GTI525 - LIVRABLE 1

VéloFacile

Équipe 02

Bouder Léo, BOUL66340201
Castonguay Simon, CASS02119909
Feghali Joseph, FEGJ82030201

Date de remise : *09/06/2025*

Table des matières

Introduction

Dans la ville de Montréal où la densité urbaine et les enjeux environnementaux continuent de croître, le vélo se présente comme un moyen de transport alternatif accessible et efficace. La popularité de ce mode de transport rend nécessaire le développement d'outils permettant aux citoyens de mieux repérer les pistes cyclables accessibles dans la ville. Dans cette optique, l'application VéloFacile a été conçue. Il s'agit d'une application web interactive qui permet aux utilisateurs de visualiser les pistes cyclables sur une carte, de consulter les statistiques des différents compteurs et d'obtenir la location de divers points d'intérêts qui peuvent être utiles aux cyclistes. Le premier livrable de ce projet repose surtout sur le développement de l'interface du projet, soit sur l'architecture logicielle utilisée, la mise en place des filtres d'informations et sur les outils utilisés pour l'esthétique de l'application.

R1. Architecture et organisation logicielle

Nous avons choisi de construire l'application utilisant Vite et Vue.js en raison de notre connaissance individuelle de ce framework, ce qui rendrait le développement plus facile. Considérant ce choix, nous utilisons aussi Vuetify pour avoir accès à des composants pratiques, rapides à implémenter et visuellement adaptable à nos besoins.

Pour rendre l'application plus facile à maintenir et à évoluer, nous avons décidé de bien implémenter Vue.js avec l'utilisation de vues et de composants pour construire nos pages. Les vues regroupent les composants nécessaires pour former nos pages et les composants contiennent toute la logique interne qui leur permet de performer leurs tâches directement. Cela permet de séparer les responsabilités proprement et de ne pas avoir de fichiers énormes et difficiles à maintenir.

L'application est donc une SPA (single page application) que nous accédons à partir d'un fichier. Les autres pages sont "rendered" dans cette seule page à l'aide d'un routeur paramétré avec nos différentes pages. Considérant notre utilisation de Vue, nous avons utilisé le routeur "vue-router" pour s'occuper de cet aspect de l'application. Ceci permet d'ajouter de nouvelles pages facilement et de rendre la navigation plus facile.

Pour garder le style le plus consistant que possible à travers l'application, nous essayons d'ajouter les déclarations CSS le plus haut possible. Au besoin, nous ajoutons du code CSS à des composants, pages ou tags HTML spécifiques pour n'affecter que ces composants et non le reste de l'application.

Pour l'instant, nous avons les fichiers de données directement dans le code source front-end qui sont lu quand la page appropriée est ouverte.

Finalement, nous avons implémenté un "store" qui nous permet de centraliser certaines informations qui seront partagées à travers divers composants de l'application. Dans notre cas, le menu qui sert à filtrer certaines pages est commun à toutes ces pages. Ceci permet de ne pas avoir à recharger la carte trop souvent. Les choix faits dans ce menu sont donc

enregistrés dans le “store” pour que les pages contenant les données puissent récupérer l’information au besoin.

R2. Implémentation du triage des données

Pour trier les informations des pages Statistiques et Points d’intérêt, nous avons implémenté des filtres dynamiques qui permettent de trier l’information selon l’année, l’arrondissement ou grâce à une barre de recherche. Pour centraliser les valeurs sélectionnées dans le menu gauche à travers l’interface, nous avons opté pour l’utilisation d’un “store” centralisé à l’aide de la fonction “reactive” de Vue.

Ce store nous permet de définir des états réactifs qui sont accessibles par tous les composants de l’application. Par exemple, lorsqu’on choisit une année et un arrondissement dans le menu gauche de la page Statistiques, les valeurs sont stockées dans le store sous le nom “year” et “arrondissement”. Par la suite, on peut faire appel à ces variables dans la page Statistique pour filtrer l’information

La réaction au changement de valeur est effectuée grâce à des propriétés calculées (*computed*) qui permettent de filtrer les données lors d’un changement de valeurs. Toujours dans le même exemple, lorsqu’une année et un arrondissement sont choisis, on peut accéder aux variables “year” et “arrondissement” dans la page Statistiques pour filtrer les données selon les champs choisis à l’aide de “filteredData” qui applique ces critères pour afficher les valeurs demandées. Ce processus est réutilisable pour l’ensemble des filtres dans l’application grâce à la centralisation amenée par le store.

R3. Couleurs et thème

Pour ce qui est du choix des couleurs et du thème global de notre application web, nous avons opté pour un thème plutôt verdoyant et donc avec différentes teintes vertes. La raison pour laquelle nous avons décidé de choisir ces couleurs est en lien direct avec le sujet de notre application web, les vélos. En effet, nous voulions des couleurs qui font référence ou qui rappellent un aspect verdure et écologique permettant de faire écho à l’environnement et l’utilisation des vélos au quotidien.
Les outils utilisés pour le choix de ces couleurs et du contraste sont les suivants :

- https://webaim.org/resources/contrastchecker/ : Pour vérifier le contraste de nos couleurs utilisées sur l’application.
- https://material-foundation.github.io/material-theme-builder/ : Pour donner des idées de thème avec des palettes de couleurs préétablies. C’est avec cet outil que nous avons notamment remarqué une palette de couleurs tournée vers le vert.

La couleur principale de l’application web est : #C5E1A5 (teinte de vert). Elle est utilisée afin de donner un aspect chaleureux et accueillant sur l’application. C’est une teinte relativement claire, ce qui permet de faire ressortir le texte inscrit. Nous l’avons notamment utilisé pour les titres comme le header, footer, champs de recherche etc…

Cela permet à la fois de faire ressortir les champs clairement les noms des différentes sections tout en étant en adéquation avec le thème de l'application.

R4. Subdivision des tâches

En ce qui concerne la répartition des tâches, nous avons décidé de diviser le travail en deux sections :

1. Le traitement et affichage de données,
2. L'affichage général et le menu.

Une personne s'est occupée d'implémenter les détails d'affichage du site, donc chaque pages et le menu. Une autre s'est occupée de récupérer les données dans les fichiers csv et de les afficher dans un tableau. La dernière faisait le lien entre ces deux parties, soit de lier les contrôles du menu de filtres et les données affichées.

Chaque membre de l'équipe tenait le reste de l'équipe au courant de ses avancements et demandait de l'aide au besoin. Ceci nous a donc aidé à parcourir les parties des autres et de se familiariser avec l'application au complet à la place de seulement notre code.

Conclusion

Pour conclure, le premier livrable de l'application VéloFacile permet de fournir une meilleure accessibilité à l'information pour les cyclistes de la ville. Son interface intuitif et ses filtres dynamiques permettent aux utilisateurs d'accéder aux informations importantes dans leur arrondissement.

Cette première version a permis de développer la base du logiciel, que ce soit au niveau de son architecture ou de l'intégration de ses premières données. Pour les prochains livrables, de nouvelles fonctionnalités seront ajoutées et d'avantages de données seront intégrées pour fournir le plus d'informations possibles aux utilisateurs.

Finalement, l'objectif de ce projet est de proposer aux cyclistes de Montréal un outil évolutif et fiable pour faciliter l'utilisation du vélo à tous les cyclistes de Montréal.