Here’s a JavaScript function using **Tesseract.js**, a browser-compatible OCR (optical character recognition) library, to extract **text from images of any common type** (JPEG, PNG, WebP, etc.).

**Step 1: Include Tesseract.js (Browser or Node.js)**

**Browser:**

Add this to your HTML:

html

CopyEdit

<script src="https://cdn.jsdelivr.net/npm/tesseract.js@5.0.1/dist/tesseract.min.js"></script>

**Node.js:**

Install:

bash

CopyEdit

npm install tesseract.js

**Step 2: JavaScript Function**

javascript

CopyEdit

// Works in both browser (with image element or File) and Node (with path or buffer)

async function extractTextFromImage(imageSrc) {

const { createWorker } = Tesseract;

const worker = await createWorker({

logger: m => console.log(m.status, m.progress) // Optional: progress logging

});

try {

await worker.loadLanguage('eng');

await worker.initialize('eng');

const { data: { text } } = await worker.recognize(imageSrc);

await worker.terminate();

return text.trim();

} catch (err) {

console.error('OCR failed:', err);

return '';

}

}

**Example Usage**

**Browser (with File input):**

html

CopyEdit

<input type="file" id="imgInput" accept="image/\*">

<script>

document.getElementById('imgInput').addEventListener('change', async (e) => {

const file = e.target.files[0];

const text = await extractTextFromImage(URL.createObjectURL(file));

console.log('Extracted text:', text);

});

</script>

**Node.js:**

javascript

CopyEdit

extractTextFromImage('path/to/image.png').then(text => {

console.log('Text:', text);

});

Let me know if you want it to support **multiple languages**, **batch images**, or if you need a **backend script** instead.

**You said:**

And a js function that will take the text from regular file types like docx or pdf and be able to feed it into an llm

**ChatGPT said:**

Here’s a JavaScript setup that extracts text from **PDF** and **DOCX** files so you can feed the result into an LLM:

**Step 1: Install Required Libraries (for Node.js)**

bash

CopyEdit

npm install pdf-parse mammoth

**Step 2: JavaScript Function**

javascript

CopyEdit

const fs = require('fs');

const pdfParse = require('pdf-parse');

const mammoth = require('mammoth');

async function extractTextFromFile(filePath) {

const ext = filePath.split('.').pop().toLowerCase();

if (ext === 'pdf') {

const dataBuffer = fs.readFileSync(filePath);

const data = await pdfParse(dataBuffer);

return data.text.trim();

}

if (ext === 'docx') {

const data = fs.readFileSync(filePath);

const result = await mammoth.extractRawText({ buffer: data });

return result.value.trim();

}

throw new Error('Unsupported file type. Only PDF and DOCX supported.');

}

**Example Usage:**

javascript

CopyEdit

extractTextFromFile('sample.pdf').then(text => {

console.log('Text for LLM:', text);

});

This gives you clean text output from DOCX or PDF, suitable for feeding directly into an LLM API or local model.

Want browser-compatible code, or support for .txt, .odt, or .pptx too?