self-service-machine-api - STEP 3

Dans cette étape nous allons mettre en place 2 routes :

- la route permettant de récupérer un produit
- la route permettant d'ajouter un nouveau produit

Route GET /api/products/id

Cette route permet de récupérer un produit en particulier.

```
productsRouter.get("/:id", (req, res) => {
  const productId = req.params.id;
  const product = products.find((product) => product.id == productId);
  const message = `Le produit dont l'id vaut ${productId} a bien été récupéré.`;
  res.json(success(message, product));
});
```

Route POST /api/products/

Cette route permet d'ajouter un nouveau produit. C'est probablement la route la plus difficile à mettre en place.

Pour commencer, il faut comprendre que le consommateur de notre API REST va devoir nous envoyer les informations du nouveau produit qu'il souhaite ajouter.

Voila à quoi pourrait ressembler un appel HTTP correspondant en utilisant curl :

```
curl -X POST http://localhost:3000/api/products -H "Content-Type: application/json" -d '{
    "name": "HamburgerVaudois",
    "price": 9.99
}'
```

Le consommateur de notre API REST nous a transmis en json les informations du nouveau produit à ajouter.

A nous de faire le nécessaire pour être capable de prendre en charge une telle requête HTTP.

```
productsRouter.post("/", (req, res) => {
  // Création d'un nouvel id du produit
  // Dans les prochains versions, c'est MySQL qui gérera cela pour nous (identifiant auto_increment)
  const id = getUniqueId(products);

  // Création d'un objet avec les nouvelles informations du produits
  const createdProduct = { ...req.body, ...{ id: id, created: new Date() } };

  // Ajout du nouveau produit dans le tableau
  products.push(createdProduct);

  // Définir un message pour le consommateur de l'API REST
  const message = `Le produit ${createdProduct.name} a bien été créé !`;

  // Retourner la réponse HTTP en json avec le msg et le produit créé
  res.json(success(message, createdProduct));
});
```

Il est important de noter que notre route est `.post()` et plus `.get()` comme précédemment.

Ensuite, nous pouvons commencer à étudier le corps de la méthode.

Nous commençons par générer un nouvel id produit.

```
const id = getUniqueId(products);
```

C'est une fonction créé par nos soins et que je vous conseille de placer dans le fichier `helper.js`.

```
...
const getUniqueId = (products) => {

  // On construit un tableau d'id de produits
  const productsIds = products.map((product) => product.id);

  // La fonction passée à reduce compare deux éléments à la fois (a et b) et
  // retourne le plus grand des deux grâce à Math.max.
  // Au final, reduce parcourt tout le tableau productsIds, compare chaque ID
  // avec l'ID maximal trouvé jusqu'à présent, et retourne l'ID le plus élevé,
  // qui est stocké dans la variable maxId.
  const maxId = productsIds.reduce((a, b) => Math.max(a, b));

  return maxId + 1;
};

export { success, getUniqueId };
```

Ensuite nous créons l'objet produit à utilisant la décomposition d'objet JavaScript.

```
const createdProduct = { ...req.body, ...{ id: id, created: new Date() } };
```

Il nous reste plus qu'à ajouter le nouveau produit à notre liste des produits.

```
products.push(createdProduct);
```

Le reste du code ne devrait pas poser de problème de compréhension.

Nous sommes prêt pour tester notre nouvel route :

```
Greg@LAPTOP-5335QTOF MINGW64 ~/OneDrive - Education Vaud/295/self-se
$ curl -X POST http://localhost:3000/api/products -H "Content-Type:
application/json" -d '{
    "name": "HamburgerVaudois",
    "price": 9.99
}'
  % Total    % Received % Xferd  Average Speed   Time    Time     Ti
me  Current
                                 Dload  Upload   Total   Spent    Le
ft  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:-
100   163  100   110  100    53   5924   2854 --:--:-- --:--:-- --:-
-:--  9588{"message":"Le produit undefined a bien été créé !","data"
:{"id":10,"created":"2023-12-27T10:59:01.758Z"}}
```

Malheureusement, il y a un problème !

Comme on peut le voir le message est : "Le produit undefined a bien été créé !"

Dans notre code, nous utilisons un `req.body` afin de récupérer les données transmises par le consommateur de notre API REST. Malheureusement, `req.body` est `undefined`.

Pourquoi ?

Lorsqu'un consommateur envoie une requête POST avec un corps de requête JSON nous devons convertir le JSON en un objet JavaScript.

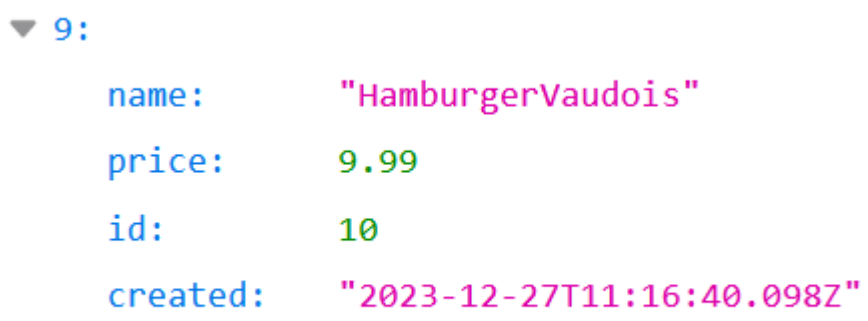
Pour ce faire, nous allons configurer le middleware `express.json()` pour analyser le corps des requêtes JSON.

```
import express from "express";

const app = express();

app.use(express.json());

const port = 3000;

...
```

Si vous testez à nouveau la commande `curl` tout devrait fonctionner maintenant !

D'ailleurs après avoir ajouté votre nouveau produit, vous pouvez refaire un HTTP GET /api/products/ dans votre navigateur.

Un nouveau produit avec l'id 10 devrait être présent :



Passons à l'étape n°4