**LIVRABLE 2 :**

**INTRODUCTION**

Dans ce livrable, nous allons expliquer tout ce qui n’a pas été précisé dans les commentaires du code. Pour mieux comprendre le fonctionnement de ce dernier, nous vous invitons à vous référer directement aux commentaires intégrés dans les fichiers sources.

**Création des fichiers TXT**

Les données présentes dans les fichiers texte ne sont pas celles issues du fichier Excel fourni initialement. Elles proviennent de différentes bases de données que nous avons jugées pertinentes pour la réalisation du projet.

Nous avons trouvé sur GitHub des données concernant les connexions et les stations, réparties dans un fichier Excel distinct, sans les coordonnées géographiques (longitudes et latitudes), mais avec une définition claire des connexions. Nous avons choisi cette base de données car elle était complète, sans connexions dupliquées, et structurée de manière simple pour être facilement exploitable en code (avec une indication sur l’orientation des connexions).

L’extraction de ces données a d’abord été réalisée avec Python, avec l’aide d’Alexandre, afin d’éviter l’utilisation de nouvelles bibliothèques que nous ne maîtrisions pas et de structurer précisément les informations souhaitées. Par la suite, pour ajouter les coordonnées géographiques, nous avons dû recourir à une autre source. Une première version des coordonnées a été ajoutée grâce à un algorithme en C#, puis complétée à l’aide d’un autre fichier Excel. Ces coordonnées seront nécessaires pour produire une visualisation graphique plus précise du graphe.

**CLASSE DRAWING**

Pour cette classe, nous avons décidé de modifier l’algorithme d’affichage du graphe. Pour rappel, dans le livrable 1, l’affichage reposait sur un système complexe simulant des forces d’attraction (entre nœuds connectés) et de répulsion (en fonction du degré de connexion), afin de répartir les nœuds de façon lisible et centrée. Bien que ce système soit intéressant, il ne garantissait pas une bonne lisibilité du graphe.

Nous avons jugé plus pertinent et plus simple d’afficher les nœuds en fonction de leur position géographique réelle (longitude et latitude). Pour cela, nous avons mis en place des calculs simples permettant d’identifier les minimums et maximums des coordonnées afin de cadrer au mieux les nœuds dans notre canvas.

Le dessin des flèches, cercles et textes ne présente pas de particularité technique majeure, si ce n’est le fait que les flèches ont été légèrement décalées pour ne pas chevaucher les cercles représentant les stations, assurant ainsi une meilleure lisibilité.

La grande majorité de ce code a été écrite sans l’aide de ChatGPT, en réutilisant certains éléments issus du code du livrable 1. L’intelligence artificielle nous a principalement servi à résoudre des erreurs ponctuelles ou à obtenir des suggestions d’approche. Par exemple, nous avons utilisé des prompts du type « Quel est le problème dans ce code ? » ou « Quelles sont les idées pour afficher un graphe orienté ? », mais jamais pour générer le code complet.

Pour voir les détails de l’image, ouvrir celui crée après le fonctionnement du code

(Image du graphe crée dans la prochaine page)

**Base des données/ SQL**

Nous avons utilisé une IA lorsque nous n’arrivions vraiment pas à corriger certaines fonctions, ainsi que pour réaliser certains affichages plus rapidement.

Nous sommes partis d’explications trouvées sur des forums pour comprendre les interactions entre SQL et C#. Nous nous sommes donc appuyés sur certains exemples de fonctions que nous avions vus.

Papalote volando en el cielo

El contenido generado por IA puede ser incorrecto.

<https://data.ratp.fr/explore/?sort=modified>

<https://data.iledefrance-mobilites.fr/explore/?disjunctive.theme&disjunctive.keyword&disjunctive.publisher&disjunctive.modified&disjunctive.features&disjunctive.language&sort=modified>

<https://github.com/vbarbaresi/MetroGit>

<https://chatgpt.com/?model=gpt-4o>

Notre Lien GITHUB :

<https://github.com/Assmodee/Liv-In-Paris>

*En cas des problèmes, venez contacter le S.A.V. au numéro :+33 0768243263*

**