# Informations générales

*NB : Cette partie du projet peut être réalisée seul ou en équipe de deux (à la fois le code et le journal de bord).*

**Nom(s) du ou des coéquipiers** : Zakaria Gannoun

**URL du référentiel sur GitHub** : https://github.com/ZakaGn/VTI-heatmap

## Description du prototype

Afin de guider ou valider nos choix de technologies de l’étape 1, nous allons développer un prototype (ou « proof of concept ») de l’application répondant au besoin présenté dans l’énoncé. Ce prototype vise à réduire le plus possible la quantité de développement requis tout en permettant d’expérimenter avec les technologies clés, c’est-à-dire celles dont on est le moins certain.

En se basant sur les résultats de l’analyse préliminaire, on veut que le prototype permette au moins :

* De lire et filtrer des données au format GTFS (provenant du fichier zip avec les données statiques)
  + Que ces données proviennent de l’API en temps réel ou seulement des données statiques, l’important est d’expérimenter avec la librairie choisie pour manipuler ces données afin de déterminer ce qu’elle peut (ou ne peut pas) faire.
* De calculer la valeur de la métrique de qualité (possiblement simplifiée en retirant les métros par exemple) pour une zone fixée d’avance (plus petite que « l’entièreté de l’ile de Montréal)
  + Bien que réduire la zone ou les plages horaires qui seront analysées ne réduit pas vraiment la quantité de code qui sera requise, ça simplifiera au moins la production de tests pour valider que les calculs sont corrects.
* De produire une carte thermique animée à partir des valeurs calculées selon la métrique de qualité
  + Ici, plusieurs options s’offrent à nous, mais la seule exigence présente dans l’énoncé du besoin est que la carte soit animée (interactive est seulement un plus).
  + Ce serait donc acceptable de choisir l’option la plus simple (par exemple MatPlotLib), mais personnellement dans ce cas je profiterais du prototype pour expérimenter avec une librairie plus complexe mais avec plus de potentiel futur (comme Bokeh), justement pour déterminer si ça en vaut la peine ou non.
  + Ceci dit, même si on choisit une librairie plus puissante que nécessaire, il suffit de générer un *heatmap* animé pour atteindre nos objectifs du prototype.

## Table des matières

[Informations générales 1](#_Toc144879183)

[Sprint #1 :](#_Toc144879184) 11 Septembre au 24 Septembre

# Sprint #1 : *11 Septembre* au *24 Septembre*

## Objectifs

Bien qu’il soit difficile de prédire le temps que prendra le développement des différentes fonctionnalités lorsqu’on utilise des technologies inconnues, il est important de se fixer des objectifs à atteindre à chaque sprint afin de planifier l’avancement du projet. Si vous ne parvenez pas à atteindre tous vos objectifs au cours d’un sprint, vous pourrez les repousser au prochain.

**Listez vos objectifs pour ce sprint (en ordre de priorité) :**

1. Collecte de données :
   1. Télécharger les données GTFS (General Transit Feed Specification) fournies par la STM depuis leur site web. Ces données contiennent des informations sur les arrêts de bus, les horaires, les itinéraires, etc.
   2. Voir la structure des données GTFS pour comprendre comment extraire les informations nécessaires.
2. Prétraitement des données :
   1. Nettoyer et traiter les données GTFS pour les rendre exploitables. Extraire les coordonnées GPS des arrêts de bus et des stations de métro, ainsi que les horaires de passage.
3. Création de la grille spatiale :
   1. Diviser l'île de Montréal en blocs de 100m x 100m. Pour chaque bloc, déterminer ses coordonnées centrales.
4. Boucle temporelle :
   1. Commencer une boucle qui itérera sur les tranches de 15 minutes tout au long de la journée de 24 heures.
5. Calcul de la qualité du service pour chaque bloc :
   1. Pour chaque bloc de 100m x 100m, effectuer les calculs suivants :
      1. Calculer le nombre total de passages d'autobus différents pour les arrêts situés dans ce bloc pendant la période de 15 minutes en cours.
      2. Additionner le nombre total de passages de métro pour toutes les stations situées dans le super-bloc entourant le bloc central (pour couvrir les blocs voisins).
6. Création de la carte thermique :
   1. Utiliser les résultats des calculs pour attribuer une couleur à chaque bloc de la grille en fonction de la qualité du service. Par exemple, utiliser une échelle de couleurs allant du rouge (mauvaise qualité) au vert (bonne qualité).
7. Animation :
   1. À chaque itération de la boucle temporelle, sauvegarder la carte thermique générée sous forme d'image statique.
   2. Une fois toutes les images statiques générées pour une journée complète, créer une animation en utilisant un logiciel d'édition vidéo ou un script pour produire un fichier GIF ou une séquence d'images animées.
8. Visualisation :
   1. visualiser l'animation pour voir comment la qualité du service évolue tout au long de la journée.

## Apprentissages effectués

Quelles ressources (sites web, vidéos, forums, etc.) avez-vous consultées au cours de ce sprint pour apprendre à utiliser les technologies choisies?  
Résumez en quelques points ce que vous avez appris (les éléments clés), puis assignez un score entre 1 et 5 à chaque ressource que vous avez consultée (et justifiez brièvement pourquoi).

|  |  |  |  |
| --- | --- | --- | --- |
| **Ressource consultée (URL)** | **Ressource consultée (URL)** | **Résumé des éléments clés appris** | **Évaluation de l’utilité (score de 1 à 5)** |
| **https://www.youtube.com/watch?v=DgLYeR56iNk** | * Python requests * Python zipFile * Python io * os * dict | * Éléments de base de Python pour télécharger des fiches d'une URL * Éléments de base de Python pour extraire des fiches .zip * Éléments de base de Python pour ouvrir des fiches | 5/5  Ces éléments de base suffisent pour accomplir cette tâche. |
| **Pandas**  **https://pandas.pydata.org/docs/user\_guide/index.html** | * read\_csv * set\_option * DataFrame * to\_datetime | * Gérer de gros fichiers CSV sans base de données | 5/5  Impressionnant, dans les fichiers csv très longs cela réduit le temps d'exécution à 90%, très utile pour le filtrage |
| **Pickle**  **https://docs.python.org/3.11/library/pickle.html** | * Dump * load | * Sérialiser et désérialiser une structure d'objet Python | 5/5  J'ai gagné beaucoup de temps en enregistrant les variables dans des fichiers |
| **Tqdm**  **https://github.com/tqdm/tqdm** | * progressbar | * Afficher la barre de progression pour les processus qui prennent beaucoup de temps | 5/5  Très utile pour calculer le temps des processus |
| **Numpy**  **https://numpy.org/doc/stable/reference/** | * Arange * zeros * radians | * Calcules numeriques et complexes dans un temps très court | 5/5  Très utile, ajoutant la prise en charge de grands tableaux et matrices multidimensionnels, ainsi qu'une large collection de fonctions mathématiques de haut niveau pour fonctionner sur ces tableaux. |
| **Seaborn**  **https://seaborn.pydata.org/** | * heatmap | * Générer une carte thermique à partir des données | 5/5  Il est très simple à utiliser et donne de très bons résultats. |
| **Shutil**  **https://docs.python.org/3/library/shutil.html** | * rmtree | * Supprimer plusieurs répertoires et leur contenu | 5/5  Très utile, même sous Linux |
| **Imageio**  **https://pypi.org/project/imageio/** | * Imread * mimsave | * Lire des images et créer une image à partir de plusieurs images | 5/5  Très utile, surtout parce que travailler avec des images est très complexe. |
| **Matplotlib**  **https://matplotlib.org/** | * figure * heatmap * title * xlabel * ylabel * xticks * yticks * invert\_yaxis * savefig * close | * Créer des cartes thermiques, très faciles et simples à utiliser | 5/5  Capable de créer des cartes thermiques complexes en écrivant peu de code |

## Fonctionnalités développées

Qu’avez-vous réussi à accomplir dans votre projet au cours de ce sprint?  
Mentionnez les fonctionnalités que vous avez implémentées même si elles ne sont pas encore entièrement fonctionnelles ou intégrées au reste du prototype.   
Vous pouvez regrouper plusieurs fonctionnalités connexes sur une seule ligne pour simplifier la lecture du tableau.

|  |  |  |
| --- | --- | --- |
| **Fonctionnalité(s) développée(s)** | **Technologie(s) utilisée(s)** | **Difficultés rencontrées** |
| * + Télécharger les données GTFS | * Python requests | * Rien |
| * + Extraire les fichie zip | * Python zipFile | * Rien |
| * + Lire des fichiers csv | * Pandas | * Moyenne, les fichiers sont tres grandes |
| * + Filtrage des données | * Pandas | * Moyenne, ce n’est pas facil de comprendre la structure des donnees |
| * + Joindre des données | * Pandas | * Moyenne |
| * + Calcules complexes de distance et lattitude | * manuellement | * Considérable, les calcules no sont pas facil de comprendre |
| * + Progressbar dans un terminal | * tqmd | * Considérable, l’utilisation de cette bibliothèque est un peu complexe |
| * + Calculer la qualité de service | * manuellement | * Très élevé, de nombreux calculs à effectuer en même temps et qui touchent différents domaines |
| * + Creation d’une carte thermique | * matplotlib | * Moyen, au début cela semble très complexe, mais en étudiant un peu cela s'est avéré assez simple. |

## Changements ou ajouts aux technologies choisies

Si au cours de ce sprint vous avez décidé de changer vos choix de technologies initiaux, ou si vous avez réalisé qu’il vous manquait des technologies utiles, listez-les ici :

* **Nouvelle technologie choisie 1** : J'ai changé de Folium vers Plotly et Seaborn, car je ne parviens pas à installer la dernière version de Folium. La dernière version est nécessaire pour implémenter toutes les fonctionnalités, comme par exemple folium.CustomPane()