Auteurs :

Bastin Julien, Duchêne François

& Raquet Damien

Date :

Le vendredi 19 mai 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 avions soumis trois propositions à notre enseignant 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. Notre site devrait aux étudiants permettre 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 :

Dans cette section, nous allons décrire comment naviguer sur le site, et les fonctions que celui-ci remplit.

D'une manière générale, toute les pages sont accessibles à tout moment, en naviguant simplement sur la barre d'outils du site. Voyons à présent en détail l'utilité de chacune.

- La page d'accueil ne remplit aucune utilité fonctionnelle, mais rends le site plus présentable.

- La page « recherche d'une bière », comme son nom l'indique, permet d'effectuer une recherche sur les bières selon certains critères : le nom, le degré d'alcool, le type ou encore par province.

- La page « points de vente à proximité » permet de localiser les établissements se trouvant à proximité, ainsi que votre position. Vous avez donc une idée précise de la distance qui vous sépare de chaque établissement. Attention cependant à ne pas trop agrandir la carte sous peine de voir les performances d'affichages grandement amoindries (nous vous conseillons de ne pas dépasser la taille de Louvain-la-Neuve).

- La page « commander une bière » vous permet de réserver une bière dans un point de vente, vous devez néanmoins être connecté avec votre compte d'utilisateur afin d'effectuer une réservation. Cette fonction n'est malheureusement pas implémentée par manque de temps de notre part. Nous avons cependant créé un écran pour vous donner une idée du résultat voulu.
- La page « contactez-nous » est présente pour permettre aux utilisateurs de nous faire part des problèmes rencontrés, ainsi que des éventuelles suggestions pour l'amélioration du design/utilité de note site.

- Les boutons s'inscrire et se connecter sont eux aussi assez explicites. Ils permettent simplement de créer un compte, et de se connecter si l'on possède déjà un compte.

Besoins utilisateurs :

Définissons maintenant de manière plus précise les besoins d'utilisateurs auquel notre site répond. Pour ce faire, nous allons utiliser le langage dit « Gherkin »

Besoin 1 :

Fonctionnalité : Recherche d'une bière

En tant qu'utilisateur

Je veux trouver une bière

Contexte :

Lorsque la recherche est effectuée

Alors chaque bière restante dans la liste a une photo et une courte description

Scénario : Recherche par mot

Lorsque l'utilisateur entre le mot "Leffe" dans le champ de recherche

Alors de la liste des bières sont retirée toutes les bières qui ne correspondent pas à "Leffe"

Scénario : Recherche par degré d'alcool

Lorsque l'utilisateur choisi un intervalle de 4,5° à 5°

Alors seules les bières avec un taux d'alcool entre 4,5° et 5° seront reprises dans la liste des propositions

Scénario : Recherche par type

Lorsque l'utilisateur choisi "bière blonde"

Alors seule les bières dont le type est "blonde" sont reprises

Scénario : Recherche par gout

Lorsque l'utilisateur choisi "Sucrée"

Alors seule les bières avec un gout "sucré" sont reprises

Scénario : Recherche groupée

Lorsque l'utilisateur choisi le mot "Leffe"

Et choisi un intervalle de 4,5° à 5°

Et choisi "bière blonde"

Et choisi "Sucrée"

Alors seule les bières ayant ces spécifications seront reprises

Scénario : Recherche impossible

Lorsque l'utilisateur entre des spécifications

Et qu'aucune bière ne correspond aux spécifications

Alors aucune bière n'est affichée dans la liste des propositions

Et l'utilisateur doit recommencer une recherche avec un autre mot

<u>Besoin 2</u>

Fonctionnalité : recherche d'un point de vente d'une bière

En tant qu'utilisateur

Je veux savoir ou me procurer une bière

Contexte :

La bière existe dans la base de données

L'utilisateur est géolocalisable

Scénario : recherche d'un point de vente quelconque

Lorsque l'utilisateur recherche un endroit où se procurer une bière

Et qu'au moins un point de vente existe

Alors tous les points de vente possibles sont donnés

Scénario : recherche par prix maximum

Lorsque l'utilisateur recherche une bière avec un prix maximum

Et qu'au moins un point de vente correspond à cette recherche

Alors les points de vente avec un prix inférieur ou égal à celui de la recherche sont donnés

Scénario : recherche par distance

Lorsque l'utilisateur recherche un point de vente dans un certain rayon de distance

Et qu'au moins un point de vente correspond à cette recherche

Alors les points de vente le plus proche sont donnés

Scénario : il n'existe aucun point de vente pour la bière recherchée

Lorsque l'utilisateur effectue une recherche

Et qu’aucun point de vente ne correspond à la recherche effectuée

Alors aucun point de vente n’est proposé

Et l’utilisateur doit faire une autre recherche

Besoin 3

Fonctionnalité : Commander une bière en ligne

En tant qu'utilisateur

Je veux commander ma bière en ligne

Contexte :

Une liste de vendeurs qui ont cette bière en stock est donnée a l'utilisateur

Et un vendeur proposant un Orval pour 3,15€

Scenario : Commande acceptée

Etant donné le client possède au moins 3,15€

Lorsque l'utilisateur choisi le vendeur

Alors la somme de 3,15$ devrait être débitée du porte-monnaie électronique

Et la somme de 3,15€ devrait être envoyée au vendeur

Et une commande doit être envoyée au vendeur

Et le montant de 1 est enlevé du nombre de bière disponibles pour ce vendeur

Scenario : Commande refusée

Etant donné l'utilisateur ne possède pas au moins 3,15€

Lorsque l'utilisateur choisi le vendeur

Alors un message d'erreur est montré à l'utilisateur

Et son porte-monnaie lui est montré afin qu'il rajoute des fonds

Architecture du système :

Dans cette section, nous allons expliquer quels outils ont été utilisés et à quelles fins. Nous ne ferons ici que survoler ces différents outils, le but étant d'avoir une vue d'ensemble sur le projet. Le système sera plus détaillé dans la section suivante.

Commençons par l'aspect de gestion des données. Toute les informations dont nous disposons sont stockées dans une base de données non relationnelles sous forme de collections. Nous avons utilisé pour ce projet le logiciel mongoDB.

Pour ce qui est de la manipulation des données, cela se fait via les différents fichiers javascript. Nous avons essayé de diviser autant que possible les tâches entre les différents fichiers afin d'avoir un code clair et facilement lisible pour une personne extérieure à l'équipe de développement.

La partie serveur quant à elle est gérée grâce à javascript elle aussi, mais nous n'avons évidemment pas tout implémenté nous-même, nous avons utilisé node.js, qui donne accès à des modules qui nous facilitent la tâche.

Passons à présent au site en lui-même. Pour créer ce site, nous avons utilisé un langage standard de la programmation web, ou plutôt une combinaison standard, le html/css. Le langage html nous permet de créer le contenu du site, au niveau de l'agencement des informations, titres, sous titres, liens entre les pages, etc. Tandis que le css permet quant à lui de gérer la mise en page, autrement dit, il est responsable de la police d'écriture, de la couleur des éléments, du cadrage, etc. Nous utilisons également le module bootstrap afin de nous aider dans l'agencement des pages web.

Conception du système

Passons à présent à une vue plus détaillée du fonctionnement de notre projet. Une fois encore, commençons par la gestion des données. Celles-ci sont stockées dans trois collections distinctes.

- La collection Beer. C'est là que sont stockées les informations propres à chaque bière comme le type, le goût, le degré d'alcool, etc.

- La collection User. C'est là que sont stockées les données relatives à tous les utilisateurs ayant créé un compte, comme leur nom, leur prénom, numéro de compte, etc.

- La dernière collection est la collection Sellplace qui contient les informations relatives aux différents points de vente, comme leur stock, leur localisation, etc. Nous n'avons pas eu le temps de l'implémenter, malheureusement. Cependant, une idée de ce à quoi elle pourrait ressembler se trouve dans le fichier pub.geoson.

Il est à noter que pour avoir une base de données d'exemple plus large, il nous aurait fallu des données toutes faites, ou bien directement tirée de sites de ventes déjà existant.

Les données, pour être manipulées, doivent être récupérées. Cela se fait grâce aux classe javascript, et un module que nous avons installé : mongoose. Mongoose sert à effectuer un lien entre la bdd et les classes javascript. Il suffit de créer un nouvel objet mongoose, et d'appeler les bonnes fonctions afin de récupérer les données qui nous intéressent. Lesdites classes javascript sont les deux classes database.js et session.js.

Pour ce qui est de la partie serveur, elle est gérée par deux classes javascript, router.js et serveur.js. L'élément router sera passé en argument à serveur.js. Elle sert à indiquer les dossiers où sont stockés les pages html à gérer, ainsi que les changements de pages. Le lien d'une page à l'autre se fait par html, mais il faut tout de même indiquer au serveur qu'il faut charger une autre page. La classe serveur vérifie que la bdd est connectée avant de lancer le serveur.

Et enfin, voyons comment sont agencées les pages html/css. Chaque page du site correspond à une page html. Nous avons rendu possible le fait de pouvoir naviguer librement entre les pages grâce à des liens dans chaque page html. Nous n'avons ajouté que deux fichier css en plus de Bootstrap. Il ne nous était pas nécessaire d'en faire plus. En effet, la plupart de nos pages ont une architecture semblable. La majorité de notre mise en page est donc dans le fichier style.css.

Chaque page contient une nav barre, et un footer. La nav barre permet de naviguer vers n'importe quelle page, depuis n'importe quelle page. Le footer sert simplement à rajouter le sigle du cours sur chaque page.

Les scripts de chaque page permettent de lier les pages html avec les classe javascript nécessaires. De plus, pour la fonction de localisation nous utilisons un lien vers le site « openstreet map » et la bibliothèque « leaflet ».

Améliorations possibles :

Commençons par les points techniques. Plusieurs fonctions font défaut à notre site. Par exemple, les comptes sont figés. Une fois qu'une personne à créer un compte, il ne peut rien changer, ni son compte bancaire, ni ses informations personnelles. De plus, notre site est assez épuré. Bien que le rendu soit présentable, il est tout à fait possible de l'améliorer, avec quelques couleurs par exemple.

De plus, étant novices en la matière, il est certain qu'il y a des optimisations possibles en termes de performances. Que ce soit les temps d'accès, ou le stockage des données. Par exemple, nous avions implémenter plusieurs classe javascript que nous comptions utiliser pour gérer les données. Nous avons finalement choisi de faire autrement. Cependant, ces classes pourraient être très utile pour de future fonction, et des manipulations plus complexe sur les données. Par exemple, la gestion des profils susmentionnées, voire un système de messages entre les utilisateurs.

D'un point de vue plus commercial, à présent. Pour l'instant les fonctions login et commander de notre site sont fictives. En effet, bien que les données circulent correctement entre les différentes pages, ces données sont fictives et non sécurisées. En effet, les adresses mail ne sont pas vérifiées, la fonction commande n'est relié à aucun réel établissement, etc. A l'avenir, notre site pourrait devenir fonctionnel.

Conclusion

A la fin de ce travail, nous ne sommes pas arrivés à atteindre tous nos objectifs. En effet, nous aurions aimé pouvoir développer un peu plus la page « commander ». Cependant, nous sommes arrivés à effectuer la géolocalisation, de même que la recherche des bières.

Lors de ce travail, nous avons rencontré beaucoup de difficulté. Tout d'abord, nous ne savions comment commencer le travail. Quelle partie du projet fallait-il faire en premier. Nous avons finalement choisi de commencer par les classes javascript, qui manipulerait nos données, car c'était à la fois ce qui nous semblait le plus pertinent, et le plus proche de ce que nous connaissions. Le langage html/css nous aura également causé quelques problèmes mineurs, mais nous sommes arrivés à les résoudre assez rapidement et à obtenir les résultats voulus.

Le point qui nous a causé le plus de problème est l'accès et la gestion de la bdd en noSQL. En effet, utiliser la bdd à l'aide de l'invite de commande fut assez simple, mais lié la bdd aux classes javascript, ainsi que récupérer les données, puis les utiliser fut un véritable challenge pour nous. D'ailleurs, c'est là la raison de notre retard et notre manque de réussite quant à la page « commander ».

Sources :

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

Modules utilisés :

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