Fichier : 01 - app.js

Tutoriel Node.js avec Express, CORS et Dotenv

Dans ce tutoriel, nous allons passer en revue un exemple de code Node.js utilisant Express, CORS et Dotenv. Nous allons expliquer chaque bloc de code en detail pour vous aider a comprendre comment ils fonctionnent ensemble.

Importation des modules

```
import express from 'express';
import cors from 'cors';
import dotenv from 'dotenv';

import aiRoutes from './routes/ai.js';
import aiServices from './config/ai-services.js';
```

Dans cette section, nous importons les modules necessaires pour notre application. Cela comprend `express` pour la gestion du serveur, `cors` pour gerer les requetes cross-origin, `dotenv` pour charger les variables d'environnement, ainsi que `aiRoutes` et `aiServices` qui sont des modules specifiques a notre application.

Configuration de l'application

```
dotenv.config();

const app = express();
const port = 3000;

app.use(cors());
app.use(express.json());
```

Ici, nous configurons notre application. Nous utilisons `dotenv.config()` pour charger les variables d'environnement. Ensuite, nous creons une nouvelle instance d'Express et definissons le port sur lequel notre serveur sera a l'ecoute. Enfin, nous utilisons `app.use(cors())` pour permettre les requetes cross-origin et `app.use(express.json())` pour analyser les requetes entrantes avec des charges utiles JSON.

Definition des routes

```
app.use('/api/ai', aiRoutes);
```

Dans cette section, nous definissons les routes pour notre application. Nous utilisons `app.use('/api/ai', aiRoutes)` pour dire a Express que toutes les requetes commencant par '/api/ai' doivent etre gerees par `aiRoutes`.

Route pour obtenir les services AI

```
app.get('/api/ai/services', (req, res) => {
  res.json({ services: aiServices });
});
```

Ici, nous definissons une route specifique pour obtenir la liste des services AI. Cette route repond a une requete GET en renvoyant un objet JSON contenant la liste des services AI.

Demarrage du serveur

```
app.listen(port, () => {
  console.log(`Server listening on http://localhost:${port}`);
});
```

Enfin, nous demarrons notre serveur en ecoutant sur le port specifie. Une fois que le serveur est pret, un message est affiche dans la console pour indiquer que le serveur est en ecoute.

Et voila ! Vous avez maintenant une meilleure comprehension de ce code Node.js. N'hesitez pas a l'experimenter et a l'adapter a vos propres besoins.