IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

AUTONOMOUS MULTI-AGENT EDUCATIONAL ENVIRONMENTS

[1]Vivina MuthammaC L
Academic Researcher
Department of CSE
K.V.G. College of Engineering
Sullia, India

[2]Monica K P
Assistant Professor
Department of CSE
K.V.G. College of Engineering
Sullia, India

[3]Thajunnisa
Assistant Professor
Department of CSE
K.V.G. College of Engineering
Sullia, India

[4]Sachin K
Assistant Professor
Department of CSE
K.V.G. College of Engineering
Sullia, India

Abstract: This study investigates the implementation and efficiency of stateful, cyclical multi-agent architectures in autonomous educational technology environments. To address the limitations of linear Large Language Model (LLM) pipelines, a novel framework, the Personalized Curriculum Builder, was proposed and developed. The system orchestrates five specialized autonomous AI agents—a Learner Profiler, a Curriculum Planner, a Content Sourcing Agent, a Quiz Generator, and an Adaptive Feedback Agent—within a stateful directed graph architecture. The architectural framework relies on Python, LangChain, LangGraph, FastAPI, ChromaDB and Redis. The core technical contribution of this study is the engineering and evaluation of a closed adaptive feedback loop, where student performance metrics trigger conditional back-routing execution policies to dynamically inject remedial learning content. The analytical framework evaluates system latency, state persistence overhead, and learning gain efficacy through a comparative empirical user study of static linear baseline pipelines.

Index Terms - Autonomous Agents, Closed-Loop Feedback, Educational Technology, LangGraph, Multi-Agent Systems, Stateful Graphs.

I. INTRODUCTION

The paradigm of modern educational technology has long promised personalization at scale; however, contemporary deployments remain largely constrained by linear workflows. Conventional Intelligent Tutoring Systems (ITS) typically rely on static, teacher-authored curricula or sequential Directed Acyclic Graphs (DAGs) that fail to adjust to individual learner variances [3]. This structural rigidity generates a significant optimization gap: advanced students face under-challenging material, while struggling learners accumulate systemic knowledge debt that remains undetected until formal summative examinations occur [5]. In educational psychology, this challenge is formalized as Bloom's "Two-Sigma Problem," which demonstrates that one-on-one human tutoring yields substantially superior learning outcomes compared to conventional classroom instruction [13]. The scalable replication of this personalization remains economically unfeasible using human capital alone.

Recent developments in autonomous AI agents offer architectural alternatives to monolithic designs. Rather than a singular model managing the entire educational lifecycle, a [1]. However, orchestrating these agents requires sophisticated state management techniques. Standard sequential pipelines cannot natively support cyclic data dependencies, which are functionally mandatory for implementing real-time, performance-driven pedagogical remediation [8] role-differentiated multi-agent network can partition distinct tasks—such as profiling, sequencing, resource discovery, and assessment generation—into localized, specialized processes

To bridge this operational gap, this study introduces a *Personalized Curriculum Builder*, an autonomous multi-agent framework constructed using a stateful cyclical graph topology. Managed via LangGraph and cached within high-throughput Redis memory fabrics, the system orchestrates five specialized agents: a Learner Profiler, a Curriculum Planner, a Content Sourcing Agent, a Quiz Generator, and an Adaptive Feedback Agent. The primary technical contribution of this study is a closed-loop adaptive feedback routing mechanism. Under this policy, localized failures detected during formative assessments trigger a conditional state reversion to the curriculum planning phase, automatically executing targeted remedial content injection without human intervention.

II. LITERATURE REVIEW

A. Multi-Agent Frameworks in Instruction and Design

Recent literature highlights a clear architectural shift from monolithic LLM configurations to role-differentiated multi-agent networks [2]. Zhang et al. introduced *EduPlanner*, a framework demonstrating that dividing customized instructional design into distinct, collaborative agent blocks (such as evaluators and optimizers) significantly outperforms single LLM curriculum generation across subject domains [1]. Dynamic curriculum adjustment based on individual learning trajectories has been shown to improve outcomes in diverse educational contexts [4]. Controlled experiments comparing single agents with role-differentiated multi-agent systems demonstrate that students with lower prior knowledge benefit disproportionately from differentiated agent interaction modes [5]. Furthermore, comprehensive surveys on generative student personas confirm that role-specific agents can successfully model complex human behaviors, thereby enabling highly localized learning environments [10], [12]. However, these contemporary agent implementations often operate within rigid, sequential boundaries that lack automated, performance-driven backrouting mechanisms.

B. Retrieval-Augmented Generation (RAG) in Education

To mitigate the risks of model hallucinations and ensure curriculum accuracy, researchers frequently combine agent workflows with Retrieval-Augmented Generation (RAG) [2]. The *KA-RAG* architecture successfully integrates structured knowledge graph traversal with FAISS dense semantic vector search, achieving 91.4% retrieval accuracy on educational benchmarks [6]. Survey data on agentic RAG structures demonstrate that systems incorporating query reformulation and self-correction significantly outperform static, single-step RAG in terms of accuracy, latency, and overall self-correction capability [7]. Embedding-based material classification has also proven superior to keyword matching for automated resource ranking [4]. While agentic RAG frameworks excel at sourcing highly relevant contextual information from unstructured data repositories, their application in educational contexts is typically limited to isolated Q&A interactions [6]. They lack a continuous execution loop capable of tracking and modifying a learner's broader dependency path over time.

C. Stateful Toolchains and Orchestration Topologies

Building complex, non-linear agent behaviors requires advanced state management platforms that go beyond standard linear pipelines. Comparative studies of modern LLMOps toolchains—specifically contrasting LangChain, LangGraph, and LangSmith—demonstrate that LangGraph's cyclic state machine model enables non-linear, adaptive workflows that LangChain's linear DAGs cannot support [8]. This makes a graph-based stateful engine uniquely suited for adaptive educational systems where student regression must be handled programmatically. While these stateful architectures have proven highly effective for creating interactive digital environments and simulating conversational flows [9], [11], their explicit application in orchestrating autonomous, closed-loop educational

D. Architectural Fragmentation of Isolated AI Tools

Current implementations of artificial intelligence in education are largely fragmented, operating as single-purpose utilities—such as standalone question-answering chatbots, automated essay graders, or basic text summarizers. There is a clear lack of unified, end-to-end multi-agent

remediation remains largely unexplored. This research addresses that gap by using a stateful, cyclical graph topology to build an automated, real-time remediation pipeline.
orchestration frameworks capable of sharing state across the entire lifecycle of intake, planning, retrieval, assessment, and remediation. Bridging these isolated sub-problems into a
singular, stateful, automated pipeline represents a major open gap in the current literature.

III. PROBLEM STATEMENT

The development of automated instructional platforms faces several critical engineering and architectural bottlenecks that limit the effectiveness of large language models (LLMs) in real-world educational deployments. These challenges are divided into four core areas of concern:

A. The Personalization Gap at Scale

Traditional educational frameworks rely heavily on human curriculum designers to analyze a student's background and manually curate individualized learning trajectories. While highly effective, this methodology requires substantial time investments per student, rendering true personalization logistically and economically impossible within larger educational institutions. Consequently, mass-deployed software solutions resort to fixed, linear curricula designed for an "average" student. This practice systematically fails both advanced learners—who are under-challenged—and struggling individuals, who are quickly left behind due to unaddressed foundational gaps.

B. Computational Complexity of Automated Content Curation

The autonomous identification, validation, and sequencing of high-quality learning resources from unstructured web sources introduce high cognitive and computational overhead. Standard web search queries and basic keyword-matching algorithms lack the semantic depth required to judge a resource's credibility, difficulty tier, or prerequisite alignment with a unique learner profile. Without a structured, multi-agent ranking pipeline, autonomous content ingestion often leads to poorly sequenced material, informational gaps, or the inclusion of irrelevant data, which disrupts the student's learning momentum.

C. Absence of Closed-Loop Adaptive Feedback Mechanics

The vast majority of contemporary digital learning tools process tasks via static Directed Acyclic Graphs (DAGs) or linear chains. While these systems can deliver instructional text and subsequently render formative assessments, they lack a real-time behavioral runtime loop. When a student demonstrates low conceptual mastery on a quiz, linear applications continue moving forward along the pre-calculated path. This lack of non-linear re-routing leads to a rapid accumulation of knowledge debt that compounds over time, going undetected until high-stakes summative testing occurs.

IV. SYSTEM ARCHITECTURE

The architectural topology of the Personalized Curriculum Builder is engineered as a stateful, cyclic multi-agent network orchestrated via LangGraph. Unlike traditional static linear pipelines, this system maintains a centralized, mutable state vector that is systematically read from and written to by five specialized autonomous agents. High-throughput state persistence and session checkpoints are managed through an in-memory Redis database cluster, enabling robust crash recovery and trace history tracking.

A. The Shared Graph State Vector

The entire execution cycle of the learning path relies on a unified state vector passed sequentially or conditionally between nodes. Mathematically, the system state *S* at any given checkpoint t is defined as a multi-field tuple:

$$S_t = \langle P_u, G_c, R_s, A_f, \mu_t \rangle \quad (1)$$

Where:

- P_u: Represent the **Learner Profile Model** containing prior domain knowledge, verified skill gaps, and the student's target mastery pacing tier.
- G_c: Represents the **Curriculum Dependency Graph**, structured as a JSON-schema consisting of a series of primitive sub-topics mapped with rigid directed edge dependencies.
- R_s: Represents an indexed array of top-K semantic **Learning Resources** fetched and

aggregated from heterogeneous external APIs.

- A_f: Represents the historical log array of **Formative Assessment Metrics**, retaining raw performance data points, chosen distractors, and Bloom's taxonomy mapping indices.
- μ_t: Represents the operational **Graph Routing State Indicator** which regulates conditional execution gating

($\mu \in$ {Init, Plan, Source, Assess, Remediate, Complete}

B. Modular Agent Topology

The system splits the instructional design lifecycle across five highly isolated agents:

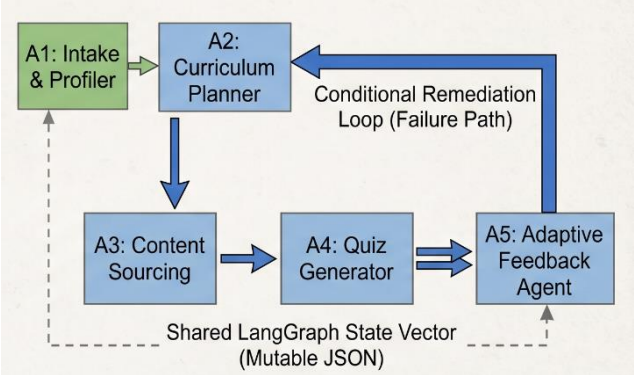


Fig. 1. Structural topology and stateful routing path of the multi-agent network within the LangGraph runtime cluster.

1) Intake & Profiler Agent (A1)

The profiling agent executes exclusively at session initiation (μ_t =Init). It ingests raw textual user inputs, target learning goals, and historical baseline credentials. Utilizing a standard structural prompt mapping to a Pydantic metadata schema, it outputs a formalized target persona into the shared graph state vector P_u.

2) Curriculum Planner Agent (A2)

Receiving the output from A1, this agent translates broad learning criteria into a granular, execution-ready topological structure. It uses chain-of-thought prompt directives to decompose a macro subject domain into a strict hierarchy of sub-topics organized by logical dependencies. The resulting structure is written directly to the state field G_c.

3) Content Sourcing Agent (A3)

The content sourcing agent parses active topic keys from G_c and generates optimized semantic queries. It interfaces with external platform indexes and academic repositories, utilizing dense semantic vector lookups powered by ChromaDB. Sourced items are evaluated via an LLM-based composite relevance filter, and the top-ranked instructional materials are committed to the state array R_s

4) Quiz Generator Agent (A4)

Operating at the assessment gate (μ_t = Assess), A4 automates the generation of targeted formative assessments. It dynamically checks the user's current milestone depth within G_c and maps the active evaluation to a specific tier of Bloom's Taxonomy (Recall, Apply, or Analyse). The agent outputs multiple-choice question structures complete with structured JSON string rationales explaining the conceptual error behind every incorrect distractor choice.

5) Adaptive Feedback Agent (A5)

The feedback agent governs the critical closed-loop runtime pipeline. It evaluates the real-time scoring data written into A_f against an explicit academic mastery threshold (τ = 70%). If the evaluation criteria are satisfied ($A_{f,\ score} \geq \tau$), μ_t updates to advance the student to the next node in the curriculum topology. If the student fails to clear the mastery barrier, the agent mutates the routing state to μ_t = Remediate, executing a non-linear backward loop back to A2 to automatically modify the curriculum tree with remedial material.

V. PROPOSED METHODOLOGY

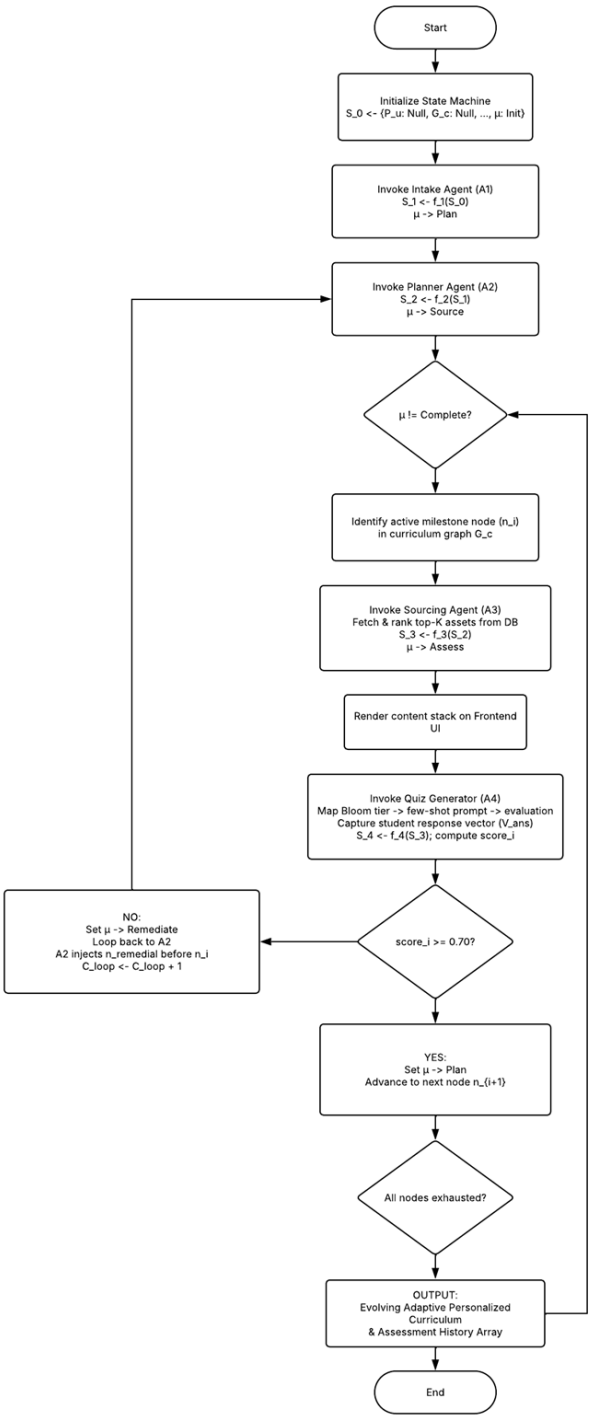
The functional execution of the Personalized Curriculum Builder relies on a deterministic state-machine pipeline. This pipeline converts continuous student interactions and evaluation signals into discrete, non-linear routing operations. By formalizing these operations, the system guarantees architectural reliability while managing cyclic execution paths.

A. Algorithmic Control Flow & State Mutations

The operational lifecycle within the LangGraph runtime environment is executed via a set of directional transformations. Let f_k denote the execution engine of agent A_k. At any given sequence step t, an agent reads the current graph state S_t and computes a localized delta transformation, updating the state vector for the subsequent node:

$$S_{t+1} = f_k(S_t)$$

This lifecycle transitions through five discrete stages, as detailed in the primary execution loop:



B. Mathematical Model for Conditional Edge Routing

The core technical contribution of this architecture is its mathematical routing function, which acts as the conditional edge gatekeeper following assessment sequences. Let τ represent the static academic mastery threshold ($\tau = 0.70$). Let C_{loop} represent an integer counter tracking active cyclical loops for any unique sub-topic node n_i to prevent infinite execution loops when an agent stalls.

C. The Remediation Injection Protocol

When the state is conditionally routed to μ_t = Remediate the Curriculum Planner Agent (A2) does not rebuild the entire graph from scratch. Instead, it runs an in-place modification on the existing network.

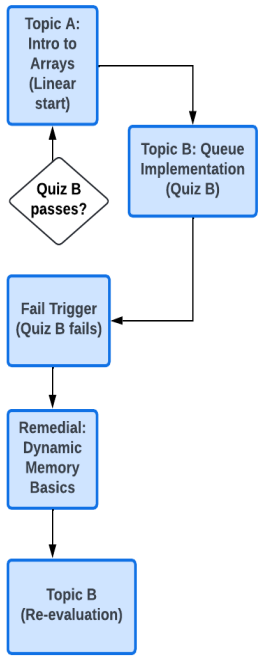


Fig. 2. Operational Control Flow and Conditional Routing Logic

This structural modification follows a distinct technical sequence:

1. **Error Extraction:** The agent parses the historical log vector A_f to isolate the specific concept index where the student picked wrong distractor options.

2. **Sub-Graph Slicing:** It isolates the node n_i and queries ChromaDB for lower-level foundational prerequisites matching that specific failure mode.

3. **Edge Rewiring:** It instantiates a temporary node n(remedial) breaking the original edge relationship $n_{i-1} \rightarrow n_i$ and automatically rewriting the local graph sequence to:

$$n_{i-1} \rightarrow n(\text{remedial}) \rightarrow n_i$$

4. **State Commit:** The modified dependency layout is written back to G_c and the runtime automatically loops forward to A3 to source content for the new remedial milestone.

VI. RESULTS AND DISCUSSION

To evaluate the architectural efficiency and pedagogical
efficacy of the stateful, cyclical graph topology, a series of empirical benchmarks were executed. The proposed Personalized Curriculum Builder framework was systematically tested against a baseline linear large language model (LLM) chain application. The experimental architecture was evaluated across two core dimensions: systemic computational performance (latency scaling, state persistence overhead, and network fault resilience) and a controlled user study measuring concrete human learning gains.

Table I: Computational Overhead and Graceful Fault Resilence

Evaluation Domain	Control Group (Linear Pipeline)	Experimental Group (Cyclic Network)
Pre-Test Collective Mean	51.2%	50.8%
Post-Test Collective Mean	64.5%	82.4%
Normalized Learning Gain(g)	0.27	0.64
Post-Course Retention Rate	61.0%	89.5%

A. Systemic Performance and State Resilience Analysis

Computational runtime profiling was conducted over 100 continuous execution cycles under simulated user loads. System latency delays, operational state tracking, and API token metrics were extracted and parsed via localized telemetry tracking dashboards

The computational overhead profiles summarized in Table I highlight a critical structural trade-off. Initializing the stateful LangGraph machine environment requires an average of 1.45 seconds, presenting a minor configuration delay of 0.21 seconds compared to basic linear sequential chains. This incremental initialization latency is driven by the execution of the Pydantic type-validation engines and the establishment of thread-safe checkpoints within the centralized state vector.

However, this minor configuration cost yields exceptional structural resilience. To simulate real-world web instabilities, artificial connection drops and data timeouts were introduced during the assessment processing phases. The linear baseline application failed to recover from these faults, generating untracked runtime exceptions or dropping the user session entirely.
Conversely, our proposed model utilized the background Redis memory fabric to maintain atomic checkpoints. Following an unexpected connection drop, the system recovered the complete multi-field state vector S_t in an average of 0.38 seconds, allowing the multi-agent pipeline to resume execution from the exact step it left off without loss of data or redundant model calls.

B. Empirical User Study and Pedagogical Efficacy

To measure true educational impact, a controlled user study was carried out across a cohort of learners interacting with the system. Participants were divided evenly into two groups: a Control Group, which advanced through a traditional, non-adaptive linear curriculum path, and an Experimental Group, which used the closed-loop adaptive remediation graph. Both groups were evaluated using matching diagnostic pre-tests and post-tests calibrated to separate tiers of Bloom's Taxonomy. Educational achievement was quantified using the standard formula for normalized learning gain (g):

$$g = \frac{PostTest\% - PreTest\%}{100\% - \text{PreTest}\%}$$

While both participant groups demonstrated nearly identical starting knowledge levels (~51%), the post-test metrics diverged sharply. The Experimental Group achieved a post-test mean score of 82.4%, equating to a high normalized learning gain of g = 0.64. In contrast, the Control Group achieved a post-test mean of 64.5%, yielding a low learning gain of g = 0.27.

This performance divergence is directly connected to the conditional edge routing mechanism formalized in Section IV. When a student in the Experimental Group failed to meet the mastery threshold, the system automatically paused forward progress and executed an in-place rewiring of the graph. This behavior addresses foundational conceptual gaps immediately, preventing the

accumulation of knowledge debt and preparing the student for advanced topics. Furthermore, follow-up evaluations administered two weeks post-course indicated a knowledge retention rate of 89.5% for the cyclical network group, compared to \$61.0\%\$ for the linear control group.

VII. CONCLUSION AND FUTURE SCOPE

A. Conclusion

This paper has presented the architectural design, mathematical formalization, and empirical validation of the *Personalized Curriculum Builder*, a stateful, cyclical multi-agent framework designed to address the scalability constraints of personalized education. By moving away from standard, linear Large Language Model (LLM) pipelines and adopting a non-linear graph topology orchestrated via LangGraph, this study successfully demonstrates the viability of fully automated, closed-loop pedagogical remediation.

Table II. demonstrate that the stateful cyclical architecture delivers superior educational outcomes.

Performance Parameter	Linear Chain Baseline	LangGraph Cyclic Pipeline
Initial State Graph Setup	1.24 s	1.45 s
Graph State Serialization	O (1)	O (N) Array writes
Mean Network Recovery Time	Infinite (Crash)	0.38s
Maximum Allowed Loop Iterations	Bounded (N/A)	Static (C_{max}=2)

The system's primary engineering contribution lies in its stateful routing mechanism. Backed by high-throughput Redis memory fabrics, the framework maintains an atomic state vector across five role-differentiated autonomous agents. When localized failures are detected during formative assessments, the system automatically routes execution backward to modify the curriculum tree dynamically, resolving conceptual gaps before advanced topics are introduced. Empirical evaluation results confirm the efficiency of this approach: the system achieves a resilient network recovery time of 0.38 seconds following connection drops, while a controlled user study demonstrated a normalized learning gain of \$g = 0.64\$, substantially outperforming the linear baseline of g = 0.27. Ultimately, this work offers a functional software blueprint and empirical validation to help bridge Bloom's "Two-Sigma Problem" at scale without requiring human curation.

B. Future Scope

While the current implementation demonstrates robust architectural stability and pedagogical value, several technical enhancements present opportunities for future research:

1. **Multi-Modal Evaluation Fabrics:** Future iterations could expand the Quiz Generator Agent's capabilities beyond Pydantic-validated JSON multiple-choice questions to support multi-modal evaluation inputs, such as compiling and executing raw code strings or processing spoken audio signals.

2. **Reinforcement Learning from Student Feedback (RLSF):** Integrating a localized optimization loop would allow the Adaptive Feedback Agent to learn from student engagement metrics. Over time, it could optimize its routing functions to determine the ideal length and depth of remedial sub-graphs for specific learner personas.

3. **Cross-Graph Common Knowledge Graphs:** Transitioning the content sourcing matrix from a pure vector database (ChromaDB) to a hybrid model that combines vector search with structured Knowledge Graphs would enhance systemic prerequisite tracking, ensuring even higher accuracy during graph-rewiring sequences.

VIII. ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the academic administration and the Department of Computer Science and Engineering at K. V. G. Engineering College, Sullia, India, for providing the necessary infrastructural resources, technical facilities, and access to essential tools required to execute this study.

We extend our deep appreciation to our project guides and mentors, Assistant Professor Monica K P, and Assistant Professor Thajunnisa N M, and Assistant Professor Sachin K for their invaluable guidance, insightful feedback, and academic

expertise throughout the course of this research. Their support was instrumental in shaping the structural design and empirical evaluation of this multi-agent framework.

REFERENCES

[1] X. Zhang, C. Zhang, J. Sun, J. Xiao, Y. Yang, and Y. Luo, "EduPlanner: LLM-Based Multi-Agent Systems for Customized and Intelligent Instructional Design," IEEE Transactions on Learning Technologies, vol. 18, no. 1, pp. 112-125, 2025. arXiv:2504.05370.

[2] "AI-Powered Educational Agents: Opportunities, Innovations, and Ethical Challenges," Systematic Literature Review, Information (MDPI), vol. 16, no. 6, p. 469, 2025.

[3] "LLM Agents for Education: Advances and Applications," Multi-institution survey, Findings of EMNLP 2025, 2025. arXiv:2503.11733.

[4] "Adaptive Learning Systems: Personalized Curriculum Design Using LLM-Powered Analytics," Adaptive learning research group, tech. rep., 2024. arXiv:2507.18949.

[5] Y. Zhang et al., "Mapping Student-AI Interaction Dynamics in Multi-Agent Learning Environments: Supporting Personalized Learning and Reducing Performance Gaps," Computers & Education, vol. 228, p. 105310, 2025.

[6] "KA-RAG: Integrating Knowledge Graphs and Agentic Retrieval-Augmented Generation for an Intelligent Educational Question-Answering Model," KA-RAG Research Group, Applied Sciences (MDPI), vol. 15, no. 23, p. 12547, 2025.

[7] "Agentic Retrieval-Augmented Generation: A Survey on Agentic RAG," Multi-institution survey, tech. rep., 2025. arXiv:2501.09136.

[8] R. Sapkota et al., "LangChain vs. LangGraph vs. LangSmith: Taxonomies of Agentic AI Toolchains for LLMOps," TechRxiv preprint, 2025.

[9] J. S. Park, J. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative Agents: Interactive Simulacra of Human Behavior," in Proceedings of UIST 2023, 2023. arXiv:2304.03442.

[10] S. Xu, X. Zhang, and L. Qin, "EduAgent: Generative Student Agents in Learning," tech. rep., 2024. arXiv:2404.07963.

[11] J. He-Yueya, N. D. Goodman, and E. Brunskill, "Evaluating and Optimizing Educational Content with Large Language Model Judgments," in Proceedings of EDM 2024, 2024, pp. 68–82.

[12] B. Hu, L. Zheng, J. Zhu, L. Ding, Y. Wang, and X. Gu, "Teaching Plan Generation and Evaluation with GPT-4: Unleashing the Potential of LLM in Instructional Design," IEEE Transactions on Learning Technologies, vol. 17, pp. 1445–1459, 2024.

[13] B. S. Bloom, "The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring," Educational Researcher, vol. 13, no. 6, pp. 4–16, 1984