hospital-referral-pro/

├── .env.example

├── README.md

├── package.json

├── /pages

│ ├── index.js

│ └── api/

│ └── upload.js

├── /components

│ └── UploadForm.js

└── /utils

└── processDocument.js

{

"name": "hospital-referral-pro",

"version": "1.0.0",

"scripts": {

"dev": "next dev",

"build": "next build",

"start": "next start"

},

"dependencies": {

"next": "latest",

"react": "latest",

"react-dom": "latest",

"axios": "^1.0.0",

"mammoth": "^1.4.19",

"pdf-parse": "^1.1.1",

"formidable": "^2.0.1",

"openai": "^4.0.0"

}

}

OPENAI\_API\_KEY=your\_openai\_key\_here

# Hospital Referral Pro Portal

Upload redacted patient documents and receive GPT-powered extractions for ICD‑10 codes, service triggers, and referral matching.

## Usage

1. Upload a PDF or Word discharge summary.

2. Optional: Enter ZIP code for localized referrals.

3. GPT extracts medical needs and matches services.

## Setup

- Copy `.env.example` to `.env.local`

- Add your OpenAI API key

- Deploy on Vercel

## File Structure

- `/pages`: Upload UI + API handler

- `/components`: UploadForm

- `/utils`: Document parser + GPT prompt builder

import UploadForm from '../components/UploadForm';

export default function Home() {

return (

<div style={{ padding: 40 }}>

<h1>Hospital Referral Pro</h1>

<UploadForm />

</div>

);

}

import React, { useState } from 'react';

import axios from 'axios';

export default function UploadForm() {

const [file, setFile] = useState(null);

const [zip, setZip] = useState('');

const [result, setResult] = useState(null);

const [loading, setLoading] = useState(false);

const handleSubmit = async e => {

e.preventDefault();

if (!file) return alert('Please select a file.');

setLoading(true);

const form = new FormData();

form.append('file', file);

form.append('zip', zip);

try {

const { data } = await axios.post('/api/upload', form, {

headers: { 'Content-Type': 'multipart/form-data' }

});

setResult(data);

} catch (err) {

alert(err.response?.data?.error || err.message);

}

setLoading(false);

};

return (

<div>

<form onSubmit={handleSubmit}>

<h2>Upload Redacted Document</h2>

<input type="file" accept=".pdf,.docx" onChange={e => setFile(e.target.files[0])} />

<input type="text" placeholder="ZIP code (optional)" value={zip} onChange={e => setZip(e.target.value)} />

<button type="submit" disabled={loading}>{loading ? 'Processing…' : 'Extract Needs'}</button>

</form>

{result && (

<div>

<h3>Extraction Result</h3>

<pre>{JSON.stringify(result, null, 2)}</pre>

</div>

)}

</div>

);

}