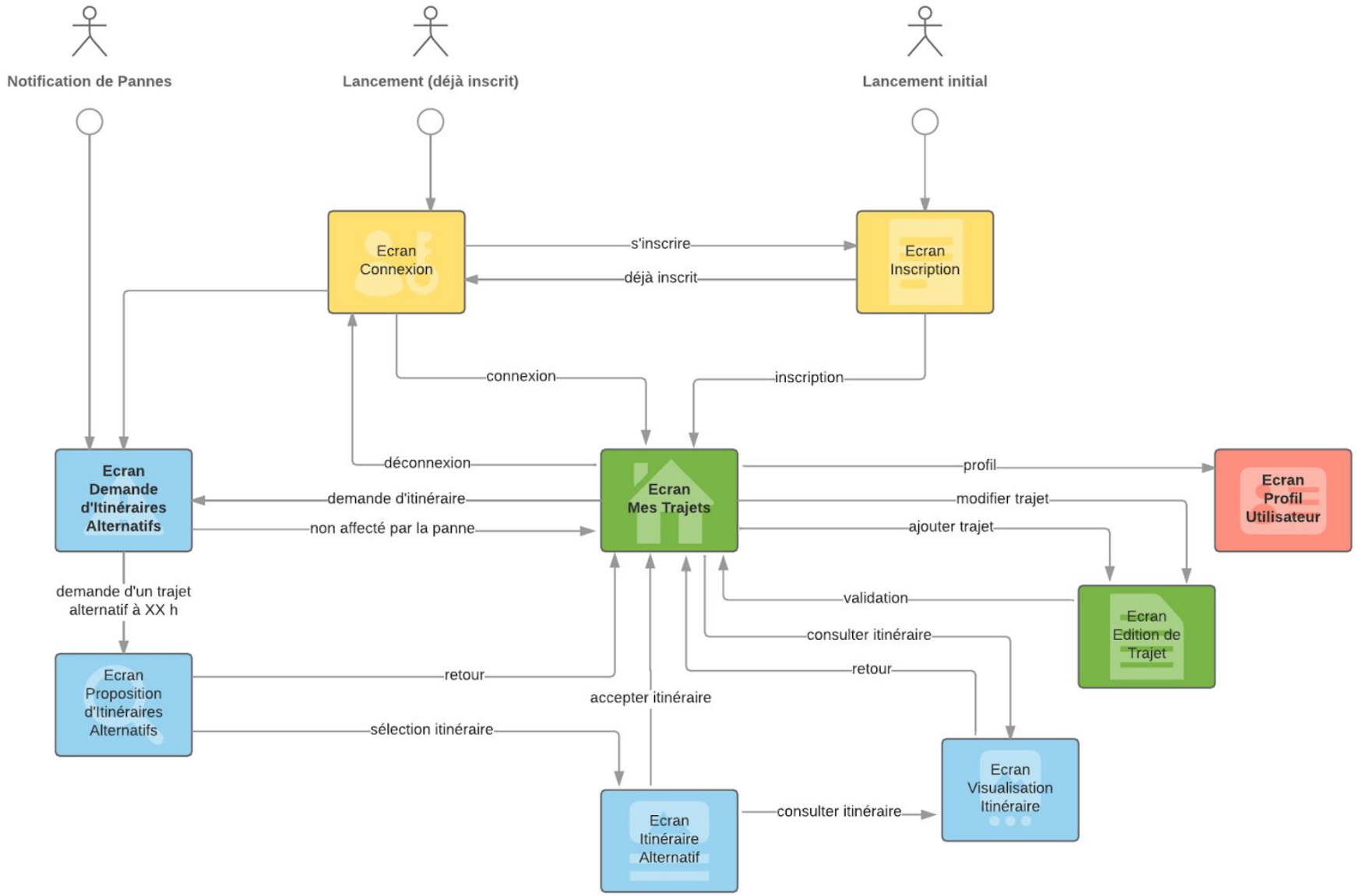
Afin de suivre les dernières tendances en application mobile, et d'exploiter au mieux Polymer, l'application sera “Flat Design”.

Le diagramme ci-dessus présente le schéma de navigation de l’application mobile que l’utilisateur final va exécuter sur son smartphone.
Les cases représentent les différents écrans de l’application. Un écran est accessible à partir des flèches, qui représentent les actions qui y mènent. Trois cas d’utilisations exécutant l’application ont été modélisés :

- **le lancement initial** : l’utilisateur exécute l’application pour la première fois. Il doit d’abord s’inscrire. Puis il pourra commencer à remplir ses trajets ;
- **les lancements suivants** : l’utilisateur exécute l’application alors qu’un compte a déjà été créé sur son téléphone. Il doit donc se connecter. Celle-ci est automatique si son compte est toujours actif sur son smartphone ;
- **la réception d’une notification** : le serveur transmet l’information de panne. L’utilisateur clique sur la notification, et l’application permet à l’utilisateur de chercher une solution.

Ces trois cas représentent seulement des interactions initiales possibles qui déclenchent le lancement de l’application. Ils ne présument en rien des actions suivantes que réalise l’utilisateur sur l‘application.

Les User-Stories suivant le diagramme détaillent le contenu, les conditions et les actions réalisables par l’utilisateur.



APPLICATION MOBILE :
SCHEMA DE NAVIGATION

Connexion / Inscription

- ➡ En tant qu'utilisateur, je souhaite pouvoir me créer un compte sur l'application.
- ➡ En tant qu'utilisateur, je souhaite pouvoir me connecter à l'application si je possède déjà un compte.
- ➡ En tant qu'utilisateur, je veux pouvoir me déconnecter.

Critères

- Après une page d'ouverture, l'utilisateur sera dirigé vers une page d'inscription lors du premier lancement de l'application. Cette page d'inscription contiendra les champs suivants : nom, prénom, email, mot de passe. En ce qui concerne le champ email et mot de passe, un contrôle sur les caractères sera fait (présence d'un @, de point pour le email, pas de caractère accentué dans le mot de passe).
- Passée la première utilisation de l'application, lorsque l'utilisateur lancera l'application, il sera dirigé vers une page de connexion si il ne s'est jamais connecté sur le périphérique utilisé. Après la première connexion, les identifiants seront stockés sur le périphérique, connectant l'utilisateur automatiquement à chaque lancement de l'application.
- La page de connexion contiendra les champs suivants : mail, mot de passe. Un lien mot de passe oublié sera disponible en dessous ces deux champs.
- Dans le cas d'un mot de passe oublié, il sera possible d'en demander un nouveau en cliquant sur le lien "Mot de passe oublié ?". Ce lien redirigera vers une page demandant l'adresse mail de l'utilisateur afin de lui envoyer un lien de réinitialisation du mot de passe, si cette adresse mail correspond bien à un utilisateur enregistré.
- Depuis la page principale, l'utilisateur aura accès à un onglet de préférences. Parmi ces onglets se trouvera un bouton permettant à l'utilisateur de se déconnecter. Lors de la déconnexion, les identifiants seront automatiquement effacés du téléphone pour éviter la connexion automatique.

Gestion du compte

- ➡ En tant qu'utilisateur, je veux pouvoir modifier mes informations.
- ➡ En tant qu'utilisateur, je veux pouvoir personnaliser mes préférences de transport

Critères

- L'onglet de préférences sera doté d'un sous-onglet "Paramètres" qui redirigera vers un formulaire contenant les champs suivants : Prénom, Nom, Adresse Mail, Nouveau mot de passe, Confirmer nouveau mot de passe.
- L'onglet de préférences sera doté d'un sous-onglet "vCub" qui redirigera vers une page proposant de désactiver l'utilisation des vCub dans les itinéraires alternatifs.

Gestion des trajets

- ➡ En tant qu'utilisateur, je veux pouvoir créer mes différents trajets quotidiens.
- ➡ En tant qu'utilisateur, je veux pouvoir visualiser l'ensemble de mes trajets.
- ➡ En tant qu'utilisateur, je veux pouvoir activer/désactiver mes trajets.
- ➡ En tant qu'utilisateur, je veux pouvoir modifier et supprimer mes différents trajets quotidiens.

Critères

- La page d'accueil de l'application sera la page des trajets. Ils seront tous affichés avec leur label, leur récurrence et un curseur « activé / désactivé ».
- Lorsqu'un curseur passe en mode « désactivé », les pannes affectant le trajet concerné ne seront plus notifiées. Un bouton de veille générale de l'application sera également présent, sauvegardant la configuration actuelle mais coupant jusqu'à réactivation toutes les notifications.
- Sur cette page se trouvera aussi un bouton « + » pour ajouter de nouveaux trajets. Une page s'ouvrira, l'utilisateur devra choisir le type de transport. Puis, il remplira deux formulaires pour les points de départ et d'arrivée. Chacun de ces formulaires disposera des champs suivants : label, ligne, arrêt. Si le trajet ne présente pas de correspondance, l'utilisateur pourra renseigner la récurrence du trajet. Une fois le trajet validé, il sera affiché sur la page principale.
- Si un trajet présente une ou des correspondances, il sera possible de les ajouter via un bouton « J'ai une correspondance » sur la page de saisie du trajet. Un nouveau formulaire apparaîtra alors doté des champs permettant d'identifier la correspondance.
- Sur la page des trajets, il y aura un bouton « modifier ». Lorsque l'utilisateur pressera ce bouton, il pourra accéder à la modification ou à la suppression de chacun de ses trajets. Si il choisit de supprimer un trajet, une vérification sera demandée puis le trajet sera supprimer. Si il décide simplement de le modifier, il sera redirigé vers le formulaire de création du trajet déjà rempli ; il pourra alors les éditer.

Notifications et gestion des pannes

- ➡ En tant qu'utilisateur, je veux recevoir une notification sur mon téléphone lorsqu'une panne est détectée sur un de mes trajets.
- ➡ En tant qu'utilisateur, je veux avoir accès à au moins un itinéraire alternatif en transport en commun lorsqu'une panne est détectée.
- ➡ En tant qu'utilisateur, je veux pouvoir sélectionner le trajet alternatif proposé qui me convient le plus.
- ➡ En tant qu'utilisateur, je veux pouvoir suivre mon trajet alternatif en temps réel depuis une carte interactive.

Critères

- Lors de l'envoi d'une alerte par le serveur concernant une panne sur le réseau de transport TBC, une notification sera envoyée au téléphone afin qu'il affiche à l'utilisateur "Attention, votre transport de XXh sur la ligne XX est actuellement hors service."
- Si l'utilisateur lance l'application depuis la notification sur son téléphone, il sera redirigé directement vers la page proposant des itinéraires alternatifs. Dans le cas contraire, un indicateur graphique (type "1" rouge) sera présent sur le trajet concerné et ce trajet sera cliquable et renverra vers la page d'itinéraires alternatifs.
- Sur la page d'itinéraire alternatif, l'utilisateur signalera si il le veut une horaire à laquelle il aura besoin d'un itinéraire alternatif. Les itinéraires s'afficheront alors sous forme de liste présentant les informations essentielles de l'itinéraire types de transport utilisés, temps estimé, nombre de correspondances.
- En cliquant sur un des trajets alternatifs, il sera redirigé vers une carte représentant le nouveau trajet. Il aura alors accès à un bouton "Démarrer", lui permettant de choisir ce nouveau trajet, et de le suivre.
- En cliquant sur "Démarrer", il pourra alors soit suivre son trajet en temps réel sur une carte (à condition que la géolocalisation soit activée), soit avoir la liste des différentes étapes du trajet. Les différentes étapes seront de la forme "Départ - Arrivée, Heure départ, Transport, Ligne". Il conviendra d'intégrer les logos des différentes lignes (A, 10 etc...) pour améliorer l'ergonomie de l'application.