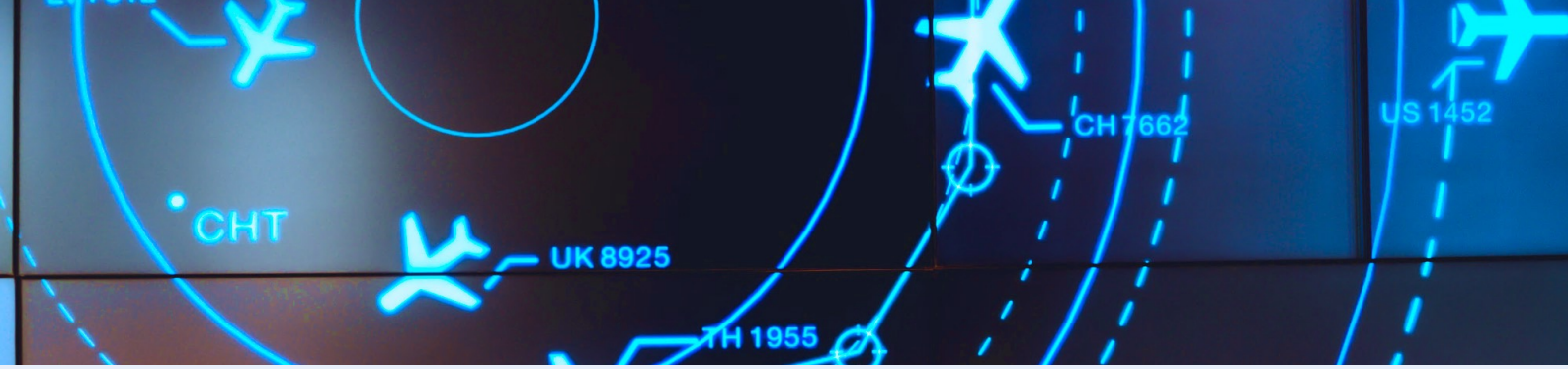
Resolute

case study –

Equipping a tracking platform for AI:

Secure, local RAG prototype with .NET and Semantic Kernel



Services

- **Technical research & feasibility**
- **Architecture consulting**
- **Custom AI solution**
- **Full-stack development (.NET)**
- **Custom Retrieval-Augmented Generation (RAG) pipeline**
- **Local AI integration**

A developer of airborne object tracking platforms turned to Resolute for help building a modern, secure AI prototype.

After a successful collaboration on both web and desktop real-time tracking systems, the client wanted to explore how artificial intelligence could **support users' queries and reference complex operational regulations**.

Resolute delivered a fully localized **Retrieval-Augmented Generation (RAG) proof of concept**, capable of answering questions from thousands of pages of regulations with traceable accuracy. Entirely built in **.NET using Microsoft's Semantic Kernel**, the solution strikes a balance between performance, control, and extensibility, with **no reliance on third-party models or external cloud APIs**.

The challenge

The client's operational domain involved **large volumes of legally binding documentation** - aviation and defense compliance, safety regulations, and other institutional standards.

The request: build an AI assistant that could parse these documents and return specific, verifiable answers.

The constraints were significant:

Source material

Vast volumes of source material - around 100 documents, each exceeding 1,000 pages, in PDF and XML formats

AI expertise

Limited internal understanding of AI capabilities

Infrastructure

Modest infrastructure, no dedicated GPU cluster

Demand

Need for full traceability and citation of results

The solution

Resolute proposed and built a complete RAG (Retrieval-Augmented Generation) solution, hosted and run locally using lightweight open-source components. It was tailored for both the client's infrastructure and the specific nature of their documents. The AI assistant follows a retrieval and response workflow designed to handle large, unstructured documents and generate accurate, reference-based answers as follows:

1

Document ingestion & chunking

To process regulatory documents efficiently, Resolute first 'tokenized' the documents by splitting thousands of pages into smaller, manageable chunks—an essential step before vectorizing and storing content in the retrieval system. The approach used was tailored to the client's constraints, but informed by several common chunking strategies, including:

- **Fixed length** (e.g., 500 tokens): Splits content at consistent intervals, useful for predictable memory handling.
- **Semantic** (paragraph-aware): Breaks text at logical boundaries like sentences or sections, preserving context.
- **Hybrid** (semantic + fixed): Combines both approaches to balance performance and meaning.
- **LLM-assisted semantic parsing**: In more advanced (but resource-intensive) cases, documents can be passed page-by-page to an LLM to extract meaning-rich segments. These can be stored separately or in parallel with manually chunked content to support multiple chunking strategies within the RAG pipeline.

To maintain context between segments, each chunk included **a 200-token overlap** (for chunk sizes of about 400 tokens), helping the retrieval model capture cross-boundary relationships and **minimize information loss**.

The solution

2

Vectorization & storage

Embeddings were created using local, open-source models and stored in a **vector database**. These high-dimensional vectors allow semantic comparison of text chunks; the system doesn't just search for keywords, but for meaning. For example, "night flight regulations" and "FAA guidelines for nighttime aviation operations" may be different words, but point to similar concepts, or else said vectors, and thus can be retrieved together. The entire vector store was kept local to maintain full control over data.

3

Prompt refinement (prompt engineering) & response generation

User prompts, sometimes phrased informally or inconsistently, were cleaned using a local LLM, then matched against the vector database to find relevant chunks. These chunks were combined with the refined prompt to generate a natural language response, including references to the source material.

The AI system handled multi-pass iterations where needed, such as resolving slang or spelling issues before querying, and refining the final output after retrieval.

All of this was designed to operate on **modest infrastructure**, utilizing lightweight models within the 3–4 billion parameter range. Despite the local constraints, the solution produced high-quality, accurate answers.

Technologies and architecture

Area	Technology
AI utilization	Microsoft Semantic Kernel (C#)
Language model	Mistral 7B, self-hosted
Embedding model	Nomic Embed
Vector storage	Qdrant
Backend platform	.NET 9, all AI logic written in C#

To align with the client's existing infrastructure and team capabilities, the AI assistant was built entirely in C# using Microsoft's Semantic Kernel, without Python. This simplified integration, reduced maintenance overhead, and kept all logic within the familiar .NET ecosystem.

All models, embeddings, and vector storage were hosted locally to minimize cost and maintain full control over sensitive data. The assistant was designed to work efficiently even on modest hardware, using lightweight models in the 3–4 billion parameter range.

The results

- **Fully functional, locally hosted AI assistant** built with Retrieval-Augmented Generation
- **Able to answer complex compliance questions** with cited references from regulatory documentation
- **Runs on local hardware** with no external model or cloud API usage
- **Extensible architecture,** ready for integration into web or desktop systems
- **Stakeholders impressed** by relevance, clarity and accuracy of results

Local-first RAG

Chunked, embedded, and queried entirely in-house

Accurate by design

Built-in overlap and citation ensure reliable answers

AI built for .NET

Powered by
Semantic Kernel and C#

Future-flexible

API-ready, LLM-pluggable, integration-ready for Azure, OpenAI, and more

Resolute

sales@resolutesoftware.com

Let's talk about your technology requirements.

Get in touch

USA

MA 01701, Framingham,
945 Concord St,

+1-617 386-9697

Deloitte. Company to watch | Tech Fast 50

Business Professional Services

Clutch ★★★★★

People First Company Award 2020 | 2021 | 2022