SCORED.TOOLS · ROUNDUP

Best AI RAG Frameworks 2026: Honest Rankings

We tested the top RAG frameworks on retrieval quality, dev experience, and production readiness. Here's what actually works in 2026.

Retrieval-augmented generation has stopped being a research curiosity. It's how most teams ship LLM features that need to ground answers in private data. The framework you pick decides whether you spend your time on the actual product or fighting plumbing.

We benchmarked the leading RAG frameworks on three things that matter: retrieval quality on a mixed corpus (PDFs, markdown, code, tables), developer experience from first install to first answer, and how cleanly they scale past the demo. No vendor input. Here's the ranking.

1. LlamaIndex — 9.2/10

Llamaindex has quietly become the default for teams whose first problem is data, not chains. The ingestion layer handles messy real-world sources better than anything else we tested — PDFs with tables, Notion exports, codebases, structured DBs all work without bespoke parsers. Query engines compose cleanly, and the recent agent workflows API removed the worst of the abstraction tax. The router and sub-question engines are still the best way to handle queries that span multiple corpora.

Best for: Teams indexing heterogeneous data sources and serving complex queries.

Pricing: Open source. LlamaCloud managed parsing is usage-based, free tier covers most prototypes.

2. LangChain — 8.5/10

Langchain is still the broadest ecosystem and the easiest place to find an integration for an obscure vector store or embedding model. LangGraph fixed most of what was wrong with the original chain abstraction — state is explicit, control flow is debuggable. The trade-off is surface area: you will inherit a lot of code you didn't write, and breaking changes still happen. If your stack is already LangChain, stay. If you're starting clean and your problem is pure retrieval, LlamaIndex is leaner.

Best for: Multi-step agent workflows where retrieval is one node among many.

Pricing: Open source. LangSmith observability is paid past a small free tier.

3. Haystack — 8.4/10

Haystack from deepset is the most production-minded framework in this list. Pipelines are typed, components are composable, and the deployment story (REST API, Docker, Kubernetes) is built in rather than bolted on. Hybrid retrieval, query rewriting, and evaluation tooling are first-class. The dev experience is more verbose than LlamaIndex, but the verbosity pays off when you need to debug a regression six months in.

Best for: Production deployments where reliability beats prototyping speed.

Pricing: Open source. deepset Cloud is enterprise pricing.

4. RAGFlow — 8.1/10

Ragflow is the dark horse of 2026. It bundles ingestion, chunking, retrieval, and a usable web UI into a single Docker compose file. The deep document understanding — table extraction from scanned PDFs, layout-aware chunking — beats most competitors out of the box. The catch is that it's opinionated: you get its pipeline or you fight it. For teams that want a working RAG app this afternoon without writing Python, it's the fastest path.

Best for: Internal knowledge bases and document-heavy use cases that need a UI.

Pricing: Open source, self-hosted.

5. txtai — 7.8/10

Txtai is the framework you reach for when LlamaIndex feels heavy. A single Python package, embedded SQLite + FAISS by default, and you can build a working RAG pipeline in about thirty lines. It also handles graph retrieval and topic modeling natively, which is rare in this space. The ceiling is lower than the bigger frameworks — fewer integrations, less tooling around evaluation — but for embedded use cases and edge deployments it's hard to beat.

Best for: Lightweight embedded RAG, edge devices, single-file deployments.

Pricing: Open source.

6. Verba — 7.5/10

Verba is Weaviate's open-source RAG application — not really a framework in the LangChain sense, more a configurable reference implementation. If you're already running Weaviate, Verba gives you a polished chat UI, document ingestion, and hybrid retrieval with minimal setup. Outside the Weaviate ecosystem it's less compelling.

Best for: Teams committed to Weaviate as their vector store.

Pricing: Open source. Weaviate Cloud is usage-based.

7. Memo — 7.3/10

Memo (formerly EmbedChain) reframes RAG as memory for agents. The API is the simplest in this list — add data, query data, done. Where it shines is conversational memory: storing user preferences and prior context across sessions, with automatic relevance scoring. As a general-purpose RAG framework it's thinner than LlamaIndex, but for agent memory specifically it's the cleanest abstraction we've seen.

Best for: Agent memory and personalization layers.

Pricing: Open source. Managed platform is usage-based.

8. Cognita — 7.0/10

Cognita from TrueFoundry wraps LangChain and LlamaIndex in a modular, API-first structure aimed at moving prototypes to production without rewriting. The UI for managing collections, embedders, and retrievers is genuinely useful. It's a wrapper, so you inherit upstream bugs, but the operational surface it adds — versioned collections, query logging, model swaps without redeploys — is worth the trade-off for teams that ship to internal users.

Best for: Internal platforms standardizing RAG across multiple teams.

Pricing: Open source. TrueFoundry platform is enterprise.

Comparison

Framework	Score	Strength	Weakness	License
LlamaIndex	9.2	Data ingestion + query composition	Smaller agent ecosystem than LangChain	MIT
LangChain	8.5	Breadth, agent workflows	Abstraction sprawl	MIT
Haystack	8.4	Production deployment	Verbose for prototypes	Apache 2.0
RAGFlow	8.1	Document understanding + UI	Opinionated pipeline	Apache 2.0
txtai	7.8	Lightweight, embedded	Lower ceiling	Apache 2.0
Verba	7.5	Polished out of the box	Weaviate-coupled	BSD 3-Clause

Framework	Score	Strength	Weakness	License
Memo	7.3	Agent memory abstraction	Thin as general RAG	Apache 2.0
Cognita	7.0	Ops surface area	Wrapper, not standalone	Apache 2.0

Final picks

- **If you're building a real product:** Llamaindex. Best ingestion, best query composition, smallest abstraction tax.
- **If you're shipping to production with reliability requirements:** Haystack. The typed pipelines and deployment story will save you.
- **If you need a working RAG app today:** Ragflow. Docker compose, walk away, have a working tool in an hour.
- **If you already have a LangChain agent:** Langchain with LangGraph. Don't rewrite working code.
- **If you want to embed RAG into something small:** Txtai. One package, no infrastructure.

The honest summary: RAG framework choice matters less in 2026 than it did two years ago. The vector store, the embedding model, and the chunking strategy will move retrieval quality more than the framework wrapper around them. Pick the one that matches how your team thinks about the problem and ship.

Generated 2026-06-04 · Read the live article at scored-tools.netlify.app/blog/best-ai-rag-frameworks-2026