HamPlan: A Retrieval-Augmented Generation System for Hamilton College Course Planning

Cade Boiney, Ken Lam, Ognian Trajanov, Benjamin Zhao
Hamilton College
Clinton, NY 13323

***Abstract*—Course planning at Hamilton College involves navigating over 800 courses with complex prerequisites and scheduling constraints. We present HamPlan, a Retrieval-Augmented Generation (RAG) system providing conversational access to course information. Using OpenAI's text-embedding-3-large for retrieval and GPT-4-turbo for generation, the system integrates course catalogs, syllabi, and department requirements. Through iterative prompt engineering driven by faculty user testing, we achieved 60% "very helpful" ratings while addressing critical challenges in conversation memory management and scope control.**

I. Motivation

Hamilton College offers over 800 courses across 28 departments with complex prerequisite chains and distribution requirements. Students struggle to navigate this complexity—comparing course sections, understanding prerequisites, and accessing information scattered across catalogs, syllabi, and departmental websites. Existing tools like Student Planning provide search functionality but lack conversational interfaces for nuanced queries such as "What economics courses cover monetary policy?" or "When does Professor Owen teach Macroeconomic Theory?" HamPlan addresses this gap by providing natural language access to comprehensive course information.

II. Data Collection

We constructed a multi-source database containing 951 text chunks (embeddings) across three complementary sources:

Course Catalog (824 chunks): We built a web scraper using Python's `requests` and `BeautifulSoup` to extract structured data from Hamilton's Fall 2025 course catalog API, capturing course codes, descriptions, instructors, meeting times, and prerequisite requirements. Each course was embedded as a single chunk.

Department Requirements (41 chunks): We scraped major and minor requirement pages for all 28 departments, extracting information about concentration pathways and proficiency standards. Each department overview was embedded as a single chunk.

Course Syllabi (57 PDFs): We solicited syllabi from faculty across departments via email campaigns. This proved more challenging than anticipated—response rates varied significantly by department, with some disciplines (Economics, Computer Science, Chemistry) contributing numerous syllabi while others showed limited participation. We worked through department chairs to request archives, but ultimately achieved coverage across 15 departments. Syllabi were processed using `PyPDF2` for text extraction and chunked into passages (minimum 50 characters, split on paragraph breaks) to preserve semantic coherence, yielding 86 total chunks.

Preprocessing: All text was embedded using OpenAI's `text-embedding-3-large` model (3072-dimensional vectors, 8000-character input limit) and cached locally using Python's `pickle` module for efficiency, avoiding repeated API calls during development and reducing query initialization time by approximately 30 seconds.

TABLE I
Dataset Composition

Data Source	Embeddings	Coverage
Course Catalog (Fall 2025)	824	All departments
Department Overviews	41	28 departments
Course Syllabi (57 PDFs)	86	15 departments
Total	**951**	—

III. Model Architecture

HamPlan follows the standard RAG architecture (Figure 1). Given a user query, we compute its embedding and perform cosine similarity search against all 951 document embeddings, retrieving the top-50 most relevant passages. These passages are concatenated with a carefully engineered system prompt and conversation history, then passed to GPT-4-turbo (temperature=0.3, max_tokens=500) for response generation. We maintain conversation history across turns, summarizing older exchanges when the conversation exceeds 6 turns to preserve context without introducing noise from distant exchanges.

IV. System Development

RAG systems using pre-trained models do not undergo traditional gradient-based training. Instead, system behavior is shaped through *prompt engineering*—iteratively refining natural language instructions provided to the model. This process mirrors training conceptually: user feedback identifies failures, prompt refinements address issues, and the system is re-evaluated. We conducted three major iterations driven by faculty testing:

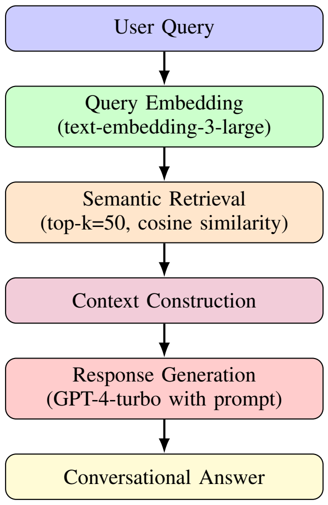
User Query
Query Embedding (text-embedding-3-large)
Semantic Retrieval (top-k=50, cosine similarity)
Context Construction
Response Generation (GPT-4-turbo with prompt)
Conversational Answer

Fig. 1. HamPlan RAG Pipeline

Iteration 1 (Baseline): Initial system prompt with basic course planning guidelines. Conversation memory threshold set to 10 exchanges before summarization.

Iteration 2 (Memory Fix): Faculty testing revealed conversation memory bugs—after 6-7 exchanges, the system conflated current questions with previous context, answering the wrong question. Investigation showed that accumulated conversation history caused the model to reference earlier queries when responding to new ones. We lowered the summarization threshold to 6 exchanges and added explicit prompt instruction: "IMPORTANT: Answer ONLY the current question, not previous queries in the conversation."

Iteration 3 (Scope Control): A Chemistry professor intentionally stress-tested the system with off-topic queries ("Who played drums for Nirvana?") and homework questions ("Which is more acidic, the alpha proton of an ester or a ketone?"). The initial system answered both using GPT-4's general knowledge. We refined the prompt to distinguish three query types: (1) *Answer*: course planning, prerequisites, schedules, and course topic overviews; (2) *Decline*: homework content and problem-solving requests; (3) *Decline*: off-topic queries unrelated to Hamilton academics.

This iterative refinement operates in prompt space rather than weight space—a form of manual RLHF (Reinforcement Learning from Human Feedback) conducted through natural language instructions.

V. Challenges

A. Data Collection Difficulties

Syllabus collection proved unexpectedly difficult. Faculty response rates varied dramatically across departments, with limited accessibility and inconsistent responsiveness slowing progress significantly. Our solution involved working through department chairs to access bulk archives and building comprehensive web scrapers to extract course catalog data, prerequisites, and major requirements as supplementary sources. Ultimately, we collected 57 syllabi across 15 departments, which were chunked into 86 embeddings. Combined with 824 catalog courses and 41 department overviews, this provided comprehensive coverage despite syllabus gaps.

B. Conversation Memory Pollution

A Computer Science professor reported the system "failed after 4 messages" during a chain of aerospace-related queries. Investigation revealed that after 6-7 conversational exchanges, accumulated context polluted responses—the system began conflating previous queries with current ones. For example, when asked "Does CS-102 have prerequisites?" after discussing several other courses, the system referenced a previously mentioned course instead of answering about CS-102 directly.

Solution: We lowered the conversation summarization threshold from 10 to 6 exchanges and added explicit prompt guidance: "Answer ONLY the current question, not previous queries in the conversation." Testing confirmed the fix—the system now correctly maintains context without conflation across extended conversations.

C. Scope Control and Academic Integrity

The Chemistry professor's stress-testing with trivia and homework questions revealed a critical academic integrity issue. While students can access general AI tools regardless, an institutionally-provided tool answering homework questions sends the wrong message and could undermine academic policies. As the professor noted: "If it's sourcing from the internet, then I worry about students asking things like which professor is better and having the chatbot use info like ratemyprofessor.com."

Solution: We implemented scope control through prompt engineering, distinguishing acceptable queries (course planning, prerequisites, topic overviews such as "What is macroeconomics about?") from unacceptable ones (homework help like "Explain supply and demand curves" or off-topic trivia). Declined queries receive: "I'm designed to help with Hamilton course planning. How can I assist you with courses or academic requirements?" Subsequent testing confirmed both the Nirvana question and chemistry homework question are now properly declined.

D. Multi-Section Display

An Economics professor noted that queries sometimes displayed only one section when multiple existed: "When I type 'When is Macroeconomic Theory taught', I only get Ann Owen's 10am section, but not her 11am section." This occurred because top-k retrieval might return only one section if embeddings were similar, and the prompt lacked explicit guidance to display all results.

Solution: We added prompt instruction: "If multiple sections of the same course appear in the information, list ALL of them with their meeting times." Subsequent testing confirmed the system now displays both of Professor Owen's sections (10am and 11am) and all sections for other multi-section courses.

VI. Results and Analysis

We evaluated HamPlan through faculty user testing with structured feedback forms providing Likert-scale ratings (1=Not at all helpful to 5=Extremely helpful) plus qualitative comments. Five faculty members across departments (Computer Science, Economics, Chemistry, Mathematics/Statistics, Music) participated in testing.

TABLE II
User Satisfaction Metrics (n=5)

Rating Category	Count	Percentage
Very helpful	3/5	60%
Moderately helpful	1/5	20%
Slightly helpful	1/5	20%
Not at all helpful	0/5	0%
Mean Rating	**3.8/5.0**	

Response Accuracy: Manual validation of system responses across test queries revealed high factual accuracy for course-related information (prerequisites, schedules, instructors). Errors were limited to missing data (courses not offered in Fall 2025) rather than incorrect generation or hallucination.

TABLE III
User Feedback Themes

Theme	n	Example
Usability	4/5	"Convenient chatbot format"
Scope Issues	1/5	Answered chemistry homework/trivia
Data Gaps	2/5	Missing courses from current semester

Key Findings: All critical feedback drove concrete improvements. The Chemistry professor's scope testing prompted refinements that now successfully decline inappropriate queries. The Music professor noted missing courses (MUS 368, 370)—investigation revealed these courses exist in Hamilton's catalog but weren't offered in Fall 2025, explaining their absence from our dataset. The Computer Science professor's reported prerequisite issue was not reproducible upon retesting, suggesting resolution through our conversation memory fixes.

The Mathematics/Statistics professor validated core functionality: "Gave good and accurate information." The Economics professor's feedback on multi-section display was directly addressed through prompt engineering. Notably, one evaluator commented: "I think this was super useful and hope we have a version of it available for students and faculty soon!"

VII. Additional Insights

Prompt Engineering as Training: Despite using transformer models with sophisticated attention mechanisms, the system required explicit prompting ("Answer ONLY the current question") to maintain proper focus. This highlights the gap between model capabilities and effective utilization—architectural features alone do not guarantee correct behavior. Our iterative prompt refinement mirrors training conceptually but operates in instruction space rather than weight space.

Production-Grade Testing: Faculty didn't treat this as an academic exercise—they stress-tested with adversarial queries (homework questions, trivia), edge cases (multi-section courses, complex prerequisite chains), and realistic planning scenarios. This rigorous evaluation surfaced bugs that typical academic testing might miss, demonstrating the value of real-world user feedback.

Academic Integrity Considerations: The Chemistry professor's concern about homework help reveals important institutional considerations for AI tools. Even when students have external access to AI assistants, institutionally-provided tools must maintain clear boundaries to avoid undermining academic policies and sending inappropriate signals about acceptable AI use in coursework.

VIII. Conclusion and Future Work

HamPlan demonstrates that production-quality RAG systems can be built using pre-trained models and prompt engineering without expensive fine-tuning. Through iterative refinement driven by faculty feedback, we achieved 60% "very helpful" ratings while resolving critical issues in conversation memory and scope control. Project code and documentation are available at https://github.com/bzhao3927/HamPlan.

The most significant opportunity for future work lies in institutional collaboration to deploy HamPlan as an official Hamilton resource. This would involve: (1) partnering with the Registrar's Office for direct API access to Student Planning data, ensuring real-time accuracy and comprehensive course coverage; (2) working with IT Services to deploy the system as an authenticated web service integrated with Hamilton's existing infrastructure; (3) establishing clear usage policies and academic integrity guidelines in consultation with faculty governance; (4) implementing student-facing deployment with proper authentication and usage monitoring.

Faculty response during testing—"I hope we have a version of it available for students and faculty soon!"—demonstrates genuine demand for this tool. Moving from proof-of-concept to institutional service would require addressing data privacy, computational costs, and ongoing maintenance, but the technical foundation is sound. Our experience demonstrates that thoughtful application of existing AI technologies can create practical tools that genuinely improve student and faculty experiences at liberal arts colleges.

References

[1] OpenAI, "GPT-4 Turbo," 2024. [Online]. Available: https://platform.openai.com/docs/models/gpt-4-turbo

[2] OpenAI, "Embeddings Guide," 2024. [Online]. Available: https://platform.openai.com/docs/guides/embeddings

[3] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," *NeurIPS*, 2020.