MuhammadOwais

File Uploads with Fetch and FormData in JavaScript

MuhammadOwais

Understanding the Basics

Before we dive into the code, let's understand the basics of file uploads in JavaScript. When a user selects a file to upload, the file is represented as a Blob (Binary Large Object) in JavaScript. Blobs encapsulate raw data, such as images or documents, and can be easily transmitted over the network.

To upload files from a web application, we'll use the Fetch API to send a POST request to a server-side endpoint. We'll also utilize the FormData interface, which provides a convenient way to construct and send form data, including files, via XMLHttpRequest or Fetch.

MuhammadOwais

File Uploads with Fetch and FormData

Now, let's dive into the code and implement file uploads using Fetch and FormData. Below is a basic example demonstrating how to upload a single file from a web form

```
const uploadFile = async (file) => {
  const formData = new FormData();
  formData.append('file', file);

  try {
    const response = await fetch('/upload', {
      method: 'POST',
      body: formData
    });

    if (response.ok) {
      console.log('File uploaded successfully!');
    } else {
      console.error('Failed to upload file:', response.statusText);
    }
  } catch (error) {
    console.error('Error uploading file:', error);
  }
};

// Example usage:
const fileInput = document.getElementById('fileInput');
fileInput.addEventListener('change', (event) => {
  const file = event.target.files[0];
  uploadFile(file);
});
```

MuhammadOwais

Handling Multiple File Uploads

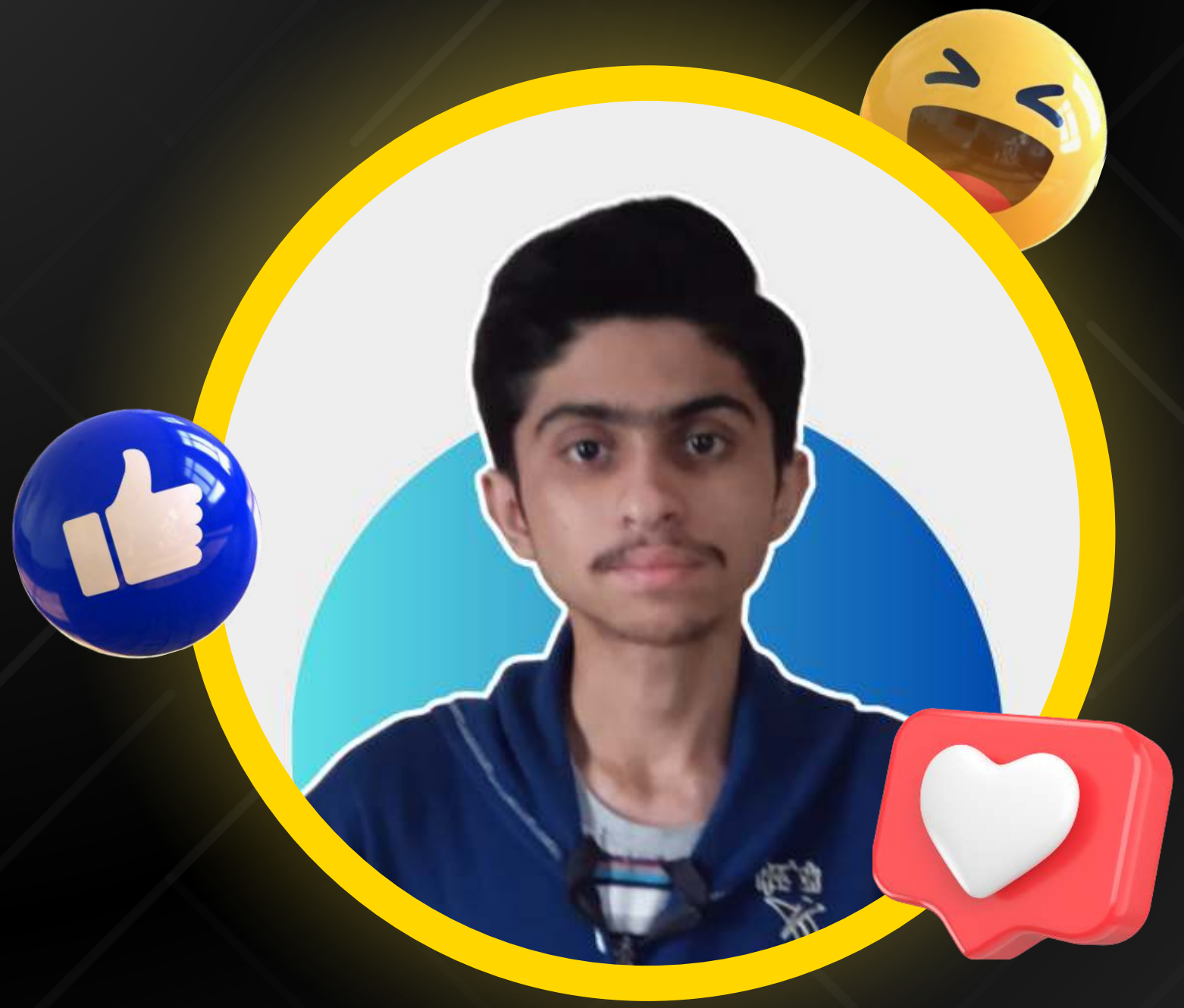
What if we want to allow users to upload multiple files simultaneously? Fear not! FormData makes it easy to handle multiple file uploads. Let's extend our example to support multiple file uploads

```
const uploadFiles = async (files) => {
  const formData = new FormData();
  files.forEach((file) => {
    formData.append('files[]', file);
  });

  try {
    const response = await fetch('/upload', {
      method: 'POST',
      body: formData
    });

    if (response.ok) {
      console.log('Files uploaded successfully!');
    } else {
      console.error('Failed to upload files:', response.statusText);
    }
  } catch (error) {
    console.error('Error uploading files:', error);
  }
};

// Example usage:
const fileInput = document.getElementById('fileInput');
fileInput.addEventListener('change', (event) => {
  const files = Array.from(event.target.files);
  uploadFiles(files);
});
```



MUHAMMAD OWAIS

FULL-STACK DEVELOPER | GRAPHIC & UI/UX DESIGNER | COPYWRITER