Présentation Soutenance P6

Construisez une API sécurisée pour une application d'avis gastronomiques

Jonathan Palacios

Sommaire :

- Présentation des différentes technologies utilisées
- Présentation des différents package utilisés
- Présentation du code + de l'API
- Les principales difficultés rencontrées

Présentation des différentes technologies

- **Express : est un Framework reposant sur Node qui facilite la création et la gestion des serveurs Node.**

Une application Express est fondamentalement une série de fonctions appelées « middleware ». Chaque élément de middleware reçoit les objets « request » et « response » peut les lire, les analyser et les manipuler.
Le middleware Express reçoit également la méthode « next » qui permet à chaque middleware de passer l'éxécution au middleware suivant,

Présentation des différentes technologies

- **Node : est le runtime ou environnement d'éxécution permettant d'écrire toute nos tâches côté serveur, en JavaScript, telles que la logiques métier, la persistance des données et la sécurité. Node ajoute également des fonctionnalités que le JavaScript du navigateur ne possède pas comme par exemple l'accès au système de fichier local.**
- **Mongoose : Outil nous mettant à disposition des fonctions complètes pour interagir avec notre base de données**
- **MongoDB Atlas : est un système de gestion de base de données orientés document**

Présentation des différents package

- **Bcrypt : fonction de hachage. En plus de l'utilisation d'un sel pour se protéger des attaques par table arc-en-ciel, bcrypt est une fonction adaptative, cad que l'on peut augmenter le nombres d'itérations pour la rendre plus lente. Ainsi, elle continue à être résiistante aux attaques par force brute malgré l'augmentation de la puissance de calcul**

- **Body-parser : pour gérer la demande HTTP POST dans Express.js . Il extrait la totalité du corps d'un flux de demandes entrantes et l'expose en « req.body ». Ce module analyse donc les données codées JSON, tampon, chaîne et URL soumises à l'aide de la demande HTTP POST**

- **Cookie-session : middleware de session simple basé sur des cookies. Une session utilisateur peut être stockées de 2 manières principales avec des cookies : sur le serveur ou sur le client. Ce module stocke les données de session sur le client.**

- **Dotenv : module sans dépendance qui charge les variables d'environnement d'un fichier .env. Le stockage de la configuration dans l'environnement est séparé du code.**

Présentation des différents package

- **Helmet : nous aide à protéger notre application de certaines vulnérabilités bien connues du web en configurant de manière appropriée des en-têtes HTTP. Collection de 9 fonctions middleware plus petite qui definissent des en-têtes HTTP liés à la sécurité. https://expressjs.com/fr/advanced/best-practice-security.html**

- **Jsonwebtoken : nous permet l'implémentation des jetons Web**

- **Cookie-session : middleware de session simple basé sur des cookies. Une session utilisateur peut être stockées de 2 manières principales avec des cookies : sur le serveur ou sur le client. Ce module stocke les données de session sur le client.**

- **Mongoose unique validator : plugin qui ajoute une validation de pré enregistrement pour les champs uniques dans un schéma Mongoose. Il rend la gestion des erreurs beaucoup plus facile car nous obtenons une erreur de validation mongoose lors d'une tentative de viol d'une contrainte unique plutôt qu'une erreur de MongoDB**

Présentation des différents package

- **Multer : Middleware node pour la gestion des données multipart/form-data qui est principalement utilisé pour le téléchargement de fichiers.**

- **Save : simple action de persistance basée sur CRUD pour stocker des objets dans n'importe quel base de données. Dans notre cas MongoDB**

Présentation du code + de l'API

Principales difficultés rencontrées

- **Imaginer la structure du code**

- **Implanter les concepts de sécurité**