JavaScript

async/await

1. async/await is a modern way to handle asynchronous code in JavaScript.
2. It makes working with **Promises** easier and code more readable.
3. **async** makes a function return a Promise.
   It allows synchronous code to execute first
4. **await** pauses the function execution until a Promise resolves.
5. Use response.ok to check if the request was successful.
6. **Error handling is simpler** (using try...catch instead of .catch()).
7. Promise.all() can fetch multiple resources in **parallel** for better performance.

fetch()

1. *fetch() is a **modern replacement for AJAX(XMLHttpRequest)**.*
2. *It is used to returns a **Promise**.*
3. *It works perfectly with async/await*

Comparison of fetch() vs. XMLHttpRequest (AJAX)

Feature	`fetch()`	`XMLHttpRequest` (AJAX)
Supports Asynchronous Requests	✅ Yes	✅ Yes
Uses Promises	✅ Yes	❌ No (callback-based)
Works with `async/await`	✅ Yes	❌ No
Auto JSON Parsing	✅ Yes (`response.json()`)	❌ No (manual `JSON.parse()`)
Simpler Syntax	✅ Yes	❌ No (more code required)
Supports Modern APIs	✅ Yes	❌ Limited
Optimized for Performance	✅ Yes	❌ No
Used in Modern Web Apps	✅ Yes	❌ No (outdated)

Making Different HTTP Requests with fetch()

1️⃣ GET Request (Fetching Data)

A GET request is used to retrieve data from the server. It is the most common HTTP request.

```
async function getData() {
  let response = await fetch("https://jsonplaceholder.typicode.com/posts/1");
  let data = await response.json();
  console.log(data);
}

getData();
```

2️⃣ POST Request (Sending Data)

A POST request is used to send data to the server. It is commonly used when submitting forms or adding new resources.

```
async function postData() {
  let response = await fetch("https://jsonplaceholder.typicode.com/posts", {
    method: "POST",
    headers: {
      "Content-Type": "application/json"
    },
    body: JSON.stringify({ title: "New Post", body: "Hello World!", userId: 1 })
  });

  let data = await response.json();
  console.log("Created Post:", data);
}

postData();
```

3️⃣ PUT Request (Updating Data)

A PUT request is used to update an existing resource on the server. It is often used when modifying data.

```
async function updateData() {
    let response = await fetch("https://jsonplaceholder.typicode.com/posts/1", {
        method: "PUT",
        headers: {
            "Content-Type": "application/json"
        },
        body: JSON.stringify({ title: "Updated Title", body: "Updated Body" })
    });

    let data = await response.json();
    console.log("Updated Post:", data);
}

updateData();
```

4️⃣ DELETE Request (Deleting Data)

A DELETE request is used to delete a resource on the server. It is typically used to remove data.

```
async function deleteData() {
    let response = await fetch("https://jsonplaceholder.typicode.com/posts/1", {
        method: "DELETE"
    });

    console.log("Deleted Post:", response.status);
}

deleteData();
```