Hausaufgabe: REST API starten und testen

Einleitung

In dieser Hausaufgabe wirst du eine einfache REST API testen. Die API erlaubt es, Todos abzurufen, neue Todos zu erstellen und bestehende Todos zu löschen. Du wirst verschiedene Anfragen durchführen und beobachten, wie die API darauf reagiert.

Abgabe

1. Schreibe zu jeder Aufgabe deine Anfrage und die Antwort der API auf.
2. Beschreibe, welche Anfragen funktioniert haben und ob dich etwas überrascht hat.
3. Falls du Fehler hattest, notiere, wie du sie gelöst hast.

Voraussetzungen

1. Stelle sicher, dass du Node.js und npm installiert hast.
2. Kopiere den Code aus der Musterlösung in eine Datei namens `index.js` oder klone das Repository: https://github.com/tomtechstarter/backend-neu/blob/main/index.js
3. Öffne das Terminal und navigiere zu dem Ordner, in dem sich `index.js` befindet.
4. Installiere die benötigten Pakete:

   ```
   npm install express cors
   ```

5. Starte die Anwendung mit:

   ```
   node index.js
   ```

Wenn alles korrekt eingerichtet ist, solltest du im Terminal die Nachricht `Express App is running on http://localhost:5050` sehen.

Aufgabe 1: Basis-Endpunkt testen

1. Öffne Postman oder einen Webbrowser.
2. Sende eine **GET-Anfrage** an den Basis-Endpunkt:
    - URL: `http://localhost:5050/`
    - **Erwartete Antwort**: `"Hello my name is Tom"`
3. Schreibe auf, ob die Antwort korrekt war. Falls nicht, überprüfe deinen Code.

Aufgabe 2: Todos nach Namen abrufen

1. Sende eine **GET-Anfrage**, um Todos mit einem bestimmten Namen zu suchen:
    - URL: `http://localhost:5050/todos/byname?name=Milch holen`
    - **Erwartete Antwort**: Das Todo-Objekt mit dem Namen `"Milch holen"`
2. Teste, was passiert, wenn du einen Namen eingibst, der nicht existiert, z.B. `?name=Unbekannt`.
3. Notiere, welche Antwort du erhältst.

Aufgabe 3: Neues Todo mit falschen Daten erstellen

1. Sende eine **POST-Anfrage**, um ein Todo ohne den erforderlichen `name` oder `userId` zu erstellen:
   - URL: `http://localhost:5050/todos`
   - **Body (JSON)**:

     ```
     {
       "name": "",
       "userId": 0
     }
     ```

   - **Erwartete Antwort**: Eine Fehlermeldung
2. Versuche, ein Todo zu erstellen, bei dem `userId` eine negative Zahl ist, und dokumentiere, was die API antwortet.

Aufgabe 4: Todo aktualisieren

1. Sende eine **PUT-Anfrage**, um den Namen eines bestehenden Todos zu ändern:
   - URL: `http://localhost:5050/todos/update?todoId=2`
   - **Body (JSON)**:

     ```
     {
       "name": "Wasser holen aktualisiert"
     }
     ```

   - **Erwartete Antwort**: Das aktualisierte Todo-Objekt
2. Probiere eine Anfrage mit einer ungültigen `todoId` (z.B. `todoId=999`) und schreibe auf, was passiert.

Aufgabe 5: Todos in verschiedenen Szenarien löschen

1. Sende eine **DELETE-Anfrage**, um ein nicht existierendes Todo zu löschen:
    - URL: `http://localhost:5050/todos?todoId=999`
    - **Erwartete Antwort**: Eine Fehlermeldung, dass das Todo nicht gefunden wurde
2. Teste das Löschen mehrerer existierender Todos in einer einzigen Anfrage und schreibe auf, welche Todos tatsächlich gelöscht wurden.