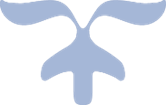




DOCUMENTATION PROJET

POUBELLE MALYRAIC

14/02/2024 – 16/09/2024



Adrien CARLES :

Mail : [adrien.carles@limayrac.fr](mailto:alexandre.escoffier@limayrac.fr)

User name GitHub : AdrienCarles

Jauffrey BICHEYRE :

Mail : [jauffrey.bicheyre@limayrac.fr](mailto:jauffrey.bicheyre@limayrac.fr) User name GitHub : JauffreyBicheyre

Valentin SIMON :

Mail : [valentin.simon@limayrac.fr](mailto:jauffrey.bicheyre@limayrac.fr) User name GitHub : Val-Smn

PROMOTION - ESI (IVERSON)

**Lien repository Git :** [**https://github.com/Val-Nost/poubelle.git**](https://github.com/Val-Nost/poubelle.git)

[**1 – Introduction 4**](#_heading=)

[**2 – Choix des technologies 5**](#_heading=)

[**3 – Choix de la méthodologie 5**](#_heading=)

[**4 – Diagrammes 6**](#_heading=)

[**5 – Difficultés rencontrées 10**](#_heading=)

[**Mode d’emploi utilisateur 10**](#_heading=h.el1wp1kpe9er)

[**Guide d'installation et de configuration 19**](#_heading=h.tko93zxsaqnt)

[Prérequis 19](#_heading=h.g2fzorwanq23)

[Démarrer l’application 19](#_heading=h.gb5fgt9pvaq6)

[Les Utilisateurs 19](#_heading=h.8p982m96bye0)

[**Algorithme et Optimisation 20**](#_heading=h.v7e6id4yxndz)

[**Revues de sprints 22**](#_heading=h.6q5s7fhkhnhf)

[**Revues de réunions 25**](#_heading=h.wm1i6x4e0eyu)

# – Introduction

Afin de mener à bien le projet dans les délais impartis nous avons établis plusieurs stratégies :

* + Lister toutes les tâches nécessaires à la réalisation du projet
  + Trier ces tâches par ordre d’importance, et les répartir sur la totalité du projet via les sprints
  + Garder un suivi de notre progression via Git, Trello, nos propres documents…
  + Faire un point régulier avec notre référent de projet

Les tâches peuvent être « communes » ou « individuelles » :

Les tâches communes sont généralement les tâches les plus techniques/difficiles. Le fait de les réaliser ensemble à souvent permis d’avancer plus rapidement, chacun ayant ses compétences et façon de coder. Ou alors les tâches communes peuvent être des tâches qui requiert l’accord des deux membres de l’équipe (base de données, planning des

tâches…)

Les tâches individuelles sont par conséquent des tâches plus simples à réaliser, ou que l’un des membres de l’équipe maîtrise correctement.

Dans tous les cas, l'entraide est un point central de notre façon de travailler, même sur les tâches dites individuelles.

*(Voir le document « Revues de sprints » pour voir la liste complète des tâches, leurs attributions par membre de l’équipe et leurs répartitions sur les différents sprints).*

Le document *« Revues de réunions »* résume l’échange régulier et constant que nous avons eu tout au long du projet avec notre référent de projet. Son retour général, ses remarques/conseils/axes d’amélioration…

Le document *« Documentation technique »* quant à lui, rentre plus dans le détail de la conception du projet, il explique comment certaines parties du site ont pu être implémentées.

Un document README est également présent à la racine du [repository git](https://github.com/Val-Nost/poubelle.git). Il explique comment accéder au site (en local ou bien en ligne au choix) et comment lancer les tests unitaires

Pour finir, ce document permet de retranscrire le déroulement de l’ensemble de ce projet. Avec les choix de méthodologies et de technologies, le rendu visuel du site et les difficultés auxquelles nous avons dû faire face.

# – Choix des technologies

Nos choix technologiques se sont faits essentiellement en fonction de nos compétences personnelles avant même le début du projet. Nous avons donc décidé d’utiliser le Framework Spring, plus précisément Spring Boot, pour le backend. Et pour le frontend, les indispensables HTML, CSS et JS. Mais nous avons aussi utilisé quelques CDN de bootstrap comme Tailwind CSS afin de faciliter le design du site ainsi que sa responsivité sur mobile et tablette.

Un **CDN** est un réseau de serveurs distribués géographiquement qui travaillent ensemble pour fournir rapidement du contenu internet aux utilisateurs. Lorsqu'un utilisateur accède à un site web, le CDN lui livre les fichiers nécessaires (comme des images, des vidéos, des fichiers CSS ou JavaScript) depuis le serveur le plus proche de sa localisation physique. Cela réduit la latence et améliore les temps de chargement du site.

Tailwind CSS est un framework CSS utilitaire qui permet aux développeurs de concevoir rapidement des interfaces utilisateur en utilisant des classes prédéfinies directement dans le HTML.

Voici la structure du projet :

* Les fichiers Java pour le Backend avec notamment
  + Les DAO pour effectuer les requêtes vers la base de données
  + Les Services qui appellent les DAO et effectue un traitement potentiel des données récupéré
  + Les Controllers qui écoutent les appels depuis le serveur, et indique les fichiers HTML à afficher selon l’endpoint
  + Les RestController, qui permettent de récupérer les données sous format JSON très utile pour afficher la carte
* Des fichiers HTML (front-end)
* Des fichiers CSS pour le style (front-end)
* Des fichiers Javascript pour certaines fonctionnalités (Menu déroulant, Tabulation, etc…)
* Une base de données utilisant le Driver MariaDB

# – Choix de la méthodologie

Nous avons choisi de travailler avec les méthodes SCRUM et KANBAN à l’aide du logiciel de gestion de projet Trello, et ce pour plusieurs raisons. La méthode SCRUM repose sur le principe des sprints. C’est-à-dire que tout le travail pour effectuer le projet est divisé en une multitude de petites tâches. Ces tâches sont réparties entre les membres de l’équipe, et étalés sur toute la durée du projet.

Pour ce qui est de la méthode KANBAN, en plus de planifier le travail dans une itération, les membres de l’équipe récupèrent la tâche la plus prioritaire dans le backlog qui est prête à être développée. Les équipes vont utiliser un tableau Kanban pour visualiser ce travail à mesure qu’il progresse dans le flux de travail. Son objectif principal est d’identifier les importantes charges de travail par rapport à la production maximale d’un individu dans le processus et de les corriger.

Avec pour notre part un total 5 sprints qui ont une durée pouvant varier de 2 à 4 semaines. A la fin d’un sprint nous réalisons une réunion avec notre référent de projet afin de lui présenter notre avancement. Il nous donne ensuite des feedbacks afin de savoir si le travail effectué lui convient ou les améliorations qui peuvent être faîtes pour que le produit corresponde d'avantages à ses besoins.

Suite à ce feedback, nous réalisons une rétrospective du sprint et un compte rendu de la réunion sur les tâches réalisées, les points positifs à souligner ainsi que les points négatifs, amenant à des propositions d’améliorations.

Cette organisation à plusieurs avantages :

**Deadlines :** Deadlines pour chaque fin de sprint durant tout le projet qui force à garder un rythme de travail variable. Afin d’être plus précis, nous avons procédé de sorte qu’au moment d’une réunion (on rappelle que l’on doit assister à 5 réunions obligatoires) nous avons assez de contenus à présenter sur les tâches effectuées chacun de notre côté sans forcément tenir compte des deadlines de sprint. Cette symbolique de “liberté” on pourrait dire est du fait que dans le cahier des charges pour la gestion de projet, il n’y avait pas de consignes spécifiques concernant la méthode “Agile” à appliquer ainsi que le rythme de travail.

**Organisation :** Vue d’ensemble du projet et priorisation des tâches via l’ordre des sprints

**Coordination :** chaque membre du binôme va réaliser une tâche que l’autre membre a besoin pour que la fonctionnalité soit complète (ex : un qui réalise le CSS d’une page et un autre qui la peaufine car il est plus familier avec celle) ainsi le membre qui peaufine la tâche l’autre membre peut se consacrer sur une autre. Cela contribue à une meilleure productivité au sein du groupe et pour le projet .

Simplement, ce n’est pas toujours le cas une tâche peut être donné sans que l’un des autres membres n’interviennent (ex : un membre s’est concentré sur la réalisation de la carte tandis que un autre membre s’est penché sur la recherche d’itinéraire)

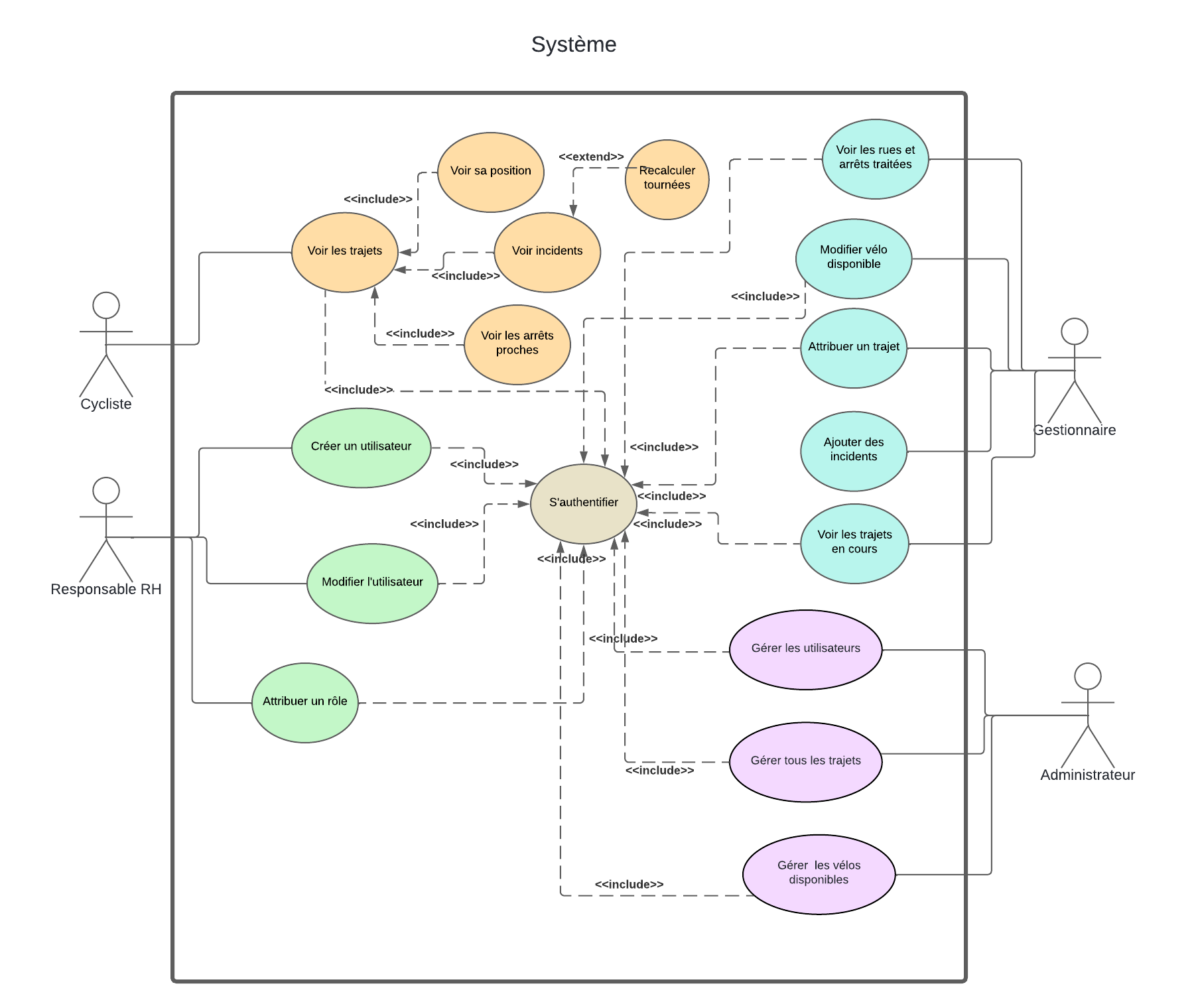
**Flexibilité :** Cette méthode de travail à la particularité d’être plutôt flexible en cas d’imprévu ou de changement de besoin. Et peut s’adapter aux emplois du temps de chacun.

Nous sommes cependant tous bien conscients des limites de cette méthode. Notamment en entreprise ou le client s’attend à des retours réguliers. D’autant plus avec une méthodologie Agile ou la durée des itérations et fixe et où un livrables est proposé à chaque fin d’itération.

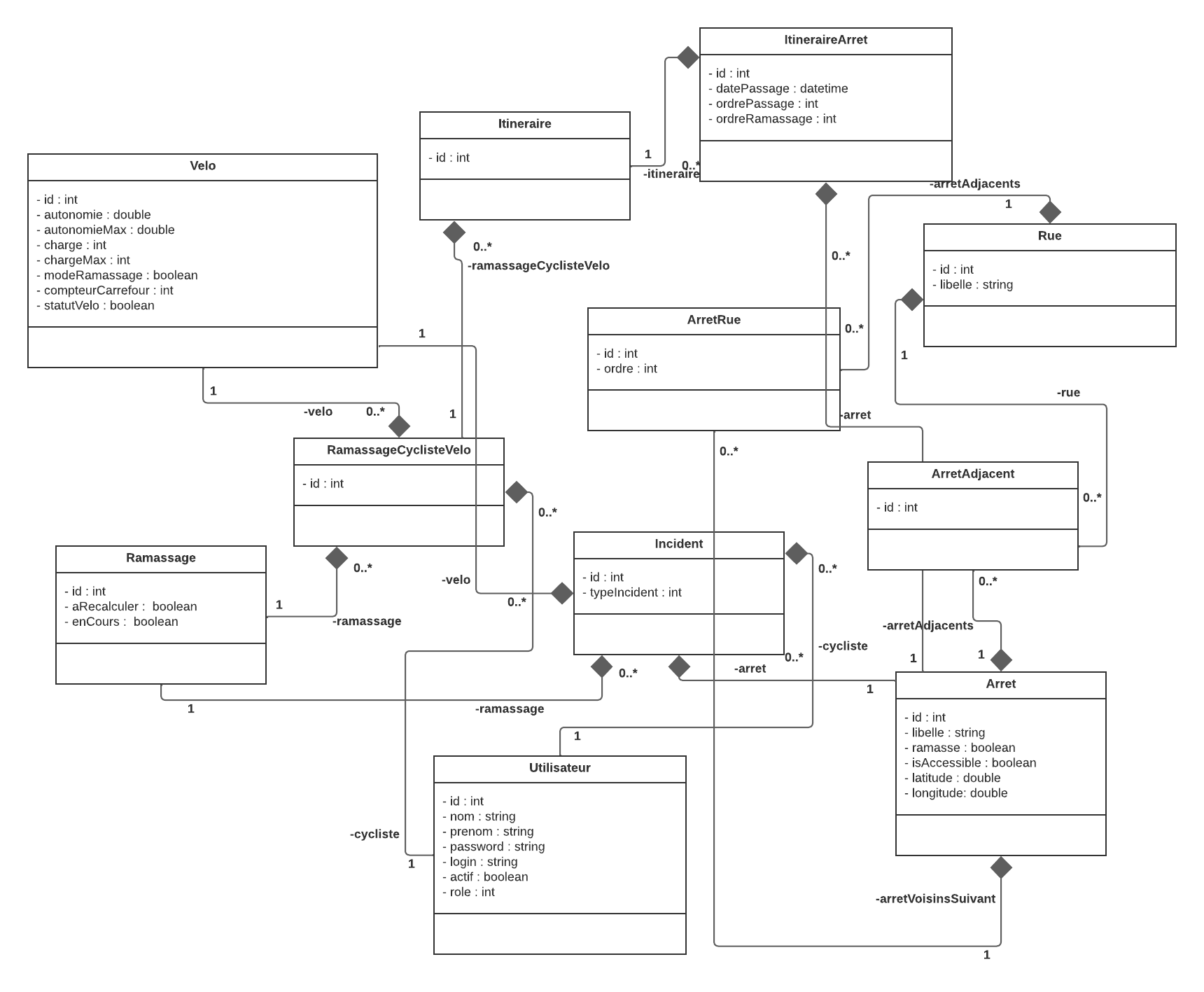
# – Diagrammes

Afin d'illustrer la conception de notre projet, nous avons réalisé plusieurs diagrammes illustrant les besoins du cahier des charges du système demandé.

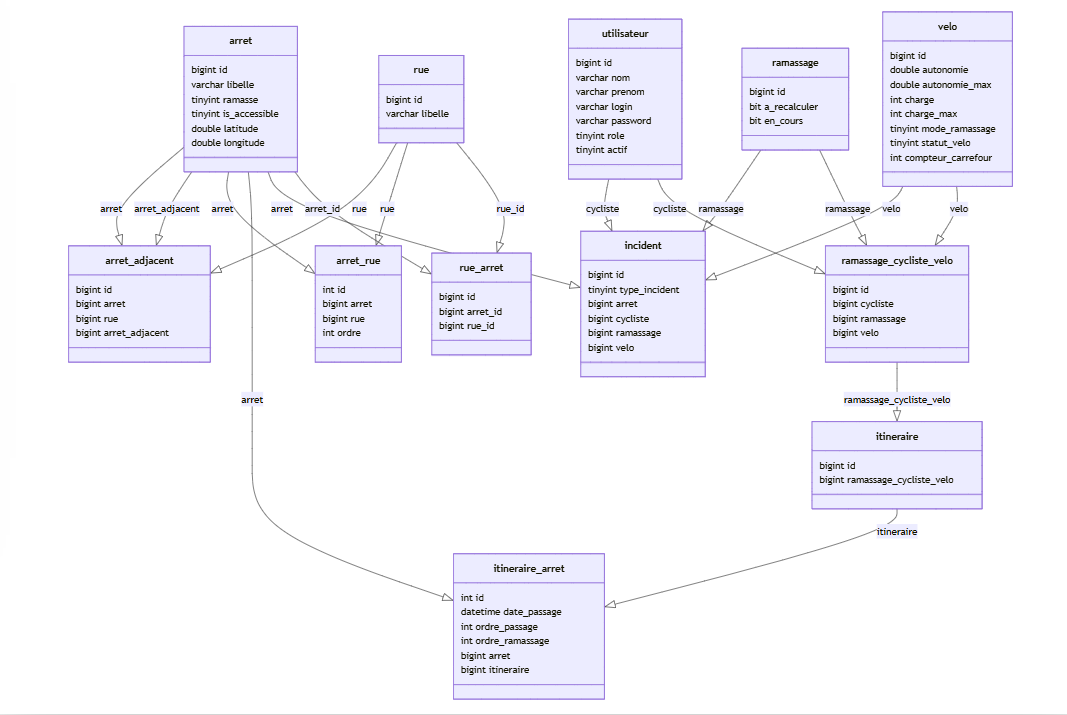
Le premier est un diagramme de cas d’utilisation montrant, au sein du système, les acteurs principaux de l’application ainsi que leurs tâches. Par exemple, pour un cycliste, il est possible de voir les trajets, la position et même les incidents lors du ramassage sur la carte.



Le deuxième diagramme met en évidence les relations des fonctionnalités du cahier des charges et du cas d’utilisation. Dès lors que l’on regarde ce diagramme on comprend les dépendances et relations dans chaque classe (outil majeur pour la construction de notre application).



Ce dernier prévoit des mécanismes pour gérer la cohérence des données, notamment via des contraintes d’intégrité référentielles et la normalisation afin d’éviter les redondances. Ce MCD est essentiel pour garantir que le système reste évolutif, maintenable et performant.



# – Difficultés rencontrées

**Modélisation de la carte et animations**

Lors du projet, une des principales difficultés a été de modéliser une carte cohérente représentant les rues et les arrêts. Bien que nous disposions de données brutes sous forme de listes, leur traduction en une représentation visuelle précise s’est avérée complexe. Il était essentiel de garantir une correspondance entre les données et la carte, en veillant à la bonne disposition des intersections, impasses et croisements multiples.

**Sélection généralisée des cyclistes pour un ramassage**

Au premier abord, la fonctionnalité pour lancer un ramassage est de sélectionner des cyclistes selon les vélos disponibles. Par exemple, si on dispose de 4 vélos disponibles et 8 utilisateurs, il sera seulement possible de lancer au maximum 4 cyclistes pour ce ramassage.

Cependant, la sélection des cyclistes nécessite d'être effectuée à la main un par un. De ce fait, nous avons procédé à l’amélioration de cette fonctionnalité en la généralisant. C'est-à-dire, un utilisateur peut cliquer sur un bouton “tout sélectionner” ou “tout décocher” et cela choisira de façon aléatoire les vélos disponibles.

La difficulté principale se centralise sur la liaison entre les vélos disponibles et les cyclistes du système.

**Calcul des itinéraires**

La mise en place du calcul des itinéraires à mis beaucoup de temps à être mise en place. Dans un premier temps il fallait établir un algorithme qui recherche la liste des chemins possibles jusqu’aux arrêts terminus. Il fallait développer une fonction récursive et aucun de nous n’était très familier avec ces dernières. Une fois cette fonction développée, il fallait définir les itinéraires à suivre pour ramasser l’ensemble des arrêts, fonctionnalité qui demande également de réaliser une fonction récursive. Une fois les itinéraires calculés, il fallait les attribuer aux cyclistes.

Même si cette fonctionnalité à mis du temps à être développée, elle reste cependant fonctionnelle mais possède quelques limites. Notamment le fait de calculer les différents chemins au début afin de choisir le plus court qui prend du temps et nos différents appareils ne sont pas assez puissants pour calculer l’ensemble des chemins rapidement. Cela met 10 minutes à calculer les chemins pour 7 rues et quelques secondes pour 3. De ce fait, cette méthode est la plus intéressante pour limiter le nombre d'aller-retour, mais est aussi trop gourmande. Nous aurions peut-être dû nous orienter sur un ramassage “brut-force” où l’on essaye tous les chemins jusqu’à ce que tous les arrêts soient ramassés.

**Recalcul des itinéraires**

Nous n’avons pas réussi à mettre en place le recalcul d’itinéraire à cause d’un problème lié au mapping Framework. En effet, même si à première vue il suffit de réutiliser la fonction pour calculer les itinéraires, c’est en réalité bien plus compliqué que cela. Il faut prendre en compte les arrêts ramassés, ainsi que les éventuels cyclistes qui seraient absents (accident corporel). Le problème étant que l’entité; liant les itinéraires aux cyclistes, ramassages et aux vélos; ne se supprime pas. Nous avons essayé différents procédés :

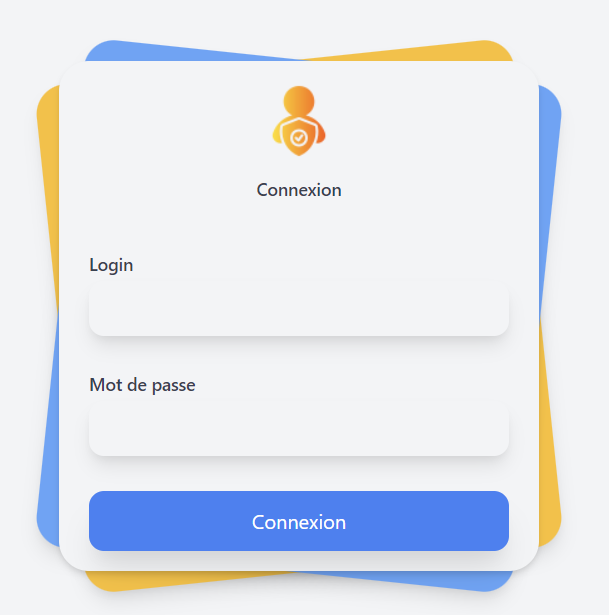
* Supprimer les itinéraires attribués aux cyclistes pour ensuite les recalculer en prenant en compte les arrêts déjà ramassés. Le problème étant que l’itinéraire ne se supprime pas. Nous soupçonnons une erreur de mapping entre les différentes entités.
* La deuxième méthode consistait à enregistrer les cyclistes et vélos utilisés, les supprimer de la base avec leurs itinéraires puis les enregistrer et calculer leurs itinéraires. Cependant, cette méthode n'a abouti à aucun résultat.

Et c’est après cette deuxième tentative que le PC de l’un des membres a été volé, avec le code utilisé pour ces deux dernières méthodes. Le temps restant jusqu’à la fin du projet ne permettant pas de développer une telle fonctionnalité, nous n’avons donc pas pu la terminer.

# Mode d’emploi utilisateur

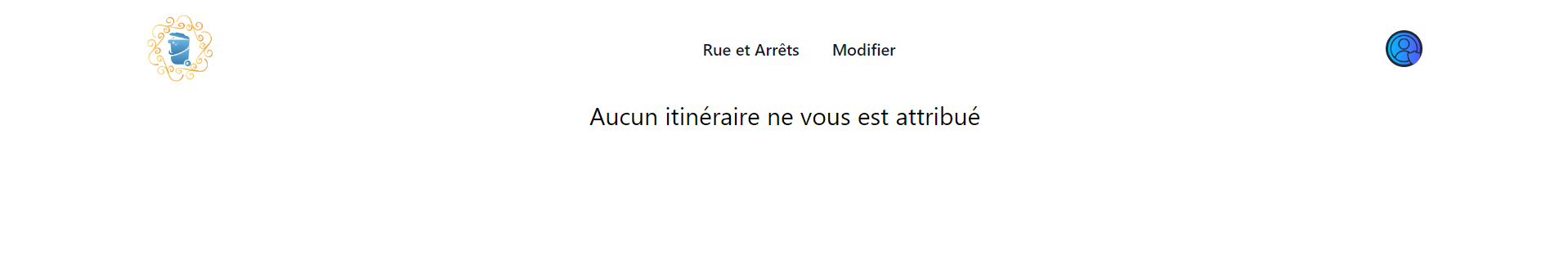
**Page commune :**

Lors de votre arrivée sur le site, vous arriverez sur cette page vous demandant votre identifiant et votre mot de passe.

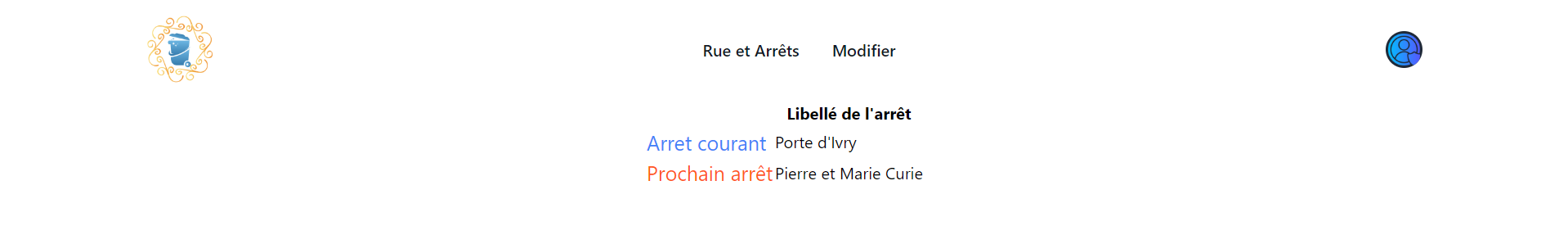


**Cycliste :**

Après s’être connecté, un cycliste se retrouvera sur cette page si aucun itinéraire ne lui est attribué :



En revanche, si un itinéraire lui est attribué, voici un aperçu de la pop-up :

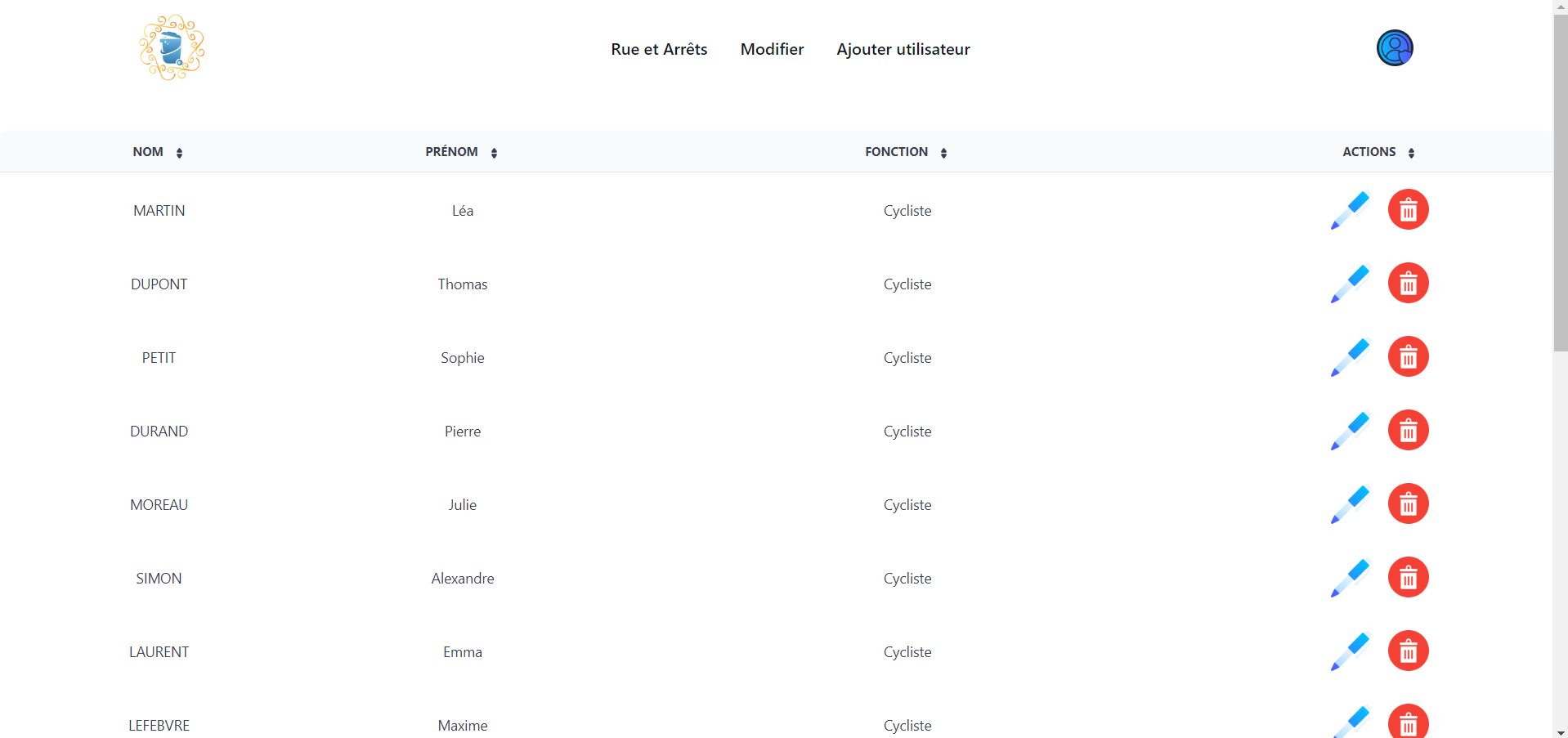


Cette page permet d’afficher les derniers arrêts qui doivent être ramassés par le cycliste.

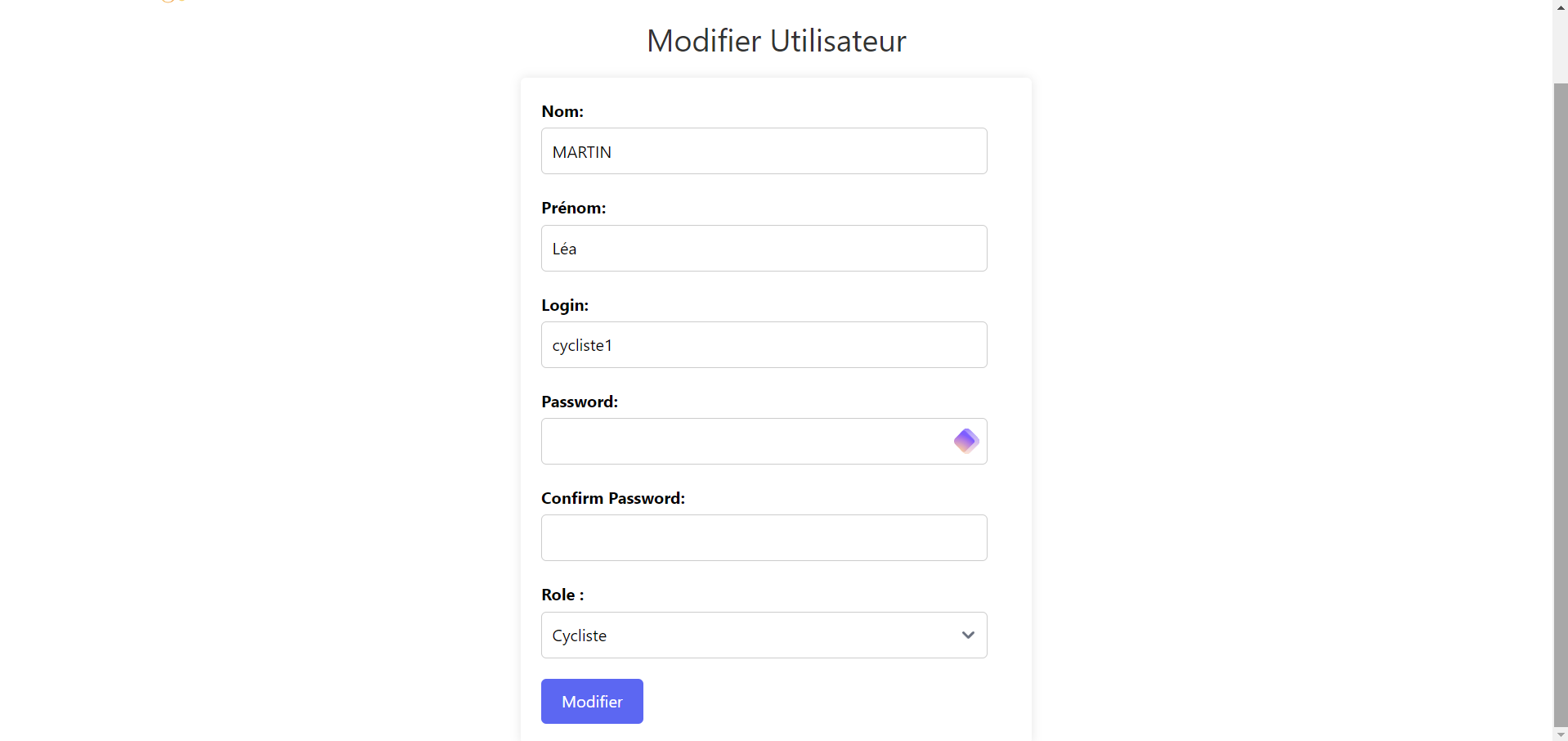
De plus, un cycliste à également la possibilité de voir les arrêts qu’il doit ramasser sur une carte accompagnée de son itinéraire.

**Responsable (RH) :**

Après s’être connecté, le responsable RH se retrouve sur la page suivante :

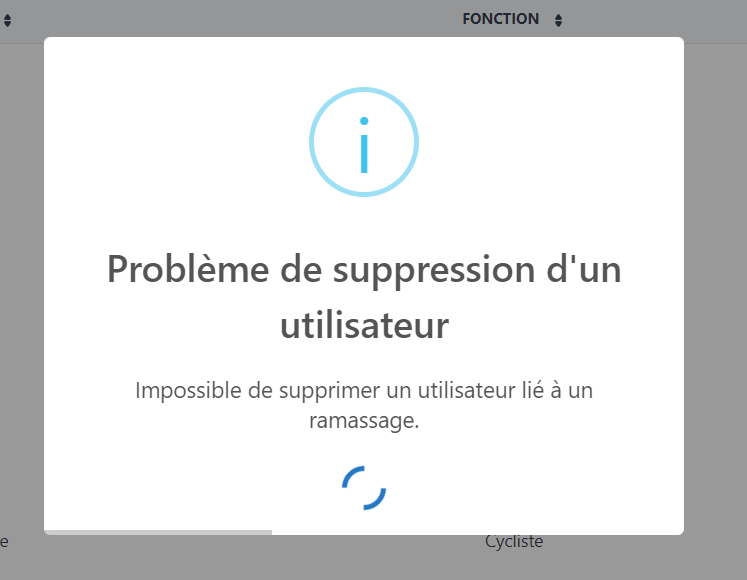


Cette page liste les utilisateurs de l’application et permet au responsable de les modifier ou de les supprimer. En cliquant sur l’icône crayon, il sera redirigé vers la page ci-dessous :



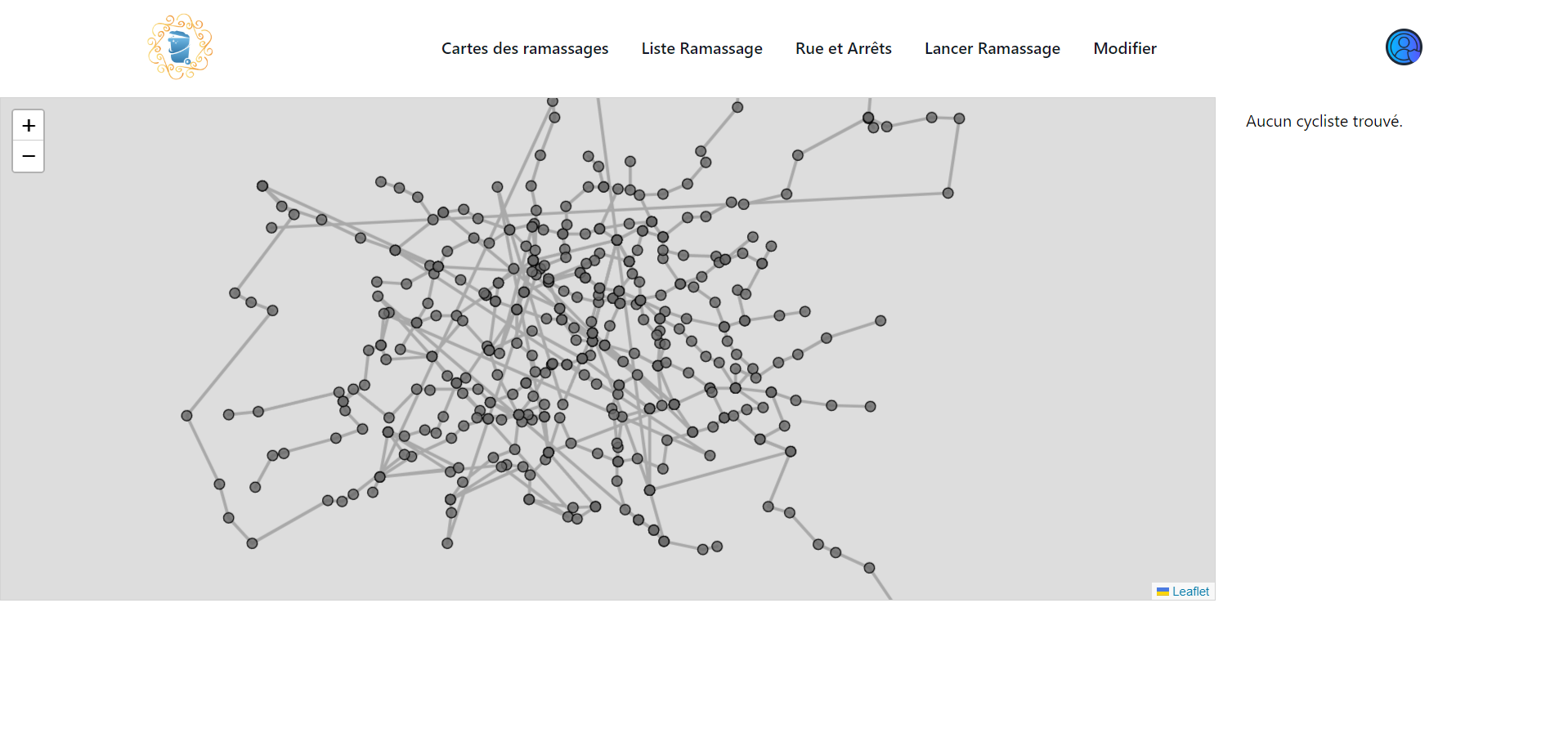
Cette page permet donc au gestionnaire de modifier les informations relatives à un utilisateur en changeant les champs du formulaire.

Contrairement à l’icône poubelle qui lui permet de supprimer un utilisateur. Dans le cas où ce dernier a déjà effectué des ramassages, il n’est pas possible de le supprimer et une pop-up s’affiche alors :

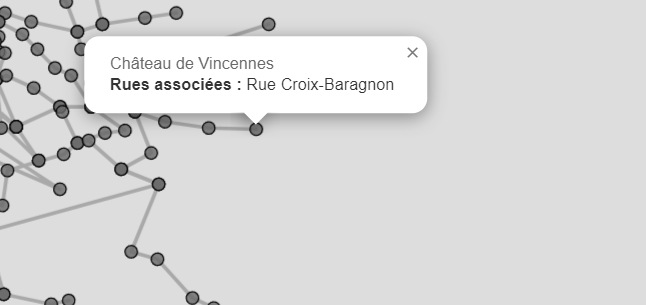


**Gestionnaire :**

Lorsqu’un gestionnaire se connecte, il peut consulter la pages représentant les itinéraires des cyclistes sur la carte présentée ci-dessous :



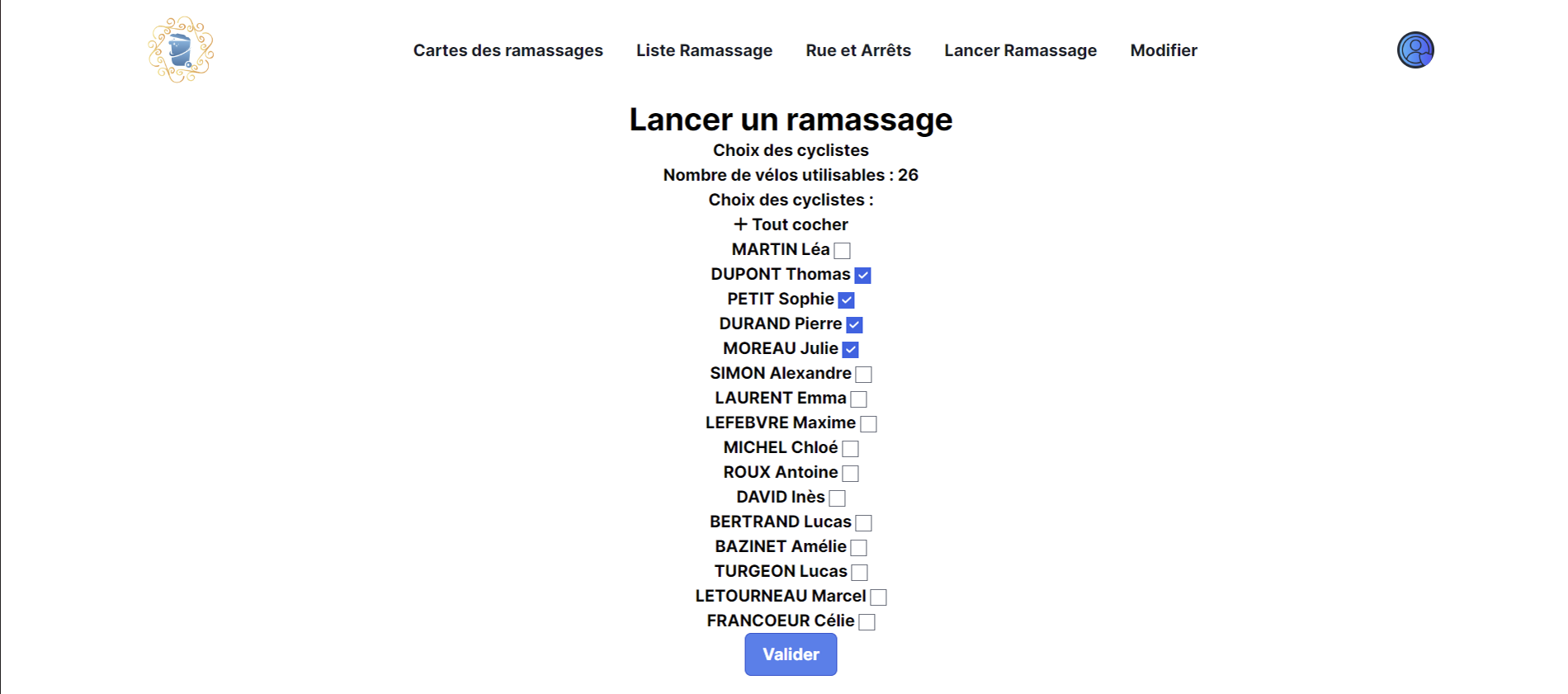
Cette page affiche la carte de la ville avec les différents arrêts ainsi que les rues qui les relient. En cliquant sur l’un des points, une pancarte apparaît indiquant le nom de l’arrêt ainsi que les rues avoisinantes :



Actuellement aucun ramassage n’est en cours donc aucun cycliste n’apparaît. Si un ramassage est en cours, la liste des cyclistes participant à ce ramassage apparaissent à droite de la carte comme ci-dessous :



Un gestionnaire a également la possibilité de lancer un ramassage, en cliquant sur l’onglet “Lancer Ramassage” en haut de la page, il est redirigée sur la page ci-dessous :



Cette dernière lui permet de sélectionner les cyclistes qui effectueront le ramassage. Une fois le calcul des itinéraires terminés, il sera conduit sur la page suivante :



Cette dernière liste les ramassages terminés ainsi que celui qui est en cours. Si le gestionnaire souhaite lancer un nouveau ramassage depuis la page “Lancer Ramassage”, un message s'affiche alors :



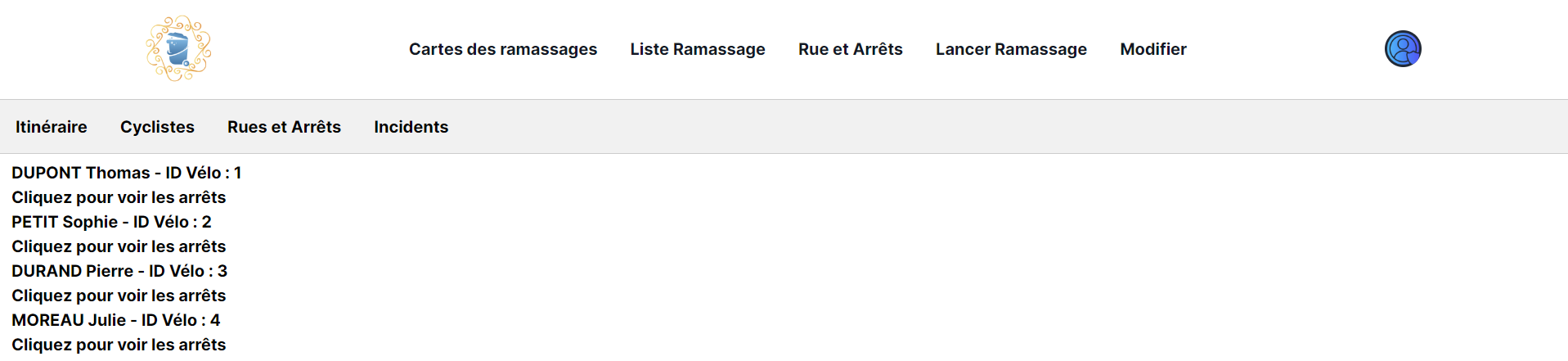
Il devra alors attendre que le ramassage en cours soit terminé avant d’en lancer un nouveau.

Revenons sur la vue listant les ramassages, en cliquant sur un ramassage, le gestionnaire à la possibilité de voir les informations liées à ce ramassage.

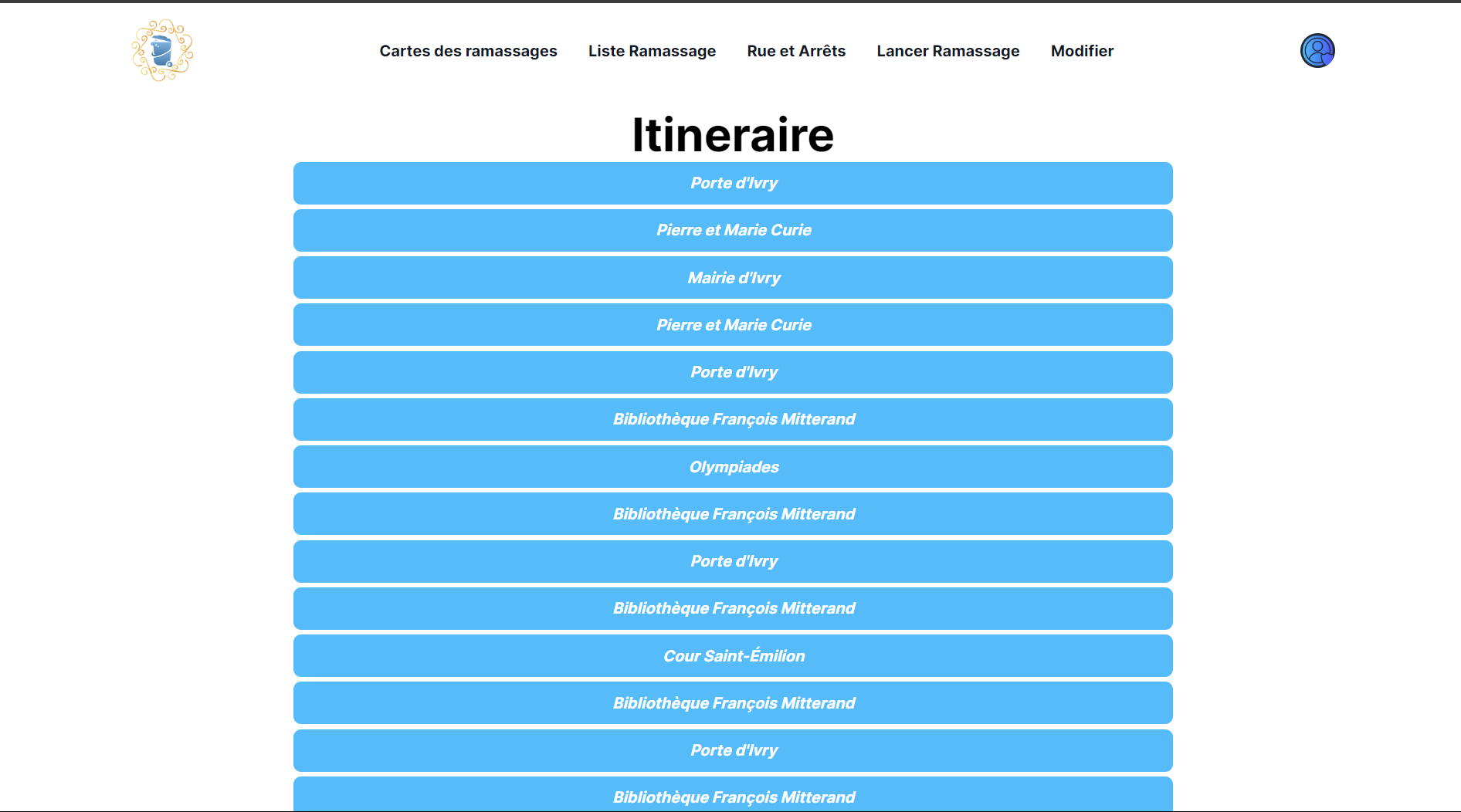
Cette page donne des informations diverses sur le ramassage.

1. **Itinéraires**

Cet onglet permet d'afficher la liste des itinéraires du ramassage :

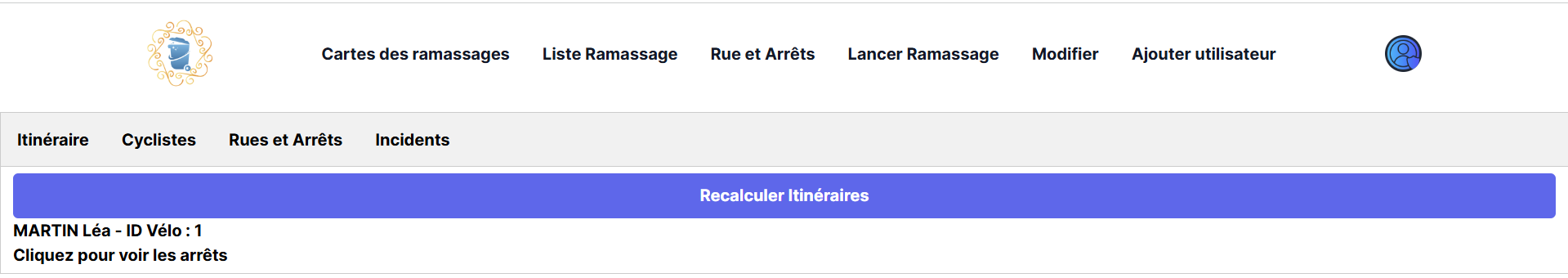


En cliquant sur l’un d’entre eux, il est possible de voir la liste des arrêts que le cycliste doit parcourir :



1. **Cyclistes**

La page des cyclistes permet d’afficher les cyclistes participants au ramassage ainsi que les vélos qui leurs sont attribués.



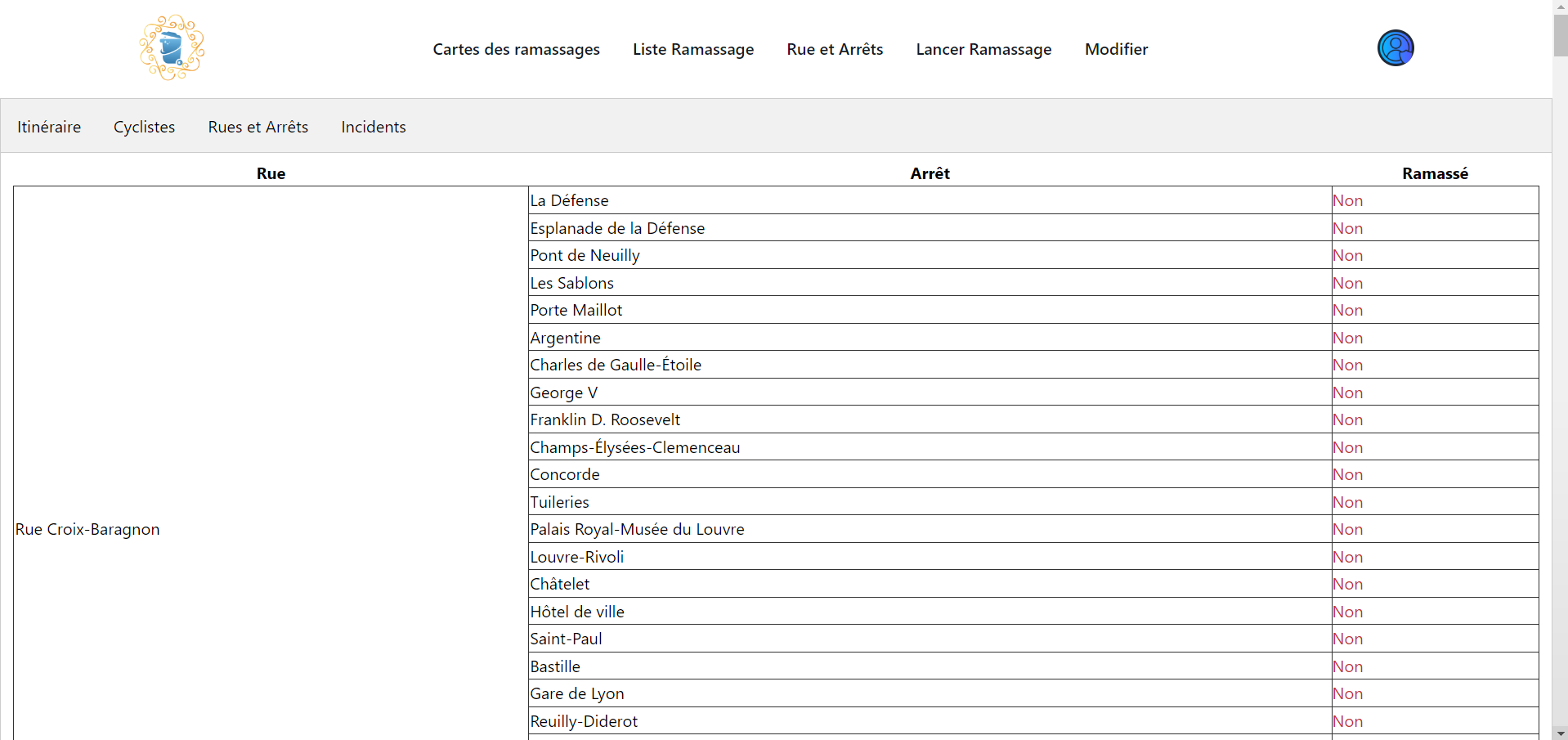
En cliquant sur le bouton “Ajouter”, un formulaire apparaît alors permettant d’ajouter un cycliste au trajet comme suit :



Après l’ajout du cycliste, il est conseillé de recalculer l’itinéraire afin d’y inclure le cycliste nouvellement ajouté.

1. **Rues et arrêts**

Cette page liste les rues et arrêts de la ville et indique si ces derniers sont ramassés ou non :



1. **Incidents**

La page des incidents affiche la liste des incidents survenu au cours du ramassage qui ont été signalés par le gestionnaire dont voici un aperçu sans aucun incident déclaré :

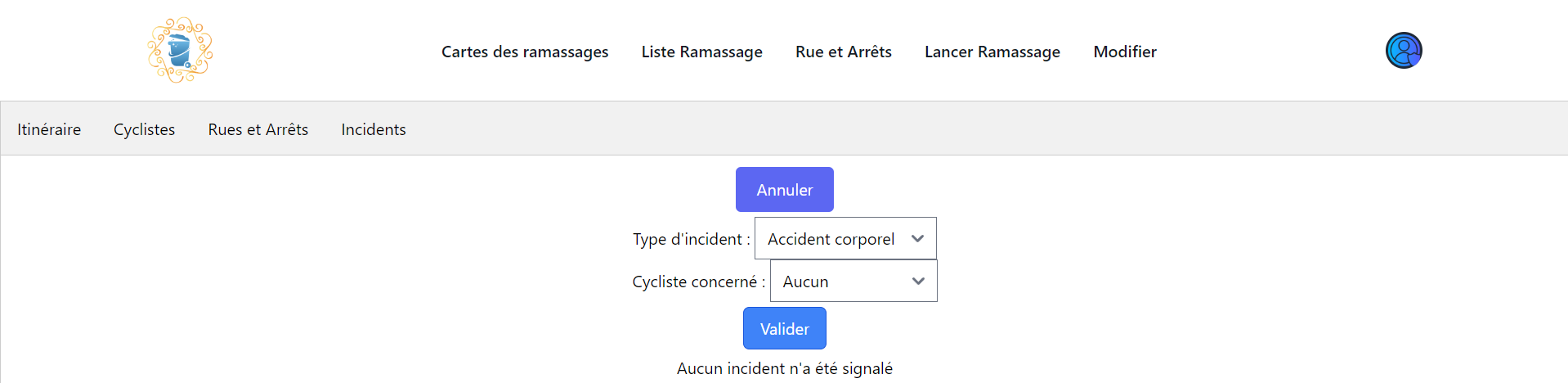


Voici un aperçu avec quelques incidents :

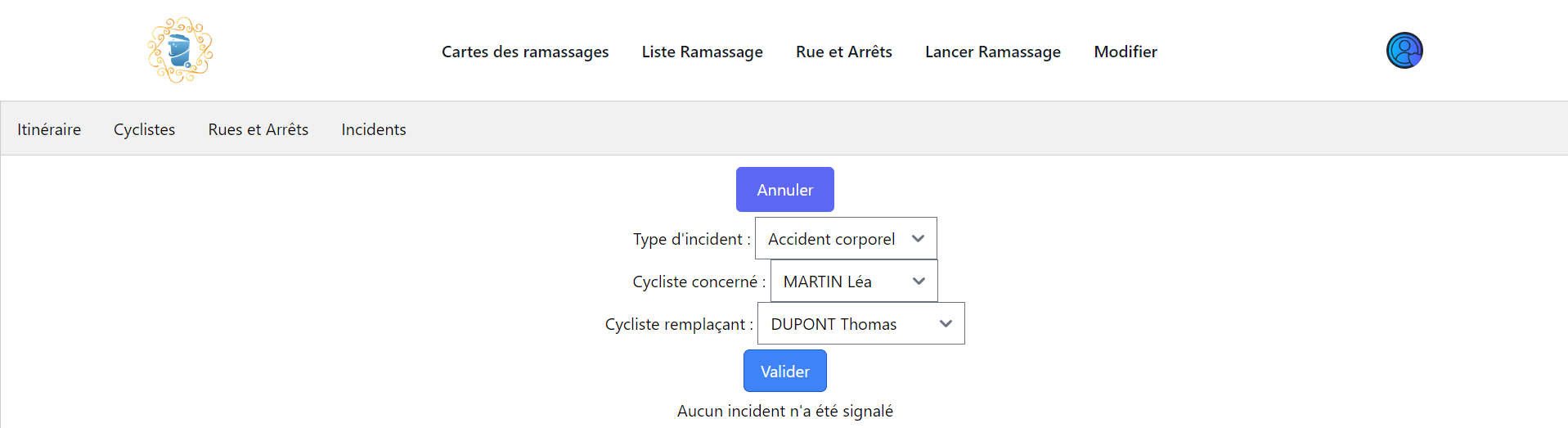


Cette page permet également de déclarer un incident survenu lors d’un ramassage. Il existe 3 types d’incidents :

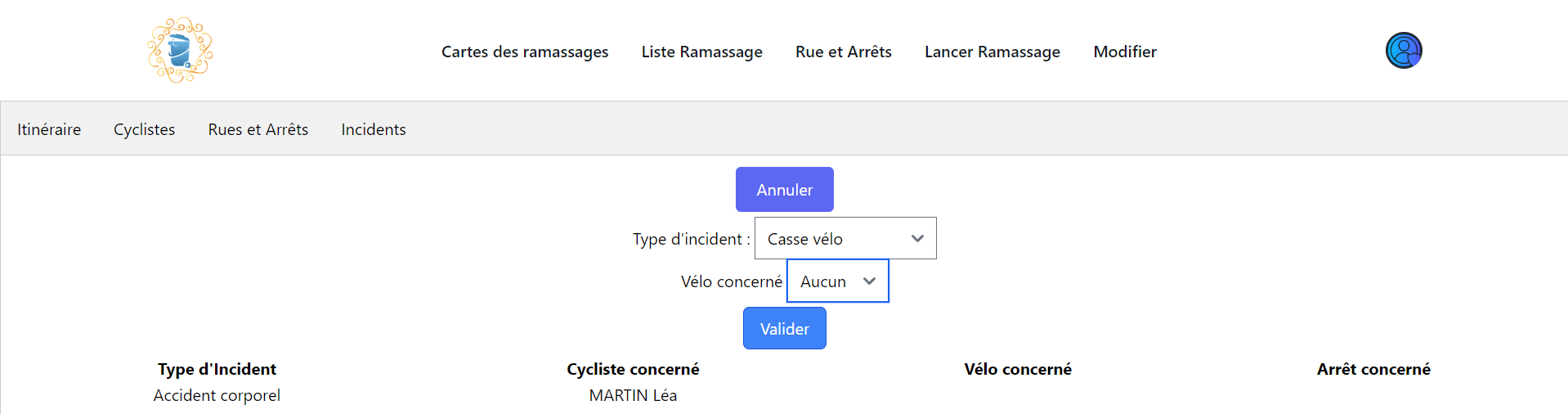
1. Accident corporel —> il faut sélectionner dans un premier temps le type d’incident. Une fois le champ sélectionné, une liste de cyclistes apparaîtra. Il s’agit de la liste des cyclistes participant au ramassage comme ci-dessous.



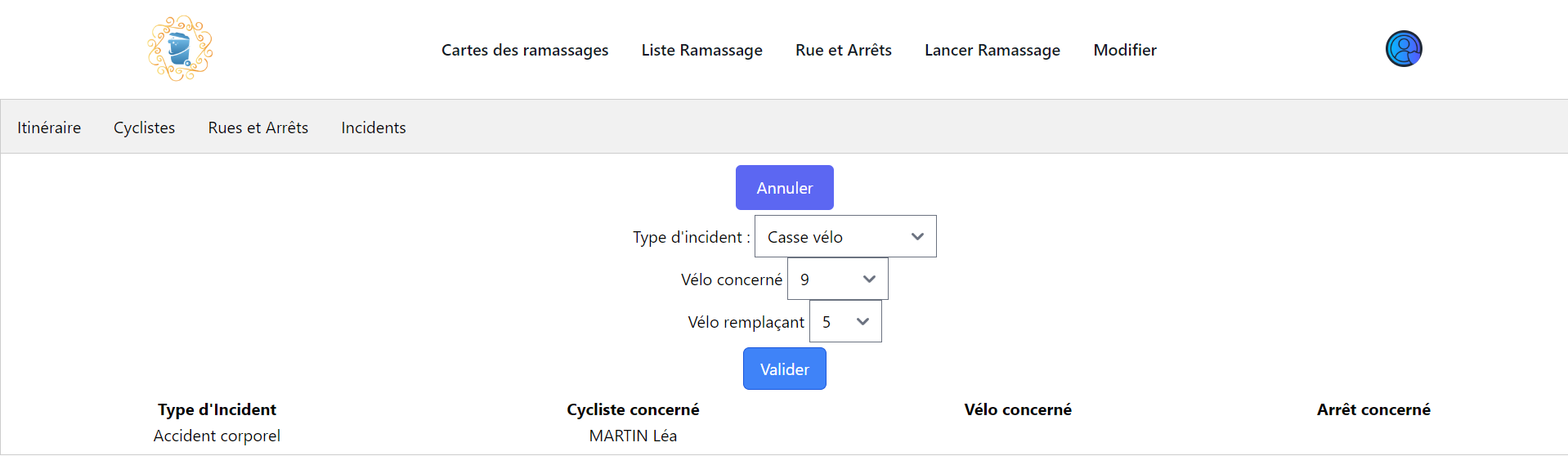
Après avoir sélectionné le cycliste victime d’un accident, il faut sélectionner le cycliste qui le remplacera dans la liste des cycliste qui apparaît après avoir sélectionné un premier cycliste comme ci-dessous :



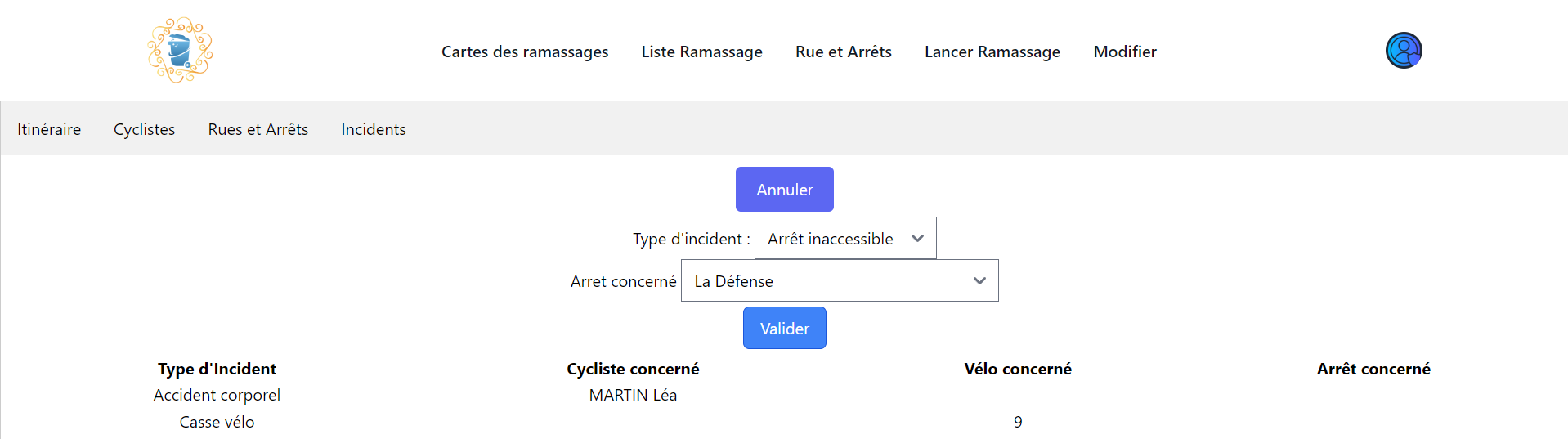
1. Casse vélo —> le fonctionnement est le même en cas de casse vélo, dans un premier temps on choisit le vélo concerné :



Puis on choisit le vélo remplaçant :



1. Arrêt inaccessible —> il suffit simplement de sélectionner l’arrêt en question :



# 

# 

# 

# Guide d'installation et de configuration

Dans un premiers temps téléchargez le fichier zip depuis le repository github.

Lien du répository : <https://github.com/Val-Nost/poubelle.git>

Extraire le contenu du projet dans un répertoire où vous voulez le ranger.

Déplacez vous dans le dossier du projet (il s’appelle poubelle normalement)

## Prérequis

java 21 : <https://download.oracle.com/java/21/latest/jdk-21_windows-x64_bin.msi> mariadb : <https://mariadb.org/download/?t=mariadb&p=mariadb&r=11.4.2&os=windows&cpu=x86_64&pkg=msi&mirror=icam>

Vérifiez que les ports 8080 et 3306 sont libres

Remarque : Le port 3306 est utilisé pour la gestion de la base de données.

Pour le lancement de l’application java spring, Le port 8080 est utilisé.

## Démarrer l’application

Ouvrez un terminal de commande.

1. Assurez-vous d'être dans le dossier racine.
2. Déplacez vous dans le dossier du projet (il s’appelle poubelle normalement)
3. Ouvrir un nouveau terminal
4. Lancer le serveur de développement local:

| path/to/openjdk-21.0.2\bin\java.exe -jar .\target\poubelle-0.0.1-SNAPSHOT.jar |
| --- |

### 

## Les Utilisateurs

Durant le développement du site nous avons créé 4 utilisateurs, (un responsable RH, un gestionnaire et un cycliste) ainsi qu’un administrateur du système.Les identifiants de connexion sont comme suit :

| Pseudo de l’utilisateur | MDP de l’utilisateur | Rôle de l’utilisateur |
| --- | --- | --- |
| cycliste1 | cycliste | Cycliste |
| rh | rh | Responsable RH |
| gestionnaire | gestionnaire | Gestionnaire |
| admin | admin | Administrateur |

# 

# Algorithme et Optimisation

En ce qui concerne l’algorithme utilisé afin d’effectuer le ramassage des trajets, nous aurions très bien pu utiliser l’algorithme de Dijkstra. Cependant, selon les besoins du client, les vélos ont seulement une capacité de 200 kg et il faut également prendre en compte l’autonomie du vélo ou encore l’inaccessibilité d’un arrêt. Tous ces paramètres rendant l’algorithme trop complexe j’ai décidé de commencer par réaliser mon propre algorithme de parcours et d’intégrer un algorithme de Dijkstra si le temps restant me le permettait. Le développement du premier algorithme et les délais du projet ne me l’ayant pas permis, je n’ai pas eu le temps d’implémenter l’algorithme de Dijkstra à la recherche d’Itinéraire. Je vais donc vous détailler les étapes de l’algorithme qui est utilisé actuellement par notre application.

Dans un premier temps, l’utilisateur de l’application (un gestionnaire ou un administrateur) doit sélectionner les cyclistes qu’il souhaite faire participer aux ramassages. Le nombre de cyclistes dépend naturellement du nombre de vélos disponibles et utilisables. Les vélos sont ensuite attribués aléatoirement aux cyclistes.

Dans un second temps, le programme va dresser la liste des chemins possibles menant aux différentes feuilles du graphes et va supprimer les boucles. Il va ensuite chercher le chemin le plus court de cette liste et va envoyer un cycliste le parcourir.

Le programme va vérifier si le cycliste a suffisamment d’autonomie afin d’effectuer l’aller-retour de ce chemin. S’il en a la capacité, le programme va alors ajouter à l’itinéraire du cycliste la liste des arrêts dans l’ordre afin d’effectuer l’aller-retour du chemin (Exemple : Porte d’Ivry -> Pierre et Marie Curie -> Mairie d’Ivry -> Pierre et Marie Curie -> Porte d’Ivry).

Une fois l’arrêt ramassé, il est retiré de la liste des arrêts du chemin possible. Le cycliste suivant est ensuite envoyé sur le même chemin afin de ramasser également les derniers arrêts du chemin. Une fois que toute la liste des arrêts du chemin est vide, les cyclistes passent au chemin suivant et ainsi de suite jusqu’à ce que la liste des arrêts de tous les chemins possibles soit vide.

En conclusion, cet algorithme sert à attribuer un itinéraire à chaque cycliste qui contient une liste d’aller-retour des chemins possibles de plus en plus réduit.

**Animation des cyclistes**

L'affichage des itinéraires se fait par le biais d’un script JavaScript qui gère l'affichage d'une carte interactive utilisant la bibliothèque Leaflet, conçue pour visualiser des itinéraires cyclistes et les arrêts associés. Il configure une carte interactive en créant plusieurs "pane" personnalisés, permettant une organisation claire des différentes couches : polylignes globales, marqueurs globaux, itinéraires individuels, et un marqueur animé représentant un cycliste en mouvement.

Les données nécessaires, telles que les arrêts et les rues, sont récupérées dynamiquement à partir d'un serveur via des appels asynchrones, puis utilisées pour afficher les arrêts regroupés par rue, avec une représentation globale sous forme de polylignes et de marqueurs interactifs munis de popups descriptives.

L'affichage par défaut présente une carte globale où les rues sont visualisées comme des polylignes, et chaque arrêt est représenté par un marqueur indiquant son nom et les rues associées. Lorsqu'un utilisateur sélectionne un cycliste, le script charge et affiche dynamiquement l'itinéraire individuel correspondant. Cet itinéraire est représenté par une polyligne verte et des marqueurs numérotés ou colorés en fonction de leur rôle (par exemple, rouge pour le point de départ).

Une animation visuelle met en mouvement un marqueur simulant le déplacement du cycliste entre les arrêts, en interpolant ses coordonnées pour offrir une transition fluide.

Le script inclut des fonctionnalités pour effacer l'itinéraire précédent avant d'en charger un nouveau, tout en conservant les éléments de la carte globale. Il ajuste automatiquement la vue pour englober tous les points pertinents et garantit une mise à jour fluide des données affichées en réponse aux interactions de l'utilisateur.

Ce mécanisme d'affichage et d'animation assure une expérience visuelle intuitive et organisée, adaptée à la navigation et à l'exploration des données cartographiques cyclistes.

# 

# Revues de sprints

Vous retrouverez dans ce document l’ensemble des tâches du projet, elles sont triées par sprint (de 1 à 9). Pour chaque sprint il y a un tableau qui comprend l’état de la tâche, son intitulé, le sprint ou elle a été originellement prévue d’être réalisée et la/les personne/s qui s’en sont chargés.

Voici les 3 états que peut avoir une tâche :

* Effectuée

(La tâche est effectuée et remplit entièrement son objectif)

* Effectuée partiellement

(La tâche est effectuée, mais une partie est dysfonctionnelle ou absente. Dans

beaucoup de cas nous ne sommes pas rendus compte tout de suite qu’une tâche été incomplète. C’est en relisant le cahier des charges que comprenons un nouvel

élément de celui-ci, ou que nous constatons un oubli de notre part). Pour chaque tâche effectuée partiellement un \* indique qu’une précision se trouve en dessous pour plus de détails.

* Reportée

(La tâche n’a pas pu être réalisée et est reportée à un moment déterminé ou non. Il peut y avoir plusieurs possibilités pour qu’une tâche soit reportée :

* + Manque de temps : les autres tâches du sprint ont pris plus de temps que prévu
  + Difficultés techniques : la tâche est complexe et nécessite plus de temps
  + Priorisation : Certaines tâches peuvent être considérés comme prioritaires, et donc prendre la place d’autres tâches qui sont alors reportées)

**Note :** La correction des bugs n’est pas compris dans la liste tâches, cela représente pourtant une part majeure du temps passé sur le projet.

Répartition par membre de l’équipe :

* Une tâche réalisée par (Adrien Carles) est notée : A
* Une tâche réalisée par (Jauffrey Bicheyre) est notée : J
* Une tâche réalisée par (Valentin Simon) est notée : V
* Une tâche réalisée par les deux membres de l’équipe est notée : A+J
* Une tâche réalisée par les trois membres de l’équipe est notée : A+J+

**Sprint 1 (14 février –> 7 mars)**

|  | Création du repository GIT | S1 | V+A |
| --- | --- | --- | --- |
|  | Réalisation du diagramme de classe | S1 | V |
|  | Réalisation du MLD/MCD | S1 | A |
|  | Réalisation du diagramme de cas d'utilisation | S1 | V |
|  | Ecriture des User Story | S1 | V+A |
|  | Réflexion sur l’algorithme des trajets | S1 | V+A |
|  | Ecriture du document de conception | S1 | A |

\*Le macro planning est incomplet car nous n’avons pas pris en compte l’ensemble des tâches du cahier des charges. \*De même pour la base de données.

**Sprint 2 (7 mars -> 4 avril)**

|  | Authentification cycliste | S2 | V |
| --- | --- | --- | --- |
|  | Authentification gestionnaire | S2 | V |
|  | Authentification RH | S2 | V |
|  | Authentification administrateur | S2 | V |
|  | Création des utilisateurs | S2 | A |
|  | Modification des utilisateurs | S2 | A |
|  | Attribution des rôles | S2 | A |
|  | Suppression des utilisateurs | S2 | A |

\*La base de données est toujours incomplète.

**Sprint 3 (4 avril -> 2 mai)**

|  | Liste rues traitées | S3 | V |
| --- | --- | --- | --- |
|  | Définition vélos disponibles | S3 | V |
|  | Modification du CSS | S3 | A |
|  | Réflexion sur l’itinéraire | S3 | A+V |

**Sprint 4 (2 mai -> 5 juin)**

|  | Amélioration du CSS | S4 | J |
| --- | --- | --- | --- |
|  | Confirmation popup pour la suppression d’un utilisateur | S4 | J |
|  | Réalisation du graphe | S4 | A |
|  | Calcul des itinéraires des cyclistes | S4 | V |
|  | Ecriture du readme pour l’utilisation du site | S4 | J |

**Sprint 5 (5 juin -> 16 septembre)**

|  | Recalcul des itinéraires | S5 | V |
| --- | --- | --- | --- |
|  | Affichage des cyclistes sur leurs itinéraires | S5 | A |
|  | Amélioration du CSS | S5 | J |

**Sprint 6 (début août -> 16 septembre)**

|  | Finalisation de la documentation | S5 | AVJ |
| --- | --- | --- | --- |
|  | Affichages du plan complets pour les gestionnaires | S5 |  |

# 

# Revues de réunions

**1ère revue de projet (7 mars 2024) :**

**Membres ayant participés :** [Valentin SIMON](mailto:valentin.simon@limayrac.fr), [Adrien CARLES](mailto:adrien.carles@limayrac.fr), Stéphane Cezera

**Objectifs de la réunion :**

* Confirmer les technologies utilisées pour le développement de l’application
  + Framework Spring (Java) pour le backend
  + HTML, CSS, JS pour le Frontend
  + MariaDB pour le SGBD
* Confirmer la méthodes de gestion de projet qui sera utilisées
  + Méthodologies Agile Scrum
  + Tableau Kanban pour effectuer le suivie
  + GitHub pour le gestionnaire de code source

**Résultats de la réunion :**

* Définition des tâches à effectuer
  + Effectuer toutes les spécifications liées à l’application
  + Commencer à travailler sur la conception
  + Créer un dépôt Git
  + Créer un tableau Kanban contenant les User Story relatives à l’application
* Choix de la durée des itérations de Sprint : 1 mois
* Choix de la date de prochaine réunion : 04/04/2024

# 

**2ème revue de projet (4 avril 2024) :**

**Membres ayant participés :** [Valentin SIMON](mailto:valentin.simon@limayrac.fr), [Adrien CARLES](mailto:adrien.carles@limayrac.fr), Stéphane Cezeras

**Objectifs de la réunion :**

* Présentation de la conception du projet
  + Présentation du document de conception
  + Présentation du diagramme de classe
  + Présentation du MCD/MLD
* Réponse aux questions
  + La base est elle l’usine de recyclage ?
* Présentations des User Story

**Résultats de la réunion :**

* Hypothèses
  + L’usine de recyclage est la base de départ et d’arrivée des vélos.
* Choix des Users Story du prochains sprints
* Préparations de la réunions suivante

**Autres remarques :**

* Justifier les associations du MCD :
  + 0.1 entre Vélo et Itinéraire
  + 0.1 entre Utilisateur et Itinéraire

**3ème revue de projet (2 mai 2024) :**

**Membres ayant participés :** [Valentin SIMON](mailto:valentin.simon@limayrac.fr), [Adrien CARLES](mailto:adrien.carles@limayrac.fr), Stéphane Cezeras

**Objectifs de la réunion :**

* Présentation de la partie gestion des utilisateurs
  + Création
  + Modification
  + Suppression
  + Login

**Résultats de la réunion :**

* Validation par le tuteur
* Planification du prochain sprint

**4ème revue de projet (5 juin 2024) :**

**Membres ayant participés :** [Valentin SIMON](mailto:valentin.simon@limayrac.fr), [Adrien CARLES](mailto:adrien.carles@limayrac.fr), [Jauffrey BICHEYRE](mailto:jauffrey.bicheyre@limayrac.fr), Stéphane Cezera

**Objectifs de la réunion :**

* Présentation de la vue sur les arrêts
* Présentations des changements apportés au css
* Présentations du début de travail effectué sur les ramassages

**Résultats de la réunion :**

* Validation par le tuteur
* Intégration de Jauffrey au groupe
* Il faut tester tout le travail déjà effectué afin de ne pas avoir de mauvaises surprise avant de développer d’autres fonctionnalitées

**5ème revue de projet (13 août 2024) :**

**Membres ayant participés :** [Valentin SIMON](mailto:valentin.simon@limayrac.fr), [Adrien CARLES](mailto:adrien.carles@limayrac.fr), [Jauffrey BICHEYRE](mailto:jauffrey.bicheyre@limayrac.fr), Stéphane Cezera

**Objectifs de la réunion :**

* Présentation du graphe modélisant toutes les rues de la ville
* Présentation des rues attitrées pour chaque cycliste
* Présentations des changements apportés au css (amélioration)
* Présentations de le lancement d’un ramassage ainsi que les détails du ramassage (attention un seul ramassage à la fois est possible)
* Annoncer les questions pour bien finaliser certains choix ou incertitudes du projet

**Résultats de la réunion :**

* Validation par le tuteur
* Corriger un problème de bouton CSS pour la valider des formulaires (c’est pas intuitif que c’est un bouton)
* Il est préférable de mettre en place que des menus déroulants pour l’ajout d’incident
* Bien noter le choix de notre algorithme et pourquoi ?
* Bien justifier l’hypothèse de la charge d’un cycliste s’il revient à la base (d’après le tuteur on recharge la batterie une fois à la base)
* Bien justifier l’hypothèse sur les incidents (si le cycliste revient à la base, passe sur le prochain ramassage ou recalcule le trajet)
* C’est fortement conseillé de réaliser un diagramme de GANTT afin de savoir notre gestion du projet dans le temps
* Il faudra prendre du temps pour mettre le site en ligne avec Azure avant l’oral (le 20 septembre)