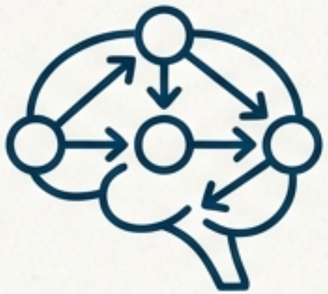
An Architectural Overview of the ‘Lord of Wisdom’ Cognitive Agent

A technical breakdown of a serverless agent designed for ingesting and teaching complex documents via Telegram.

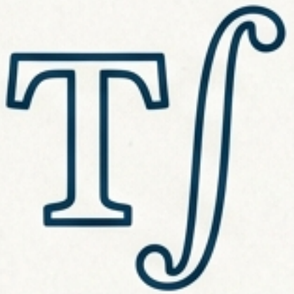
Resilient RAG

An ingestion pipeline built to prevent silent failures in a serverless context.



Agentic Logic

A reasoning engine using LangGraph for contextual conversation and state management.

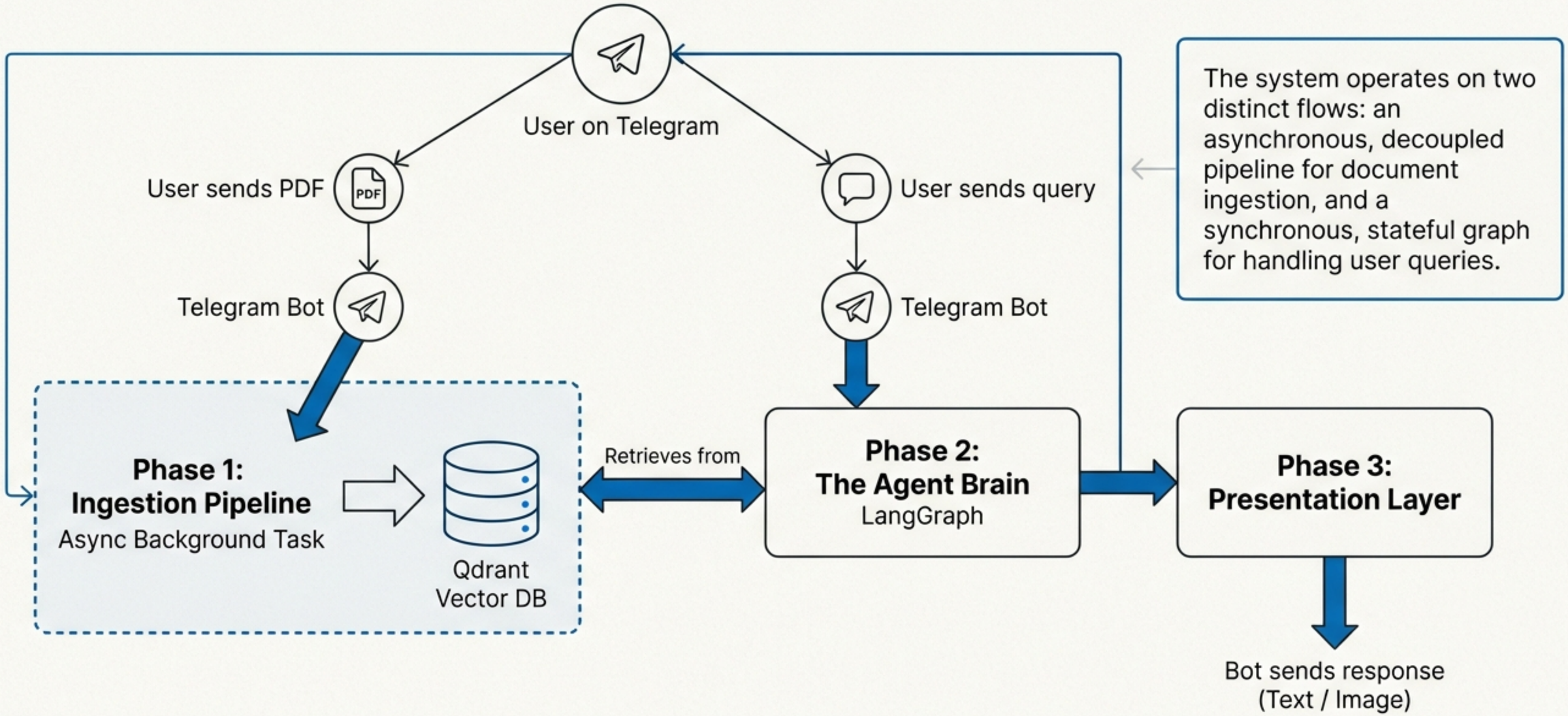


Multi-Modal Output

Intelligent rendering of text and mathematical notation.

Version: 1.0.2 | Deployment: Google Cloud Run (Serverless)

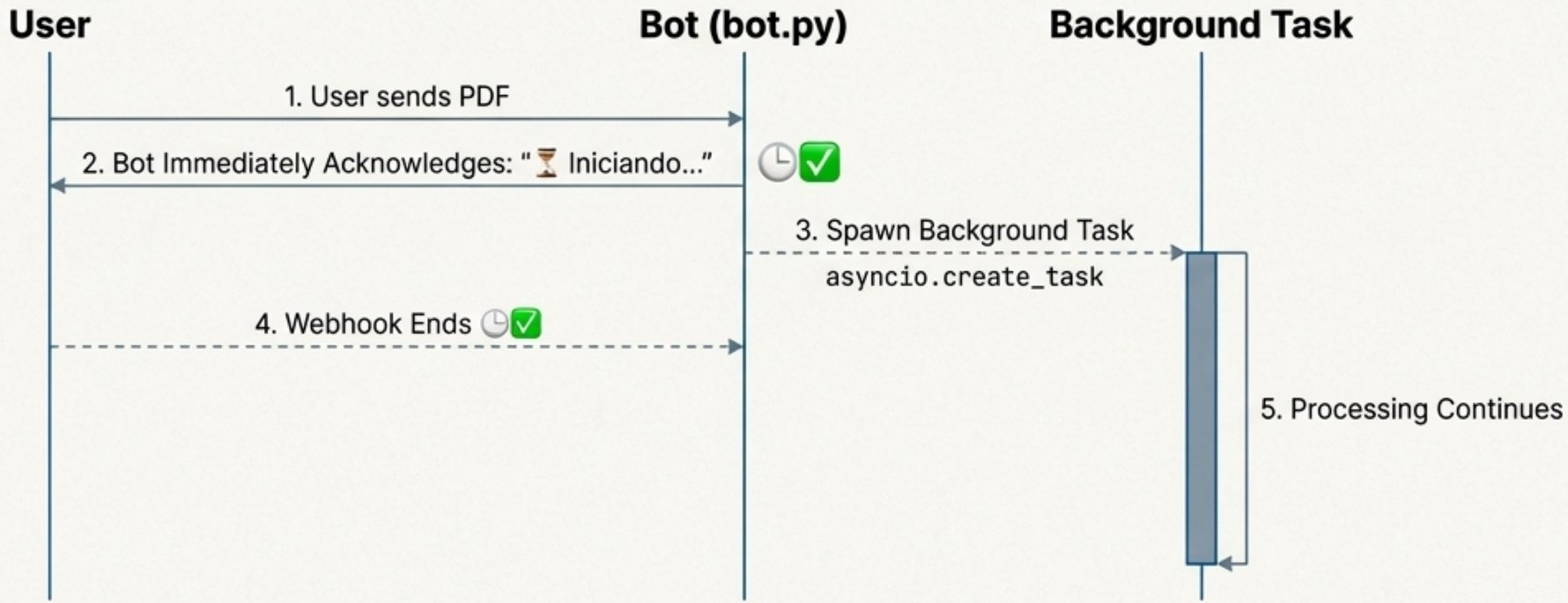
High-Level System Flow



Phase 1: The Senses - Engineering for Reliable Ingestion

Core Problem Addressed: The “Silent Failure” of Serverless Webhooks. Standard webhook processing can time out during heavy tasks like document ingestion, leaving the user with no feedback.

The Solution: Decoupled Background Processing



Key Takeaway: This design decouples the heavy processing from the webhook response, **ensuring both immediate user feedback and successful completion of long-running tasks.**

Phase 1: Ingestion State Tracking & Optimization

The ‘X-Ray’ Registry

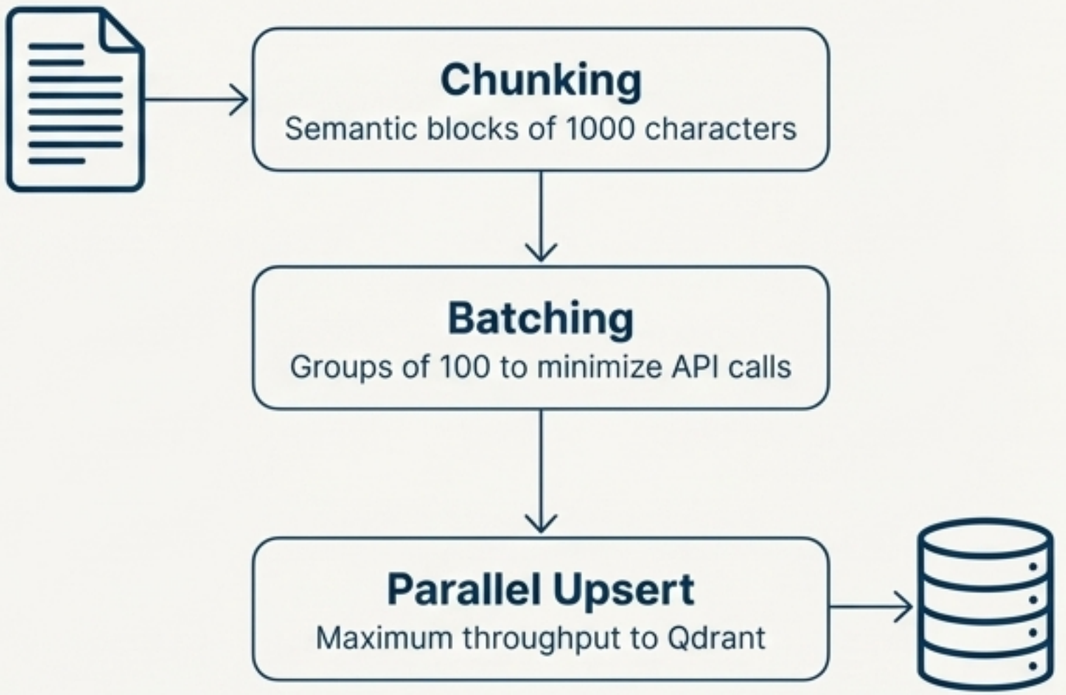
A thread-safe global dictionary (`task_registry`) maps a user's `chat_id` to their current ingestion status.

task_registry

```
'12345': "Downloading..."
'67890': "Extracting..."
'54321': "Embedding Batch 5/50"
```

Provides real-time status updates and enables the 'Short-Circuit Firewall' (covered in Phase 2).

Optimized Data Processing



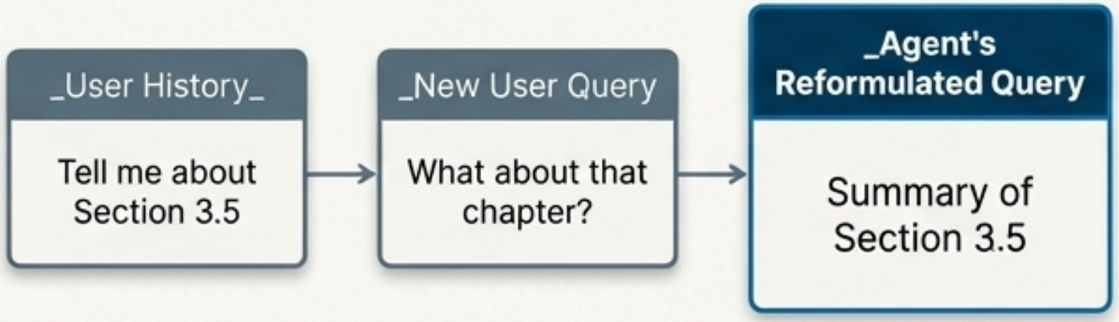
Supporting Infrastructure: The Cloud Run container is configured with **2GiB RAM** and **`--no-cpu-throttling`** to prevent Out-of-Memory (OOM) errors during this intensive process.

Phase 2: The Brain - An Agent Built on LangGraph

The agent's reasoning is built on LangGraph, a framework for creating stateful, multi-step agentic applications using a directed cyclic graph.

Contextual Memory & Query Reformulation

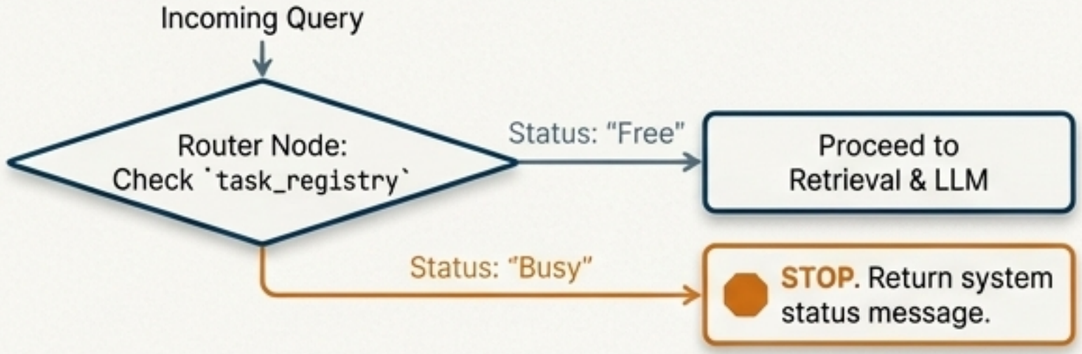
The bot maintains a rolling window of the **last 5 messages** for each user.



Seamless context maintenance without the user having to to repeat themselves.

The "Short-Circuit" Firewall

RAG bots often hallucinate when asked about status (e.g., "Are you done?") because a search for "done" yields no relevant documents.



Before retrieval, a Router Node in the graph checks the `task_registry`. If the status for that user is "Busy," the graph execution **STOPS** and returns a system status message without ever querying the LLM.

Phase 2: The ‘Charismatic Tutor’ Persona

Goal: To create a seamless, human-like teaching experience, moving beyond the typical 'According to the document...' responses of standard RAG bots.

Method: Strict Prompt Engineering: The Generator LLM node is given a precise set of instructions to shape its personality and output style.

Internalize Knowledge: "Treat the `context retrieved` from the `database` as **YOUR OWN MEMORY**."

Ban Robotic Language: A strict negative constraint: "Never say "According to the context" or similar phrases."

Speak Directly: 'The `definition of curl` is ..." instead of "The document defines `curl` as..."

Standard RAG Response

User Query: What is the definition of curl?

Bot Response: According to the provided context, the document defines the curl of a vector field **F** as a vector operator that describes the infinitesimal rotation of **F**.

Agent's Persona Response

User Query: What is the definition of curl?

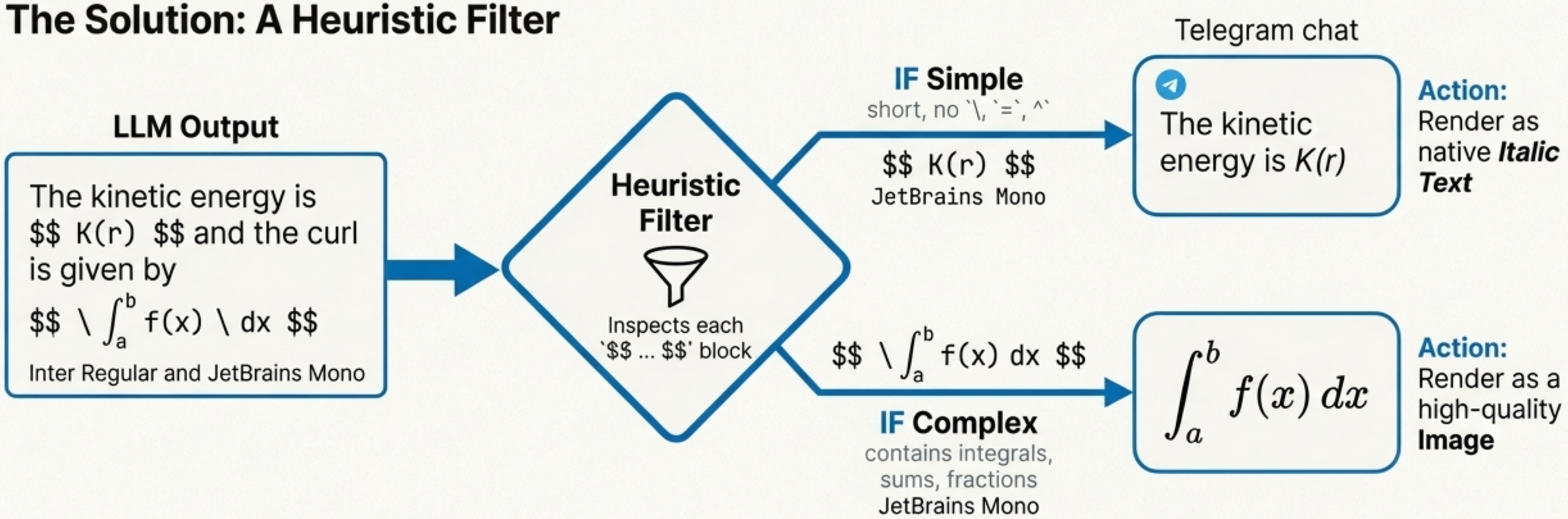
Bot Response: The curl of a vector field is a vector operator that describes its infinitesimal rotation. Imagine placing a small paddle wheel in the field; the curl is the vector that points along the axis of rotation and whose magnitude is the speed of that rotation.

Phase 3: Smart Rendering for Mathematical Notation

The Challenge

Complex mathematical equations sent as plain text in Telegram are unreadable. Sending every formula as an image clutters the chat.

The Solution: A Heuristic Filter



Deployment & Operations on Google Cloud Run

A custom PowerShell script (`scripts/deploy.ps1`) automates deployment to Google's serverless platform.

```
gcloud run deploy lord-of-wisdom \
    --source . \
    --region us-central1 \
    --allow-unauthenticated \
    --memory 2Gi \
    --cpu 2 \
    --no-cpu-throttling
```

Essential. Required to handle memory-intensive PDF parsing without causing Out-of-Memory crashes.

Performance. Provides two virtual CPUs to accelerate parallel processes like vector generation.

MANDATORY. This is the most critical flag. By default, Cloud Run freezes CPU after an HTTP response is sent. This flag allows the CPU to remain active, ensuring the background ingestion tasks can complete successfully.

Technology & Environment Stack

Core Technologies

- **Language & Frameworks**: Python, LangGraph
- **Data Storage**: Qdrant (Vector Database)
- **AI Models**:
  - OpenAI API (Embeddings & Vision)
  - DeepSeek API (Core 'Brain' LLM)
- **Rendering**: Matplotlib (Agg Backend)
- **Deployment**: Google Cloud Run, Docker
- **Automation**: PowerShell

Environment Variables

- `TELEGRAM_BOT_TOKEN`: The Telegram bot interface.
- `OPENAI_API_KEY`: For text embeddings.
- `DEEPSEEK_API_KEY`: For the primary reasoning model.
- `QDRANT_URL`: The long-term memory store.