UVSQ
université PARIS-SACLAY

UFR des Sciences - Versailles
M1 Informatique - AWS - MIN17217
Github du projet
Site web de notre projet

Compte Rendu d'AWS
Groupe 17 : Site de réservation d'hôtels - Hotel Resort

Thivagini SUGUMAR
Gwennael CANNENPASSE
Salah AOUCHICHE
Yasser DRIF

Versailles, le 15 mai 2022

Table des matières

1 Introduction

Dans le cadre de notre Master 1 Informatique, nous devons effectuer un projet à travers l'UE "Application web et sécu". Consistant à faire un site web, nous avons choisi de réaliser un site de réservation d'hôtels qui s'intitule Hotel Resort.
Ce projet nous a permis de se familiariser avec de nombreux concepts et outils de développement web. Notre projet repose sur une bonne implémentation et gestion d'une base de données ainsi qu'une bonne méthodologie de recherches et une gestion de projet adéquate dûe à nos connaissances très limitées en développement web.

2 Description du projet

Avec Hotel Resort, l'utilisateur peut planifier son prochain séjour. Il a la possibilité de rechercher un hôtel selon des critères : la ville de destination, la date d'arrivée et la date de départ. Une fois la recherche lancée, il pourra visualiser l'ensemble des hôtels disponibles.
Si l'utilisateur veut réserver, il devra donc choisir un hôtel parmi ceux qui sont présents sur la page. Pour l'aider, nous avons décidé de lui offrir un ensemble de fonctionnalités détaillées pour chaque hôtel : la description de l'hôtel, les services fournis, les photos, les étoiles de l'hôtel et les avis d'autres utilisateurs. Une fois l'hôtel choisi, l'utilisateur pourra sélectionner une chambre. Arrivé à cet étape, il ne lui restera plus qu'à réserver en saisissant ses informations personnelles, puis en passant au paiement. Une fois le paiement effectué, il aura la possibilité d'obtenir une facture qui confirmera sa réservation.

L'intérêt de passer par notre site et non par celui d'un de nos concurrents (ex : Booking) est le fait de pouvoir cumuler des points de fidélité. A chaque réservation effectuée sur notre site, l'utilisateur gagnera des points de fidélité. Arrivé à 1000 points, il obtiendra un bon de -40% sur sa prochaine réservation sur notre site web.

3 Technologies web utilisées

3.1 Front-end

3.1.1 HTML/CSS

Pour réaliser les pages, nous avons utilisé des frameworks afin de rendre l'affichage du site uniforme et de simplifier le code. Nous avons donc intégrer les librairies suivantes dans nos pages HTML :

- Bootstrap : pour afficher les items des pages,
- Animate : pour ajouter de l'animation,
- Slick : pour ajouter des slides sur les pages,
- Leaftlet : pour afficher une Map sur les pages.

3.1.2 Javascript

Pour donner un peu de dynamisme à nos pages, nous avons exploité des librairies qui sont les suivantes :

- JQuery/Bootstrap : pour gérer les événements et rendre interactive les pages,
- Wow : pour ajouter des animations lors du défilement des pages,
- Slick : pour ajouter des slides animés sur les pages,
- SideBar : pour ajouter un menu latéral lorsqu'on réduit la page,
- Dropzone : pour télécharger un pdf sur la page.

3.1.3 Fonts

Pour avoir un affichage plutôt accueillant et "user-friendly", nous avons choisi d'utiliser des librairies qu'on a intégré dans les pages HTML. Ces librairies sont :

- Font Awesome : icônes pour illustrer les pages,
- Flaticon : icônes pour illustrer les pages,
- Favicon : icônes qui se placent à côté du nom de l'onglet,
- Google : police personnalisée pour les pages.

3.1.4 Vues ejs

Nous avons utilisé les vues (fichiers .ejs) car ils offrent un moyen plus simple de générer du HTML de manière dynamique, et aussi car on les utilise pour répondre aux requêtes des clients, donc les pages sont des pages dynamique où les données proviennent de diverses sources, donc il devient trop facile d'y insérer des données dynamiques.

3.2 Intermédiaire du Front-end et du Back-end

3.2.1 L'API REST

Pour la connexion Back-end/Front-end, nous avons utilisé l'API REST, qui est un intermédiaire logiciel (Client/Server) permettant la communication entre les différent clients et le serveur. Nous avons choisi d'utiliser cette API, car elle :

- Offre un moyen de communication standardisé entre le client et les données, avec des requêtes gérées via HTTP,
- Utilise un système à couches, invisible pour le client, pour la sécurité donc elle nous permet d'éviter le problème de l'injection de données,
- Utilise des routes, qui pemettent un mappage des requêtes HTTP(get, post, put, delete..) aux actions de client sur le site.

3.3 Back-end

3.3.1 Node.js

Pour la partie Back-end, nous avons utilisé node.js ainsi que de nombreux modules :

- Le module express : s'occupera de notre serveur, pour créer des API. Il prend en charge les détails essentiels du backend tels que les sessions, le traitement des erreurs et le routage.
- Le module Mongoose : est une bibliothèque de modélisation des données d'objets pour MongoDB et Node.js, elle gère les relations entre les données, fournit une validation de schéma et est utilisée pour traduire les objets dans le code et la représentation de ces objets dans MongoDB.
- Le module body-parser : analyse les données codées soumises à l'aide d'une requête HTTP POST.
- Le module nodemon : Il enveloppe l'application Node, surveille le système de fichiers et redémarre automatiquement le processus.

3.3.2 MongoDB

Pour la gestion de la base de données, nous avions fait le choix de partir sur une base de données relationnelle à cause de nos trop nombreuses relations. Ce choix nous a permis de diviser l'ensemble des informations et de les mettre en relation : les chambres, les commentaires, les comptes, les réservations. Cependant, pour des raisons de simplicité d'hosting de base de données, nous avons décidé d'utiliser *MongoDB* et donc de partir sur un système de base de données non relationnelle.

3.3.3 Nodemailer

Enfin, pour gérer le rendu de facture, nous avons décidé d'envoyer un mail à l'utilisateur. Pour cela, nous avons utilisé le module *nodemailer*.

4 Points délicats du projet

4.1 Liaison entre le back-end et le front-end

La partie Middle-end notamment la transmission d'information entre le Back et le Front a été l'un des points difficile de ce projet. L'utilisation d'un système NoSql est l'un des points difficile. En effet, nous avons due, en grande partie, repenser à notre base de données mais surtout dû adapter nos requêtes déjà écrites en SQL en langage MongoDB. C'est ce dernier point qui a été le plus compliqué.

4.2 Système de commentaires des clients

Pour gérer les commentaires, nous avons décidé de créer une table dans notre base de données. Un commentaire est donc lié à un utilisateur et un hôtel. Lors de l'affichage des commentaires, nous effectuons une requête pour récupérer les commentaires d'un hôtel en particulier.

4.3 Gestion de la base de données d'images des hôtels

Afin d'afficher les images correspondants à chaque hôtel, nous avions d'abord eu l'idée de les enregistrer dans notre base de données. Cependant cela aurait pris trop de place. Par la suite, nous avons pensé à mettre le lien de chaque image dans la base de données. Mais après l'avis de notre chargé de projet, nous avons conclu que cela n'était pas la bonne méthode.
Nous avons donc fait le choix d'organiser nos images dans les dossiers correspondants à chaque hôtel et d'utiliser une simple fonction de comparaison du nom d'hôtel et du nom de dossier pour récupérer et afficher sur la page. Cependant pour des raisons de manque de temps, nous ne l'avons pas implémenté à présent.

4.4 Hébergement du site

Pour héberger notre site nous avions pensé à utiliser Heroku. Or nous avons vu que récemment il y avait des problèmes de sécurité avec les dépôts Github. Afin d'éviter tout soucis, nous sommes donc partis sur Replit.com qui nous permet d'exécuter notre code et d'obtenir un lien https. Cependant ce lien n'est pas constant, il faut en effet que le code du repository soit relancé par moment.

5 Structure du projet

Notre projet est composé des dossiers suivants :

- **Pages :** ce dossier contient l'ensemble des pages HTML,
- **SQL :** ce dossier contient l'ensemble des définitions de la base de données ainsi qu'un fichier contenant l'ensemble des requêtes SQL que nous utilisons,
- **Node Modules :** ce dossier contient toutes les dépendances installées pour notre projet,
- **Routes :** ce dossier contient la définition des différentes routes afin que le client interagisse avec notre base de données MongoDB.

Il contient aussi les trois fichiers suivants :

- **package.json** : liste les dépendances qui nous indique les versions appropriées à installer pour le projet,
- **package-lock.json** : garde la trace de toutes les modifications dans package.json ou Node Modules et nous indique la version exacte du paquet installé,
- **Server.js** : il contient le programme principal du projet (comme point d'entrée principal).

6 Conclusion

Pour conclure, notre projet nous permet bien de réserver une chambre d'hôtel en fonction de la ville souhaitée et des dates de séjours de l'utilisateur. Il gère l'ensemble des informations (description, services fournis, photos, étoiles et avis d'autres utilisateurs) de chaque hôtel ainsi que leur disponibilité.

6.1 Organisation

Grâce à une bonne organisation et une communication constante, notre projet s'est bien déroulé sans problème. En ce sens, nous avons programmé deux réunions dans la semaine :

- Une réunion chaque lundi pour se répartir les tâches,
- Une réunion chaque mercredi afin de mettre en commun les avancées de chacun et de rédiger le compte-rendu de la semaine.

6.2 Amélioration

Notre projet reste tout de même améliorable. Nous pourrions ajouter quelques fonctionnalités telles que :

- un chat en ligne : mise en place d'un bot pour que le client puisse se renseigner ou un espace d'échange avec d'autres utilisateurs présents sur le site,
- une création d'utilisateur "hôtelier" sur le site qui permettrait de rentrer eux-même les informations de leurs établissements,
- une option pour changer la langue du site,
- une option pour convertir la devise,
- une possibilité d'authentification sur notre site avec les réseaux sociaux grâce à leur API.

Références

— MongoDB : *https ://www.mongodb.com/docs/drivers/node/current/*
— Hevodata : *https ://hevodata.com/learn/mongodb-join-two-collections/*
— Bootstrap : *https ://getbootstrap.com*
— Font Awesome : *https ://fontawesome.com/*
— Javascript : *https ://www.tutorialsteacher.com/javascript/javascript-date*
— HTML/CSS : *https ://www.w3schools.com/html/html*$_c$*ss.asp*