AnythingLLM Setup Guide: Chat With Your Documents Locally

February 8, 2026 · by Mark Bartlett

Download this guide as PDF

> **Quick Answer:** AnythingLLM is the fastest way to chat with your documents locally. Install Ollama, pull a model (qwen2.5:7b), download the AnythingLLM desktop app, point it at localhost:11434, create a workspace, drag in your files. Five minutes from download to asking questions about your PDFs. It handles embedding, chunking, and retrieval automatically — no Python, no LangChain, no configuration files. You need 8GB+ VRAM for a good experience (chat model + embedding model running simultaneously). Works on CPU but expect 5-10 second response times.

📚 **Related:** Local RAG Guide · Open WebUI Setup · Ollama Setup · Best LLMs for RAG · Planning Tool

You have a stack of PDFs — research papers, contracts, technical docs, meeting notes. You want to ask questions and get answers that reference the actual content. Not hallucinated summaries. Actual quotes from actual pages.

That's RAG (Retrieval Augmented Generation), and normally it requires Python, an embedding pipeline, a vector database, and enough LangChain boilerplate to make you question your career choices.

AnythingLLM skips all of that. It's a desktop app — 54K+ GitHub stars, MIT license — that connects to your local Ollama instance and gives you point-and-click document chat. Upload files, ask questions, get answers grounded in your documents. No coding required.

Here's how to set it up and get useful results.

What You Need

Hardware

Component	Minimum	Recommended
RAM	8 GB	16 GB

Component	Minimum	Recommended
VRAM	None (CPU mode)	8 GB+
Disk	10 GB free	20 GB+ (models + embeddings)
CPU	Any x86_64 or ARM64	4+ cores

AnythingLLM itself is lightweight. The hardware requirements come from running the LLM and embedding model alongside it. With 8GB VRAM, you can run a 7B chat model and an embedding model simultaneously. On CPU only, expect 5-10 second response times for a 7B model.

Software

- **Ollama** (or LM Studio) — your local inference engine
- **A chat model** — Qwen 2.5 7B or Llama 3.1 8B recommended
- **An embedding model** — nomic-embed-text (pulled automatically or manually)

Step 1: Install Ollama and Pull Models

If you already have Ollama running, skip to Step 2.

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

# Pull a chat model (pick one)
ollama pull qwen2.5:7b        # Best general-purpose at this size
ollama pull llama3.1:8b        # Alternative

# Pull an embedding model
ollama pull nomic-embed-text   # ~275 MB, fast, good quality
```

Verify Ollama is running:

```
curl http://localhost:11434/api/tags
```

You should see your pulled models listed. If you get a connection error, run `ollama serve` in a separate terminal.

For detailed Ollama setup, see the beginner's guide.

Step 2: Install AnythingLLM

Desktop App (Recommended)

Download from anythingllm.com — available for Windows, macOS, and Linux.

- **Windows:** Run the .exe installer
- **macOS:** Drag to Applications (requires macOS 13.6+ for Apple Silicon)
- **Linux:** AppImage — make it executable and run

The desktop app includes a built-in embedding model (all-MiniLM-L6-v2), so you can start without pulling nomic-embed-text. But the Ollama embedding model is better — configure it in the next step.

Docker (Alternative)

```
docker pull mintplexlabs/anythingllm

docker run -d -p 3001:3001 \
  --add-host=host.docker.internal:host-gateway \
  -v anythingllm_data:/app/server/storage \
  --name anythingllm \
  mintplexlabs/anythingllm
```

Then open `http://localhost:3001` in your browser.

The `--add-host` flag lets the container reach Ollama on your host machine. Without it, AnythingLLM can't connect to localhost:11434.

Step 3: Connect to Ollama

On first launch, AnythingLLM walks you through setup:

1. **LLM Provider:** Select "Ollama" → set URL to `http://localhost:11434` (or `http://host.docker.internal:11434` if running Docker)

2. **Chat Model:** Pick your model from the dropdown (qwen2.5:7b or llama3.1:8b)
3. **Embedding Provider:** Select “Ollama” → pick `nomic-embed-text`
4. **Vector Database:** Leave as default (LanceDB — built-in, no setup)

That’s the core configuration. Everything else can stay at defaults.

Step 4: Create a Workspace and Upload Documents

Create a Workspace

Click “New Workspace” and give it a name — something descriptive like “Research Papers” or “Project Docs.” Each workspace has its own document collection and chat history.

Upload Documents

Click the upload icon in your workspace and drag files in. Supported formats:

Format	Quality	Notes
PDF	Good	Text-based PDFs work great. Scanned PDFs need OCR first
TXT / Markdown	Excellent	Cleanest results
DOCX	Good	Formatting stripped, text preserved
Web URLs	Good	Paste a URL, AnythingLLM scrapes the content
Code files	Good	.py, .js, .ts, .md, etc.
CSV	Fair	Parsed as text rows

After uploading, AnythingLLM automatically:

1. Splits your document into chunks (default: 1,000 tokens)
2. Generates embeddings for each chunk using your embedding model
3. Stores the embeddings in the vector database

Large PDFs (100+ pages) take a minute or two to process. You’ll see a progress indicator.

What Happens Under the Hood

When you ask a question, AnythingLLM:

1. Embeds your question using the same embedding model
2. Searches the vector database for the most relevant chunks
3. Sends those chunks + your question to the chat model
4. The model answers based on the retrieved context

This is why answers reference your actual documents instead of making things up — the model sees the relevant text before responding.

Step 5: Start Chatting

Type a question in the chat box. Good first questions:

- "Summarize this document"
- "What are the key findings?"
- "What does section 3 say about [topic]?"
- "List all dates and deadlines mentioned"

AnythingLLM shows which document chunks it retrieved for each answer. Check these — if the retrieved chunks are irrelevant, the answer won't be good regardless of how capable your model is.

Settings That Matter

Most defaults are fine. These are the ones worth adjusting.

Chunk Size

Default: 1,000 tokens. **When to change:** If your documents have very short sections (emails, notes), reduce to 500. If they have long continuous arguments (legal documents, research papers), increase to 1,500-2,000.

Smaller chunks = more precise retrieval but risk missing context. Larger chunks = more context per retrieval but risk including irrelevant text.

Top K (Number of Retrieved Chunks)

Default: 4. **When to change:** Increase to 6-8 if answers seem incomplete or miss relevant information. Decrease to 2-3 if answers include too much off-topic content.

More chunks means more context for the model, but also more noise and slower responses (larger prompt).

Chat Mode

AnythingLLM has two modes:

- **Chat:** Normal conversation with RAG. The model uses retrieved documents plus conversation history.
- **Query:** Pure question-answering. No conversation history, just document retrieval + answer. Better for factual lookups, worse for follow-up questions.

Use Chat mode by default. Switch to Query mode when you need precise, standalone answers without conversational drift.

Embedding Model

If you started with the built-in embedder (all-MiniLM-L6-v2), consider switching to nomic-embed-text via Ollama. It produces better retrieval quality, especially for technical documents.

Important: Changing your embedding model requires re-embedding all documents. AnythingLLM will prompt you to do this. It's not a quick operation on large collections.

Tips for Better Results

Use Descriptive Questions

Bad: "Tell me about the budget" Good: "What is the total projected budget for Q3 2026 according to the financial report?"

Specific questions produce better embedding matches, which produce better retrieval, which produce better answers.

One Topic Per Workspace

Don't dump 50 unrelated documents into one workspace. Create separate workspaces for separate topics — "Tax Documents," "Research Papers," "Project Specs." This keeps the vector database focused and retrieval accurate.

Check Your Sources

AnythingLLM shows which chunks were retrieved for each answer. If the citations look wrong, the answer is probably wrong too — even if it sounds confident. Always verify claims against the source chunks.

Pre-Process Messy PDFs

Scanned PDFs, PDFs with complex tables, and PDFs with multi-column layouts produce poor chunks. If you have important documents in these formats, convert them to clean text or markdown first. Tools like `pdftotext` or Marker (AI-based PDF extraction) produce much cleaner input.

Model Choice Matters

For RAG specifically, instruction-following ability matters more than raw intelligence. A 7B model that reliably quotes source text beats a 14B model that paraphrases and adds its own interpretation. Qwen 2.5 7B and Llama 3.1 8B are both good choices. See the best models for RAG guide for detailed comparisons.

Limitations

Table Extraction

PDF tables are AnythingLLM's weakest point. Complex tables with merged cells, multi-line entries, or nested headers often get garbled during text extraction. If table data matters, consider extracting tables separately and uploading them as CSV or markdown.

Very Large Document Sets

With hundreds of documents, embedding and retrieval slow down. The built-in LanceDB handles thousands of documents fine, but processing time during upload scales linearly. Budget 1-2 minutes per 100 pages for initial embedding.

Multi-Document Reasoning

"Compare the conclusions of Paper A and Paper B" requires the retriever to pull relevant chunks from both documents simultaneously. This works sometimes but isn't reliable — the retriever might grab chunks from only one document. For cross-document analysis, summarize each document separately first, then ask comparison questions in a fresh chat.

Embedding Model Lock-In

Embedding models are set system-wide in AnythingLLM, not per workspace. If you switch embedding models, every workspace needs re-embedding. Pick your embedder once and stick with it.

Fixed Ollama Timeout

AnythingLLM has a built-in 5-minute timeout for Ollama responses. If your model is slow (CPU inference, very large models), long responses may get cut off. Not an issue with 7-8B models on GPU, but can bite you with larger models on limited hardware.

AnythingLLM vs Alternatives

Feature	AnythingLLM	Open WebUI	PrivateGPT
Primary strength	RAG / documents	Chat interface	Customizable RAG
Setup difficulty	Easy (desktop app)	Easy (Docker)	Moderate (Python)
RAG quality	Good (built-in)	Basic (add-on)	Good (LlamaIndex)
Agent support	Yes (built-in skills)	Yes (tools/functions)	Limited
Multi-model switching	Yes	Yes (better)	Yes
UI quality	Good	Best	Basic
GitHub stars	54K+	120K+	20K+
Best for	Document Q&A	General chat	Dev-friendly RAG

Pick AnythingLLM if your primary goal is chatting with documents. It has the best out-of-the-box RAG experience with zero coding.

Pick Open WebUI if you want a general-purpose chat interface that also does RAG. Better for daily chat use, model switching, and community features.

Pick PrivateGPT if you're a developer who wants full control over the RAG pipeline and is comfortable with Python.

Best Use Cases

Searching personal notes: Upload your markdown notes, journal entries, or meeting minutes. Ask "When did I last discuss the API redesign?" and get actual dates and context.

Technical documentation: Upload API docs, README files, or internal wikis. "How do I authenticate with the /users endpoint?" returns the actual documented process.

Research paper review: Upload 5-10 papers on a topic. "What methods did these studies use for data collection?" pulls relevant methodology sections from across your collection.

Contract review: Upload a contract and ask "What are the termination clauses?" or "What happens if delivery is late?" Note: for legal decisions, always verify against the source document — don't rely solely on AI interpretation.

Code documentation: Upload your codebase as text files. "Where is the database connection configured?" finds the relevant files and shows the configuration.

The Bottom Line

AnythingLLM is the path of least resistance for local document chat. No Python, no Docker (unless you want it), no configuration files. Install the desktop app, connect to Ollama, upload your files, start asking questions.

The RAG isn't perfect — table extraction is weak, multi-document reasoning is inconsistent, and the embedding model choice is global rather than per-workspace. But for the five minutes it takes to set up versus the hours you'd spend building the same thing with LangChain, it's hard to argue against starting here.

If you outgrow it, you'll know exactly what you need from a custom solution — because AnythingLLM will have taught you what RAG can and can't do with your specific documents.

📚 **Go deeper:** Local RAG Guide · Best LLMs for RAG · Ollama Setup Guide · Open WebUI Setup

Get notified when we publish new guides.

Subscribe — free, no spam

Source: https://insiderllm.com/guides/anythingllm-setup-guide/