Exercise 1 - Async/Await with Multiple API Calls to JSONPlaceholder

Objective: Fetch data about users and their posts from JSONPlaceholder using async/await and display them in a given HTML structure.

Preparation:

- Observe the provided HTML and CSS, focusing on the users' container and individual user structures.
- Familiarize yourself with the endpoint `https://jsonplaceholder.typicode.com`, specifically the `/users` and `/posts?userId=USER_ID` endpoints.
- You can use some of the code from the second exercise in the 'Promises' section.

Step-by-Step Guide:

1. Understand the HTML Structure:

- Locate the container for users (`#users`) and recognize the structure for individual users and posts.
- Identify the classes and IDs in the HTML that will be targeted with JavaScript.

2. Set Up the Base URL:

- Define a constant that will store the base URL.

3. Create Async Function to Fetch Users:

- Write an async function called `fetchUsers`.
- In this function, use the fetch method to request data from the `/users` endpoint.
- Await the response and parse it into JSON, then return it.

4. Create Async Function to Fetch Posts for Each User:

- Write another async function called `fetchPosts` that accepts a user ID.
- Use the fetch method to request data from the `/posts` endpoint using the provided user ID.

- Await the response, parse it into JSON, and return it.

5. Main Async Function to Display Users and Posts:

- Create a main async function called `displayUserAndPosts`.
- Inside this function, include the following steps:
  - Use a try block to handle errors.
  - Fetch all users by calling the `fetchUsers` function and await the result.
  - Iterate over the users and do the following for each user:
    - Create a container for the user and fill it with details such as the name and email.
    - Append this user container to the main users' container.
    - Within a nested try block, fetch posts for the current user by calling the `fetchPosts` function and passing the user's id, and await the result.
    - Iterate over the posts, creating a container for each post and fill it with details like title and body.
    - Append these post containers to the user's posts container.
    - Handle any errors related to fetching posts with a catch block.
  - Handle any errors related to fetching users with a catch block.
- Call this main function to execute the code.

Guidelines:

- Utilize try/catch blocks to gracefully handle errors.
- Regularly test your code for correctness.

Challenges:

- Add loading indicators visible during data fetching.

Note:

- Use browser Developer Tools for debugging and network inspection.
- Focus on understanding the modern async/await syntax for handling promises and organizing code.