Lesson 4: Fetch API with Promises and Async/Await

Lesson Duration: ~ 2 hours

Lesson Goals:

- Understand the basics and use of JavaScript's Fetch API.
- Distinguish between handling fetch requests using Promises vs. async/await syntax.
- Reinforce DOM manipulation by dynamically updating webpage content based on API responses.

---

1. Introduction to Fetch API (20 min.)

- Explanation:
  - What is the Fetch API?
  - Typical use-cases: retrieving data from APIs.
  - Handling responses and errors.

---

2. Demo 1: Fetching with Promises (10 min. [student] + 10 min. [take up])

- Demonstration:
  - Students will see how to make an API request using `.then()` and `.catch()`.
- Task:

```
document.getElementById("btn").addEventListener("click", function () {
  // Fetch data from GitHub API (user: octocat)
  // Handle HTTP response statuses and errors
  // Parse and display JSON data on the page
});
```

- Discussion:
  - Discuss the limitations or potential complexity when handling multiple promises (callback hell).

---

3. Demo 2: Fetching with Async/Await (10 min. [student] + 10 min. [take up])

- Demonstration:
  - Use modern async/await syntax for cleaner asynchronous code.
- Task:

```
document.getElementById("btn").addEventListener("click", async function () {
  // Fetch data from GitHub API (user: octocat)
  // Handle HTTP response statuses and errors
  // Parse and display JSON data on the page
});
```

- Discussion:
  - Compare the readability and ease of maintenance between Promises and async/await.

---

4. Interactive Task: GitHub User Search (30 min. [student] + 20 min. [take up])

- Reinforce concepts learned in previous demos:
  - DOM manipulation
  - Error handling
  - Fetch API with async/await
- Task Description:
  - Students build a simple GitHub user search:
    - Retrieve username from input
    - Display user avatar, username, bio, and GitHub profile link
    - Include loading and error messages to enhance user experience
- Sample Implementation:

```
searchBtn.addEventListener("click", async () => {
  // Handle empty input error
  // Display loading state
  // Fetch user data from GitHub
  // Dynamically create and insert elements (avatar, username, bio, profile link)
  // Handle and display errors clearly
});
```

---

5. Recap & Questions (10 min.)

- Review key takeaways:
    - Using Fetch API
    - Comparing Promises and async/await
    - Dynamic DOM updates based on fetched data
- Q&A session for any unresolved questions or troubleshooting.