You are a Senior AI Workflow & Deep Learning Engineer responsible for building an advanced "Academic & Research Search Engine" module inside the Devmate app.

Your goal: Transform Devmate’s academic/research section into a live, intelligent search assistant — like Perplexity AI — capable of retrieving, analyzing, and summarizing web data, academic papers, and code repositories with citations.

■ CORE FUNCTIONALITY Build a Retrieval-Augmented Generation (RAG) pipeline with the following stages:

1■■ Intent Analysis (Query Understanding) - Detect what kind of question the user is asking (academic, code, research, or hybrid). - Use GPT-5 or Gemini API for understanding context and intent.

2■■ Multi-Query Web Retrieval - Execute 2–3 parallel queries to web search APIs (Tavily, Serper.dev, Perplexity Search, or Google Custom Search). - Retrieve 5–10 top results with titles, text snippets, and URLs. - Remove duplicates and low-quality sources.

3■■ Neural Ranking / Filtering - Rank snippets using semantic similarity (via GPT embedding scoring or LLM judgment). - Prefer high-credibility sources (scholar, government, educational, documentation).

4■■ Passage Extraction / Chunking - Extract only the relevant paragraphs or facts from each snippet. - Chunk long text into smaller segments for summarization.

5■■ Summarization & Answer Generation - Feed the extracted facts into GPT-5 (or Gemini) for synthesis. - Combine results into one coherent, factual, citation-based answer. - Use the instruction: “Summarize only using information provided; include citations in [number] or [source] format.”

6■■ Citation and Grounding - Attach each output statement with its reference URL. - Format output as: “Answer text … [Source: example.com]”

7■■ Output to Devmate UI - Display the answer inside a styled “Research Mode” window in the UI. - Include: - Main answer (summary) - List of sources (clickable) - Option: “Open in full research view” for detailed summaries.

■ ENHANCEMENTS & FEATURES - Add toggle “Ask the Web / Research Mode” in Devmate’s UI. - Add Model Selector (GPT-5, Gemini 2.5 Pro). - Add Feedback buttons (“Helpful / Not accurate”) → store results for improvement. - Add Insight cards showing short summaries per source. - Add Query Expansion: reformulate the user query into 2–3 related search terms before retrieval. - Add Hallucination Guard: LLM must only respond based on retrieved content; do not invent facts.

■ DEVELOPMENT INSTRUCTIONS Language: Node.js (or TypeScript) Framework: Next.js (backend) + React (frontend) Storage: Replit DB for logs + API keys Styling: Tailwind CSS + Shadcn UI Key APIs allowed: - OpenAI API (GPT-5 / GPT-4o) - Google Custom Search / Tavily / Serper.dev - Gemini 2.5 Pro API (optional hybrid mode)

■■ WORKFLOW TO IMPLEMENT

Step 1: Create /api/research endpoint - Input: user query, selected model - Process: a. Analyze intent b. Expand query c. Retrieve results via search API d. Extract top snippets e. Generate final summary with citations f. Return structured JSON

Example Output: { "summary": "AI models like GPT-5 use RAG pipelines for grounded answers...", "sources": [{ "title": "OpenAI Docs", "url": "https://platform.openai.com/docs" }, { "title": "Perplexity AI Architecture", "url": "https://blog.bytebytego.com/p/how-perplexity-built-an-ai-google" }] }

Step 2: Create React UI Component - Search bar for user queries - Toggle for “Ask the Web” - Display results with citations and expandable cards

Step 3: Integrate into Devmate’s main dashboard - When user switches to “Research/Academic Mode,” use the RAG backend - If “Ask the Web” is off, fallback to normal chat mode.

■ PROMPT LOGIC (For Summarizer) Prompt: You are a professional academic summarizer. You will receive multiple web snippets with sources. Read them carefully, identify verified facts, and synthesize them into one clear and concise answer. Do not hallucinate. Only use the text provided. Include citations in square brackets with their source URL.

■ Example Expected Output:

Question: What is Retrieval-Augmented Generation in AI research?

Answer: Retrieval-Augmented Generation (RAG) is an AI technique that combines information retrieval with language model generation. It first fetches relevant documents from external sources and then generates responses grounded in that evidence. This improves factual accuracy and transparency compared to pure generative models. [Source: huggingface.co] [Source: blog.bytebytego.com]

Sources: 1. https://huggingface.co/blog/rag 2. https://blog.bytebytego.com/p/how-perplexity-built-an-ai-google

---

■ GOAL Build and integrate the entire workflow end-to-end so Devmate’s academic section behaves like a research-oriented Perplexity engine: - Smart search - Credible summarization - Citation display - Hybrid model use - Seamless UI integration

Output: Fully functional Node.js + React implementation inside Replit with modular backend and UI ready for deployment.