Rapport de Projet

1. Répartition des tâches

Pour mener à bien ce projet, nous avons adopté pour une organisation collaborative avec des responsabilités bien définies et une approche modulaire.

Dans un premier temps, nous avons analysé ensemble le sujet et les consignes associées. Pour clarifier nos idées, nous avons identifié les grandes étapes du projet et les éléments clés en les notant sur une feuille blanche. Cela nous a permis de structurer nos réflexions et de visualiser les priorités.

La répartition des tâches a été réalisée en fonction des compétences et des intérêts de chacun. Sam a pris en charge le codage du script Shell, car il dispose des outils nécessaires pour exécuter ce type de programme sur son ordinateur, et l'idée d'apprendre un nouveau langage l'attirait particulièrement. Jules et Hugo, de leur côté, se sont concentrés sur le langage C, qu'ils maîtrisent mieux. Ensemble, ils ont implémenté les fonctions de base de l'AVL ainsi que le fichier header.

Sam a également assumé la responsabilité du fichier **main.c**, en guidant Jules et Hugo dans le traitement des fichiers triés par le script Shell et en contribuant à sa rédaction. Par ailleurs, Hugo s'est chargé de rédiger le fichier **ReadMe**, tandis que Jules s'est occupé des commentaires dans le code C pour assurer une documentation claire. Sam a ensuite recherché et corrigé d'éventuelles erreurs dans le code Shell, tandis que nous avons collaboré tous les trois pour identifier et résoudre les problèmes rencontrés en C.

Le script Shell s'est avéré un peu difficile à maîtriser au départ, notamment pour le tri des données, car c'est un langage nouveau pour nous. En revanche, le langage C a été plus accessible grâce à notre expérience antérieure avec les fonctions AVL.

Enfin, Discord a été un outil précieux pour coordonner notre travail. Nous avons utilisé cette plateforme chaque semaine pour faire le point sur nos codes respectifs, partager nos avancées en temps réel et nous entraider face aux éventuels obstacles.

2. Planning de réalisation

- **Semaine du 25/11 :**

  - Etude et compréhension des consignes du projet
  - Création du dossier et ajout des fichiers nécessaires dans GitHub

- **Semaine du 02/12 :**

  - Début du fichier en Shell :
    - Ajout de la fonction d'aide

- Vérification des paramètres entrés lors de l'exécution
- Compilation + vérification
- Création des dossiers 'tmp' & 'graphs'

- Début C :
  - Implémentation des fonctions d'un AVL dans le fichier station.c
  - Analyse du fichier temporaire envoyé par le Shell

- **Semaine du 09/12 :**

- Corrections minimes du code en Shell
- Traitement du fichier, trié par le Shell, en C
- Création du fichier header "station.h" pour la gestion de l'AVL (prototypes des fonctions et structures)

- **Semaine du 16/12 :**

- Création du fichier main.c :
  - Traitement des données triées par le Shell.
  - Création d'un fichier dans le dossier 'tests'
- Génération de graphiques
- Tests sur la robustesse du code et les éventuels problèmes liés au code
- Le code génère un message d'erreur en cas de surconsommation (stations où la charge dépasse la capacité)

3. Limitations fonctionnelles

Malgré nos efforts pour couvrir tous les aspects requis du projet, certaines limitations ont été identifiées :

1. Temps de traitement :
   - Le fichier de données CSV est volumineux (+150 Mo), imposant des contraintes de performance. Les traitements Shell et C peuvent être longs pour des requêtes étendues (par exemple, l'analyse de toutes les stations LV).
   - Avec GnuPlot, le temps de traitement pour afficher les graphiques est plus ou moins important en fonction des types de stations demandés.
2. Génération de graphiques :
   - Dans le cas des LV, les graphiques sont très denses à cause du nombre de stations, empêchant de voir précisément les détails de chaque station.
3. Fonctionnalité :
   - Il est recommandé que le programme s'exécute sur un environnement Linux pour fonctionner correctement car la compatibilité est limitée sur Windows en l'absence d'outils (Git, MakeFile, etc).