Hướng dẫn kết nối API cơ bản giữa Vue.js và JSON Server

1. Giới thiệu chung

JSON Server là một công cụ nhẹ giúp tạo API giả lập nhanh chóng từ file JSON. Trong hướng dẫn này, chúng ta sẽ xây dựng một ứng dụng Vue.js kết nối với JSON Server để thực hiện các thao tác CRUD cơ bản

2. Cài Đặt JSON Server

B1: Cài đặt JSON Server

Trước tiên, cần cài đặt JSON Server bằng npm:

```
npm install -g json-server
```

B2: Tạo file db.json

Tạo một file db.json trong thư mục dự án với nội dung sau:

```
{
    "posts": [
        { "id": 1, "title": "Bài viết đầu tiên", "content": "Nội dung bài viết 1" },
        { "id": 2, "title": "Bài viết thứ hai", "content": "Nội dung bài viết 2" }
    ]
}
```

B3: Khởi động JSON Server

Chạy lệnh sau để khởi động server:

```
json-server --watch db.json --port 3000
```

Lưu ý: trường hợp lỗi Failed to connect to server thì chạy lệnh sau:

```
json-server --watch db.json --port 3000 --host 0.0.0.0
```

Sau khi chạy, API sẽ có sẵn tại http://localhost:3000/posts.

3. Thiết lập Vue.js

B1: Tạo dự án Vue

```
npm create vue@latest vue-jsonserver-app
cd vue-jsonserver-app
npm install
```

B2: Cài đặt Axios: Axios là thư viện giúp gửi HTTP request dễ dàng:

```
npm install axios
```

4. Ví dụ thực hiện CRUD đơn giản với JSON Server

a. Ví dụ lấy danh sách bài viết

Mở `src/components/PostList.vue` và thêm đoạn mã sau:

```
<script setup>
import { ref, onMounted } from 'vue';
import axios from 'axios';

const posts = ref([]);
const fetchPosts = async () => {
    try {
        const response = await axios.get('http://localhost:3000/posts');
        posts.value = response.data;
    } catch (error) {
        console.error('Lỗi khi lấy dữ liệu:', error);
    }
};

onMounted(fetchPosts);
</script>

<template>
    <div>
        <h2>Danh sách bài viết</h2>
        <ul>
            <li v-for="post in posts" :key="post.id">
                {{ post.title }} - {{ post.content }}
            </li>
        </ul>
    </div>
</template>
```

b. Ví dụ Thêm bài viết mới:

```
<script setup>
import { ref } from 'vue';
import axios from 'axios';

const title = ref('');
const content = ref('');
const addPost = async () => {
    if (!title.value || !content.value) return;
    try {
        await axios.post('http://localhost:3000/posts', {
            title: title.value,
            content: content.value
        });
        title.value = '';
        content.value = '';
```

```
    } catch (error) {
        console.error('Lỗi khi thêm bài viết:', error);
    }
};
</script>

<template>
    <div>
        <h2>Thêm Bài Viết</h2>
        <input v-model="title" placeholder="Tiêu đề">
        <textarea v-model="content" placeholder="Nội dung"></textarea>
        <button @click="addPost">Thêm</button>
    </div>
</template>
```

c. Ví dụ Cập nhật bài viết

```
const updatePost = async (id) => {
    try {
        await axios.put(`http://localhost:3000/posts/${id}`, {
            title: 'Tiêu đề mới',
            content: 'Nội dung mới'
        });
    } catch (error) {
        console.error('Lỗi khi cập nhật:', error);
    }
};
```

d. Ví dụ Xóa bài viết

```
const deletePost = async (id) => {
    try {
        await axios.delete(`http://localhost:3000/posts/${id}`);
    } catch (error) {
        console.error('Lỗi khi xóa:', error);
    }
};
```

Kết luận: Sinh viên tham khảo cách thực hiện và mở rộng ứng dụng bằng cách hoàn thiện các chức năng mà bài Assignment yêu cầu