PVCC
Projet Pluridisciplinaire d'Informatique Intégrative 1A S2 2022/2023

ChargeMate

24 Mai 2023

BOUCHADEL Maxence
GONCALVES Florian
JEANJACQUOT Thomas
VATRY Etienne

Table des matières

1 Introduction

Le sujet du projet pluridisciplinaire d'informatique intégrative du second semestre portait sur la transition vers une mobilité plus respectueuse de l'environnement. En effet, à partir de 2035, les voitures à moteur thermique ne pourront plus être mises en vente en Europe. Dans ce contexte, les véhicules électriques apparaissent comme une alternative prometteuse et le but de notre projet était de faciliter l'utilisation de celle-ci.

Notre cahier des charges était divisé en deux parties. En premier lieu, nous devions créer un outil permettant de parcourir un trajet en voiture électrique en France d'un point A à un point B, puis d'afficher les stations de recharge auxquelles il fallait s'arrêter afin d'atteindre notre point d'arrivée sans tomber en panne de batterie. Nous avons ajouté d'autres fonctionnalités à cette partie du programme afin que l'utilisation soit optimale (batterie au départ, seuil de batterie qu'on ne veut pas dépasser pour assurer la pérennité de celle-ci...).

Dans un second temps, nous avons dû implémenter un module de simulation afin d'évaluer la charge du réseau de bornes de recharge en fonction d'un ensemble d'usagers fictifs. Ce simulateur gère un ensemble d'usagers en prenant en compte leur nombre, leur parcours et les caractéristiques de leurs véhicules. À chaque étape du temps, il calcule le taux de charge des bornes de recharge sur le territoire, identifie celles sur lesquelles des files d'attente se forment (et propose, si possible, des chemins alternatifs aux véhicules pour éviter ces files d'attente). Afin de faciliter l'implémentation nous avons supposé que tous les véhicules roulent à la même vitesse sur le territoire entre deux bornes pour simplifier le processus. Ce simulateur est destiné aux investisseurs et aux autorités pour les aider à évaluer la charge du réseau de bornes de recharge et à prendre des décisions en conséquence.

2 Gestion de projet

2.1 Organisation générale

Afin de réaliser ce projet, nous avons essayé d'appliquer au mieux les différents éléments vu en cours de gestion de projet. Pour commencer notre groupe était composé de quatre membres :

— Maxence BOUCHADEL
— Florian GONCALVES
— Thomas JEANJACQUOT
— Etienne VATRY

Nous avons au lancement du projet désigné Etienne VATRY en tant que chef du projet. Il avait pour rôle de motiver les membres lors de la réalisation du projet en leur donnant les différentes tâches à réaliser. Il avait de plus la mission de fixer et préparer les réunions en avance pour qu'elles soient les plus productives possible et d'assurer leur bon déroulement en les dirigeant. Notre secrétaire durant ses réunions a été Thomas JEANJACQUOT qui est celui qui maitrisait le mieux le langage LateX. Dès le début du projet, nous avons établi un diagramme de Gantt que nous avons suivi le plus méticuleusement possible. Les taches étaient plutôt claires et détaillés et Etienne était là pour coordonner le tout lors de nos réunions hebdomadaires.

Afin de répartir les tâches à chacun, Etienne se basait sur le diagramme de Gantt réalisé au début du projet, qui permettait de connaître les grandes étapes à réaliser. En plus du digramme, une to do list plus détaillée a été créée sur le drive partagé afin d'y détailler beaucoup plus précisément les tâches à réaliser. Cette liste avait pour avantages d'être beaucoup plus flexible que le diagramme de Gantt, de pouvoir être modifiable facilement par tout le monde mais également de pouvoir suivre plus aisément les avancées de chacun. Cette liste permettait aussi d'appliquer une démarche de PDCA. En effet, les grosses tâches étant définies en réunion et sur le diagramme de Gantt, chacun pouvait réaliser ce qui était prévu puis Etienne pouvait venir "Checker" si ce qui avait été réalisé était complet ou pouvait, grâce à la liste, ajouter des améliorations possibles.

Ainsi grâce à cette organisation chacun savait exactement ce qu'il avait à faire et ce qu'il devait modifier.

2.2 Les moyens de communication

Afin de pouvoir assurer une bonne communication dans le groupe, différents moyens ont été mis en place. Nous avons commencé par la création d'un Google Drive permettant la création, l'échange et modifications simples des différents documents nécessaires au projet autre que le code. Ceci a donc permis la création de la to do list que chacun pouvait consulter et modifier

simplement, mais également la consultation des comptes rendus ainsi que des préparations des réunions contenant l'ordre du jour, auxquels chacun pouvait rajouter des éléments qu'il souhaitait évoquer en réunion.

Nous avons également créé un serveur Discord, permettant de communiquer à la fois par écrit et par vocal. Cette dernière option nous a beaucoup servi pour réaliser des réunions en distanciel mais également pour travailler ensemble lorsque nous le désirions .

Pour finir, nous avons également créé un groupe sur Snapchat, étant le moyen de communication le plus utilisé par les différents membres, bien qu'il ne soit pas idéal pour le travail, il permettait de pouvoir contacter rapidement tout le groupe et a permis de renforcer les liens et la bonne entente au sein de l'équipe.

2.3 Organisation des réunions

Comme dit précédemment, les réunions étaient fixées par le chef du groupe qui prenait soin de trouver un créneau disponible à chacun. Sur ce premier point la tâche n'était pas toujours très aisée. En effet les membres appartiennent à 3 groupes de TD différents, apprennent 3 langues secondaires différentes et pratiquent également des sports différents, ceci avait donc pour conséquence de générer des emplois du temps également très différents.

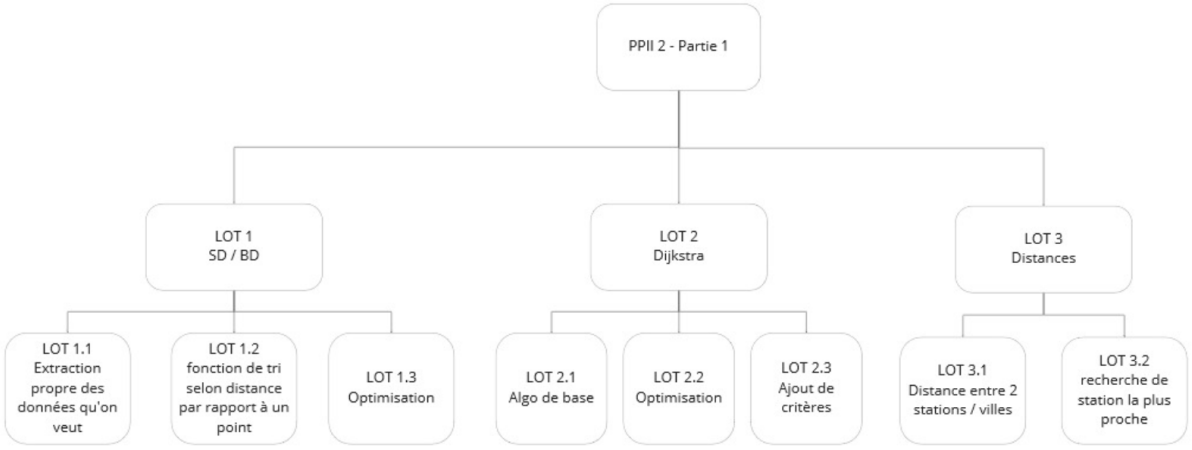
Une fois un créneau trouvé, Etienne préparait l'ordre du jour de la réunion afin que chacun soit au courant du déroulement de la réunion et chacun était libre de rajouter quelque chose. Une réunion se déroulait généralement sous le même schéma. Nous commencions par voir ce que chacun avait réalisé depuis la dernière réunion et vérifions si tout ce qui avait été noté dans la ToDo Liste avait bien été réalisé. Cette phase permettait également de revenir sur les difficultés rencontrées ou qui persistaient encore afin qu'on puisse y apporter une solution collectivement. Par la suite, nous réfléchissions aux différentes tâches à attribuer à chacun et complétions la ToDo liste pour la prochaine réunion.

Les réunions ont eu lieu chaque semaine jusqu'à la fin du projet. Cependant, nous nous retrouvions aussi lors des pauses pour discuter des difficultés rencontrées et aussi s'entraider. Cette organisation nous convenait plutôt bien, chacun pouvait avancer sur ses tâches et demander de l'aide à n'importe quel moment.

2.4 WBS

Pour réaliser notre diagramme de Gantt nous avons réalisé un Work Breakdown Structure qui nous a permis de répartir en différents lots de travail notre projet. Ceci permettant de pouvoir répartir le travail équitablement et de pouvoir estimer les différentes charges de travail.

Voici le WBS réalisé :



NOMS		ETIENNE	MAXENCE	THOMAS	FLORIAN
LOT 1	LOT 1.1	A/C	R		
	LOT 1.2	A/C	R		
	LOT 1.3	A/C	R		
LOT 2	LOT 2.1	A/C	I	R	I
	LOT 2.2	A/C	I	R	I
	LOT 2.3	A/C		R	
LOT3	LOT 3.1	A			R
	LOT 3.2	A			R

Légende :

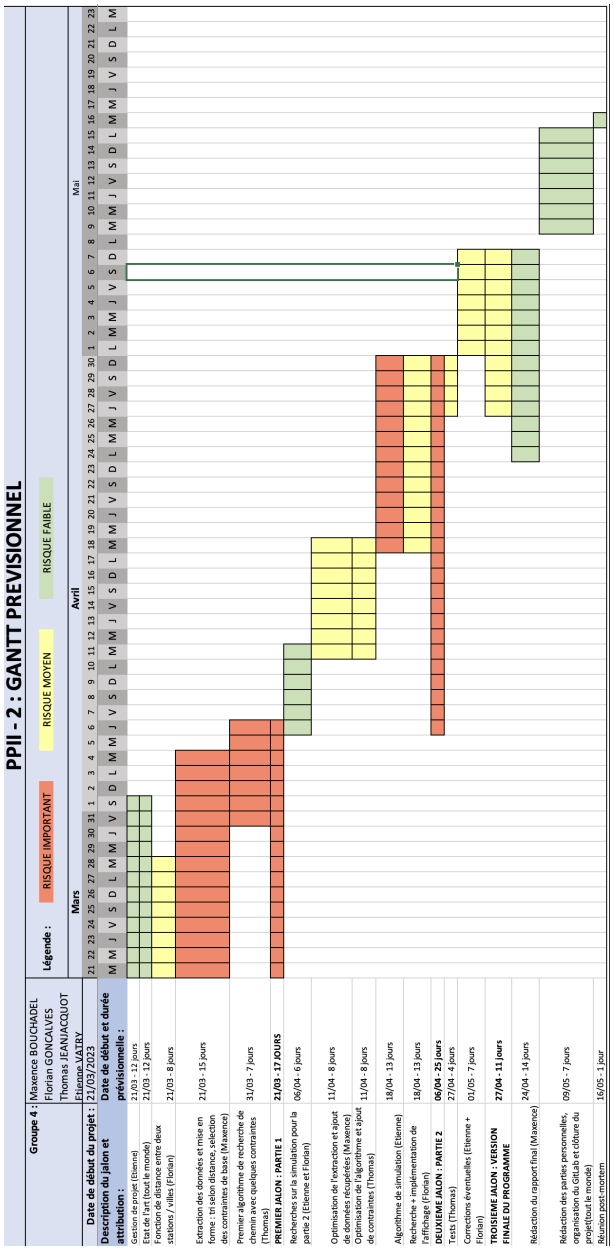
- R = réalise
- A = autorité
- C = conseille
- I = informe

Ce travail a également servi de base pour la réalisation du diagramme de Gantt.

2.5 Diagramme de Gantt

Nous avons réalisé au lancement du projet notre diagramme de Gantt qui détaillait les différentes tâches à accomplir ainsi que les échéances associées. Nous avons suivi ce diagramme de manière méthodique tout au long du projet, en veillant à actualiser régulièrement les informations et à y apporter des modifications lorsque cela était nécessaire. Grâce à ce suivi attentif et rigoureux de notre diagramme de Gantt, nous avons pu garantir la bonne progression du projet et atteindre les objectifs fixés dans les délais impartis.

Voici la version finale de notre diagramme de Gantt (celui-ci est aussi disponible sur le gitlab du projet car il est compliqué de le rendre lisible dans ce rapport) :



2.6 Matrice SWOT

Nous avons réalisé la matrice SWOT de notre projet afin d'identifier ses forces et ses faiblesses et savoir comment les tourner en notre faveur.

Voici notre matrice SWOT :

	Facteurs Positifs	Facteurs Négatifs
Diagnostic Interne	**FORCES**	**FAIBLESSES**
	• Le projet répond à une demande croissante de solutions de recharge pour les véhicules électriques. • Le système proposé est capable de prendre en compte les caractéristiques spécifiques de chaque véhicule, offrant ainsi une expérience utilisateur complète et personnalisée. • La simulation proposée permettra de prédire la charge du réseau en fonction d'un ensemble d'usagers, aidant ainsi les investisseurs et autorités à prendre des décisions éclairées.	• Les hypothèses simplificatrices (chemin direct entre toutes les bornes de recharge, tous les véhicules roulent à la même vitesse) pourraient affecter la précision de la simulation. • L'adoption généralisée des véhicules électriques pourrait prendre du temps, ce qui pourrait limiter la portée du projet à court terme.
Diagnostic Externe	**OPPORTUNITÉS**	**MENACES**
	• Les gouvernements et les investisseurs sont de plus en plus enclins à soutenir les solutions de recharge pour les véhicules électriques, offrant ainsi un marché potentiel pour le projet. • Il est possible d'étendre les fonctionnalités du système pour inclure des fonctionnalités de paiement, de réservation de bornes, etc. • Le système pourrait être adapté à d'autres types de véhicules électriques tels que les vélos ou les scooters électriques.	• La concurrence sur le marché des solutions de recharge pour les véhicules électriques est intense, avec de nombreux acteurs majeurs déjà établis. • Les réglementations et les politiques gouvernementales en matière de recharge des véhicules électriques pourraient changer rapidement, affectant la demande pour le projet. • Les avancées technologiques pourraient rendre obsolète le système développé, limitant ainsi sa durée de vie utile.

Dans le cadre de notre projet, nous avons rencontré quelques défis. L'un des principaux problèmes est lié à l'utilisation d'hypothèses simplificatrices dans notre modèle de simulation. Bien qu'elles facilitent le développement du modèle, ces hypothèses - comme l'existence d'un chemin direct entre toutes les bornes de recharge et une vitesse de conduite constante pour tous les véhicules - peuvent en effet compromettre la précision des prédictions de notre système. En outre, nous avons conscience que l'adoption généralisée des véhicules électriques peut prendre plus de temps que prévu. Cette adoption lente pourrait limiter la portée de notre projet à court terme, en réduisant le nombre de personnes qui utilisent notre système de recharge dans l'immédiat.

De plus, nous avons échangé avec d'autres groupes d'étudiants pour recueillir leurs idées et recommandations sur la manière d'optimiser efficacement l'utilisation de nos bases de données.

Concernant les menaces, nous avons pris en compte la périodes des partiels et des vacances afin de réaliser notre diagramme de Gantt, nous avons préféré ne pas ou peu remplir certaines de ces périodes afin de ne pas prendre

de retard. En effet, nous nous sommes mis d'accord que pendant la période de partiels, aucun de nous ne serait vraiment en état d'avancer sur le projet donc nous avions prévu de finir le projet avant le début de ceux-ci.

Nous nous sommes également mis d'accord de l'importance de communiquer dans le groupe à la fois sur ce qui allait mais aussi et surtout sur ce qui n'allait pas. C'est maintenant notre second projet en groupe, nous avons trouvé important que chacun puisse s'exprimer librement comme durant le premier projet, cela a permis de corriger immédiatement les problèmes et changer de comportement au lieu d'attendre que la situation se dégrade.

3 Notre application

3.1 État de l'art

3.1.1 Contexte

La Commission européenne a récemment pris la décision d'interdire la mise sur le marché des véhicules à moteurs thermiques à partir de 2035. Cette décision marque une étape importante dans la transition vers des solutions alternatives, et les véhicules électriques sont actuellement en plein essor. Pour soutenir cette transition, des réseaux de stations de recharge sont en cours de déploiement à travers tout le territoire.

Dans ce cadre, le projet vise à fournir aux utilisateurs (conducteurs de véhicules électriques, autorités de régulation et acteurs économiques) un ensemble de fonctionnalités qui faciliteront le déploiement et le dimensionnement d'un réseau de recharge adapté aux besoins. L'objectif principal est de créer une plateforme innovante et conviviale qui répondra aux enjeux spécifiques liés aux véhicules électriques et qui tirera parti des meilleures pratiques déjà présentes sur le marché, tout en étant peu exploité et peu connu.

L'application permettra aux conducteurs de véhicules électriques de localiser facilement les stations de recharge les plus proches, de vérifier leur disponibilité en temps réel et de planifier efficacement leurs trajets en fonction de ces informations. De plus, elle fournira des fonctionnalités avancées telles que des alertes en cas de panne de courant, des statistiques de consommation d'énergie et des conseils pour optimiser l'autonomie des véhicules.

Pour les autorités de régulation et les acteurs économiques, l'application offrira des outils de gestion et de suivi du réseau de recharge. Ils pourront

ainsi visualiser les points de recharge existants, analyser les données de fréquentation, planifier de nouvelles installations en fonction des besoins de la population et évaluer l'impact environnemental du réseau.

En encourageant l'utilisation des véhicules électriques et en facilitant l'accès à des stations de recharge fiables et efficaces, cette application contribuera à réduire les émissions de gaz à effet de serre et à promouvoir une mobilité plus durable. De plus, elle sensibilisera les utilisateurs aux avantages environnementaux et économiques de la conduite électrique, et encouragera ainsi l'adoption de ces véhicules dans le cadre de la transition énergétique.

Dans nos recherches approfondies, nous avons étudié plusieurs applications existantes dans le domaine des véhicules électriques et des stations de recharge. Parmi celles-ci, nous avons examiné de près des applications populaires telles que "ChargePoint", "PlugShare" et "Electrify America".

3.1.2 Liste des applications déjà existantes

Après avoir réalisé ce travail de recherche, nous avons ensuite comparé, à l'aide du tableau suivant, les différentes applications existantes sur le marché avec leurs principales options :

Nom de l'application	Recharge Rapide	Itinéraire	Paiement depuis l'application	Itinéraire alternatif	Occupation de la station
PlugShare	OK	OK	-	-	OK
ChargePoint	OK	-	OK	-	OK
Blink Charging	OK	-	OK	-	OK
Tesla Supercharger	OK	OK	OK	OK	OK

Légende : OK : Présence de la fonctionnalité et - : Absence de la fonctionnalité

3.1.3 Problématique

La problématique qui nous était posé était la suivante :

Comment concevoir un programme en C pour optimiser le trajet des véhicules électriques selon les stations de recharge et fournir une simulation permettant d'anticiper les demandes sur le réseau pour un déploiement efficace ?

3.2 Présentation générale

Comme souligné dans l'introduction, l'objectif de notre projet est d'accompagner la transition vers une mobilité plus respectueuse de l'environnement en facilitant l'utilisation des véhicules électriques. Pour ce faire, notre programme met à la disposition des utilisateurs un ensemble de fonctionnalités qui les guident dans l'utilisation de leur véhicule électrique et contribuent au développement de réseaux de recharge efficaces.

Le premier module de notre programme permet à un utilisateur de planifier un itinéraire en voiture électrique de n'importe quel point A à un point B sur le territoire français. L'outil prend en compte les caractéristiques spécifiques de chaque véhicule, telles que la capacité de la batterie et la consommation d'énergie, et propose un itinéraire optimisé qui indique les stations de recharge nécessaires pour compléter le trajet. De plus, nous avons enrichi cette fonctionnalité avec des paramètres supplémentaires pour répondre aux besoins individuels des utilisateurs, tels que la conservation d'un certain niveau de charge de la batterie et la limitation de la durée de la recharge.

Le deuxième module de notre programme est un simulateur de réseau de recharge, conçu pour aider les investisseurs et les autorités à comprendre et à anticiper les demandes sur le réseau de recharge. Le simulateur est capable de gérer un ensemble d'usagers fictifs, en tenant compte de leurs caractéristiques spécifiques et de leurs parcours. À chaque intervalle de temps, il évalue l'occupation des stations de recharge et identifie celles qui sont susceptibles de générer des files d'attente.

3.3 Les bases de données

Pour notre projet, nous avons utilisé trois bases de données différentes : `cities.csv`, `station.csv`, et `voiture.csv`. Ces fichiers contiennent respectivement les informations sur les villes, les stations de recharge et les voitures que nous avons utilisées dans notre simulation.

Ces fichiers de données ont été trouvés gratuitement sur Internet. Le format CSV (Comma-Separated Values) est un choix judicieux pour plusieurs

raisons :

— **Simplicité et universalité** : Le format CSV est largement utilisé et est reconnu par une grande variété de logiciels, qu'il s'agisse de bases de données, de tableurs ou de langages de programmation. Cela rend le partage et l'importation de données facile et directe.

— **Facilité de lecture et d'écriture** : Les fichiers CSV peuvent être lus ou écrits par une grande variété de langages de programmation, dont le C. Cela a facilité l'importation des données dans notre programme.

— **Compact et efficace** : Contrairement à d'autres formats de données tels que JSON ou XML, CSV est un format très compact. Il n'a pas besoin de balises de nom pour chaque valeur, ce qui signifie que les fichiers CSV occupent moins d'espace pour la même quantité de données.

Avant l'utilisation de ces bases de données, nous avons dû les traiter afin de supprimer les données qui ne nous intéressaient pas ou qui étaient présentes en double. Par exemple, pour les stations, nous avons remarqué qu'avant le traitement, nous avions 35 000 stations, et qu'après la suppression des doublons, nous sommes passés à environ 12 000 stations disponibles. Ceci s'explique par le fait que certaines stations possédaient, par exemple, 4 prises de courant, et donc la même station était présente 4 fois dans la base de données au lieu d'être présente seulement une fois.

Cependant, il est à noter que le format CSV peut ne pas convenir à tous les types de données. Par exemple, si nous avions eu besoin de stocker des données plus complexes ou hiérarchiques, nous aurions pu envisager d'utiliser un format différent, comme JSON.

id	department_code	insee_code	zip_code	name	slug	gps_lat	gps_lng
1	01	01001	01400	L'Abergement-Clémenciat	l abergement clemenciat	46.156781992032	4.9246992031872
2	01	01002	01640	L'Abergement-de-Varey	l abergement de varey	46.010085625	5.4287591666667
3	01	01004	01500	Ambérieu-en-Bugey	amberieu en bugey	45.958409392265	5.3759920441989

FIGURE 1 – cities.csv

id_station_itinerance	code_insee_commune	coordonneesXY	puissance_nominale	consolidated_longitude	consolidated_latitude	consolidated_commune
FRELCPBLOHM	68042	[7.50290400,47.60821400]	225	7.502904	47.608214	Blotzheim
FRELCPECUSM	69081	[4.78311100,45.78565100]	150	4.783111	45.785651	Écully
FRELCPSBARS	49267	[-0.50261700,47.48171600]	150	-0.502617	47.481716	Saint-Barthélemy-d'Anjou
FRELCPSBARS	49267	[-0.50261700,47.48171600]	150	-0.502617	47.481716	Saint-Barthélemy-d'Anjou

FIGURE 2 – stations.csv

Model Name	Range (km)	Efficiency (Wh/km)	Fast Charge (km/h)	Brand
Tesla Model Y Long Range Dual Motor	435	172	670	Tesla
Tesla Model 3	380	151	630	Tesla
Tesla Model Y	345	167	580	Tesla

FIGURE 3 – voitures.csv

3.4 Les structures de données utilisées

Pour réaliser notre projet, nous avons utilisé plusieurs structures de données pour modéliser les différents éléments nécessaires à notre simulation. Voici une brève description de chaque structure :

- `station_t` : Cette structure contient des informations sur une station de recharge, y compris ses coordonnées, sa puissance de recharge, son identifiant, son nombre de prises disponibles et un pointeur vers la prochaine station dans la liste chaînée. La structure contient aussi un tableau d'entier correspondant au nombre de véhicules dans la station au cours du temps. Cela permet de regrouper toutes les informations utiles en un seul objet, ce qui facilite la gestion et l'accès à ces informations. Elle est également utile pour le passage d'arguments car si vous devez passer des informations sur une station de recharge à une fonction, vous pouvez simplement passer la structure "station" en tant qu'argument, ce qui permet de transmettre toutes les données nécessaires en une seule opération.
- `coordinates_t` : Cette structure contient deux variables de type double pour stocker les coordonnées géographiques (latitude et longitude).
- `car_t` : Cette structure stocke les informations concernant une voiture, incluant un identifiant, le nom du modèle, la portée (en km) et l'efficacité énergétique (en Wh/km).
- `AutoPath_t` : Cette structure encapsule toutes les informations nécessaires pour un trajet donné. Elle comprend les noms des villes de départ et d'arrivée, un pointeur vers la structure de la voiture, l'état initial et actuel de la batterie, une référence à la structure du trajet, et des informations sur l'état de progression de la voiture.
- `city_t` : Cette structure est utilisée pour stocker les informations sur une ville, y compris son nom et ses coordonnées géographiques.
- `journey_t` : Cette structure contient des informations détaillées sur un

trajet, y compris les étapes (sous forme de coordonnées), les conditions de chaque étape, l'état de la batterie au début de chaque étape, et le nombre total d'étapes.

— `carre_t` : Cette structure sert de liste chaînée pour stocker les stations de recharge dans un certain carré de la grille. Elle contient un pointeur vers la première station de la liste et un compteur du nombre total de stations.

Ces structures nous ont permis de modéliser de manière précise et flexible les différentes composantes de notre simulation, facilitant ainsi l'implémentation des différentes fonctionnalités de notre programme.

3.5 Première Partie : L'itinéraire

La première partie consistait à fournir pour tout usager désirant se rendre d'un point A à un point B, un trajet pour son véhicule électrique. La fonction devait prendre en compte les spécifications de la voiture de l'utilisateur pour lui proposer un trajet dédié à son automobile. A ces fonctionnalités de base, on pouvait ajouter de plus quelques options que nous détaillerons dans la suite.

La fonction principale pour cette partie se nomme itinerary. Cette fonction utilise les fonctions load_cars et load_stations afin de charger et créer les listes pour les voitures et les stations. Après cela, le programme demande à l'utilisateur d'entrer les différentes options de son trajet. On retrouve évidemment en premier lieu, la demande des villes de départ et d'arrivée. Une fois le nom des villes saisies, le programme appelle la fonction find_city qui associe à un nom de ville, ses coordonnées GPS.

Ensuite, le programme demande à l'utilisateur d'entrer le nom du modèle de sa voiture afin de récupérer toutes les informations concernant son véhicule. La fonction fin_car_by_model réalise cette tâche. Elle vérifie que la voiture est bien présente dans le tableau des modèles de véhicule électrique puis, renvoie la ligne qui correspond au bon modèle afin que l'on puisse ensuite récupérer l'autonomie et l'efficacité de la voiture. Ces informations étant nécessaires pour le calcul de l'itinéraire.

Une fois les coordonnées des villes acquises ainsi que les caractéristiques du véhicule, le programme demande à l'utilisateur différentes options. Il commence par demander le niveau de charge actuel de la voiture, le niveau de batterie minimum en dessous duquel l'utilisateur ne veut pas se retrouver

(pour éviter par exemple qu'une station soit déjà occupée lorsqu'il faut recharger le véhicule ou bien si la station est en panne par exemple).

Nous avons également ajouté une option qui demande à l'utilisateur combien de temps il veut attendre au maximum à une borne de recharge.

Une fois toutes ces données collectées, le programme appelle soit la fonction print_itinerary qui permet d'afficher uniquement le trajet à réaliser avec lieu de départ et d'arrivée, ainsi que les stations à utiliser avec le temps de trajet total et la distance.

Si l'on a choisi d'actualiser la position de la voiture toutes les X minutes. Alors le programme appelle la fonction print_itinerary_by_time qui renvoie de plus la position de la voiture à chaque instant X.

Une fois le programme terminé, il supprime la liste de voiture ainsi que la liste des stations afin d'éviter de générer des fuites de mémoires.

Nous avons réalisé des fonctions de test pour la totalité de nos fonctions afin de vérifier qu'elles réalisaient bien les tâches attendues. Tous les tests sont disponibles dans le git du projet.

```
Entrez le nom de la ville de départ: Nancy
Entrez le nom de la ville d'arrivée: Paris
Entrez le modèle de votre voiture: Tesla Model 3
Entrez le niveau de charge de votre batterie (en %): 79
Sous quel niveau de batterie ne voulez vous pas passer en dessous (en %)? 20
Actualiser la position des voiture toute les x minutes (0 si non voulu): 0
Combien de temps maximum voulez vous attendre à une station (en minutes)? 10


Calcul de votre itinéraire en cours ........

Votre trajet fait 282 km
Votre voiture à une autonomie de 380 km.

Le chemin à suivre sera:
    Départ: [48.689646,6.173720]
    Rechargement 1 à la station [48.821800,3.138290], pendant 153 minutes et puissance station (
18000 watt)
    Arrivée: [48.864049,2.331053]
```

FIGURE 4 – Exemple d'itinéraire de Paris à Nancy avec une Tesla Model 3

3.6 Seconde Partie : La simulation

Etienne s'est occupé en majorité de cette partie du projet. La réalisation de la simulation s'est découpée en deux partie : une partie initialisation et une partie réalisation.

La partie initialisation concerne, comme son nom l'indique, l'initialisation de la simulation. Pour ce faire, la fonction « iniAutoPath_simu » a été créée. Cette dernière a pour but de créer un tableau qui va contenir toutes les entités de la simulation. On commence donc par allouer toute la mémoire dont on va avoir besoin. Ensuite, on charge les fichiers contenant les voitures et les villes pour récupérer les données dans deux tableaux distincts.

On choisit ensuite, pour chaque entité, les données nécessaires au hasard : villes de départ et d'arrivée, modèle de la voiture, niveau de batterie de départ, niveau de batterie minimum et le temps d'attente maximum dans une station. On obtient ainsi un tableau contenant toutes les entités et leur données initialisées.

La fonction « iniAutoPath_simu » fonctionne avec une complexité en O(n), n étant le nombre d'entités dans la simulation. En effet, on ne fait qu'une boucle sur l'ensemble des éléments qu'on veut créer.

Concernant la gestion de la mémoire, on alloue de la mémoire pour trois tableaux : celui qui va contenir nos entités, un pour les villes et un pour les voitures. Ces deux derniers sont libérés dès la fin de la fonction. Ainsi, on ne garde en mémoire que les données utiles.

Vient ensuite la partie réalisation. Cette dernière correspond à la réalisation des trajets de chaque entité. C'est la fonction « travelAutoPath_simu » qui va gérer cela.

Pour ce faire, on calcule le trajet de chaque entité l'un après l'autre. Avant de calculer le trajet de chaque entité, on va charger les données du fichier contenant les stations de recharge en France. On va ranger ces dernières dans un tableau de listes de la façon suivante : on divise la France en 256 carrés.

On crée un tableau à 256 cases, chaque case correspondant à un carré de la France. On parcourt ensuite le fichier contenant les stations. Pour chaque station, on calcule l'indice du carré où elle se trouve (si la station n'est pas en France métropolitaine ou en Corse, on l'ignore). Une fois l'indice trouvé (on l'appelle n), on va placer la station à la suite de la liste chaînée située à la case n du tableau.

Ainsi, chaque case du tableau contiendra une liste chaînée de toutes les stations qui sont dans le carré correspondant à l'indice de la case du tableau. Ce tri se fait à l'aide de la fonction « load_station_simu ». Le calcul du trajet se fait de la façon suivante :

- **Cas 1** : l'utilisateur possède assez de batterie pour rouler 10 min ET l'arrivée est assez proche pour y accéder en 10 min ou moins. Dans ce cas là, on dit que l'utilisateur est directement arrivé : c'est le cas le plus simple. On met à jour la batterie restante après le trajet.
- **Cas 2** : l'utilisateur possède assez de batterie pour rouler 10 min, mais l'arrivée est trop loin. On rajoute donc une étape dans le trajet de la voiture, cette étape correspondant aux coordonnées de la voiture après 10 min de route dans la direction de l'arrivée. On met aussi à jour la batterie de la voiture. Ici, c'est la fonction « point_on_line_simu » qui va donner la position de la voiture après 10 min de route.
- **Cas 3** : l'utilisateur ne possède pas assez de batterie pour rouler 10 min. Dans ce cas-là, il faut trouver une station pour recharger. On va donc appeler la fonction « find_neares_station_simu » pour trouver la station la plus proche. Là encore, on va distinguer plusieurs cas :
  - **Cas 3.1** : il existe une station dans le carré où la voiture se trouve actuellement et la voiture possède assez de batterie pour y accéder. Dans ce cas, on dit que la voiture va à cette station pour recharger. On met à jour la batterie en prenant en compte la batterie consommée pour arriver jusqu'à la station. Une fois arrivé à la station, on calcule (à l'aide de la macro « TEMPS_RECHARGEMENT », qui prend en paramètres la batterie de la voiture, sa capacité et la puissance délivrée par la station) le temps de rechargement nécessaire pour un rechargement à 100. On divise ensuite ce temps par 10 pour avoir le nombre de cycle de 10 min de rechargement nécessaires. Si le temps de rechargement est plus long que celui indiqué par l'utilisateur (temps d'attente max à une station), on rechargera aussi longtemps que le temps voulu. Ainsi, pour chaque cycle de 10 min de rechargement, on mettra à jour la batterie et on indiquera qu'on est en train de recharger.
  - **Cas 3.2** : il existe une station dans le carré où la voiture se trouve actuellement mais la voiture ne possède pas assez de batterie pour y accéder. Dans ce cas on arrête le calcul du trajet et on le comptabilise comme une erreur.
  - **Cas 3.3** : il n'existe pas de station dans le carré où on se trouve.

Dans ce cas on arrête le calcul du trajet et on le comptabilise comme une erreur. Pour chaque entité, le calcul du trajet peut s'arrêter dans trois cas différents : soit on est arrivé à destination, soit une erreur a été relevée (voir les cas 3.2 et 3.3), soit on a fait trop d'étapes (on limite ainsi le nombre d'étapes et donc le temps de calcul). Dans les deux cas précédents, on gardera en mémoire le fait que l'utilisateur n'est pas arrivé à destination. La fonction « travelAutoPath_simu » est linéaire : en effet, le temps de chargement des stations est constant, de même que le temps de calcul du trajet (on limite le nombre d'étapes pour chaque trajet, ce qui rend la recherche d'étapes constante). Concernant la mémoire, on remplit le tableau créé par « initAutoPath_simu », ce qui ne nous fait pas consommer plus de mémoire. De plus, le tableau contenant les stations est supprimé dès la fin de la fonction, permettant de libérer la mémoire allouée le plus tôt possible.

Enfin, pour ce qui concerne le côté état du réseau en France, on procède de la manière suivante :

- Lors de la création du tableau de listes de stations, on garde en mémoire l'identifiant de la station ainsi que son nombre de prises disponibles. A côté de cela, on initialise un tableau d'entiers de même taille que ceux qui vont contenir les différentes étapes dans AutoPath.
- On modifie ce tableau dans travel_AutoPath. Pour ce faire, à chaque fois qu'on récupère une station avec « find_nearest_station_simu », on incrémente de 1 la case correspondant à notre numéro d'étape dans le tableau d'entier décrit ci-dessus. A l'inverse, on décrémente la case suivant le rechargement pour simuler le départ d'une voiture
- Enfin, on récupère le tableau de stations modifié qu'on va parcourir pour afficher les stations où il y a une trop grande file d'attente à des instants t définis à l'avance. Ces derniers sont définis dans ChargeMate : on demande à l'utilisateur de rentrer une durée pendant laquelle il veut voir l'état des stations, ainsi qu'une heure de début. Ainsi, on est capable de montrer quelles sont les stations avec trop de file d'attente (on a choisi de n'afficher que celles ayant deux fois plus de voitures voulant charger que de prises).

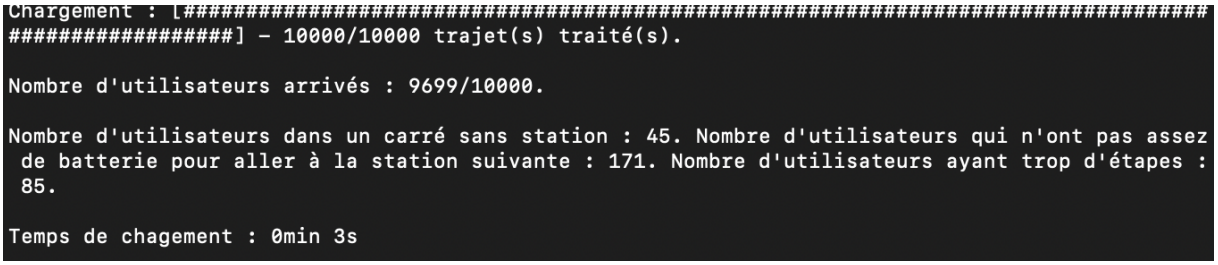


FIGURE 5 – Exemple de simulation pour 10 000 véhicules

3.7 Test

Voici l'ensemble de nos test pour ce projet :

- **"test_distance" :** Calcule la distance entre deux coordonnées géographiques et vérifie si la distance calculée est proche de la distance réelle entre Paris et Marseille.
- **"test_coordtocarre" :** Vérifie si la fonction "coordtocarre" attribue correctement l'indice de la case correspondant aux coordonnées géographiques données.
- **"test_load_stations" :** Charge les stations de recharge à partir d'un fichier et vérifie si le tableau de carrés contenant les stations n'est pas nul. Vérifie également si le nombre de stations dans le carré correspondant à Paris est supérieur à zéro et si le nombre de stations dans le carré correspondant à l'océan est nul.
- **"test_station_append" :** Vérifie si la fonction "station_append" ajoute correctement une nouvelle station au carré donné.
- **"test_find_city" :** Utilise la fonction "find_city" pour rechercher les coordonnées géographiques d'une ville donnée et vérifie si les coordonnées obtenues sont correctes.
- **"test_find_nearest_station" :** Recherche la station de recharge la plus proche des coordonnées géographiques spécifiées et vérifie si la station renvoyée est correcte.
- **"test_point_on_line" :** Calcule les coordonnées d'un point situé à une distance spécifiée le long d'une ligne définie par deux coordonnées géographiques et vérifie si les coordonnées du point calculé sont correctes.
- **"test_load_cars" :** Charge la liste des voitures à partir d'un fichier et vérifie si les voitures sont correctement chargées dans le tableau.

- **"test_find_car_by_model" :** Recherche une voiture dans la liste en utilisant le modèle de voiture et vérifie si l'index retourné est correct.
- **"test_battery_level" :** Calcule le niveau de batterie restant après un trajet et vérifie si le niveau de batterie calculé est correct.
- **"test_waiting_time" :** Calcule le temps d'attente nécessaire pour recharger une batterie et vérifie si le temps d'attente calculé est correct.

— **"test_distance_before**$_b$*attery_limit*" : *Calculeladistancemaximalepouvanttreparcourueavan*

3.8 Nos algorithmes

3.8.1 Problématiques et solutions

Le problème principal que nous avons rencontré est un problème d'optimisation. En effet, nos fichiers de données sont gigantesques et si nous n'optimisions pas le programme il aurait été impossible de réaliser la simulation.

Pour palier ce problème, nous avons décidé de diviser la france en un nombre de carré défini au préalable. L'optimisation du projet avec l'utilisation des carrés se révèle être une approche pratique et efficace. En regroupant les stations de recharge électrique en carrés, on peut réduire la complexité et améliorer les performances des opérations de recherche et de comptage des stations dans une région donnée (cette approche ne permet cependant pas de passer de O(n) à O(log(n)).

En utilisant les carrés, on divise l'espace géographique en régions plus petites et délimitées. Cela permet de réduire la quantité de données à traiter lors de l'analyse des stations de recharge. Plutôt que de parcourir toutes les stations une par une, on peut se concentrer uniquement sur les carrés pertinents pour une région donnée, ce qui accélère les opérations de parcours du tableau de stations.

3.9 Points d'améliorations

Malgré le fait que nous sommes fiers de notre application, nous nous rendons compte qu'il y a plusieurs points que nous aurions aimé améliorer :

— Concernant les bases de données, nous avons remarqué que celle-ci avaient parfois des données manquates. Avec un peu plus de temps, nous aurions pris le temps de les modifiers ou compléter.

— Bien que notre application soit fonctionnelle, elle n'est en l'état pas utilisable en application réelle à cause des simplifications que nous avons effectués, il aurait été interessant dans un second temps de tenter de supprimer au fur et à mesure ces simplifications afin de se rapprocher au plus possible de la réalité.

— Enfin, nous aurions aimé faire un affichage plus détaillé avec une carte de la france interactive et les trajets directement affichés dessus.

4 Guide d'utilisation

L'utilisation de notre application est vraiment simple, tout d'abord il faut utiliser un terminal et se placer dans le répertoire /src de notre dépot gitlab.

Par la suite il suffit simplement de taper la commande "make ChargeMate" et l'application se lance. Apparait ensuite un menu textuel où il suffit de sélectionner la fonctionnalité que nous souhaitons utiliser.

Taper 1 : lance l'itinéraire classique. Il faut ensuite rentrer une ville de départ, une ville d'arrivée, un modèle de voiture, la batterie initiale, le seuil de batterie en deçà duquel on ne veut pas descendre, l'actualisation ou non des étapes du trajet et le temps d'attente maximum dans une station.

Taper 2 : lance une simulation permettant d'afficher le rendu des trajets. Il faut ensuite rentrer un nombre d'entités.

Taper 3 : lance une simulation permettant d'évaluer les files d'attentes dans les différentes stations. Il faut ensuite rentrer un nombre d'entités puis, dans un second temps, la durée pendant laquelle on veut évaluer les files d'attentes dans les stations et l'heure de début.

Taper 4 : ferme l'application.

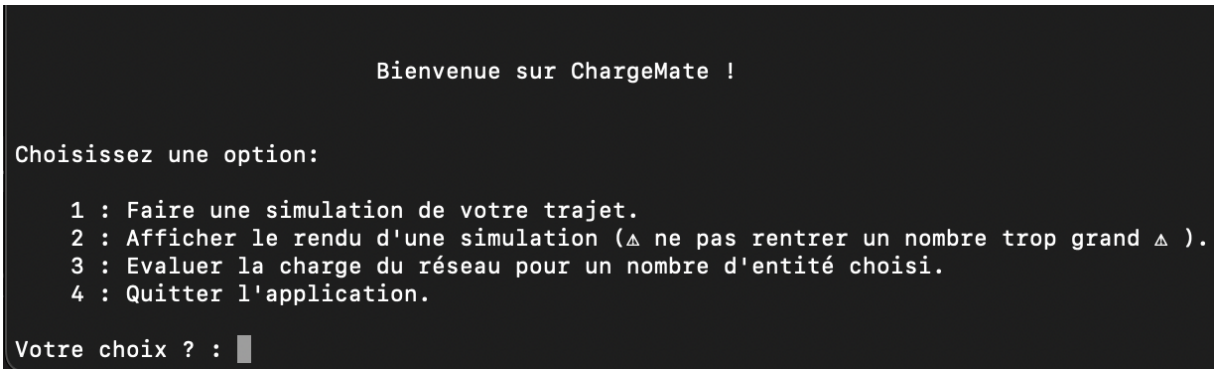


FIGURE 6 – Menu textuel

5 Conclusion

5.1 Conclusions personnelles

5.1.1 Conclusion de Florian

Tout d'abord, j'aimerais remercier mes camarades pour la bonne entente au sein du groupe tout au long du développement du projet, ce qui a rendu agréable la réalisation de ce dernier.

De plus, je tiens à souligner l'organisation efficace du groupe, chaque membre a rapidement su se placer sur les différents axes du projet grâce à des réunions efficaces et bien menées par le chef de projet.

Le commencement du projet m'a permis de revoir et d'approfondir mes connaissances dans le langage C. Au début, cela a été un peu troublant car c'était la première fois que j'appliquais de manière concrète mes connaissances sur ce langage. Rapidement, j'ai su m'adapter et très vite rendre une fonction qui permet de calculer la distance entre deux points, en me basant sur une fonction similaire que j'avais fais au premier semestre en python. Cela m'a motivé par rapport aux difficultés rencontrées et j'ai pu poursuivre le développement du projet de manière enthousiaste.

La seconde partie du projet a été très intéressante pour moi car je devais réaliser une fonction qui devait découper la France en carrés afin d'optimiser les recherches et gagner du temps de calcul. J'ai cependant eu beaucoup de mal à l'implémenter car le fichier de base de données des stations n'était pas correctement formaté.

Au niveau des connaissances personnelles, le projet m'a permis d'approfondir mes connaissances et de manipuler les bases de données en C. Ce projet était le second avec les membres de ce groupe, et les habitudes que nous avions pris ensemble au premier semestre se sont faites ressentir. Ce fut un plaisir de travailler sur ce projet de manière efficiente, avec des camarades motivés.

5.1.2 Conclusion de Maxence

Pour moi, ce second projet PP2I est une réussite. Nous avons réussi à atteindre nos objectifs en temps voulu, dans une ambiance sereine et conviviale. Étant donné que nous avions déjà réalisé le premier projet ensemble, l'équipe se connaissait déjà et nous étions conscients des forces et des faiblesses de chacun.

Dès l'annonce du sujet, nous avons commencé à travailler, ce qui nous a permis de gagner du temps et d'implémenter un maximum de fonctionnalités avant la date limite du 24 mai. Étienne, en tant que chef de projet, a été vraiment excellent et nous a permis de progresser dans les meilleures conditions possibles.

Durant la première partie du projet, j'étais chargé de trouver les différentes bases de données que nous utiliserions ainsi que les structures de données à implémenter. Puis, lors de la seconde partie du projet, je me suis occupé de certains documents de gestion de projet afin d'avoir un dossier clair et complet à rendre.

En conclusion, j'aimerais dire que ce projet m'a vraiment apporté de nouvelles connaissances. Il m'a permis de mettre en pratique ce que nous avions appris durant le cours de Structures de Données de manière concrète avec une application réelle.

5.1.3 Conclusion d'Etienne

De mon point de vue, ce projet s'est beaucoup mieux déroulé que le premier PP2I (même si ce dernier c'était déjà très bien passé). En effet, comme nous avions déjà travaillé ensemble sur un projet informatique, nous savions quoi faire pour ne pas répéter les mêmes erreurs que nous avions faites.

Je voudrais notamment souligner le fait que nous avons réussi à nous en tenir au Gantt prévisionnel, à savoir finir les deux parties séparément avant la fin des vacances de Pâques. Grâce à cela, nous avons pu éviter d'être sous pression pendant les semaines de partiels, ce qui nous a permis d'aborder ces dernières plus sereinement.

Ce projet m'a beaucoup plus plu que le premier. En effet, le C est un langage que j'apprécie particulièrement, ce qui n'a fait que me rendre plus enthousiaste à travailler dessus. La répartition du travail s'est très bien organisée : il était convenu que Thomas, Florian et Maxence fassent la partie 1 de façon très complète pendant que je faisais la partie gestion de projet. Grâce à leur travail, j'ai pu commencer la partie 2 en ayant déjà une bonne base de travail et en n'ayant qu'à réadapter les fonctions créées par mes camarades à une simulation. Je n'ai pas rencontré de difficulté particulière, et voir le projet avancer aussi vite et aussi régulièrement était très satisfaisant. Pour conclure, je dirais donc que le projet était vraiment intéressant et qu'il m'a permis de mettre en application tout ce que j'avais appris du premier projet et d'apprendre de nouvelles choses sur la programmation d'une application. C'est un expérience qui me servira sûrement dans mes prochains projets !

5.1.4 Conclusion de Thomas

Je suis très content du déroulement de ce deuxième projet. Nous avons fait le choix de garder le même groupe qu'au premier semestre. En effet, lors du premier projet nous avions identifié les points forts et les points faibles de notre groupe, et nous avons pensé que nous avions les capacités de faire du bon travail ensemble et de mettre à profit les connaissances que nous avions sur chacun pour travailler encore plus rapidement.

Nous avons donc dès le départ décidé de modifier les choses qui nous avaient fait défaut. En commençant par nommer un chef de groupe Etienne, qui était assisté par un secrétaire que je représentais. Ceci nous a permis de commencer très tôt le travail et ainsi de se retrouver avec une version fonctionnelle de notre application longtemps avant la date de rendu du projet. Ceci nous a permis de pouvoir ajouter des fonctionnalités supplémentaires, de corriger les différents bugs, de réaliser l'entièreté des tests, ... Tout ceci à contribuer à la bonne entente et la cohésion du groupe mais également à un sentiment de travail bien fait.

Nous avons donc pu apporter des améliorations en termes de performance pour notre application, ce qui a été permis par plus de sessions de travail en groupe au cours desquelles on pouvait donner des conseils aux autres pour réaliser leur partie et également créer une cohérence dans le programme.

5.2 Conclusion générale

À l'issue du projet, nous avons réalisé une application en C axée sur la transition écologique, passant des voitures thermiques aux voitures électriques, répondant à plusieurs problématiques.

Afin de garder constamment un œil sur nos objectifs, nous avons utilisé différents outils de gestion de projet vus lors du MOOC GDP, comme par exemple un diagramme de Gantt, qui a permis de recenser le niveau d'avancement des différentes tâches. De plus, la création du WBS nous a permis de diviser les objectifs en lots plus simples, ce qui a fluidifié le travail en général. Nous noterons également que nous avons su nous adapter rapidement lorsqu'il fallait réaffecter quelqu'un à une autre tâche, afin d'optimiser le travail en fonction des compétences de chacun.

Au niveau des fonctionnalités escomptées, nous sommes plutôt fiers de notre travail. Nous avons respecté l'ensemble du cahier des charges. Notre application permet d'afficher un trajet d'un point A à un point B avec n'importe quel véhicule électrique et indique où le conducteur doit s'arrêter pour recharger la voiture s'il en a besoin. De plus, la partie simulation est très fonctionnelle et optimisée, ce qui permet de réaliser de grandes simulations en peu de temps.

Nous aurions aimé avoir un peu plus de temps pour permettre d'afficher les différents trajets sur une carte de la France, mais le délai était un peu trop court avec les examens et le temps qu'il nous restait.

Malgré les simplifications que nous avons effectuées (trajet à vol d'oiseau, vitesse constante...), nous pensons que notre application répond à la problématique qui nous a été proposée.

Finalement, ce projet a permis à chacun de développer de nouvelles compétences et de mettre en pratique ce qui a été vu tout au long du semestre. Nous en tirons de solides enseignements qui nous seront utile pour nos futurs projets.

6 Annexes

6.1 Comptes rendus de réunions

6.1.1 Réunion du 21 Mars 2023

Compte-rendu n°01
Réunion du Mardi 21 Mars 2023

Type de la réunion : Réunion de lancement	Rédigé le 21 Mars 2023 Lieu : Réalisée dans la salle S.0.13
Heure de début : 13h	Heure de fin : 14h

Objet : Lancement du projet

Présents :	
BOUCHADEL Maxence	GONCALVES Florian
JEANJACQUOT Thomas (secrétaire)	VATRY Etienne

Ordre du jour :

- Relecture du sujet
- Concentration sur la partie 1 :
  1. deadline
  2. Création et répartition des lots de travail
- Partie 2 : si il y a trop de monde pour la partie 1, une personne pourra commencer la partie 2

Échanges :

Nous avons commencé par relire ensemble le sujet pour se le remémorer. Suite à cela, Etienne nous a reformulé les consignes de la première partie du projet. Il faut donc réaliser un programme C qui demande en entrée le nom de la ville de départ, de la ville d'arrivée et le modèle de la voiture. Ensuite le programme doit nous renvoyer la liste des stations de recharge auxquelles l'utilisateur doit s'arrêter pour recharger sa voiture électrique.

Nous avons ensuite consulté les bases de données qui nous ont été fournies. Il a été question de savoir si l'on pouvait utiliser ces bases de données directement via SQL, et donc de savoir comment l'utiliser en C ou alors de devoir tout stocker dans une structure de données adaptée que nous aurions définie. Pour répondre à cette question, Florian a décidé d'envoyer un mail à Monsieur Festor afin d'être sûr sur de la marche à suivre

Nous nous sommes mis d'accord sur les différentes tâches à réaliser :

1

Thomas va devoir commencer à réaliser une première version d'un algorithme de calcul de plus court chemin dans un graphe (algorithme de Djikstra) qui nous permettra, une fois les bases de données implémentées, de réaliser la première partie de projet.

Florian s'occupe des fonctions de calcul de distances entre 2 villes/stations et de recherche de stations les plus proches des villes. Il doit adapter la fonction qu'il avait déjà réalisée pour le projet précédent.

Maxence s'occupe de l'extraction des données (soit en C soit SQL) suivant les données qu'on veut : véhicule (id, nom, capacité, consommation), station (id, nom, puissance disponible, vitesse de recharge, coordonnées), ville (id, nom, coordonnées).

Etienne s'occupe de faire les documents de gestion de projet notamment le WBS pour développer tous les lots de travail ainsi que de faire le diagramme de Gantt.

Une courte réunion est prévue Vendredi à 10h pour faire un point sur l'avancement. Il a également été décidé de faire beaucoup plus de sessions de travail en commun autre que les réunions afin de pouvoir avancer ensemble et de pouvoir échanger directement avec les autres en cas de problème.

To Do List :

Description	Délai	Personne
Commencer algo PCC	Mardi 28/03	Thomas
Modifier son algo de distance en C	Mardi 28/03	Florian
Création de structure pour les BDs	Mardi 28/03	Maxence
WBS, Gantt	Mardi 28/03	Etienne

Prochaine Réunion : Mardi 28 Mars 2023 pour une réunion d'avancement.

6.1.2 Réunion du 05 Avril 2023

Compte-rendu n°02
Réunion du Mercredi 5 Avril 2023

Type de la réunion : Réunion de lancement	Rédigé le 5 Avril 2023 Lieu : Réalisée dans la salle S.0.13
Heure de début : 8h30	Heure de fin : 9h30

Objet : Réunion d'avancement

Présents :	
BOUCHADEL Maxence	GONCALVES Florian
JEANJACQUOT Thomas (secrétaire)	VATRY Etienne

Ordre du jour :

- Mise au point sur l'avancement de chacun
- Choix de l'algorithme de calcul du trajet
- Définitions des contraintes

Échanges :

Nous avons commencé par faire un point sur l'avancement de chacun depuis la dernière réunion.

Florian a commencé par nous présenter sa fonction Distance qui permets de calculer la distance entre 2 points en donnant leurs coordonnées GPS. Il a ensuite créé une fonction qui permet de donner la station la plus proche d'un point.

Maxence a téléchargé les bases de données et les a rendues exploitables en C. Il a ensuite modifié la fonction de Florian afin créer la fonction FindCity qui permet de renvoyer la station la plus proche en rentrant le nom d'une ville.

Thomas a réfléchi à plusieurs moyens de réaliser la fonction principale et les a exposées pendant la réunion afin que l'on puisse choisir la meilleure option.

Etienne a réalisé les différents documents de gestion de projet comme le diagramme de Gantt, la matrice Swott, ...

Nous avons définies les principales contraintes à respecter pour notre fonction :

- Lieu départ / arrivée
- Niveau de batterie de départ

1

— Modèle de la voiture
— Temps d'attente aux stations (en fonction de la puissances des chargeurs)
— Le niveau de batterie à l'arrivée des stations
— Niveau minimum de charge de la batterie
— Vitesse moyenne

Nous avons réfléchi également aux sorties de la fonction que nous pourrons stocker dans une structure de donnée (chemin_t) :

— Listes de stations/ villes à visiter (donner le temps de recharge à chaque station)
— Distance totale
— Durée totale
— Niveau de batterie à l'arrivée
— Temps d'attente moyen dans les stations

To Do List :

Description	Délai	Personne
Faire le programme principale	Dimanche 16/04	Thomas
Faire le programme Findsation	Dimanche 16/04	Maxence
Faire la fonction qui en lui donnant une adresse exact permet de renvoyer ses coordonnées	Dimanche 16/04	Florian
Recherches sur simulation et affichage	Dimanche 16/04	Etienne et Flo
Recherche de station dans un certain angle	Mercredi 12/04	Florian

Prochaine Réunion : A déterminer

6.1.3 Réunion du 13 Avril 2023

Compte-rendu n°03
Réunion du Jeudi 13 Avril 2023

Type de la réunion : Réunion d'avancement	Rédigé le 13 Avril 2023 Lieu : Réalisée dans le local du BDS
Heure de début : 13h	Heure de fin : 14h

Objet : Réunion d'avancement

Présents :	
BOUCHADEL Maxence (secrétaire)	GONCALVES Florian
JEANJACQUOT Thomas	VATRY Etienne

Ordre du jour :

- Mise au point sur l'avancement de chacun
- Recherche station dans un certain angle afin d'optimiser le programme
- Fonction Findstation
- Recherche sur la simulation et affichage

Échanges :

Nous avons commencé par mettre en commun ce que chacun avait fait depuis la dernière réunion.

Maxence a parlé de la fonction Findstation et de ses difficultés a mettre au point une contrainte permettant de réduire le temps de parcours de la liste de station et l'optimisation du programme.

Thomas a parlé de son idée, diviser la France en plusieurs zones et chaque stations sera associé à une zone. Cela ne semble pas être trop complexe à mettre en place et nous ferait gagner un temps non négligeable sur la rapidité du programme.

Etienne a continué la gestion de projet et a échangé avec d'autres groupes pour savoir leur organisation autour de ce projet

Florian a parlé de ses idées pour la simulation mais pour le moment il essaie d'optimiser le programme principal avant de se lancer dans celle-ci.

To Do List :

Description	Délai	Personne
Fonction pour diviser la France en zone et maj de la base station	Jeudi 20/04	Florian
Optimiser le programme principal	Jeudi 20/04	Thomas
Optimisation de FindStation en utilisant le fait que la France soit diviser en différentes zones	Jeudi 20/04	Etienne
Matrice SWOT, Charte de projet, affichage de la simulation	Jeudi 20/04	Maxence

Prochaine Réunion : Jeudi 20 Avril 2023 pour une réunion d'avancement.

6.1.4 Réunion du 21 Avril 2023

Compte-rendu n°04
Réunion du Mercredi 21 Avril 2023

Type de la réunion : Réunion d'avancement	Rédigé le 21 Avril 2023 Lieu : Réalisée sur Discord
Heure de début : 16h45	Heure de fin : 17h30

Objet : Réunion d'avancement

Présents :	
BOUCHADEL Maxence	GONCALVES Florian
JEANJACQUOT Thomas (secrétaire)	VATRY Etienne

Ordre du jour :

— Faire le point sur l'avancement de chacun

Échanges :

Nous avons commencé par faire un point sur l'avancement de chacun depuis la dernière réunion.

Florian a réussi à quadriller la France en 64 carrés et a attribué à chaque station le carré qui lui correspond. Il a également créé la fonction qui permet d'associer à des coordonnées un carré. Il va à présent modifié la fonction find_nearest_station pour qu'elle utilise les carrés

Thomas a réalisé les fonctions print_itinerary qui permet d'afficher l'itinéraire que va parcourir une voiture entre une ville A et une ville B, en demandant plusieurs paramètres comme le modèle, la batterie actuelle et la batterie minimum. Il a également réalisé la fonction print_itinerary_by_time qui permet d'afficher en plus la position des voitures toutes les X minutes. Il doit à présent rajouter quelques fonctionnalités supplémentaires comme les temps de pause ou bien le temps d'attente maximum.

Etienne a commencé la réalisation de la simulation, il peut à présent générer une liste aléatoire de voiture avec différents statuts, permettant de sauvegarder les étapes de chaque voiture (modèle, batterie, batterie min, ville départ, ville arrivée). Il doit ajouter l'utilisation de la fonction itinerary à son programme pour calculer l'itinéraire pour chaque voiture. Il attend donc que la fonction soit optimisé afin de faire cet ajout.

1

To Do List :

Description	Délai	Personne
Faire les différentes modifications détaillé en réunion	Dimanche 30/04	Thomas
Commencer le rapport et l'affichage final 30/04	Maxence	
Finir la fonction de carrés et l'utiliser sur les autres fonctions	Dimanche 30/04	Florian
Continuer la simulation	Dimanche 30/04	Etienne

6.1.5 Réunion du 17 Mai 2023

Compte-rendu n°06
Réunion du Mercredi 17 Mai 2023

Type de la réunion : Réunion d'avancement	Rédigé le 18 Mai 2023 Lieu : Réalisée dans le local du BDS
Heure de début : 13h	Heure de fin : 14h

Objet : Réunion d'avancement

Présents :	
BOUCHADEL Maxence (secrétaire)	GONCALVES Florian
JEANJACQUOT Thomas	VATRY Etienne

Ordre du jour :

- Mise au point sur l'avancement final du projet
- Avancement du rapport
- Test des fonctions
- Affichage du trajet/simulation

Échanges :

Nous avons commencé par mettre en commun ce que chacun avait fait depuis la dernière réunion.

Maxence a parlé du rapport, celui-ci a bien avancé, il manque juste les parties de chacun et les conclusions personnelles (état de l'art terminé).

Thomas a montré les différents tests qu'il avait implémenté, il manque seulement les test pour les fonctions de la simulation.

Etienne a paufiné la simulation est a améliorer l'affichage du trajet dans le terminal (barre de chargement pour la simulation, mise en forme du trajet)

Florian a parlé de l'affichage, et trouve cela compliqué mais pense que nous pouvons tenter d'en faire un avant la deadline du projet.

To Do List :

Description	Délai	Personne
Chacun doit faire sa conclusion personnelle	Vendredi 19/05	Tout le monde
Faire les test de la simulation	Mercredi 24/05	Thomas
Afficher un menu dans le terminal pour sélectionner si on veut une simulation ou un trajet	Mercredi 24/05	Etienne
Finir le rapport, conclusion, intégration des images et de l'explication de code de chacun	Mercredi 24/05	Maxence
Relire le rapport et faire un affichage pour notre programme	Mercredi 24/05	Florian

Prochaine Réunion : Mercredi 24 Mai 2023 pour une réunion finale