From Paper Chaos to AI-Powered Search

Building an Intelligent Document Management System

The Problem

> I had 10 years of documents - tax returns, medical records, contracts, receipts - all scanned into PDFs. The built-in search was useless.
>
> Searching "insurance" wouldn't find documents titled "Geico Policy 2023" or scanned forms where the OCR mangled "insurance" into "insuranc3".
>
> **I needed a system that could:**
>
> - Actually read poor-quality scans
> - Understand what documents were *about*, not just match keywords
> - Auto-organize everything without manually tagging 3,000+ files

The Solution

I built **two custom services** that extend **paperless-ngx** (open-source document management):

> **What I Built vs. What I Used:**
> *Base Platform (not my code):* paperless-ngx - handles document

storage, UI, basic OCR
My Custom Services: paperless-gpt (Go) + paperless-chroma (Python)

Layer 1: Intelligent OCR (paperless-gpt)

Traditional OCR choked on faded receipts, handwritten notes, multi-column layouts, and low-resolution scans.

I built **paperless-gpt** - a Go service that routes documents through multiple OCR engines:

- **GPT-4 Vision** for complex layouts
- **Google Document AI** for forms
- **Ollama** (local models) for privacy-sensitive docs

A worker pool processes documents concurrently. Each document gets clean, accurate text - not garbled output from basic OCR.

Layer 2: Auto-Classification

Once text is extracted, the system uses LLMs to:

- Generate meaningful titles ("Geico Auto Insurance Renewal - March 2023")
- Apply relevant tags automatically
- Identify correspondents (who sent it)
- Extract document dates

No more manual organizing. Drop a document in, get it classified in seconds.

Layer 3: Semantic Search (paperless-chroma-integration)

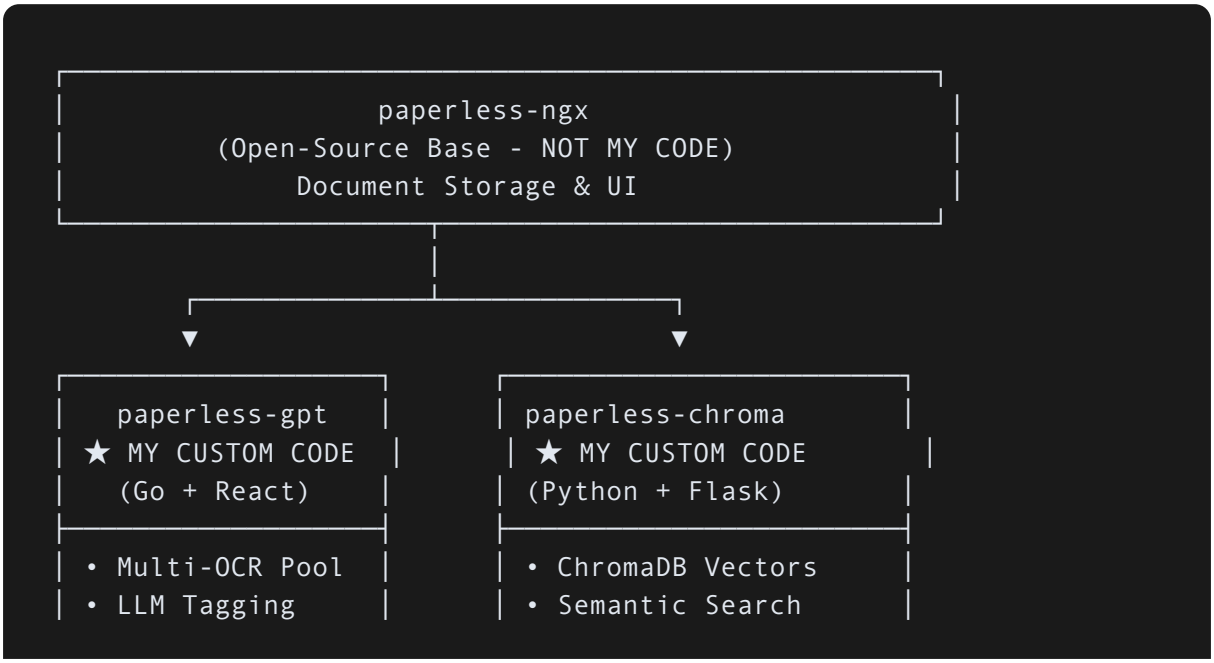
The final layer adds **concept-based search** using ChromaDB (vector database):

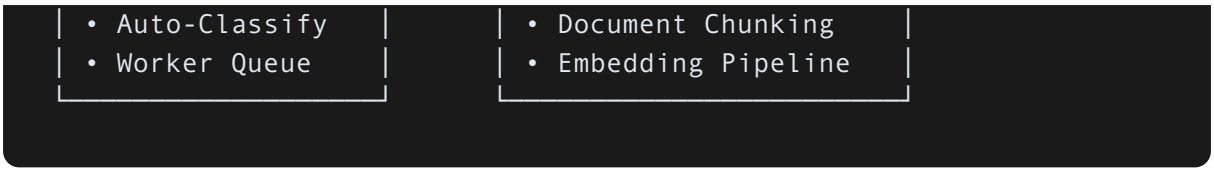
Query: "car accident paperwork"
Finds: Insurance claims, repair estimates, police reports - even if none contain those exact words

Query: "money I owe"
Finds: Credit card statements, loan documents, invoices marked "due"

Documents are chunked into overlapping segments, converted to embeddings via sentence transformers, and stored in ChromaDB. Searches match *meaning*, not keywords.

The Architecture





Technical Highlights

Multi-Provider OCR - Abstracted interface lets me swap OCR engines per document type. Medical forms go to Google Document AI. Personal notes stay local with Ollama.

Worker Pool Architecture - Go's concurrency handles batch processing. 4 workers process documents simultaneously with job status tracking and retry logic.

Vector Search - BAAI/bge-base-en-v1.5 embeddings with 1000-character chunks and 200-character overlap. Finds conceptually related documents across 3,000+ files in milliseconds.

Docker Compose Orchestration - Three services (paperless-ngx, paperless-gpt, ChromaDB) coordinated via compose. Single `docker-compose up` deploys the entire stack.

The Result

3,000+
Documents Auto-Organized

90%+
OCR Accuracy on Poor Scans

0
Manual Tagging Required

<1s
Semantic Search Response

> What used to take hours of manual organization now happens automatically. I drop documents in a folder; AI handles the rest.

Technologies Used

Go Python React Flask GPT-4 Vision Google Document AI Ollama Sentence Transformers ChromaDB Docker Docker Compose paperless-ngx