EVALUATION

TP1

Construire une API REST

Production

- Compressez votre projet au formation zip. Ne pas mettre le répertoire node_modules.
- Déposez le fichier dans Teams.
- Nommage du fichier : Nom_Prenom_apiRestNodeJS.zip.

node-express-api

Construire une API REST

- Objectif 1 : A partir du support de cours, développer l'API qui implémente les routes suivantes :
  - GET /parkings — Lister l'ensemble des parkings
  - GET /parkings/:id — Récupérer les détails d'un parking à partir de son id
  - POST /parkings — Créer un parking
  - PUT /parkings/:id — Modifier les détails d'un parking à partir de son id
  - DELETE /parkings/:id — Supprimer un parking à partir de son id

- Objectif 2 : Tester l'accès à l'API REST via l'outil Postman

- Objectif 3 : Evolution de l'application en prenant en compte les réservations de parkings
  - Cf. le cahier des charges présenté dans les slides suivants.

Construire une API REST

Objectif 3 : cahier des charges

- La société exploitant des parkings de longue durée prend des réservations de la part de ses clients.
- A la racine de votre projet, créez un fichier au format JSON nommé reservations.json (fichier disponible dans Teams).
- Modifiez le fichier parkings.json afin de prendre en compte les réservations. Un parking peut avoir plusieurs réservations.

```
[
   {
      "id": 1,
      "name":"Parking 1",
      "type":"AIRPORT",
      "city":"ROISSY EN FRANCE",
      "reservations": [1, 2]
   },
……
]
```

reservations.json

```
[
   {
      "id":1,
      "clientName": "Bob Marley",
      "vehicle": "car",
      "licensePlate": "ED432EF",
      "checkin":"2024-08-21T06:00:00Z",
      "checkout":"2024-08-27T06:00:00Z"
   },
   {
      "id":2,
      "clientName": "Stevie Wonder",
      "vehicle": "car",
      "licensePlate": "AB213CD",
      "checkin":"2024-08-20T06:00:00Z",
      "checkout":"2024-08-27T06:00:00Z"
   },
   {
      "id":3,
      "clientName": "Roger Federer",
      "vehicle": "car",
      "licensePlate": "EB123KJ",
      "checkin":"2024-08-01T06:00:00Z",
      "checkout":"2024-08-17T06:00:00Z"
   },
……
]
```

Construire une API REST

- Définition des ressources

- Vous devez définir les fonctionnalités suivantes :
  - Créer une réservation d'une place dans un parking — CREATE
  - Afficher l'ensemble des réservations — READ
  - Afficher les détails d'une réservation en particulier — READ
  - Afficher les détails des réservations d'un parking en particulier — READ
  - Modifier les détails une réservation d'une place dans un parking — UPDATE
  - Supprimer une réservation d'une place dans un parking — DELETE

Ces opérations sont plus communément appelées CRUD (CREATE, READ, UPDATE, DELETE).
Dans cet exemple, votre API dispose de 2 ressources : le Parking et la Réservation.

Construire une API REST

- Création des routes
- Définissez les routes à partir de la définition des ressources présentée dans le slide précèdent.
- Au sein du fichier index.js, créez les différentes routes.
- Testez chaque route à l'aide de Postman.
- Lorsque vous créez ou supprimez une réservation, vous devez :
  - Vérifier que l'id de la réservation existe. Si l'id n'existe pas, vous devez retourner un code HTTP 404 et le message "Réservation non trouvée."
  - Vérifier que l'id du parking existe. Si l'id n'existe pas, vous devez retourner un code HTTP 404 et le message "Parking non trouvé."
  - Mettre à jour les réservations du parking concerné.
- Lorsque vous affichez ou modifiez les caractéristiques une réservation, vous devez :
  - Vérifier que l'id de la réservation existe. Si l'id n'existe pas, vous devez retourner un code HTTP 404 et le message "Réservation non trouvée."

Vous pouvez utiliser console.log() pour afficher une réservation ou un parking dans la console de Visual Studio Code.

Construire une API REST

- Création des routes

  - GET /reservations

    Afficher l'ensemble des réservations

  - GET /parkings/:id/reservations

    Afficher les détails des réservations d'un parking en particulier (parking identifié par son id)

  - DELETE /parkings/:id/reservations/:id1

    Supprimer à partir de son id une réservation d'une place dans un parking (parking identifié par son id)

Construire une API REST

- Création des routes

  - POST /parkings/:id/reservations/

    Créer une réservation d'une place dans un parking(parking identifié par son id)

```
{
        "id": 7,
        "clientName": "Bob Marley",
        "vehicle": "car",
        "licensePlate": "ED432EF",
        "checkin": "2024-08-21T06:00:00Z",
        "checkout": "2024-08-27T06:00:00Z"
}
```

Construire une API REST

- Création des routes

  - POST /parkings/:id/reservations/idreservation

    Créer une réservation d'une place dans un parking (parking identifié par son id)

```
{
        "clientName": "Bob Marley",
        "vehicle": "car",
        "licensePlate": "ED432EF",
        "checkin": "2024-08-21T06:00:00Z",
        "checkout": "2024-08-27T06:00:00Z"
}
```

Construire une API REST

- Création des routes

  - PUT /reservations/:id

    Modifier les détails une réservation d'une place dans un parking (parking identifié par son id)

```
{
        "clientName": "Bob Marley",
        "vehicle": "car",
        "licensePlate": "ED432EF",
        "checkin": "2024-08-21T06:00:00Z",
        "checkout": "2024-08-27T06:00:00Z"
}
```