APPLICATION WEB VÉLO+

par
KAMIL TALEB - TALK29089800
SARA AISSAT - AISS22529904
ANTOINE BOUCHER - BOU12039801
NAZIM FERRAT - FERN25019709

RAPPORT DE PROJET PRÉSENTÉ À MARCOS DIAS DE ASSUNCAO

MONTRÉAL, LE VENDREDI 03 JUIN 2022

ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

GTI525-01

TABLE DES MATIÈRES

LIVRABLE 1 DU LABORATOIRE VÉLO+

L'objectif principal du cours GTI525, Technologies de développement Internet, est d'en apprendre davantage sur la conception, le développement et le déploiement d'applications web dynamiques. Cela comprend le choix d'une architecture appropriée, la conception HTML ainsi que la programmation côté client et côté serveur. Le laboratoire central de ce cours est d'ailleurs la mise en pratique de tous ces concepts répartis sur 3 livrables.

Il s'agit de mettre en place une application web permettant aux utilisateurs de vélos de visualiser de différentes manières les données de comptage des pistes cyclables de la ville de Montréal. L'application comprendra également un deuxième affichage d'informations sur les points d'intérêt pertinents pour les cyclistes tels que les fontaines d'eau et les ateliers de réparation de vélos. Les livrables 2 et 3 viendront compléter l'offre de service par l'ajout de fonctionnalités supplémentaires.

Le premier livrable de ce laboratoire, quant à lui, se limite à l'élaboration de la structure statique du site, ainsi qu'à l'affichage des données des compteurs de vélos et des points d'intérêt. La liste des compteurs de vélos, des fontaines à eau et leur géolocalisation seront extraits à partir de fichiers CSV.

En effet, ce rapport traitera des principales décisions prises lors de la réalisation de cette application web. Il sera d'abord question du choix de cadriciel accompagné d'une présentation de l'architecture logicielle et de l'organisation avec le rôle des différents éléments du code JavaScript. Ensuite, une brève description de l'algorithme de tri utilisé pour les différentes possibilités d'affichage des comptages de vélos suivra. Enfin, la répartition des tâches au sein de l'équipe sera abordée pour détailler l'ensemble des réalisations effectuées par chacun des membres.

L’architecture logicielle et l’organisation

Dans le cadre de ce laboratoire, les équipes avaient la responsabilité de choisir parmi les différents cadriciels et librairies existants, ceux qui seraient au cœur de leur implémentation. En ce qui nous concerne, Angular et ReactJS ont été nos principaux choix. Ce sont deux cadriciels très utilisés pour le développement web et il était donc pertinent pour nous de nous y familiariser. Suite à une délibération collective sur le framework qui serait le plus facile à maîtriser, car nous sommes tous plus ou moins débutants en développement web, c'est sur ReactJS que notre choix s'est arrêté. Étant le cadriciel qui nous semblait le plus utilisé à travers le monde du développement web, nous aurions forcément accès à un plus large éventail de sources d'information au cours de notre apprentissage. En outre, notre responsable de laboratoire est plus à l'aise avec cet environnement de développement, ce qui nous assure une aide plus ciblée en cas de problématique lors de l’implémentation.

Lorsque nous travaillons avec ReactJS, nous utilisons des classes JSX. C'est un ajout à la syntaxe JavaScript qui est un mélange entre HTML et JavaScript. Nous n'avons donc pas de classes HTML à proprement dit, mais plutôt une intégration du code HTML dans le JavaScript. Enfin, en plus des classes JSX, nous avons des classes CSS qui nous permettent de façonner notre page selon le visuel souhaité.

Notre application web est subdivisée en plusieurs composants, eux-mêmes fractionnés en composants plus petits, comme le montre la figure 1.1.

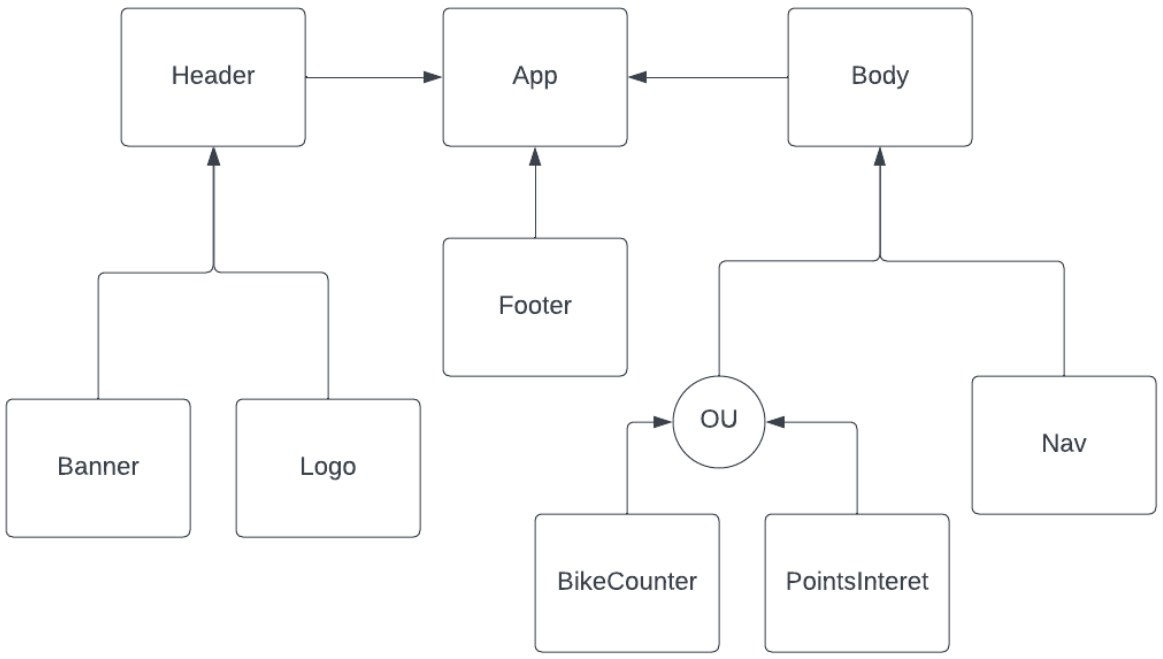


Figure 1.1: Aperçu du schéma de l’application web

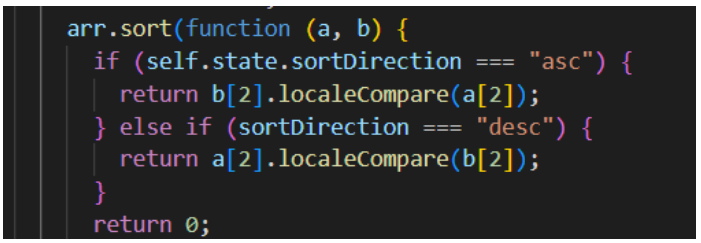
Tout d'abord, notre application contient un header qui contient à son tour un logo et une bannière. Ensuite nous avons un Body qui contient le menu, soit la Nav, ainsi que les classes *BikeCounter* et *PointsInteret*. Le composant Nav, possède deux boutons : Comptages de vélos, et points d'intérêt. Ces boutons permettent d'interchanger, entre les deux composantes, l'affichage de la section principale du Body selon le choix de l'utilisateur. Notre application possède aussi une classe CSVParser, qui permet de parse nos fichiers CSV grâce au framework papaparse. Cela nous permet d'accéder aux informations des fichiers CSV de manière plus intuitive dans le code afin de les afficher dans les tableaux de nos deux composants, *BikeCounter* et *PointsInteret*.

Dans nos classes *BikeCounters* et *PointInterets*, il y a différentes fonctions pour chacun des attributs, ces fonctions permettent de filtrer de manière ascendante ou descendante selon le choix de l'utilisateur. Ces deux classes contiennent aussi une fonction initialisation qui permet d'initialiser le header avec les différents attributs ainsi que de remplir le tableau avec les données du fichier CSV qui a été parse précédemment. Dans la classe Body, il y a deux fonctions qui se nomment respectivement *setDisplayBikeCounte*r et *setDisplayPointsInteret*, les appels de ces fonctions permettent d'afficher sur l'écran de l'utilisateur les données qu'il souhaite observer.

L'algorithme de tri

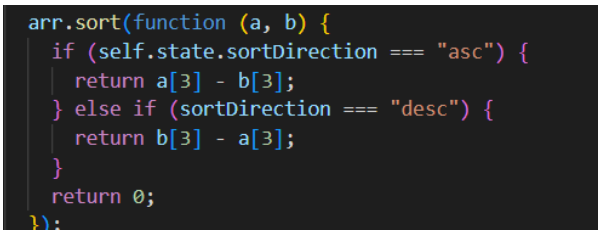
Pour le tri des différentes valeurs contenues dans le tableau, nous avons utilisé la fonction .sort(), cette fonction utilise différents types de tri algorithmique, puisque dans notre cas, nous utilisons le V8 engine de JavaScript, cela signifie que nos colonnes seront soient triées avec un *quicksort* (tri rapide) ou avec un *insertion sort* (tri par insertion). Puisque dans notre cas, notre tableau contient énormément d'informations, le Quicksort sera priorisé comme démontré dans l'article *Sorting arrays in JavaScript* par *Scott Robinson*. Le tri rapide (*QuickSort*) est un algorithme utilisant un pivot qui partitionne les données reçues en sous-partie ordonnée, faisant ainsi du pivot la plus grande valeur de la sous-partie de gauche et la plus petite valeur de la partie de droite. Cette étape est ensuite répétée de façon récursive en parcourant chacune des sous-parties des tableaux et en plaçant le pivot au bon endroit jusqu'à ce que toutes les données soient placées dans un ordre croissant ou spécifié.

La fonction *.sort()* reçoit en paramètre une autre fonction de comparaison qui a en paramètre deux valeurs a et b, et qui retourne une valeur qui va déterminer l'ordre du tri. Dans notre cas, nous avons ajouté une condition pour savoir si l'utilisateur veut que le tri se fasse de manière ascendante ou descendante. Une autre exigence à respecter est le fait que nous pouvons trier un tableau de valeurs numériques ou alphabétiques. Dans le cas de valeurs alphabétiques, nous devons aussi utiliser une fonction de JavaScript nommée *.localeCompare* qui permet de vérifier si cette chaîne de caractères se trouve avant ou après la chaîne de caractères comparée comme démontré dans la figure 2.1 présentée ci-dessous.



```
arr.sort(function (a, b) {
  if (self.state.sortDirection === "asc") {
    return b[2].localeCompare(a[2]);
  } else if (sortDirection === "desc") {
    return a[2].localeCompare(b[2]);
  }
  return 0;
```

Figure 2.1: comparaison alphabétique des chaînes de caractères



```
arr.sort(function (a, b) {
  if (self.state.sortDirection === "asc") {
    return a[3] - b[3];
  } else if (sortDirection === "desc") {
    return b[3] - a[3];
  }
  return 0;
});
```

Figure 2.2: comparaison numérique pour la fonction de comparaison du array.sort()

La répartition des tâches

Étant donné que le laboratoire de ce cours s'étend sur l'entièreté de la session, notre équipe a préconisé la répartition des tâches en groupe de 2. De cette façon, nous étions moins susceptibles de nous retrouver face à une implémentation faite par un membre qui ne fait plus partie du projet. Un peu comme nous l'avons vu dans le cadre du cours LOG210, Analyse et conception de logiciels, nous avons opté pour un *facteur bus* minimal de 2.

Tout d'abord, la gestion de ce premier livrable du laboratoire a été confiée à Sara, en raison de la facilité qu'elle aurait à joindre les membres de l'équipe. Ayant un second travail d'équipe avec chaque membre, cela lui permettait d'avoir un retour supplémentaire de la part de chacun. Ainsi, elle avait pour rôle de s'assurer que chaque élément de l'énoncé soit respecté avant la remise. Kamil et Nazim quant à eux étaient principalement responsables de l'aspect visuel de l'application web, tout en assurant un code propre et lisible. Antoine était chargé d'appliquer les bonnes pratiques de ***React***, le cadriciel choisi par l'équipe. Il était responsable d'effectuer des recherches complémentaires et de débriefer l'équipe à ce sujet.

En ce qui concerne les tâches, ces dernières ont d'abord été divisées en deux sections principales, à savoir le développement de la structure de base de l'application web, et la lecture des fichiers CSV. La première tâche a été confiée à Sara et Kamil, et la seconde à Antoine et Nazim. Une fois les premières étapes achevées, l'équipe s'est réunie pour se répartir à nouveau les tâches. Nous en étions au contenu des différents composants. La gestion de l'en-tête, du pied de page et de l'affichage des menus a été assignée à Sara et Kamil, et l'affichage du contenu des fichiers CSV dans les tableaux a été confié à Antoine et

Nazim, conformément à leur tâche précédente. La majorité des tâches ayant été réalisées, l'équipe a travaillé de concert pour finaliser les derniers éléments, à savoir l'algorithme de tri et les ajouts visuels le concernant. Kamil et Antoine ont pris en charge ces tâches tandis que Nazim et Sara ont entamé la rédaction du rapport.

Enfin, le premier livrable touche à sa fin, et l'équipe a atteint chacun des objectifs fixés. Le travail a été effectué en temps opportun et l'accent a été mis sur le soutien mutuel tout au long du projet. L'équipe s'est réunie régulièrement deux fois par semaine, soit le mercredi après-midi en plus de la période de laboratoire du vendredi. Cela nous a permis de favoriser la communication et de partager nos connaissances afin de mener à bien ce mandat.

CONCLUSION

En sommes, durant ce laboratoire, nous avons pu réaliser la mise en place du front-end d'une application web, qui sert à présenter les informations sur les compteurs de vélos ainsi que sur les points d'intérêts qui ont été extraites à partir de fichiers CSV.

Pour la réalisation de ce projet, nous avons décidé d'utiliser le cadriciel ReactJS. Tout d'abord, nous avons effectué la partie supérieure de la page nommée header, qui affiche le logo ainsi que la bannière, ensuite nous avons décidé de s'occuper du bas de page nommé footer, qui affiche les différentes informations sur les différents coéquipiers. De cette manière, nous avons commencé à apprendre l'utilisation de ReactJS, ainsi que la modélisation avec CSS, ensuite nous avons commencé à travailler sur le body qui était un peu plus dur à réaliser. Dans cette tâche, nous avons rencontré quelques difficultés comme le fait de changer l'affichage entre les compteurs de vélos ainsi que les points d'intérêt. Puisque nous étions des débutants en ReactJS, nous avions encore de la difficulté à comprendre comment fonctionnent les concepts de state et de props. Mais, grâce aux recherches effectuées par les collaborateurs sur ces différents éléments, nous avons réussi à implémenter l'affichage en parallèle de ces modules. Nous avons essayé de faire une application qui peut être facilement modifiée, car dans de prochaines itérations, nous aurons à appliquer différents changements, comme par exemple l'ajout d'un back-end, ou l'ajout de nouveaux types de points d'intérêts.

ANNEXE + LISTE DE RÉFÉRENCES

Shehroz Azam. Linux Hint. (2021). What is the difference between JS and JSX. Repéré à

https://linuxhint.com/what-is-difference-between-js-and-jsx/#:~:text=JS%20is%20simply%20a%20scripting,easier%20to%20understand%20for%20users.

Derek Austin. (2019). Sorts in 60 seconds: Speedy JavaScript Interview Answers on Sortng. Repéré à

https://medium.com/coding-at-dawn/sorts-in-60-seconds-speedy-javascript-interview-answers-on-sorting-acb72bdea8a2

Scott Robinson. (2020). Sorting arrays in JavaScript. Repéré à

https://stackabuse.com/sorting-arrays-in-javascript/

JavaScript Array sort: Sorting Array Elements

https://www.javascripttutorial.net/javascript-array-sort/#:~:text=The%20sort()%20method%20allows,first%20and%20largest%20value%20last

MDN contributors. (2022). String.prototype.localeCompare()

https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Global_Objects/String/localeCompare