Auteurs :

Bastin Julien, Duchêne François

& Raquet Damien

Date :

Le **** 25 aout 2017

BeerSearch

Professeur : Mr. Sebastian Gonzales

Travail réalisé dans le cadre du cours LSINF1212 « projet d'approfondissement en informatique »

Donné à l'Université Catholique de Louvain.

Table des matières :

Introduction

Dans le cadre du cours « projet d'approfondissement en informatique », il nous a été demandé de réfléchir à une application web qui pourrait être utile soit à titre personnel, soit pour aider des proches, des connaissances, voire une communauté, soit dans un but de citoyenneté participative.

Nous vous avions soumis trois propositions afin de valider l'application que nous allions développer. Comme celles-ci ont été acceptées, nous avons donc décidé de nous lancer dans la construction d'un site web de recherche de bières. Notre site devait permettre aux étudiants de voir quels bars disposent en stock de leur bière favorite, via un outil de recherche. Le second objectif était de leur permettre de de trouver aisément un bar dans les environs. Enfin, le site devait leur laisser l'occasion de réserver une bière de leur choix dans un bar, à condition de s'être créer un compte, et d'être connecté.

Nos motivations pour ce projet sont assez claires, en tant qu'étudiants. Nous apprécions faire la fête et boire un coup avec des amis. Or si la volonté est souvent présente il arrive de ne pas savoir où aller pour passer un bon moment. Cette application pourra nous aider, ainsi que bien d'autres étudiants, à ne plus autant hésiter sur notre destination.

Fonctionnalité :

Nous allons ici vous décrire les fonctionnalités que notre site peut accomplir. Les trois objectifs initialement prévus ont tout trois étés atteints. Ainsi, les personnes utilisant le site ont accès à la page d'accueil, la fonction de recherche des bières et la carte des environs. La commande de bières, quant à elle, est réservée aux utilisateurs connectés. Commençons par les fonctions accessibles à tous.

La page d'accueil n'a aucune utilité particulière, si ce n'est rendre le site plus présentable. La page de recherche permet de rechercher une bière particulière selon 3 critères : le goût de la bière, le degré d'alcool ou la région. En plus d'une recherche nominale, bien sûr. Elle renverra alors une liste des bières correspondant à votre recherche, ainsi que diverses informations à leur sujet, tel que le degré d'alcool ou le lieu de production. La fonction de localisation permet de localiser l'utilisateur, puis d'afficher sur une carte sa position actuelle, ainsi que les points de vente à proximité, avec leur nom.

Notre site permet également aux utilisateurs réguliers de se créer un compte afin de pouvoir commander différentes bières. La création d'un compte se fait assez simplement. Il faut remplir les champs avec les informations correspondantes. Les champs « adresse mail » et « mot de passe » sont les deux champs qui permettent d'identifier un utilisateur, ou du moins les deux champs nécessaires pour se connecter. Les utilisateurs peuvent se connecter et se déconnecter à tout moment.

Une fois connecté, l'utilisateur a accès aux fonctions suivantes.

D'abord, la gestion de compte. Les utilisateurs peuvent à tout moment modifier leurs informations de compte en cliquant sur le bouton « bonjour XXX ». Cela les redirige vers une page avec un premier aperçu de leur informations générales. Ils peuvent, en utilisant le menu de gauche, voir leur historique de commande, ou avoir une vue d'ensemble de leur données personnelles et les modifier à volonté. De plus, les utilisateurs qui le souhaitent peuvent supprimer leur compte.

Ensuite, la création de commande. Cela lui permet de créer une commande d'un nombre choisi de bières depuis différents points de vente. Il est tout à fait possible de réinitialiser sa commande. Une fois l'utilisateur satisfait de sa commande, il clique sur le bouton commander, et il reçoit un message confirmant sa commande si son solde est suffisant, sinon, il reçoit un message lui précisant le montant de sa commande, ainsi que le solde restant, et la commande n'est pas confirmée tant que l'utilisateur n'a pas renfloué son compte ou réduit sa note.

Choix des outils de conception :

Pour développer ce site, nous avons utilisés les différents outils ci-dessous.

Commençons par la base de données. Il nous a été demandé d'utiliser une base de données non relationnelle. Etant donné qu'un système nous a été présenté en cours, et que nous avons assez peu d'expérience en tant que développeur, nous avons décidé d'utiliser ledit système. Il s'agit de MongoDB. Pour notre site, nous avons utilisé la dernière version de MongoDB.

Pour ce qui est de la transition des données entre la bdd et les pages web, nous avons utilisé le langage javascript, et plus précisément le module mongoose. Nous avons également utilisé ce langage pour créer et gérer la partie serveur du site, à l'aide de Node.js et du module express. Ces deux modules nous ont facilité la tâche pour la partie code du site. Nous avons décidé d'utiliser javascript car il est particulièrement adapté aux applications web, bien que son utilisation ne s'y limite pas.

Les derniers outils à présenter sont html, css et Bootstrap. Ces trois outils nous ont permis de développer le rendu et les liens entre les différentes pages web. Html se charge de la partie des liens entre les différentes pages, ainsi que la structure des pages et des textes, tandis que css se charge de la mise en pages desdites structures et textes. Enfin, nous avons utilisé Bootstrap. Il s'agit d'un module javascript qui nous permet de lier les différentes pages javascript avec les pages html. Pour notre site, nous avons utilisé les versions html5/css3/Bootstrap 3, car il s'agit des versions les plus stables, à défaut d'être les plus récentes.

Nous avons également utilisé des fichiers de type « ejs » et non pas « html » pour créer les pages. Cela nous permet d'utiliser du javascript directement dans les pages html. Nous aurions pu utiliser angular à la place. Cependant bien qu'angular soit beaucoup plus poussé, et offre donc plus de possibilités, il est également plus difficile d'utilisation pour des novices. Les fichiers « ejs » étaient donc un bon compromis entre efficacité et simplicité.

La gestion des données :

Pour notre site, nous avons décidé d'utiliser un fichier json afin de remplir la bdd avec une liste de bière assez conséquente. En effet, le système de mongo permet de sauvegarder des fichier json directement dans la bdd, pour peu que la syntaxe du fichier soit correcte. Nous avons donc utilisé une liste de Wikipédia des bières belges, et avons inséré le fichier dans la bdd (après avoir transformé le fichier xtml en fichier json).

Comme dit précédemment, les données sont stockées dans une bdd à l'aide de MongoDB. Afin de pouvoir accéder, manipuler ou supprimer ses données, nous avons utilisé un module de node.js appelé mongoose. Mongoose fonctionne de la manière suivante : il faut créer à l'aide de classe javascript des objets qui correspondent aux données de la bdd, puis créer une connexion avec la bdd, et enfin lier les différences objets à la bdd.

Dans notre application, nous avons trois classes, situées dans le dossier 'model'. Ces classes représentent les bières, les points de vente et les utilisateurs. Les informations de connexion à la bdd sont quant à elles dans la classe session.js. Ainsi, chaque fois que nous voudrons utiliser des données de la bdd, nous devrons créer un nouvel objet, ou modifier les données d'un objet déjà existant, puis appliquer les requêtes correspondantes grâce aux données de ces objets dans la bdd. Nous avons choisi d'utiliser mongoose car il s'agit de l'outil le plus répandu. Il est à noter que les modifications de bdd se font dans les méthodes router.post, puisque ce sont ces méthodes qui envoient les réponses aux demandes faites par l'utilisateur.

Petite précision au sujet du modèle « pdv ». Il était initialement prévu dans le but de manipuler les informations de la fonction de géolocalisation. Cependant, cela nous aurait demandé de repenser et recréer la quasi-totalité du fonctionnement de cette partie du site. Nous avons décidé de faire l'impasse dessus, pour nous concentrer sur l'achèvement du site, avant de retravailler les parties fonctionnelles.

Partie serveur :

Parlons à présent de la partie serveur de notre site. Tout se passe dans deux classes clés. La première est la classe server.js, et la seconde est la classe router.js.

Commençons par la classe server.js. Celle-ci effectue deux opérations au démarrage du serveur. D'abord elle ouvre la base de données, ensuite, elle connecte les différentes adresses localhost.

La classe router.js elle contient toutes les fonctions nécessaires à la navigation entre les pages. De même, c'est dans cette classe que sont définies les fonctions permettant le

transfert des données, que ce soit en réception des informations envoyées par les utilisateurs ou les informations à renvoyer pour qu'elle soit affichées par les fichiers « ejs » décrit plus bas. Globalement, le fichier router contient deux grandes catégories de fonctions : les fonctions get et les fonctions post.

Commençons par les méthodes get. Celles-ci servent à s'assurer que chaque page web est bien liée à une url spécifique. Par exemple, lorsqu'un utilisateur change de page à l'aide de la 'navbar', ou par un autre moyen une fonction get est appelée. Celle-ci va regarder quelle est l'url à laquelle l'utilisateur veut se rendre, et va renvoyer la page ejs correspondante qu'il faut afficher, et avec quelles informations.

Voyons maintenant les fonctions post. Chaque fois que l'utilisateur appuie sur un bouton, ou effectue une opération qui nécessite un transfert ou une manipulation de données, une requête post est envoyée au serveur. Celui-ci va alors regarder la méthode post correspondante et effectuer les opérations correspondantes.

Pour illustrer ce fonctionnement, prenons un exemple simple : la création d'un compte. Vous êtes sur la page de création de compte, vous remplissez les données, puis clickez sur le bouton 'créer un compte'. Les données que vous avez entrées vont être transmise au serveur. Le serveur va alors utiliser la méthode post correspondante. Cette méthode va créer un nouvel objet 'user', puis l'insérer dans la base de données. Une fois les données mises à jour, la méthode vous renvoie à la page d'accueil, avec l'information que vous êtes maintenant connecté (dans notre code, cette information revient souvent, et est stockée sous forme booléenne dans la variable 'isLog'). Le système va alors réafficher la page d'accueil, mais en mettant à jour les deux icones dans la barre de navigation, l'une indiquant que vous êtes maintenant connecté, l'autre vous permettant de vous déconnecter.

Mise en page du site :

Pour la mise en page du site, nous avons utilisé le html 5 et le css 3. Pour chaque page web, un page « ejs » doit être créer. Dans notre site, presque toutes les pages contiennent une 'navbar' ainsi qu'un 'footer'. Afin d'uniformiser les 'navbar' des différentes pages, et de faciliter les modifications, nous avons mis la structure dans un fichier à part. Ainsi, lorsque nous devrons modifier la barre de navigation, il suffira de modifier ce fichier, et non toutes les pages du site. Il en va de même pour le 'footer'.

Passons maintenant à la partie css. Nous avons créé un seul fichier css, nommé style.ccs, pour la gestion de la mise en page. Cela nous a permis d'inclure un seul fichier css pour toutes les pages, et ainsi pouvoir réutiliser certaines structures.

Pour nous permettre d'effectuer quelque opération en javascript, et d'ainsi éviter la multiplication de code illisible ou de page inutile, nous avons utilisé non pas des fichiers « html », mais des fichiers « ejs ». La différence est que les fichier « ejs » nous permettent d'exécuter du code javascript directement, sans avoir à être lié à une autre classe javascript.

Toutes les lignes de commandes <% xxxxxx %> indique qu'il faut considérer le texte comme du javascript et pas du html. Par exemple, c'est grâce à ça que nous pouvons effectuer les opération « if/else » pour le bouton commander. Cela explique également comment nous utilisons une même version de barre de navigation, en effectuant un simple « include ».

Pour ce qui est des fonctions de localisation utilisé pour notre application, nous avons utilisé les informations du site « open street map », et la bibliothèque « leaflet ». < à préciser>

Améliorations possibles du site :

Nous sommes arrivés à atteindre tous les objectifs initialement prévus. Cependant, notre site, comme tout programme ou application, est perfectible. Lors du développement, nous avons pensé à certaines améliorations, d'ordre technique et visuelles, que l'on pourrait apporter au site.

Commençons par l'aspect technique. La gestion des comptes est une partie importante du site, cependant, avant de penser à ajouter des fonctions, ce serait un bon début que de vérifier les données d'entrées. Par exemple, vérifier que le format du numéro de compte bancaire ou l'adresse mail soit conforme, etc.

Au niveau de l'esthétique, notre site pourrait également s'améliorer. Par exemple, ajouter les images des bières commandées ou recherchées, voire simplement quelques images de fond que l'on pourrait faire varier.

Pour ce qui est des fonctionnalités, nous pourrions remplacer la page d'accueil par une page de suggestion. Il est également envisageable de créer un système de message entre les utilisateurs, qui pourraient alors organiser leur soirée directement depuis le site.

Conclusion

Nous sommes très satisfaits du résultat final de notre site. Lors de la première version, nous nous étions mal organisés et avions sous-estimé la difficulté du projet. Cela a mené à un cruel manque de temps qui nous a permis d'aboutir à un résultat, convenable, mais pas satisfaisant. Lors du développement de cette seconde version, nous avons essayé de tenir compte des conseils que vous nous aviez donné en juin.

Pour terminer ce projet, nous avons rencontré quelques difficultés pour réaliser ce que nous voulions. Notamment, le transfert des informations entre les différentes pages a été un exercice long à maîtriser, car nous n'avions pas entièrement compris son fonctionnement lors de la première réalisation. Cependant, une fois le concept connu, la répétition fut assez aisée.

Lors de note premier passage, vous nous aviez demandé de modifier la fonction de géolocalisation de telle sorte que les données utilisées soit celle de points de vente venant de la base de données. Cependant dans l'implémentation de cette page, nous utilisons un script ajax qui nous facilite grandement la tâche. Pour utiliser les données de la bdd, il nous aurait fallu en changer, et réécrire tout le code. Non seulement cela nous aurait pris énormément de temps, mais nous ne sommes même pas sûr que cela soit techniquement possible en javascript.

En fin de compte, bien que cela nous ait demandé un effort supplémentaire, nous sommes satisfaits d'avoir eu l'occasion de pouvoir terminer complètement le site. Lors de la première présentation, nous avions un sentiment de frustration. Nous étions parvenus à obtenir quelque chose de présentable, mais nous aurions voulu pouvoir présenter le site tel que nous voulions qu'il soit, et non pas tel que nous sommes arrivés à le bricoler. Le fait de le repasser en aout nous a donné cette opportunité. Nous l'avons saisie et sommes arrivés à un résultat qui nous satisfaits beaucoup plus car nous avons atteint nos trois objectifs principaux.

Sources :

- La base de données d'exemple de bières provient de wikipédia

Modules utilisés :

- Mongoose
- Mocha
- node.js
- yarn
- Express
- Bootstrap
- Openstreet map