



Team 10: Smart Document Retrieval & The AI-Assisted Build

Pacera Hack Days 2026

Aleksejs, Nikolajs, Olegs, Yurii

```
d = document_retriever.query(prompt)
}

# Document query AI other data
ai build_pipeline.run_ai_assisted()

= 1("Text")
return "com test")

build_pipeline.run_ai_assisted()
=> Processing 1.2GB data...
=> Optimizing vectors...
=> Optimizing system...
=> Build successful. 98% accuracy.
```

We used AI to build a better way to use AI.

The Vision: Velocity Above All

“Use anything and everything that speeds us up.”

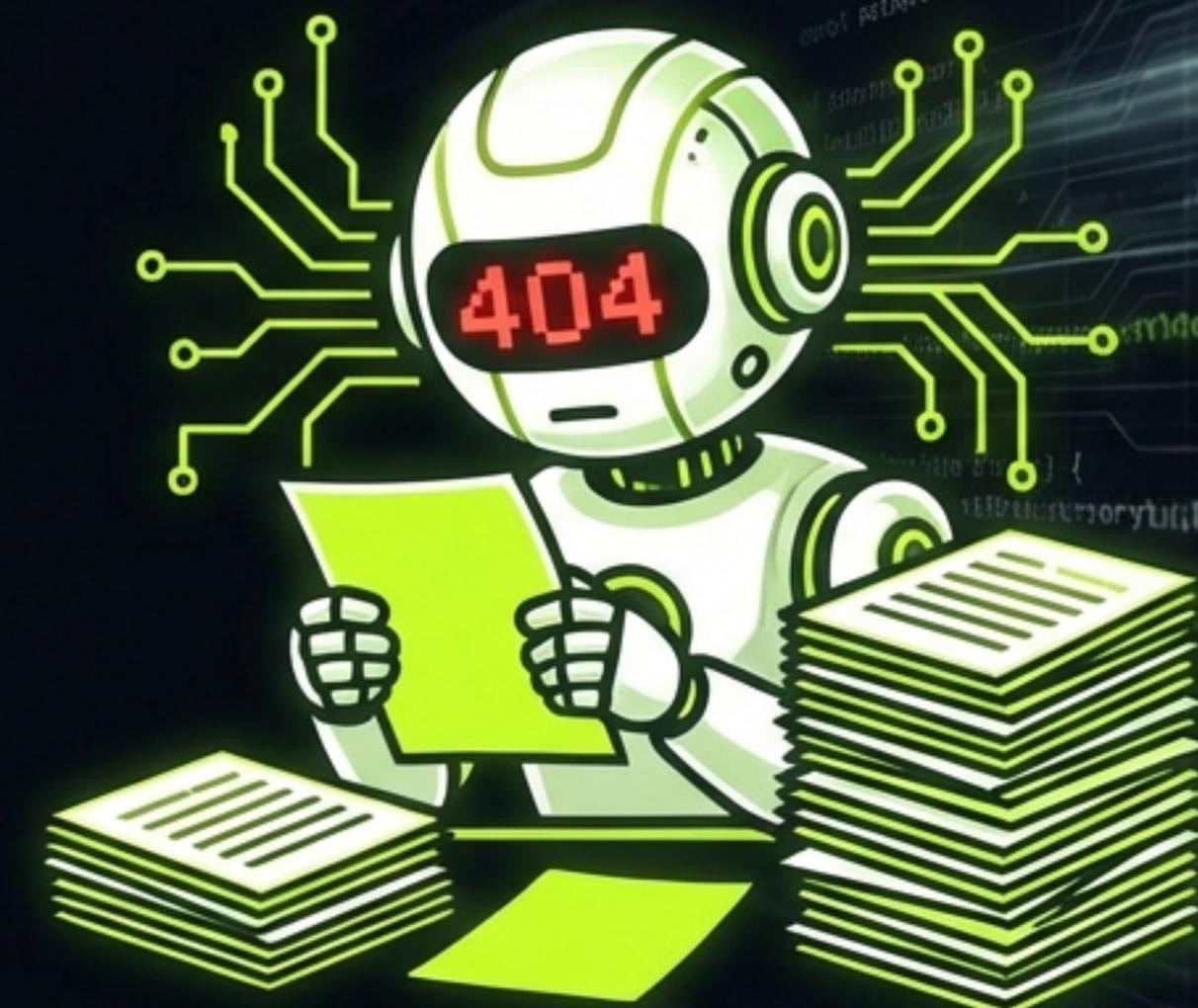
Strategy: Orchestrate AI agents to bypass traditional coding bottlenecks.



The Problem: Data Overload & LLM Amnesia

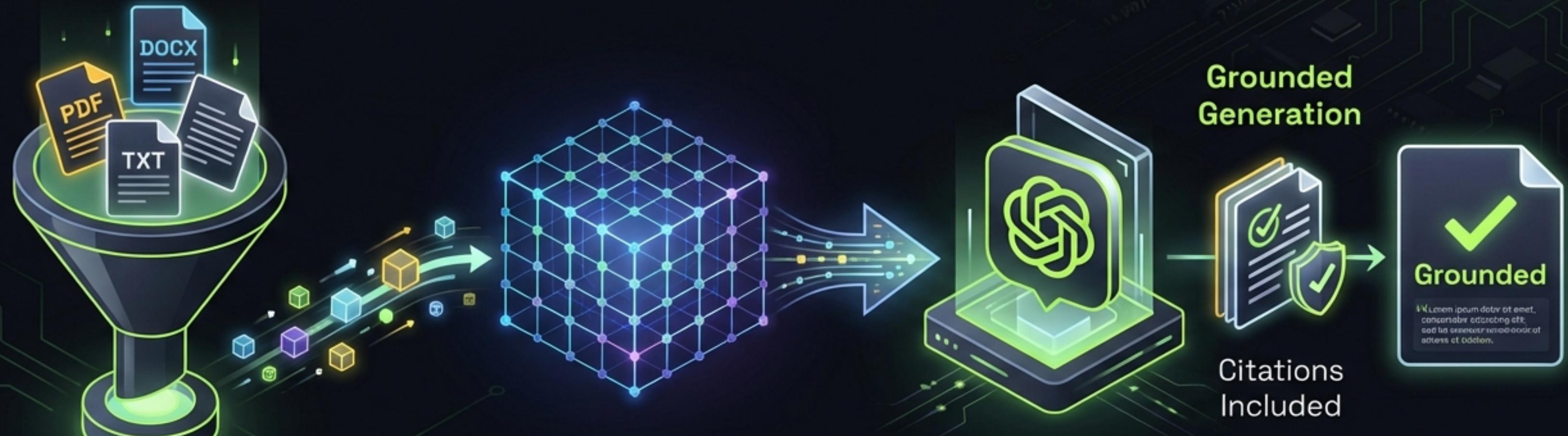


The Black Box: Opaque Collections.



Context Limits & Hallucinations.

The Solution: Retrieval-Augmented Generation (RAG)



1. Index:
Chunk & Embed

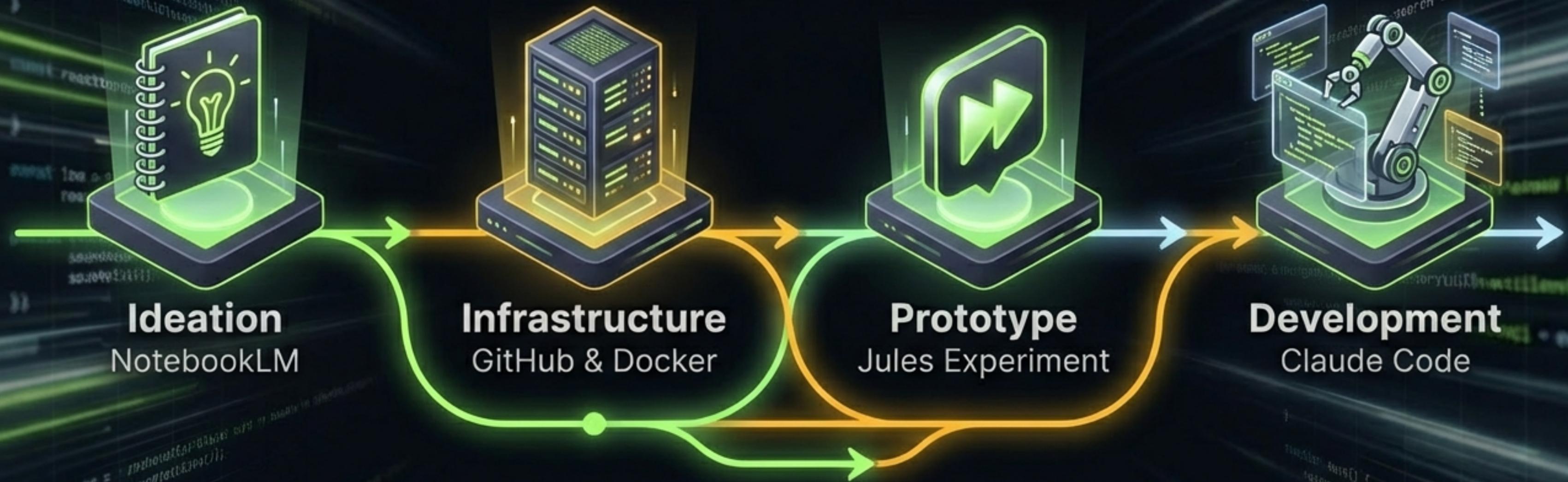


2. Retrieve:
Semantic Search



3. Generate:
Grounded Answers

The Build Strategy: AI as a Team Member

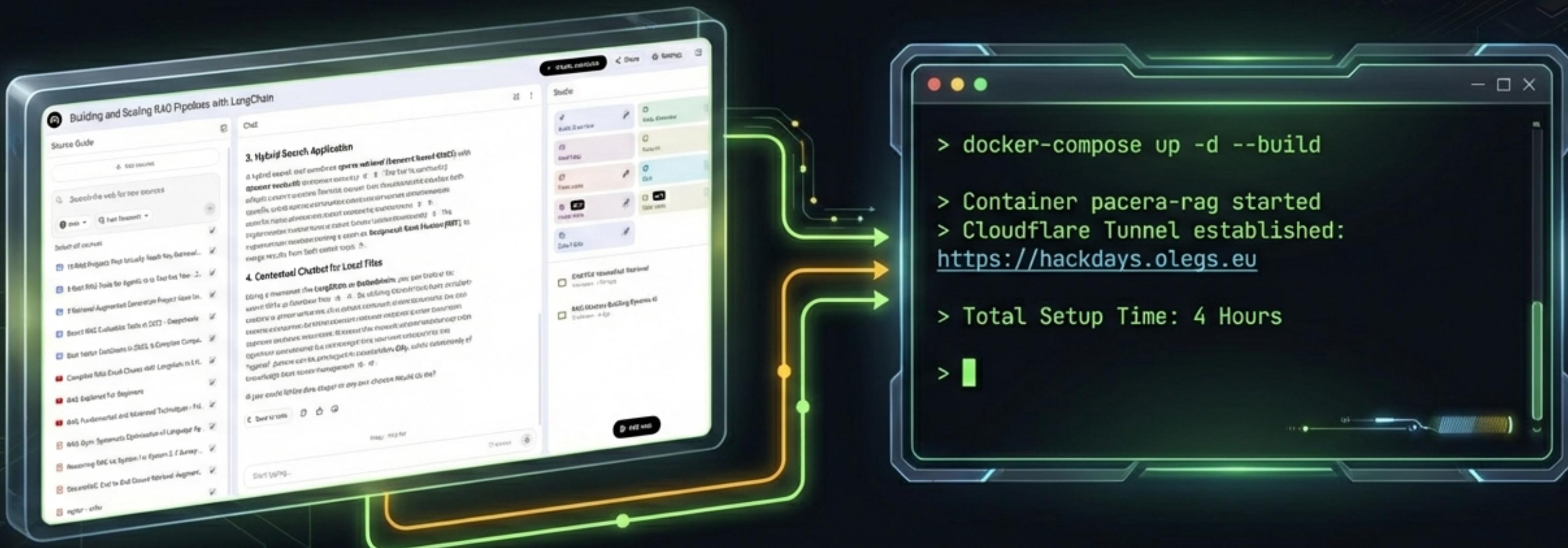


AI was used for everything—from **ideation** to deployment.

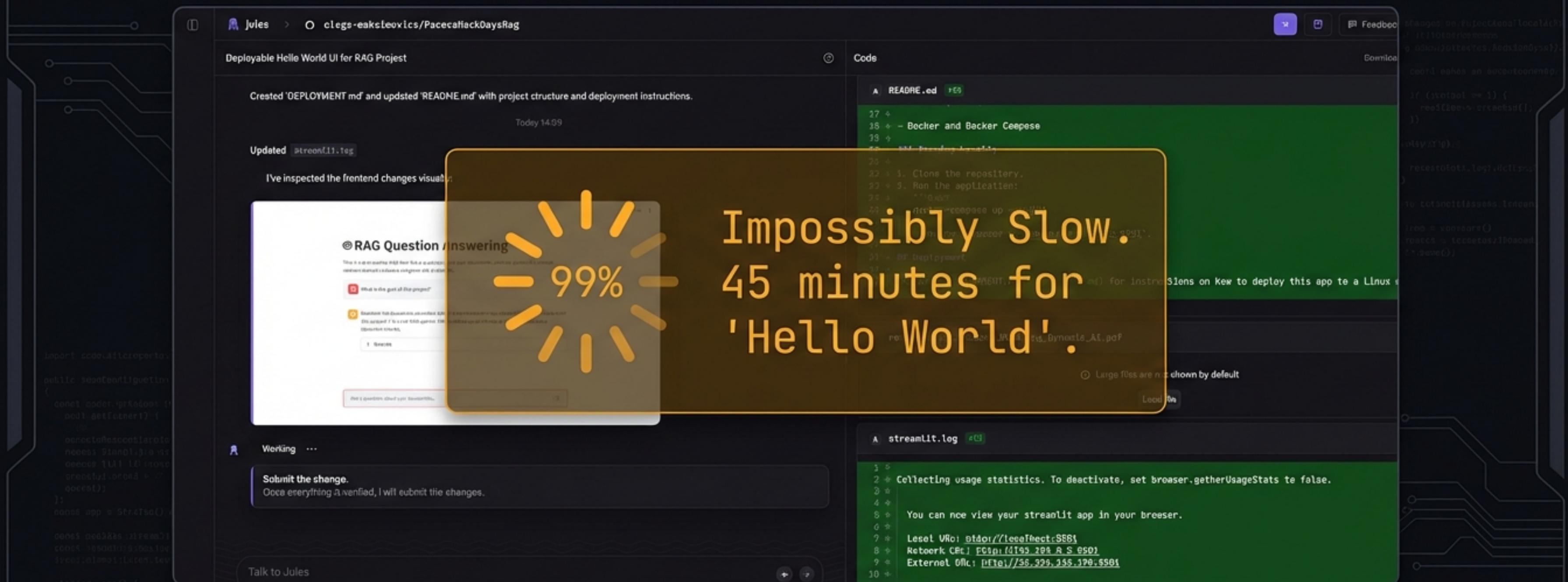
Step 1 & 2: Instant Ideation to Deployed Infra

NotebookLM:
Ideation & RAG Concepts

Deployment:
Infrastructure as Code



The Failure: The ‘Jules’ Experiment



Insight: Not all AI tools are ready for rapid prototyping.

The AI Developer Tool Showdown

Claude Code

9/10

JetBrains Mono

The MVP. Excellent
planner & implementer.

Google
Antigravity

8/10

JetBrains Mono

Copilot on steroids.
Generous limits.

GitHub
Copilot

7/10

JetBrains Mono

Standard, capable.

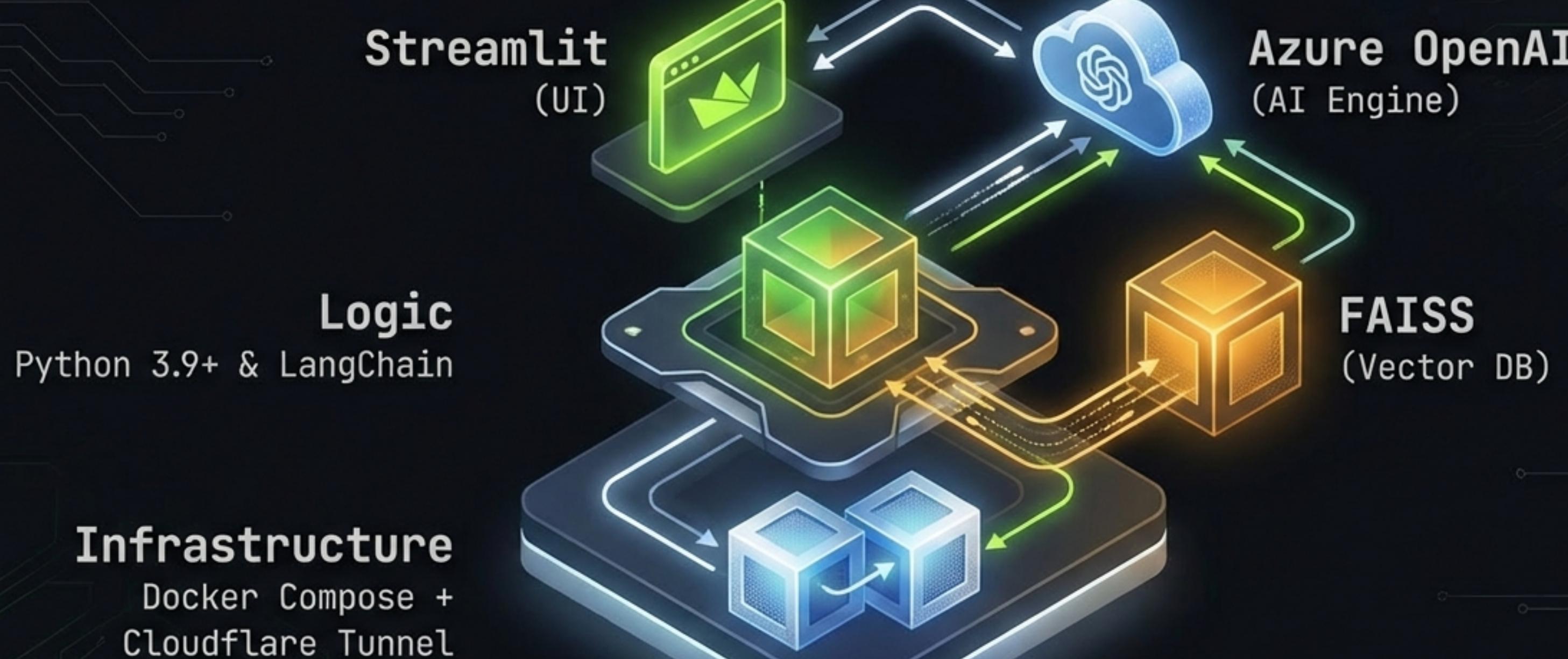
Google Gemini
Assistant

0/10

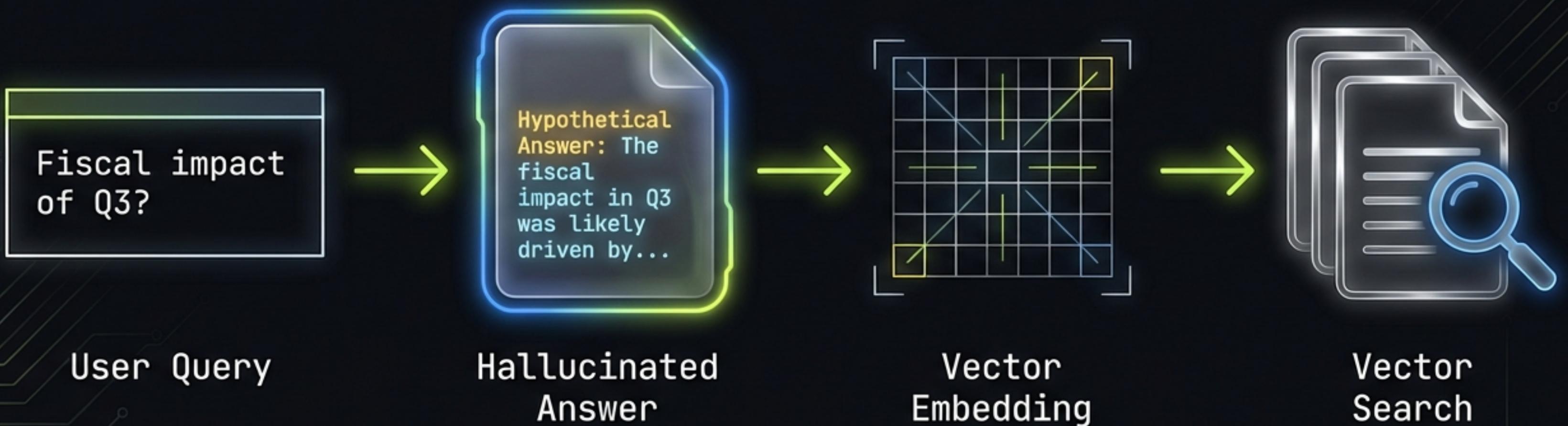
JetBrains Mono

More often doesn't
work.

Under the Hood: The Tech Stack

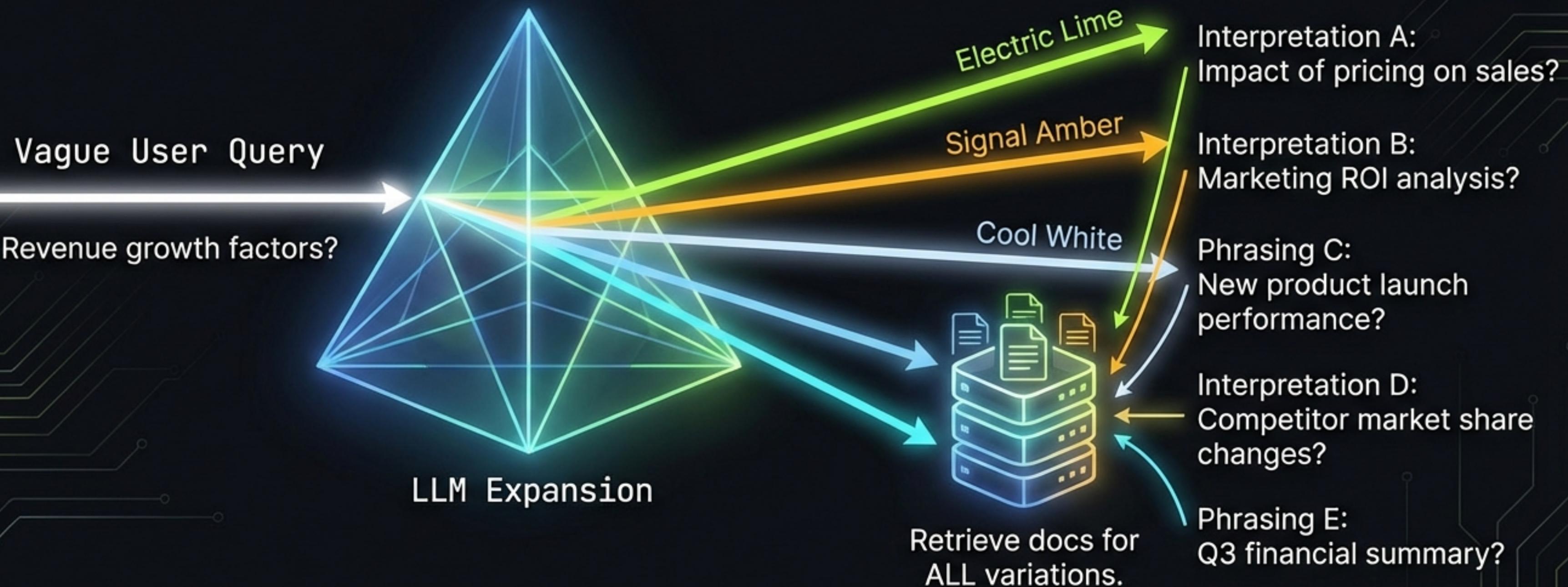


Solving Vocabulary Mismatch: HyDE



Hypothetical Document Embeddings: Search for what the answer **looks** like.

Solving Ambiguity: Multi-Query Expansion



Significantly increases recall by addressing multiple potential interpretations of the user's intent.

The Result: Smart, Grounded Retrieval

Knowledge Base

Search Mode: Hybrid (Vector + Keyword)

Upload files to process into chunks

Ask a question about your documents...

Process & index Documents

Version: 1.0.35

© RAG Question Answering

This is an interactive RAG interface. Upload documents in the sidebar to process them into chunks, then ask questions about their content.

how many people are in the team and what tools are they planning to use ?

There are 4 people: Nikolajs, Olegs, Yurii, and Aleksejs. Planned tools: - Nikolajs: Claude Code - Olegs: <https://jules.google/kplans> - Yurii: Gemini Code Assist - Aleksejs: Claude Code ?, continue.dev

Sources

- Source: rag-project-overview.md (Section: Success Criteria (what does good look like))
- Source: team-ai-usage.md (Section: New tools we want to try)

Grounded Answer

Source Citations
(No Hallucinations)

Interactive Chat

NotebookLM

Performance Metrics



Retrieval
Relevance
(>90%)



Latency
(<5s)



Accuracy
(>85%)



Scalability
(1,000+ Docs)

Key Learnings & Surprises



What Worked

RAG is efficient.
Claude Code is a productivity beast.



The Surprise

Autonomous planning capabilities of agents are higher than expected.



The Challenge

Vocabulary mismatch (User vs. Docs) is the hardest problem.
Solved by HyDE.

Ready to Deploy



github.com/AleksejsKudrins/PaceraHackDaysRag

TEAM 10: ALEKSEJS • NIKOLAJS • OLEGS • YURII