Local AI for Accounting and Tax: Keep Your Financial Data Off the Cloud

March 6, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** Local AI handles transaction categorization, client letter drafting, receipt OCR, and Q&A over tax documents — all without your financial data leaving your machine. It cannot do tax calculations reliably (LLMs hallucinate numbers), can't file returns, and doesn't replace professional judgment on complex positions. Best setup for most practitioners: Ollama + Open WebUI + Qwen 3.5 9B on 8GB VRAM for general work, or Qwen 3.5 35B-A3B on 24GB VRAM for heavier analysis. Add a RAG pipeline over IRS publications for tax research. 16GB system RAM minimum, GPU recommended but not required for smaller models.

More on this topic: VRAM Requirements · Best Local LLMs for Mac · Open WebUI Setup · Run Your First Local LLM · Planning Tool

In February 2026, a federal judge ruled that documents generated through a consumer AI tool lost attorney-client privilege because the platform's terms allowed the provider to use inputs for training and disclose data to regulators. The defendant had typed legal strategy into Anthropic's Claude. The court said that was equivalent to telling a third party.

If you're an accountant or tax professional typing client financials into ChatGPT, the same logic applies. Cloud AI terms of service typically reserve the right to store, process, and train on your inputs. A 2025 KPMG survey found that 46% of US workers have uploaded sensitive company data to public AI platforms. The AICPA has explicitly warned against sharing client information with AI tools without proper safeguards.

Local AI sidesteps the problem. Your data never leaves your machine. No third-party terms of service. No training on your inputs. No "we may disclose to regulators" clause. The model runs on your hardware, processes your documents in memory, and the results stay on your SSD.

Here's what that looks like for accounting and tax work.

What local AI can actually do for you today

These are tasks I'd trust a local model to handle right now, with appropriate human review.

Categorize transactions from bank exports

Export your transactions as CSV. Feed them to a local model with a prompt like "categorize each transaction as one of: office supplies, travel, meals, software, professional services, other." Qwen 3.5 9B handles this well. It won't be perfect on every line item, but it'll get 80-90% right and save you hours of manual sorting. Review the output, fix the edge cases, move on.

Draft client communications

Engagement letters, status updates, year-end summaries. Give the model context about the client situation and ask for a draft. The writing quality from current 9B models is good enough to work from. You'll edit it, but the first draft takes minutes instead of an hour.

Summarize tax law changes

Drop a new IRS notice or revenue ruling into the chat and ask for a plain-language summary. Local models handle this well because it's comprehension, not calculation. "What changed in the 2026 standard deduction rules?" is a question a local model can answer from the document you provide.

Extract data from receipts and invoices

This is where vision models earn their keep. Qwen 3.5's small models (0.8B through 9B) are natively multimodal — they handle images without a separate vision adapter. Drag a photo of a receipt into Open WebUI, and the model reads the vendor, date, amount, and line items. Accuracy varies with image quality, but for typical printed receipts it's solid.

```
# Send an image to Qwen 3.5 9B for receipt extraction
ollama run qwen3.5:9b "Extract the vendor name, date, total, and line items from this receipt" <
```

For higher accuracy on messy or handwritten receipts, Qwen3-VL models handle low-light, blurred, and tilted images better than the standard multimodal models. The 7B variant runs on consumer GPUs.

Q&A over tax documents with RAG

This is the one worth setting up properly. Load IRS publications, state tax guides, and your own internal memos into a local RAG (retrieval-augmented generation) pipeline. Then ask questions

in plain English: “What are the 2026 Section 179 deduction limits?” The model searches your local document store, finds the relevant passage, and answers based on the actual source text.

I’ll cover the setup below.

What local AI cannot do (be honest with yourself here)

Tax calculations

LLMs hallucinate numbers. They will confidently tell you a depreciation schedule adds up to a figure it doesn’t. They will apply the wrong tax rate to the wrong bracket. They will carry errors forward through multi-step calculations without flinching.

Never trust a language model’s math output for anything you’d put on a return. Use a spreadsheet, use your tax software, use a calculator. The model is a research assistant and a drafting tool, not an adding machine. The Stanford Law School paper on LLMs as tax attorneys found that general-purpose LLMs can produce plausible but factually incorrect tax guidance.

File or submit anything

No local LLM has API integrations with the IRS, state tax systems, or e-filing platforms. None are coming. Filing is a separate process that happens in your tax software. AI helps you prepare the work. It doesn’t submit it.

Replace professional judgment

A model can summarize a revenue ruling. It can’t tell you whether your client’s situation falls on the right side of a gray area. Complex tax positions require professional judgment, knowledge of the client’s full picture, and willingness to defend the position under audit. A model that read some IRS publications doesn’t have any of that.

Use local AI to speed up the routine work. Apply your own judgment to the work that matters.

Best models for financial work

Model	VRAM (Q4)	Best For	Speed
Qwen 3.5 9B	~6.6 GB	General financial Q&A, transaction categorization, client letters, receipt OCR	Fast — recommended backbone
Qwen 3.5 35B-A3B	~17 GB	Complex analysis, longer documents, multi-step research	112 tok/s on RTX 3090
Qwen 3.5 27B	~17 GB	Heavier reasoning, tax law interpretation	Moderate
QwQ-32B	~20 GB	Complex reasoning, multi-step tax analysis	Moderate
DeepSeek-R1-Distill-14B	~9 GB	Chain-of-thought reasoning on complex questions	Moderate

Qwen 3.5 9B is the starting point. It handles categorization, drafting, summarization, and vision tasks. Tool calling works in Ollama v0.17.6+, so you can build workflows where the model calls functions to look up data or write files.

If you have 24GB VRAM, the 35B-A3B MoE variant is worth trying. It runs at 112 tok/s on an RTX 3090 because only 3B parameters are active per token — faster than most dense 7B models. The trade-off is slightly lower quality on hard reasoning compared to the dense 27B, but for financial document work the speed matters more.

```
# Install or update Ollama
curl -fsSL https://ollama.com/install.sh | sh

# Pull the recommended model
ollama pull qwen3.5:9b

# Or the speed option for 24GB VRAM
ollama pull qwen3.5:35b-a3b
```

The setup: Ollama + Open WebUI

The practical stack for most practitioners is Ollama for the model backend and Open WebUI for the interface. Open WebUI gives you a ChatGPT-style browser interface with drag-and-drop document upload, conversation history, and RAG built in.

```
# Start Ollama
ollama serve

# Run Open WebUI (Docker)
docker run -d -p 3000:8080 \
  --add-host=host.docker.internal:host-gateway \
  -v open-webui:/app/backend/data \
  --name open-webui ghcr.io/open-webui/open-webui:main
```

Open `http://localhost:3000`, select your Qwen 3.5 model, and start chatting. Upload a CSV of transactions by dragging it into the chat window. Upload a receipt image the same way.

For a full walkthrough, see our Open WebUI setup guide.

If you'd rather avoid the command line entirely, LM Studio offers a desktop GUI. Download it, search for "Qwen 3.5 9B," click download, and start chatting. No terminal, no Docker.

Sample prompts

These are ready to copy and paste. Modify the details for your situation.

Categorize transactions:

```
Here are transactions from my client's business checking account.
Categorize each one using standard Schedule C categories
(advertising, car/truck, office, supplies, utilities, meals,
professional services, other).
Format as a table with columns: Date, Description, Amount, Category, Notes.

Transactions:
[paste your CSV data here]
```

Draft a client memo:

```
Write a brief memo to my client explaining:
- Their total estimated tax liability for 2025 is approximately $X
- They underpaid quarterly estimates by approximately $Y
- They may owe a penalty under IRC Section 6654
- Options to reduce the penalty (annualized income method, etc.)

Keep it professional but not intimidating. They're a sole proprietor
who runs a small landscaping business.
```

Research a tax question:

```
My client is a freelance photographer who works from a dedicated
room in their apartment. They want to claim the home office
deduction. Explain the simplified method vs regular method,
square footage limitations, what expenses qualify, and how it
affects depreciation. Cite the relevant IRC sections and IRS
publications.
```

Reconciliation helper:

```
I have two lists of transactions -- one from the bank statement
and one from QuickBooks. Find the discrepancies:

Bank statement transactions:
[paste bank data]

QuickBooks transactions:
[paste QB data]

List any transactions that appear in one but not the other,
and any amount mismatches for matching transactions.
```

RAG for tax research

RAG is what turns a general-purpose model into a tax research tool. Instead of relying on the model's training data (which may be outdated or incomplete), you feed it the actual source documents and it answers from those.

What to load

- IRS Publication 17 (general tax guide)
- Publications relevant to your practice (535 for business expenses, 946 for depreciation, etc.)

- State tax guides for your jurisdiction
- Your firm's internal memos and position papers
- Recent IRS notices and revenue rulings

How to set it up

Open WebUI has built-in RAG. Upload PDFs through the Knowledge section, and they get chunked, embedded, and stored locally. When you ask a question, the system retrieves relevant chunks and feeds them to the model as context.

For better results:

- IRS publications have dense, cross-referenced sections. 512-token chunks with 50-token overlap work for most tax documents. If the model keeps missing context that spans two sections, increase the chunk size.
- Open WebUI supports local embedding models through Ollama. Pull `nomic-embed-text` for a good general-purpose option that runs on CPU.
- Don't load every IRS publication at once. That creates noise. Load the publications relevant to your current work and add more later.

```
# Pull a local embedding model
ollama pull nomic-embed-text
```

The result: you type "what are the meal deduction rules for 2026 business travel?" and the model searches your local copy of Publication 463, finds the relevant section, and gives you an answer with the source passage cited. All on your machine.

Tools worth knowing about

TaxHacker is a self-hosted accounting app built for freelancers and small businesses. Upload receipts and invoices, and it uses AI to extract dates, amounts, vendors, and line items into a structured database. It currently works with OpenAI and Mistral APIs, with local LLM support coming. If you're comfortable with Docker, it's a solid way to organize client receipts. The data stays on your server.

Hardware reality

Setup	What You Need	What You Get
Minimum	16GB RAM, no GPU	Qwen 3.5 9B on CPU — slow (3-5 tok/s) but works for short tasks
Comfortable	16GB RAM + 8GB VRAM (RTX 3060)	Qwen 3.5 9B on GPU — fast, handles daily work
Ideal	32GB RAM + 24GB VRAM (RTX 3090)	Qwen 3.5 35B-A3B at 112 tok/s, comfortable RAG with large document sets
Mac	M-series with 16GB+ unified memory	Qwen 3.5 9B runs well through MLX. See our Mac guide

If you already have a desktop with a recent NVIDIA GPU, you probably have enough hardware to start. A used RTX 3060 12GB runs $170-200 and handles everything in this guide except the larger models.

CPU-only inference works for occasional use — a quick question about a tax rule, a short letter draft. For batch processing hundreds of transactions or doing RAG over large document sets, a GPU makes the difference between usable and painful.

For exact VRAM math, see our VRAM requirements guide.

The privilege argument

The Heppner ruling goes further than the specific facts of that case. The court applied a simple principle: if the platform's terms let the provider see your data, you don't have a reasonable expectation of confidentiality. That expectation is a requirement for both attorney-client privilege and work-product protection.

Tax professionals have similar confidentiality obligations. Section 7216 of the Internal Revenue Code makes it a criminal offense for tax preparers to disclose client information without consent. When you paste a client's K-1 data into ChatGPT, the data goes to OpenAI's servers, gets processed under their terms, and may be retained. Whether that constitutes "disclosure" under 7216 is a question nobody wants to test in front of a judge.

Local AI makes that question disappear. The data stays on your hardware. There is no third party. No disclosure, no consent question, no terms of service to parse.

Who this is for

Solo practitioners who want AI help with routine work but can't justify the risk of sending client data to the cloud. Small firms that want document analysis and drafting assistance without paying for enterprise AI tiers with confidentiality agreements. Freelancers doing their own books who want transaction categorization and receipt extraction without a subscription.

This is not a replacement for tax software, for professional judgment, or for an accountant. It's a private research assistant that happens to run on your desk. It makes the boring parts faster. The hard parts are still yours.

```
# The 5-minute start
curl -fsSL https://ollama.com/install.sh | sh
ollama pull qwen3.5:9b
ollama run qwen3.5:9b
```

Get notified when we publish new guides.

Subscribe — free, no spam