Le 16/05/2024

Eval – NoSQL MongoDB

H3 - Hitema

Consignes

- **Date de rendu maximum** : le 16 Mai 2024 17h00

- **Modalité de rendu** :
    1. créer un dépôt github et réaliser un push de l'ensemble des réponses
    2. remplir le formulaire disponible à l'adresse suivante :
    3. https://forms.gle/ozm9zWCZwmaAei9y7

ou

https://docs.google.com/forms/d/e/1FAIpQLScwt3tBhvJVB9HZXM2p0OaiWJNP5_quQ4tvXVwJtwbCPyQdZg/viewform

- **Evaluation** : la note de l'exercice est noté à côté de l'énoncé

Vous disposez d'internet, du support et de notes de cours transmise sur github
Il s'agit d'un travail individuel

Enoncé (noté sur 20 points)

La mairie de Paris souhaite créer une API pour faciliter la consultation de ses promenades.

Elle dispose d'une base de données en format JSON quelle vient de réaliser et elle vous la met à disposition (voir le fichier annexe : paris.json) .

Enfin, elle a choisi MongoDB comme système de Gestion de Base de Données pour stocker les informations et NodeJS avec les module Mongoose et Express JS pour faire communiquer des applications clients avec la base de données

1 Créer une base de donnée intitulée « Paris » dans MongoDb Atlas

2 Créer une collection nommé « balades » dans la base de données « Paris »

3 Importer l'ensemble des données du fichier JSON dans cette collection

4 Créer un projet NodeJS avec les modules Mongoose et Express JS permet de réaliser un CRUD sur la collection « balades » :

1. la route http://localhost:1235/all (GET) doit permettre de lister toutes les balades disponibles en base de donnée
2. la route http://localhost:1235/id/:id (GET) doit permettre d'afficher une seule balade via son identifiant unique / si l'id est invalide l'API doit retourner un message d'erreur
3. la route http://localhost:1235/search/:search (GET) doit permettre d'afficher les ou les balades dont les lettres saisies dans:search sont présentes soit dans la clé « nom_poi » soit dans la clé « texte_intro »
4. la route http://localhost:1235/site-internet (GET) affiche l'ensemble des balades qui disposent d'une clé « url_site » qui a une valeur non null
5. la route http://localhost:1235/mot-cle (GET) affiche l'ensemble des balades qui disposent de plus de 5 mots clé dans leur clé « mot_cle »
6. la route http://localhost:1235/publie/:annee afficher l'ensemble des balades publiés lors de l'année passé en paramètre d'url. Les balades devront être triées de la plus ancienne à la plus récente
7. la route http://localhost:1235/arrondissement/:num_arrondissement (GET) doit permettre de compter le nombre de balade pour l'arrondissement écrit dans la clé:num
8. la route http://localhost:1235/synthese (GET) afficher par arrondissement le nombre de balades disponibles
9. la route http://localhost:1235/categories (GET) afficher les différentes catégories distinctes de balades disponibles en base de donnée
10. la route http://localhost:1235/add (POST) permet de créer une nouvelle balade : Attention le s clés « nom_poi » / « adresse » et « categorie » sont obligatoires
11. la route http://localhost:1235/add-mot cle/:id (PUT) permet d'ajouter un mot clé à une balade existante dans sa clé « mot_cle », Attention le mot clé ajouté en doit pas être en doublon avec les mots clé existant. Attention si l'id est invalide afficher une message d'erreur

12. la route http://localhost:1235/update-one/:id (PUT) permet de mettre à jour une balade via son id, Attention si l'id est invalide afficher une message d'erreur
13. la route http://localhost:1235/update-many/:search (PUT) permet de mettre à jour « nom_poi » de plusieurs balades distante si leur « texte_description » contient les lettres dans la clé search qui est dans l'url
14. la route http://locahost:1235/delete/:id (DELETE) permet de supprimer une balade via son id , Attention si l'id est invalide afficher une message d'erreur

Ne pas hésiter à utiliser l'extension Thunder Client de Visual Studio Code pour tester chacune des routes du projet