# Human-in-the-Loop Math Routing Agent — Final Proposal

## Overview We propose an Agentic■RAG system that behaves like a mathematics professor: it first consults an in■house knowledge base (KB), and only if needed, routes to web search through MCP servers. A human■in■the■loop feedback layer continuously improves solution quality (bonus: DSPy). The stack ships with a FastAPI backend and a lightweight React UI.

## Architecture (Agentic Workflow) 1. **AI Gateway Guardrails (Input/Output):** The API enforces topic scoping (Math■only) and basic PII filters pre■inference; and validates outputs post■inference (topic■safe, length caps). We align this with modern AI gateways that provide rule■based guardrails at the edge (e.g., Cloudflare AI Gateway's Guardrails). Citations: Cloudflare AI Gateway Guardrails (Feb 2025).

2. **Routing:** A LangGraph■like controller (implemented here as a lightweight router) decides:
a) **KB path:** retrieve with TF■IDF/Qdrant and solve using a math solver.
b) **Web path:** if KB confidence is low, call an **MCP** server (Tavily/Exa) to search + extract, then synthesize a step■by■step solution.
c) **Refuse:** if nothing reliable is found.

3. **Knowledge Base & VectorDB:** We include a minimal KB (algebra/calculus) and TF■IDF store for the demo; production swaps to **Qdrant** with embeddings.
Citations: Qdrant docs.

4. **Web Search via MCP (Required):** Use **Tavily MCP** or **Exa MCP** as Model Context Protocol servers to perform real■time search and structured extraction. This keeps the agent's external tools sandboxed and observable.
Citations: Tavily MCP, Exa MCP, Awesome MCP Servers (directory).

5. **Solver:** A symbolic core (SymPy) covers common tasks (solve, differentiate, integrate) and formats steps like a teacher. When external docs are available, they inform the explanation (not the final numeric algebra which is verified by SymPy).

6. **Human■in■the■Loop (HITL):** The UI collects 1■to■5 ratings and free■text critiques. Feedback is stored (SQLite) and periodically compiled with **DSPy** to refine the "MathExplainer" module (self■improving prompts/programs).
Citations: DSPy.

## Guardrails (Input & Output) — Approach & Rationale - **Input Guardrails:** topic■scope to Mathematics; lightweight PII regex; out■of■scope refusal. This reduces prompt■abuse and keeps the system educational. - **Output Guardrails:** reject non■math or

**unsafe content, cap verbosity, and surface citations for web■sourced answers. - **Why this approach?** Guardrails at the “gateway” are easy to audit and can run before model calls—reducing cost and risk, per AI gateway best practices.**

**Knowledge Base - **Dataset:** Demo KB contains core tasks (linear equations, quadratics, derivatives, basic integrals). For production, seed with JEE/AMC/CEM math archives and lecture notes, embedded into Qdrant. - **Example in■KB questions to try:** 1) “Solve for x: 2x + 5 = 17” 2) “Differentiate f(x) = sin(x)” 3) “Integrate x^2 dx”**

**Web Search / MCP Setup - **Servers:** - Tavily MCP (search/extract/crawl) - Exa MCP (web/academic/code search) - **Strategy:** Use MCP tools to: (i) search, (ii) extract focused snippets, (iii) attribute sources. We only use web to support explanations; algebra is solved/verified with SymPy to avoid copying errors. - **Example out■of■KB questions:** 1) “State and explain the Binomial Theorem (short).” 2) “What is the definition of a convergent sequence?” 3) “Give the formula for the sum of the first n natural numbers and prove it.” (These trigger MCP search, then a SymPy■verified reasoning where applicable.)**

**Human■in■the■Loop Routing - **Flow:** User → Guarded Input → Router → (KB or MCP) → SymPy solution → Guarded Output → Feedback capture (rating+comment). - **Learning:** Periodically export (question, solution, rating≥4) and use **DSPy** to compile an improved “MathExplainer”. Redeploy the compiled module (or prompt) to the solver agent. This is objective■aligned and simple to operate.**

**[Bonus] Benchmark: JEEBench - **Dataset:** JEEBench (EMNLP 2023) — 515 challenging IIT■JEE problems across Math, Physics, Chemistry. - **Script:** Included `backend/scripts/run_jeebench.py` expects a JSONL subset; evaluates symbolic equivalence via SymPy. - **Notes:** Because many JEE problems are long■form and multi■step, we recommend scoring both final correctness and step quality (via rubric / human sampling).**

**Deliverables - **Source code:** FastAPI backend (+ stubs for MCP), React UI, KB builder, DSPy training sketch, JEEBench runner. - **Demo:** Run FastAPI (`uvicorn app.main:app`), then `npm start` in frontend. Ask KB questions first, then toggle to out■of■KB concepts to observe MCP (stub in this archive). - **PDF (this file)** and **README** included.**

**References - Cloudflare AI Gateway Guardrails (2025-02): https://blog.cloudflare.com/guardrails-in-ai-gateway/ - Qdrant Documentation: https://qdrant.tech/documentation/ - Tavily MCP Server: https://github.com/tavily-ai/tavily-mcp / https://docs.tavily.com/documentation/mcp - Exa MCP Server: https://github.com/exa-labs/exa-mcp-server / https://exa.ai - Awesome MCP Servers directory: https://mcpservers.org/ - DSPy (Stanford): https://dspy.ai/ ; https://github.com/stanfordnlp/dspy - JEEBench (EMNLP 2023): https://github.com/dair-iitd/jeebench ; https://arxiv.org/abs/2305.15074**