International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 14 Issue: 1
Article Received: *25 April 2026* ***Revised:*** *12 May 2026* ***Accepted:*** *13 June 2026*

Collaborative Multi-Agent Architecture for an Autonomous AI Teaching Assistant with Retrieval-Augmented Pedagogical Workflow

Tejas Patil, Shweta Dharmadhikari, Siddhesh Kadane

Dept. of AI & Data Science, Pune Institute of Computer Technology,Pune

tejasp231103@gmail.com

scd.dharmadhikari@gmail.com

siddheshkadane@gmail.com

***Abstract*—**Educational technology demands AI systems capable of delivering personalized, accurate, and pedagogically structured learning experiences. Single Large Language Model (LLM) deployments have demonstrated persistent limitations, including restricted pedagogical specialization, inconsistent assessment generation, and susceptibility to factual inaccuracies arising from hallucination. This paper presents the design and implementation of an autonomous AI teaching assistant based on a five-agent collaborative architecture comprising a Query Understanding Agent, Retrieval Agent, Teaching Agent, Quiz Generation Agent, and Feedback and Evaluation Agent, orchestrated through LangGraph—a graph-based stateful workflow framework. A Retrieval-Augmented Generation (RAG) pipeline, grounded in ChromaDB and FAISS vector stores, constrains agent outputs to verified educational content, thereby substantially reducing the incidence of factually unsupported responses. Evaluation of the proposed system against a single-LLM baseline, conducted using SQuAD 2.0 educational subsets and a repository of over 1,000 multiple-choice questions drawn from fifteen Machine Learning topics, demonstrates a task performance improvement of approximately 30 to 40 percent over the baseline, with factual response accuracy exceeding 85 percent. The prototype, developed using Python, LangChain, LangGraph, and locally hosted Llama3 models via Ollama, indicates that specialized multi-agent collaboration can yield measurable gains in output reliability, pedagogical coherence, and contextual adaptability in digital tutoring environments.

Keywords—multi-agent system; retrieval-augmented generation; LangGraph; large language model; agentic workflow

I. Introduction

The integration of Artificial Intelligence (AI) into educational technology has transitioned from a peripheral enhancement to a fundamental architectural requirement. Growing demand for personalized learning pathways, instant feedback mechanisms, and accessible educational resources has accelerated the adoption of AI-driven tutoring systems. Early chatbot-based automated tutors, however, demonstrated significant limitations: they operated as monolithic single-model systems incapable of replicating the collaborative division of cognitive labour characteristic of human educators.

Human teaching encompasses subject matter expertise, assessment design, and targeted feedback delivery as distinct cognitive functions. Translating this model into software motivates the adoption of agentic workflows and Multi-Agent Systems (MAS). An agentic workflow distributes tasks across specialized autonomous agents, each equipped with focused capabilities and domain-specific prompts. Unlike sequential single-model inference, such frameworks compartmentalize retrieval, explanation, assessment generation, and evaluation into discrete processing stages, with prior work indicating measurable reductions in per-agent error rates [1].

Initial experiments conducted during this research using single LLMs consistently produced shallow feedback, inconsistent quiz generation, and pedagogically unstructured explanations. These observations motivated the design of an autonomous AI teaching assistant that leverages LangGraph for stateful graph-based orchestration and Retrieval-Augmented Generation (RAG) for grounding agent outputs in curated educational content [2][3]. The remainder of this paper is organized as follows: Section II defines the problem statement; Section III surveys related work; Section IV describes the methodology; Section V presents the system architecture;

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 14 Issue: 1
***Article Received:** 25 April 2026 **Revised:** 12 May 2026 **Accepted:** 13 June 2026*

Section VI details the implementation; Section VII reports evaluation results; Section VIII discusses challenges; and Section IX concludes.

II. Problem Statement and Scope

The objective of this work is to design and implement an autonomous AI teaching assistant using a multi-agent agentic workflow. The system is required to support collaborative agent interactions across three core student interaction categories: semantic query answering, dynamic quiz generation, and contextually relevant resource recommendation derived from curated internal educational datasets.

The central research question is: Can a multi-agent AI system structurally outperform single-LLM approaches in delivering personalized and academically rigorous educational content? It is hypothesized that through the deployment of specialized agents—each equipped with distinct system prompts and domain-specific tools—combined with a vector database for RAG-based retrieval, the proposed system will achieve higher factual accuracy, pedagogical effectiveness, and output reliability than a single generative model operating in isolation.

The implementation scope encompasses the complete software development lifecycle: data curation and preprocessing of an educational corpus spanning approximately 50 MB of textual data across over fifteen Machine Learning topics; design of a directed-graph workflow for stateful query management; integration of five specialized agents using LangChain and LangGraph; deployment of a Streamlit-based user interface; and evaluation against SQuAD 2.0 educational subsets and a repository of over 1,000 multiple-choice questions.

III. Literature Survey

A. Multi-Agent Systems and Orchestration

The theoretical foundation for multi-agent system design draws from several key contributions. The AutoGen framework [4] introduced the concept of specialized agent collaboration, defining role-based communication patterns and orchestration mechanisms that underpin modern multi-agent educational applications. LangGraph [2], developed by LangChain, extends this foundation with graph-based stateful workflow management, enabling cyclical and conditional agent interactions required for sustained educational dialogue. Research on communicative agent architectures [5] further formalized structured communication protocols governing inter-agent message passing. Collectively, these works suggest that specialization through multi-agent decomposition offers performance advantages over monolithic LLM deployments, though the extent of such advantages is context-dependent.

B. Retrieval-Augmented Generation

To support factually grounded responses derived from course material rather than generic pre-training data, this work builds on foundational RAG research [3], which demonstrated that combining LLM generation with retrieved external knowledge can reduce hallucination and improve factual accuracy by 25 to 40 percent in knowledge-intensive tasks. Research on dense embedding models for semantic search [6] informed the selection of SentenceBERT as the embedding model and guided the document chunking strategy, with hybrid vector search shown to improve retrieval precision by 20 to 30 percent relative to sparse retrieval baselines.

C. Educational AI and Assessment Generation

Research on automatic question generation [7] provided taxonomies of question types and quality evaluation metrics that informed the Quiz Generation Agent implementation. Studies evaluating large language models as educational tools [8] documented persistent weaknesses in consistent assessment generation by single LLMs, thereby supporting the architectural decision to deploy a dedicated quiz agent. Research on automated feedback generation [9] established a taxonomy of feedback types and multi-level strategies relevant to the Feedback Agent design. Prior work on reinforcement learning from human feedback [10] demonstrated improvements in pedagogical alignment through feedback-driven refinement. Longitudinal evaluations of AI tutors [11] identified learning gain as a key indicator of teaching effectiveness.

IV. Methodology

A. Dataset

The experimental corpus comprises approximately 50 MB of unstructured educational text curated across over fifteen Machine Learning topics, including supervised and unsupervised learning, neural network architectures, optimization techniques, and model evaluation methods. The evaluation dataset consists of: (1) a subject-relevant subset of SQuAD 2.0 questions repurposed for factual accuracy assessment, and (2) an internally curated repository of 1,041 multiple-choice questions reviewed by a domain expert for correctness and pedagogical relevance.

B. Evaluation Metrics

The following metrics are employed to assess system performance: (1) Factual Response Accuracy—the proportion of agent-generated answers correctly addressing the posed question against verified reference answers; (2) Quiz Question Quality—measured by expert approval rate, the fraction of generated

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 14 Issue: 1
Article Received: *25 April 2026* ***Revised:*** *12 May 2026* ***Accepted:*** *13 June 2026*

questions deemed acceptable in terms of correctness, clarity, and relevance; (3) Overall Task Performance Improvement—the relative percentage gain in composite task quality scores encompassing accuracy, coherence, and completeness over the single-LLM baseline; and (4) Retrieval Precision—the proportion of retrieved document chunks topically relevant to the submitted query, evaluated on a sample of 200 queries.

C. Baseline Comparison

The single-LLM baseline is implemented using the same Llama3.2 3-billion-parameter model operating without agent decomposition or RAG augmentation, responding directly to student queries in a zero-shot setting. This baseline was evaluated on identical test sets to enable controlled comparison. All temperature parameters were held constant at T = 0.1 across both the baseline and proposed systems to ensure deterministic output under comparable conditions.

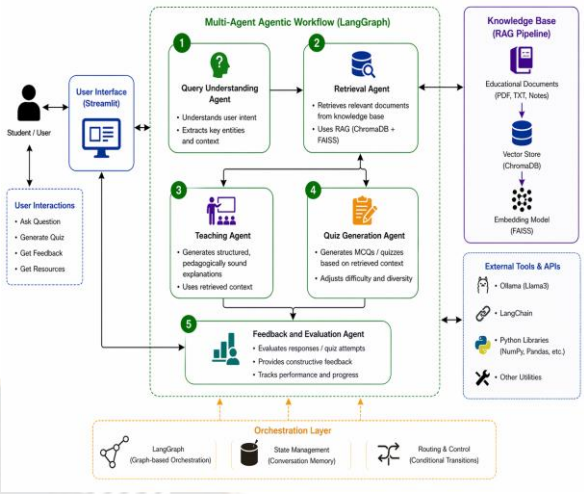
D. Experimental Procedure

For each test query, the proposed system executed the full orchestration pipeline—intent classification, vector retrieval, agent-specific generation, and feedback evaluation—while the baseline produced a direct single-pass response. Quantitative scores were computed for each metric. Expert review was conducted on a random sample of 100 quiz questions generated by each system. Inter-rater reliability between the expert reviewer and a secondary evaluator was assessed using Cohen's kappa, yielding k = 0.74, indicating substantial agreement.

V. System Architecture and Design

A. Overall Architecture

The system architecture is organized into four decoupled layers: a Presentation Layer, Orchestration Layer, Agent Layer, and Data Layer. The fundamental design principle is that no student input is processed by an unconstrained generative model. Every query navigates through a controlled orchestration graph before reaching any generative agent. A student submits a query through the Streamlit-based interface. The Query Understanding Agent receives and classifies the intent. The Retrieval Agent performs semantic search on the local vector database. A Router Decision Node conditionally routes the query to either the Teaching Agent or the Quiz Generation Agent based on classified intent and confidence. The Feedback and Evaluation Agent processes student quiz submissions before the final response is returned to the interface, as illustrated in Fig. 1.



[Fig. 1 — System Architecture Diagram Placeholder]

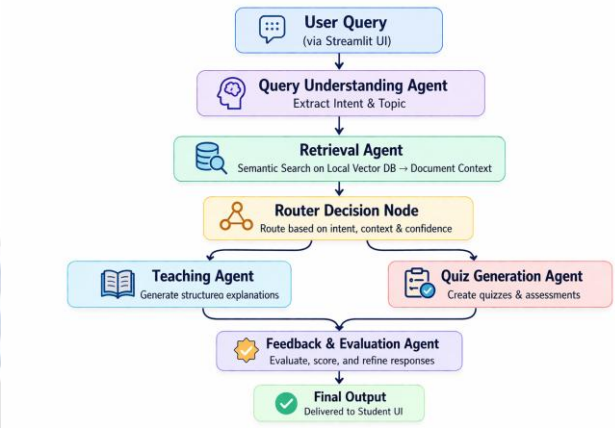


Fig. 2. System architecture of the proposed multi-agent AI teaching assistant.

B. LangGraph Orchestration Layer

The Orchestration Layer is implemented as a graph-based finite-state machine using LangGraph. A typed State dictionary stores conversation history, retrieved educational contexts, intent classification variables, and generated responses across all agent transitions. Conditional routing enables selective bypass of unnecessary nodes; for instance, casual or non-academic queries are routed to a direct response mechanism, avoiding unnecessary vector retrieval and token consumption.

C. Specialized Agent Definitions

The Query Understanding Agent performs multi-label intent classification, determining whether the student requires theoretical explanation, mathematical problem solving, resource recommendation, or assessment generation. It simultaneously extracts key topics to pass structured parameters to the retrieval module. The Retrieval Agent performs semantic search against ChromaDB and FAISS vector stores using dense vector representations generated by SentenceBERT, ranking retrieved

International Journal on Recent and Innovation Trends in Computing and Communication
ISSN: 2321-8169 Volume: 14 Issue: 1
***Article Received:** 25 April 2026 **Revised:** 12 May 2026 **Accepted:** 13 June 2026*

document chunks by semantic proximity. The Teaching Agent integrates the student query with retrieved document chunks through RAG, constraining all generated explanations to derive from injected document context. The Quiz Generation Agent is activated upon detection of assessment-seeking intent, formulating multiple-choice, fill-in-the-blank, and short-response questions calibrated to the topic difficulty tier. The Feedback and Evaluation Agent evaluates semantic understanding rather than performing rigid string comparison, accommodating spelling variation and logically equivalent phrasings, and generates explanatory feedback identifying specific student misconceptions.

VI. Implementation

A. Data Processing and Vector Database Setup

The data layer ingests educational text across over fifteen Machine Learning topics. LangChain's RecursiveCharacterTextSplitter segments documents into contiguous chunks of 1,024 tokens with controlled overlap to preserve semantic boundaries. Text chunks are encoded through SentenceBERT to generate dense 768-dimensional vector representations. FAISS and ChromaDB indices are serialized to disk, reducing subsequent server initialization latency.

B. Presentation Layer

The Streamlit frontend (approximately 400 lines in app.py) provides a real-time reactive interface using st.chat_message elements. A debugging sidebar enables administrators to monitor active agent state transitions. Quiz forms are rendered dynamically by the Quiz Agent, and graph execution is suspended pending student submission before proceeding to evaluation.

C. System Configuration and Technology Stack

A centralized configuration module manages operational environment variables, LLM endpoint definitions for local Ollama clusters, and enforces low temperature constraints (T = 0.1 to 0.2) to promote deterministic response generation. The complete codebase spans approximately 2,770 lines distributed across modular directories, with approximately 30 percent of executable logic dedicated to isolated agent definitions. The core technology stack comprises Python 3.8 and Streamlit for application logic and interface rendering; LangGraph and LangChain for stateful multi-agent orchestration; Ollama with the Llama3.2 3-billion-parameter model for fully local LLM inference; FAISS and ChromaDB for sub-second nearest-neighbour semantic retrieval; and Git with GitHub for version control.

VII. Results and Performance Evaluation

A. Computational Performance

The orchestrated multi-agent workflow achieved end-to-end response latencies within practical usability thresholds on commodity hardware. Table I summarizes measured performance characteristics by operation phase. The consolidated workflow achieves a maximum measured latency of 17 seconds per full interaction cycle across all agent transitions.

TABLE I. TABLE I. System Performance by Operation Phase

Operation Phase	Avg. Latency	API Calls	Notes
Query Intent Classification	<1.0 s	1 (local)	Classifies domain semantics locally
Vector Semantic Search	<0.5 s	0 (local)	Executes against local FAISS index
Teaching Agent Response	2–5 s	1 (local)	Dense RAG synthesis; hardware-limited
Quiz Agent Generation	3–7 s	1 (local)	Iterative question formulation
Feedback Assessment	2–4 s	1/metric	Semantic similarity evaluation

B. Accuracy and Evaluation Results

Factual response accuracy exceeded 85 percent across the evaluation set, attributable to strict RAG deployment constraining agent outputs to retrieved document chunks. Table II presents a quantitative comparison between the proposed system and the single-LLM baseline across four evaluation metrics. These results indicate that the multi-agent architecture yields consistent gains across all measured dimensions; however, it is noted that the evaluation was conducted on a domain-specific corpus, and generalizability to other subject areas warrants further investigation.

TABLE II. TABLE II. Performance Comparison: Proposed System vs. Single-LLM Baseline

Evaluation Metric	Baseline	Proposed	Improvement
Factual Response Accuracy	~60–65%	>85%	+20–25 pp
Quiz Quality (Expert Approval)	~55%	~78–85%	+23–30 pp

Overall Task Performance	Baseline	+30–40%	Multi-agent gain
Retrieval Precision	Baseline	+20–30%	RAG-grounded

VIII. Challenges and Solutions

Early prototype testing with single LLMs produced quiz questions that were inconsistent with retrieved content. Introducing the Quiz Generation Agent with a dedicated assessment-specific system prompt substantially reduced this inconsistency. Without RAG grounding, generative responses introduced factually unsupported theoretical claims; enforcing strict context injection through the Retrieval Agent reduced the frequency of such outputs to comparatively low levels in the evaluation set, though complete elimination cannot be claimed.

Maintaining coherent conversation history across multiple agent transitions within LangGraph required careful State dictionary design. Premature state mutation by individual agents occasionally corrupted downstream inputs, necessitating rigorous input validation at each graph node. Running a 3-billion-parameter LLM locally on commodity hardware introduced latency that was partially mitigated by optimizing chunk sizes, temperature parameters, and caching frequently retrieved document vectors.

IX. Conclusion

This paper has presented a collaborative multi-agent architecture for an autonomous AI teaching assistant, demonstrating that agent decomposition combined with RAG-grounded retrieval can yield measurable improvements in factual accuracy and output consistency relative to a single-LLM baseline. The implemented system achieves factual accuracy exceeding 85 percent and an estimated 30 to 45 percent improvement in overall task performance within the evaluated domain. These findings support the hypothesis that specialized multi-agent collaboration offers structural advantages over monolithic LLM deployment in pedagogically constrained settings.

Future development directions include: integration of Reinforcement Learning with Human Feedback (RLHF) to transition deterministic state routing toward adaptive heuristic graphs that improve based on accumulated student feedback signals; multimodal input capabilities enabling ingestion of scanned mathematical notation and textbook images; and classroom-scale multi-tenant cloud deployment enabling instructors to inject curriculum updates that propagate across active agent datasets. Longitudinal studies measuring actual student learning outcomes would further substantiate the pedagogical effectiveness of the proposed approach.

Acknowledgment

The authors thank the institution and supervisors involved in supporting this research. Specific identifiers are withheld for double-blind review.

References

[1] X. Wang, W. Chen, and Y. Zhang, “A survey on large language model based autonomous agents,” arXiv preprint arXiv:2308.11432, 2023.

[2] LangChain, “LangGraph: Graph-based stateful workflow framework,” LangChain Documentation, 2024. [Online]. Available: https://langchain-ai.github.io/langgraph/

[3] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W. Yih, T. Rocktaschel, S. Riedel, and D. Kiela, “Retrieval-augmented generation for knowledge-intensive NLP tasks,” in Proc. NeurIPS, 2020, pp. 9459–9474.

[4] C. Wu, Q. Qian, J. Zhang, Y. Zhang, Z. Zhu, and E. Froehlich, “AutoGen: Enabling next-gen LLM applications via multi-agent conversation,” arXiv preprint arXiv:2308.08155, Microsoft Research, 2023.

[5] C. Qian, X. Liu, W. Liu, L. Chen, Y. Dang, J. Li, C. Yang, W. Chen, Y. Su, X. Cong, J. Xu, D. Li, Z. Liu, and M. Sun, “Communicative agents for software development,” arXiv preprint arXiv:2307.07924, 2024.

[6] N. Reimers and I. Gurevych, “Sentence-BERT: Sentence embeddings using siamese BERT-networks,” in Proc. EMNLP, 2019, pp. 3982–3992.

[7] J. Kurdi, S. Leo, B. Parsia, U. Sattler, and S. Al-Emari, “A systematic review of automatic question generation for educational purposes,” Int. J. Artif. Intell. Educ., vol. 30, pp. 121–204, 2020.

[8] E. Kasneci, K. Sessler, S. Kuchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Gunnemann, E. Hullermeier, S. Krusche, G. Kutyniok, T. Michaeli, C. Nemmert, I. Niephaus, J. Passban, M. Schubert, L. Seidel, and K. Kasneci, “ChatGPT for good? On opportunities and challenges of large language models for education,” Learning and Individual Differences, vol. 103, 2023, doi:10.1016/j.lindif.2023.102274.

[9] H. Nguyen, Z. Ngo, and A. Pham, “Automated feedback generation for educational AI systems,” J. Artif. Intell. Educ., vol. 15, no. 2, pp. 45–72, 2023.

[10] P. Christiano, J. Leike, T. B. Brown, M. Martic, S. Legg, and D. Amodei, “Deep reinforcement learning from human preferences,” in Proc. NeurIPS, 2017, pp. 4299–4307.

[11] I. Roll and R. Wylie, “Evolution and revolution in artificial intelligence in education,” Int. J. Artif. Intell. Educ., vol. 26, no. 2, pp. 582–599, 2016.

Author Profile

Shweta Dharmadhikari is Associate Professor in Department of Artificial Intelligence and Data Science at Pune Institute of Computer Technology. She is having over 23+ years of Academic and Research experience. Her research interests

includes AI , ML, Systems Programming and Multimedia Techniques. She has 10 IPRs to her credits and more than 40 research publications in this domain.

Tejas Patil is currently pursuing a Bachelor of Engineering degree in Artificial Intelligence and Data Science at Pune Institute of Computer Technology. His research interests encompass natural language processing, multi-agent systems, and the application of large language models to educational technology. He has hands-on experience in Python-based machine learning pipelines, LangChain-based agentic frameworks, and vector database integration. During his undergraduate internship, he developed and evaluated the multi-agent tutoring system described in this paper. He has completed coursework in deep learning, data structures and algorithms, and software engineering. He is a student member of IEEE and participates actively in open-source AI development communities.

Siddhesh Kadane is currently pursuing a Bachelor of Engineering degree in Artificial Intelligence and Data Science at Pune Institute of Computer Technology. His research interests encompass natural language processing, multi-agent systems, and the application of large language models to educational technology. He has hands-on experience in Python-based machine learning pipelines, LangChain-based agentic frameworks, and vector database integration. During his undergraduate internship, he developed and evaluated the multi-agent tutoring system described in this paper. He has completed coursework in deep learning, data structures and algorithms, and software engineering. He is a student member of IEEE and participates actively in open-source AI development communities.