Exam Final – Technologie web

Prof. TMIMI Mehdi

Objectif

L'objectif de ce projet est de développer un web service en **Node.js** permettant de stocker et de récupérer les racines carrées des nombres dans une base de données **JSON (fichier)**. Le web service doit être implémenté en utilisant uniquement les modules **http** et **fs**.

1. Conception de la base de données JSON

Il est nécessaire de définir une structure adaptée pour stocker les nombres et leurs racines carrées dans un fichier JSON.

Question : Donnez un exemple d'une base de données JSON contenant au moins deux nombres et leurs racines carrées.

Exemple attendu : Une correspondance entre un nombre et sa racine carrée, comme `25 => 5` et `9 => 3`.

2. Configuration du Web Service

2.1. Route GET

- **Endpoint ou Path :** `/numbers/:number`
- **Méthode :** `GET`
- **Description :**
  - Lorsqu'un client envoie une requête `GET /numbers/25`, le serveur doit retourner **uniquement** la racine carrée sous forme de texte brut (exemple : `"5"`).
  - Si le nombre demandé n'existe pas dans la base de données, le serveur retourne **404 Not Found**.

2.2. Route POST

- **Endpoint ou URL:** `/numbers`
- **Méthode :** `POST`

- **Données requises :**
  - **Body** : Contient le nombre à ajouter.
  - **Header `racine`** : Contient la racine carrée du nombre.
- **Comportement :**
  - Si le nombre existe déjà dans la base de données, ou si les données sont **manquantes**, ou si la racine carrée est **incorrecte**, la réponse est :
    - **Status :** `400 Bad Request`
    - **Format de réponse : JSON**

    ```
    { "error": "Message indiquant la nature de l'erreur" }
    ```

  - Si l'ajout est réussi :
    - **Status :** `201 Created`
    - **Format de réponse : JSON**

    ```
    { "number": 25, "squareRoot": 5 }
    ```

3. Contraintes techniques

- Utiliser **uniquement** les modules natifs de Node.js :
  - `http` pour gérer les requêtes et réponses.
  - `fs` pour la gestion de la base de données JSON.
- Assurer la **persistance des données** en mettant à jour le fichier JSON à chaque ajout.
- Gérer correctement les **erreurs** (exemple : fichier JSON corrompu ou introuvable, mauvais format de données, etc.).

4. Travail demandé:

- **Donnez la structure de votre base de données JSON** contenant au moins **deux nombres** et leurs **racines carrées**.
- **Implémentez un serveur HTTP** en utilisant le module **`http`** de Node.js et en respectant la **configuration demandée**.

Ajout d'un Client Web

Afin de permettre aux utilisateurs de visualiser les nombres et leurs racines carrées, nous allons développer un **client web** simple en **HTML**. Ce client devra envoyer une requête au serveur pour récupérer les données et les afficher dans un tableau.

Hypothèse :

On suppose que **le serveur est hébergé en local** et accessible à l'adresse :

`http://localhost:3000`

Toutes les requêtes HTTP doivent être envoyées vers cette adresse.

Code de base fourni :

```html
<!DOCTYPE html>
<html lang="fr">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Liste des Nombres et Racines Carrées</title>
</head>
<body>
    <h2>Liste des Nombres et de leurs Racines Carrées</h2>
    <table>
        <thead>
            <tr>
                <th>Nombre</th>
                <th>Racine Carrée</th>
            </tr>
        </thead>
        <tbody id="tbodyTable">
            <!-- Les données seront insérées ici dynamiquement -->
        </tbody>
    </table>
</body>
</html>
```

Exemple de structure de ligne `<tr>` attendue après ajout dynamique :

Le JavaScript devra générer dynamiquement des lignes similaires à celle-ci :

```html
<tr class="number-row">
  <td>25</td>
  <td name="racine-carre">5</td>
</tr>
```

Consignes obligatoires :

- Interdiction d'utiliser `innerHTML` pour insérer des éléments dans le tableau.

- Vous devez impérativement utiliser :

- **`document.createElement`** pour créer les éléments `<tr>` et `<td>`.
- Mehdi.tmimi@usmba.ac.ma
- **`setAttribute`** pour ajouter l'attribut "`name`".
- **`classList.add`** pour ajouter la classe "`number-row`".
- **`appendChild`** pour ajouter les éléments au DOM.
- **Utiliser `fetch`** pour envoyer une requête GET au serveur et récupérer les données.

Travail demandé aux étudiants :

1. **Ecrivez le code JavaScript** pour envoyer une requête GET au serveur et récupérer la liste des nombres et leurs racines carrées.
2. **Créez dynamiquement** les éléments `<tr>` et `<td>` en respectant l'exemple fourni.
3. **Ajoutez des classes et des attributs personnalisés** aux éléments créés.
4. **Affichez les erreurs** en cas d'échec de la requête HTTP en utilisant la fonction 'alert' de javascript.