RAG'n'Roll: A Local RAG System for Fresh Technology News (FAISS + Sentence-Transformers + Ollama)

Krishna Kirit Maniyar

Practical Data Science CS667 / Data Science
Seidenberg School Of Computer Science and Information Systems – Pace University
Instructor: Prof. Yin Yiqiao
km14292n@pace.edu

Project Summary

RAG'n'Roll is a fully local Retrieval-Augmented Generation (RAG) pipeline that:

- Ingests **fresh technology news** via RSS (Google News, NYTimes Tech, The Verge)
- Extracts + cleans article text (Trafilatura + custom cleanup)
- Stores articles locally in **SQLite**
- Builds embeddings using **Sentence-Transformers (all-MiniLM-L6-v2)**
- Indexes embeddings with **FAISS** for fast similarity search
- Generates grounded answers with **Ollama (llama3.2)** and **citations**

Objectives

- Build an end-to-end **local** RAG system (no paid LLM APIs).
- Make "**today**" verifiable by showing **published timestamps**.
- Add one-click **Refresh News** (ingest → embed → rebuild index).
- Improve answer quality by removing website boilerplate/junk.

Motivation

Technology news changes quickly. If the dataset is static, answers feel outdated. RAG improves trust by grounding responses in retrieved sources, but only if ingestion and indexing refresh regularly. This project focuses on a lightweight, practical pipeline runnable on a laptop.

System Architecture (Required Screenshot)

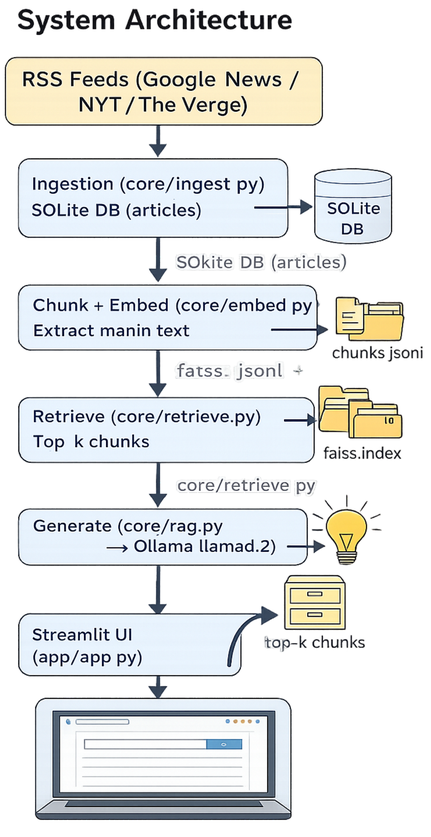

Figure 1:RSS → SQLite → chunk+embed → FAISS retrieve → Ollama generate → Streamlit UI.

Data Ingestion (RSS → Articles)

Feeds used:

- Google News RSS (Technology)
- NYTimes Technology RSS
- The Verge RSS

Ingestion process:

1. Parse feed entries (feedparser)
2. Fetch article HTML (requests)
3. Extract main text (trafilatura)
4. Clean boilerplate/junk lines (custom `clean_text`)
5. Store in SQLite: `articles(url,title,source,published,text`

Key improvement: Higher daily coverage (e.g., `limit_per_feed=20`) so the index has enough "today" signal.

Embedding + Indexing (SQLite → FAISS)

Chunking: Overlapping chunks (e.g., 1200 chars, 200 overlap) preserve context.
Embeddings: `all-MiniLM-L6-v2` produces dense vectors (normalized).
Index: FAISS `IndexFlatIP` enables fast Top-k similarity search.
Metadata: Each chunk stores `title, source, url, published` for citations + recency display.

Text Cleaning (Polish Upgrade)

Cleaning removes navigation menus and boilerplate to improve:

- Embedding quality
- Retrieval relevance
- Snippet readability in the UI

Answer Generation (RAG Prompting)

Prompt rules:

- Answer **only** from retrieved sources
- Cite sources as `[1],[2],...`
- If sources are insufficient, say so

Local model: Ollama runs `llama3.2` locally to generate grounded answers.

Streamlit App (User Workflow)

User flow:

1. Ask a question
2. Retrieve Top-k evidence chunks (FAISS)
3. Generate answer with citations (Ollama)
4. Show sources + **Published timestamps**
5. Click **Refresh News** to rebuild the index

UI Snapshot (Required)

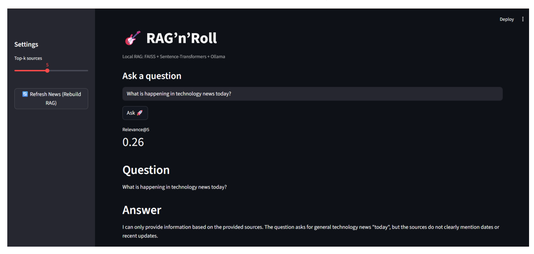

Figure 2:Streamlit UI: question input + Top-k control + Refresh button.

Answer + Sources Snapshot (Required)

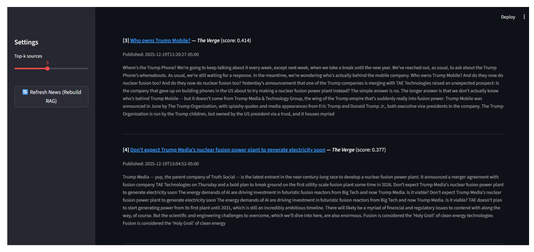

Figure 3:Grounded answer with citations and verifiable recency via **Published** dates.

Key Result / Demo Outcome

The system produces **citation-grounded** answers from freshly ingested technology news. Displaying **Published timestamps** makes "today" claims verifiable and improves trust.

Lightweight Evaluation (Relevance@k)

A simple **Relevance@k** score is computed using keyword overlap between the user query and retrieved snippets. This is not a full IR benchmark, but it provides fast feedback during interactive testing.

Refresh Logs Snapshot (Required)

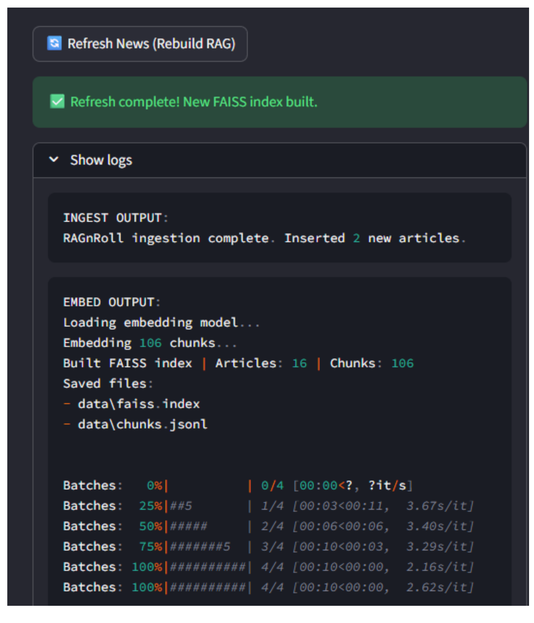

Figure 4:Refresh workflow logs: ingestion + embedding + index rebuild.

Conclusion

RAG'n'Roll demonstrates a practical local approach to grounded question-answering over fresh technology news:

- Automated ingestion keeps the knowledge base up-to-date
- Retrieval + citations reduces hallucination risk
- Published timestamps improve transparency for "today" questions

Limitations

- RSS coverage depends on feed availability and extraction quality
- Evaluation metric is heuristic (not human-labeled relevance)
- Some sites block scraping or rely on dynamic rendering

Future Work

- Time-based filtering (last 24h / last 7 days)
- Reranking (cross-encoder) for higher top-k precision
- Topic clustering + daily trend summaries

How to Run (GitHub Readers)

Commands:

- `python -m core.ingest`
- `python -m core.embed`
- `streamlit run app/app.py`

Note: Requires Ollama installed and the model pulled (e.g., `ollama pull llama3.2`).

Tools & Technologies

Python FAISS Sentence Transformers Ollama Streamlit SQLite RSS Feeds

Contact / Repo

- GitHub: github.com/krishnam229/RAG-n-ROLL
- Email: km14292n@pace.edu