iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 13 **Issue:** V **Month of publication:** May 2025

DOI: https://doi.org/10.22214/ijraset.2025.70592

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

Personalized E-Learning Assistant Using Knowledge Graph and LangGraph-Orchestrated Multi-Agent Framework

Chitresh Gyanani[1], Owais Salauddin Khan[2], Arjun Singh[3], Sameer Asthana[4]
[1, 2]*B. Tech CSE, Final Year, Greater Noida Institute of Technology, Greater Noida, India*
[3, 4]*Asst. Professor, Dept. of CSE, Greater Noida Institute of Technology, Greater Noida, India*

Abstract: ***The design of an e-learning system incorporating a personalized instructional approach, surpassing conventional pedagogical methodologies, is presented in this study. Many adaptive tutorial systems don't consider students' existing knowledge and past learning well enough. To fix this, the system uses knowledge graphs and multi-agent systems. The main focus is to build an academic system that can record students' likes learning styles, and grades in a clear way. A semantic graph model has been set up in Neo4j to store and manage this data. The system uses modular agents run by Lang Graph, to work with the knowledge graph. This allows it to change learning paths, teaching content, and make extra materials that fit each student. As students use the system more and take quizzes, each round of learning gets better to stay relevant and teach well. Adding Django makes the system more flexible and able to work with different front-end apps. This approach stands out because it's forward-thinking, can change, and aims to make learning better over time making it a strong choice compared to other options.***
Keywords: Knowledge Graph, Lang Graph, Lang Chain, Personalized Learning, Multi-Agent Systems, Neo4j, OpenAI, Gemini, Django REST

I. INTRODUCTION

A. Limitations of Current E-Learning Platforms

A lot of today's platforms dish out unchanging content in a set curriculum often missing the mark when it comes to meeting the wide range of what individual students need. Students come from all sorts of educational backgrounds and face their own unique challenges so there's a growing need for systems that can adjust to personal learning needs on the fly. The go-to model for most e-learning platforms sticks to a strict, step-by-step curriculum structure—often called a "track and field" model. Even when they throw in a bit of personalization, like letting you pick some classes or move at your own speed, the core teaching setup is still boxed in by a template the course makers came up with beforehand. This rigid setup is a big roadblock to dealing with how different learners think, what they already know how they learn, and what's going on in their educational world. Students differ a lot in how well they know tasks how fast they learn, what drives them how they like to get information (by seeing, hearing, or doing), and how their brains work. Teaching everyone the same way doesn't work for these differences. This can make students less interested, remember less, and not learn as well as they could. Also, many ways that claim to be personalized don't work that well. They focus on simple likes or what's popular instead of looking at how students act and learn.

B. Outlined approach

A personalized approach to learning shouldn't start with a set curriculum. It should change based on how each student interacts with the material adapting as they progress. While we've seen the rise of modular curriculum, they often still stick to a standard pace and order. This creates a one-size-fits-all path that goes against the basic ideas of deep personalization. These methods assume all students move forward at the same rate, which goes against the main goals of adaptive education. To tackle these shortcomings, experts suggest a custom e-learning helper that combines a knowledge graph to show student-specific info and an agent-based system to deliver personalized content and tests on the fly. Instead of creating a course with a fixed path, they've built a learning assistant that adjusts in real time based on each student's aims, actions, and grasp of the material. At the heart of this approach is a semantic knowledge graph that charts the student's academic journey showing mastered ideas tough spots, concept links, and recommended future topics. This setup doesn't just store data; it offers a living changing picture of the learner. Adding to this setup, Lang Graph manages a flexible multi-agent setup where each agent gets a certain job. Jobs include pinpointing what you don't know yet suggesting what you should learn next, making quizzes, and explaining stuff in a way that makes sense to your brain.

Big language models get used to make on-the-fly custom questions clear explanations, and recap bits of info that match up with how good the learner is and the way they like to learn. This setup aims to flip the usual way of doing things on its head making the course change shape to fit the learner, not the other way around. It doesn't matter if a student digs visual help, comparisons simple step-by-step guides, or hands-on tests, the teaching stuff gets made just for them. We're moving away from one-size-fits-all teaching to a smarter, answering-back tutoring kind of thing that's all about helping each person get ahead and get it.

II. LITERATURE SURVEY

Knowledge graphs are now super useful tools for showing what's what in specific areas of know-how connecting all the dots in neat data setups. When it comes to learning, these graphs are great at laying out tough subject ladders, stuff you got to know first, and how learners move on up, which is important for learning that changes to fit you. Bhuvanesh and the crew showed us how putting knowledge graphs to use in specific areas can be super helpful. They built this medical chatbot that drives home how useful graphs are when you're juggling live chats that need to be smart about the topic. Their research highlights how packing info that's specific to a field into graph layouts can level up the smarts of the answers for fancy needs. When you set up all that basic know-how and the fancy terms in a way that's like a tree, it gets way clearer how complex teaching systems are because they mix so many different things [1]. Aberbach and colleagues took a look at how graph-based setups can make learning more of a solo gig and tailor-made for each person. They showed that you can tweak how peeps learn on graph setups based on what they need, which is pretty slick for changing the game to fit different folks [2]. This smart setup changes up how tough or easy things are how fast you go through stuff, and how into it students get all by watching how different everyone is at learning. That way, everyone gets their own kind of learning party, no matter how good they are or what kind of brain busters they face. Zhao and the team worked on making the usual stiff learning systems livelier. They did this by mixing in knowledge structures that are pretty well-organized. Their research tackled the problem when set-in-stone lessons lead to gaps in what students learn. They suggested a way to keep school standards but still let students learn in ways that fit them best, thanks to these things called knowledge graphs [3]. The framework they came up with shows that it's possible to meet the big-picture educational needs and still be cool with what each student can do and likes.

The growth of agent-based systems and the mingling of big language models have pushed personalized learning tech forward. Stuff like LangChain and LangGraph have been super helpful in this rise. LangChain hooks up language models with other info, memory stuff, and cool tech tools. This setup lets people build tricky, one-way systems that take care of making prompts holding on to what's going on, and keeping outputs making sense during different learning times [4]. Folks say this kind of flow is super important when you're making teaching things that need learning over a bunch of sessions and pulling info from all sorts of places. LangGraph takes things up a notch by bringing in graph-based control flow, so you can build complicated workflows for agents [5]. It's different from just linking stuff one after the other because LangGraph can handle tasks side-by-side or in loops just like how teaching and learning go in all sorts of directions. Managing how things change from one agent to another makes it possible to set up smart feedback loops and make learning routes that change on the fly, which is super important when you want to get personal with the teaching. Major advancements happened in retrieval-augmented generation (RAG) setups. Zerhoudi and Granitzer came up with PersonaRAG where they put in agents all about the user right into the usual retrieval systems [6]. This setup gets better at responding and more personal by breaking up work based on special skills. They got agents handling their own biz—fixing up queries, picking documents, and making responses. Shi et al. crafted the "ERAGent" [7] setup which boosts the precision, speed, and tailored nature of outputs from language models. This setup shifts its search methods depending on how complex the query is and what the user has done before. It weighs the trade-offs between how well it does its job and the goodness of what it finds. This way of doing things shows why it's super important to have retrieval systems that can react in education. Getting info that fits the context and is super relevant matters a ton for doing well in learning. Thway et al. showcased how they rolled out a chatbot named Professor Leodar [8] at a uni using some nifty RAG setup. They wrote down how this tech helps students get more into their studies and keeps fake news at bay. They ran into a few pickles though, like keeping the convo on track making sure answers matched up, and figuring out how to measure if students were happy. What they found was pretty cool—these RAG thingies might be better than your standard brainy bots when it comes to teaching stuff right and making sure chats go smooth. Zhou and team dug into the problem of incomplete knowledge graphs in educational RAG situations [9]. They suggested ways to keep retrieval good when knowledge structures change or aren't built yet. They came up with methods to measure how complete graphs are and change how systems fetch information to stay useful in areas where knowledge keeps growing or isn't there yet.

Now when talking about teaching putting AI into the mix needs to respect the way lessons are planned. Niyozov and pals pointed out that language models being useful for learning isn't just about giving out facts [10]. It's also about whether they fit well into the teaching plans and help students learn.

They pushed for AI to go hand in hand with teaching methods aiming for students to get involved and not just soak up info. Bijanov and the team kept digging into the topic by checking out how mixing old-school teaching with the latest tech helps students learn on their own terms [11]. They point out that tech should give traditional learning a boost rather than take over. They flagged up how critical it is to have responses and checks that make learners think about their thinking and control their own study time. They're big on the idea of smart combos of instructor guidance and tech help as the best recipe for school success. Together, these findings lay a solid groundwork to create flexible learning tech. The use of "knowledge graphs" to show learning in a structured way, the teamwork of multiple agents to handle tasks that change, and the use of language models to make content that fits a person's needs show a good path forward. Taking what these early studies offer, the current effort tries to better digital lessons that change based on the student. New ways have been suggested that use graphs to customize the experience and keep students hooked.

III. PROPOSED SYSTEM

This suggested setup includes a custom e-learning helper with different parts connected to give learning that changes to fit each student. The heart of this setup is a Neo4j-based Knowledge Graph. It's not just a space for data but a smart setup that grabs important details about the student, the study stuff, and how they get along.

A. Knowledge Graph Design

The knowledge graph features important bits like nodes. These stand for big parts of the learning world (see Table 1 and Table 2). A Student node holds info on learners, stuff like their name, what they study, and how pumped they are to learn. There's more in the mix, including Subject nodes (think “Data Structures”, “Machine Learning”), Topic ones (like “Graph Algorithms”), and things tied to them such as Strengths, Challenges, Learning Goals, and Learning Style. When it comes to testing what they know, Quiz and Question nodes play a part. And for the stuff they use to learn, that’s where Resource nodes come into play.

These things all link up cause of ways they're related, like STUDIES, IS_GOOD_AT, STRUGGLES_WITH, and RECOMMENDED_BY. Having this kind of organized setup lets us keep an eye on how different ideas depend on each other. This helps us figure out what learners need and spot the right stuff to teach them next or help 'em out when they're stuck.

1) Knowledge Graph Schema

This schema outlines a personalized learning ecosystem where a student interacts with subjects, quizzes, learning preferences, and personalized resources.

TABLE I. Nodes in the Knowledge Graph

Node	Attributes	Description
Student	id, name, email, grade	Represents a learner.
Subject	id, name, description	Topics or academic areas studied by the student.
Strength	id, topic, description	Areas where the student excels.
Challenge	id, topic, description	Topics or habits the student struggles with.
Learning Style	id, choice	Preferred learning approach.
Resource	title, type, link	Study materials or tools recommended to a student.
Quiz	title, difficulty	A set of questions used to assess the student.
Question	text, topic	Individual quiz questions.
Answer Option	text, is_correct	Multiple-choice options for each question.
Attempt	timestamp	Represents a student’s quiz attempt.
Result	score, feedback	Outcome of an attempt.

TABLE II. Relationships in the Knowledge Graph

Relationship	***From***	***To***
STUDIES	Student	Subject
IS_GOOD_AT	Strength	Student
STRUGGLES_WITH	Challenge	Student
WANTS_TO_STUDY_BY	Student	Learning Style
RECOMMENDED	Resource	Student
ASSIGNED_TO	Quiz	Student
HAS_QUESTION	Question	Quiz
HAS_OPTION	Option	Quiz
HAS_RESULT	Attempt	Result
ATTEMPTED	Student	Question

B. LangGraph Orchestrated Agents

The orchestration of system behaviour is handled by a multi-agent architecture implemented using Lang Graph. Each agent operates autonomously with a defined functional scope, facilitating modular task execution across the platform (see Figure 1).

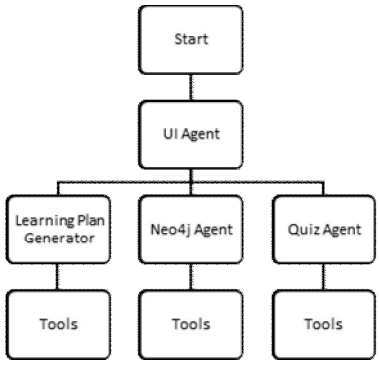


Fig. 1. Agent Architecture

The User Interaction Agent acts as the initial point of contact, interpreting user intent and delegating the request to the appropriate downstream agents. The Graph Agent interfaces with the Neo4j knowledge graph, enabling real-time data querying and updates. Subsequently, learner profile data, goals, and historical performance are retrieved and processed by the Learning Plan Agent, which generates tailored and adaptive learning pathways. The Quiz Agent is responsible for producing assessments aligned with both the learner's current knowledge and areas of difficulty.

C. LLM Integration for Personalization

The intelligence of these agents is further augmented through the integration of Large Language Models (LLMs) such as OpenAI's GPT and Google's Gemini. These models are utilized for various tasks, including the generation of quiz questions, the formulation of learning plans in natural language, and the evaluation of student-submitted answers.

Prompt templates are dynamically adjusted using the learner-specific data stored within the knowledge graph. This enables the production of highly contextual and personalized outputs, ensuring content relevance and learner engagement.

D. API Integration

System-wide coordination is achieved through the use of Django REST Framework APIs, which act as communication bridges between the frontend application and backend logic. These APIs are employed for capturing learner data, delivering personalized learning plans and assessments, recording quiz responses, and updating system states.

The architecture is designed with modularity in mind: each agent specializes in a particular task, while the knowledge graph provides a unified, centralized view of the learner's academic journey. This modular and data-centric design facilitates scalable personalization, ensuring both flexibility and maintainability in the e-learning platform.

IV. METHODOLOGY

A particular methodology has been crafted for the development and operation of this e-learning assistant which starts with Graph Schema Design (see Figure 2). In this initial step, a holistic model is created which documents all components of the learner's academic history, including their academic information (degree and subjects undertaken) and performance data (quiz history and grades). Furthermore, it also covers pedagogical information such as learning styles, goals, and even challenges. The structure of the knowledge graph gives a great guarantee of consistency with the data, enables meaningful queries and supports retrieval that is tailored in a highly personalized way. For instance, when trying to determine the topics with which a student struggles, the system can query the graph for all "STRUGGLES_WITH" relationships to determine coverage per student.

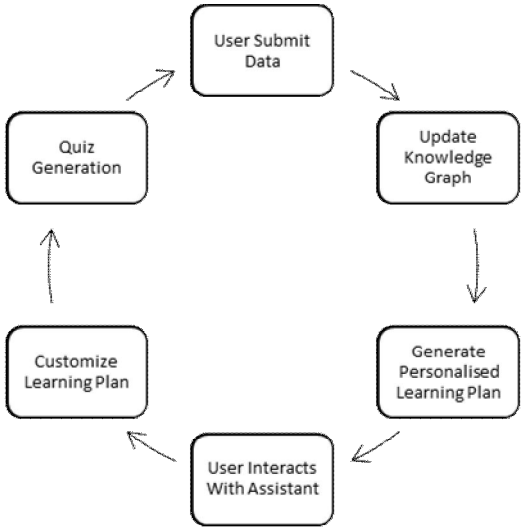


Fig. 2.System Workflow

Fig. 3.

A. Onboarding and Data Ingestion

Students start the process with Form-Based Data Ingestion by completing an onboarding form where they provide a wealth of information for data collection, including academic history, learning subjects of interest as well as challenges, resources preferred, and motivating factors. This detailed information undergoes thorough shaping to guarantee that it is in a compatible JSON format with the knowledge graph prior to being uploaded into Neo4j database.

B. Agent Configuration using langGraph

Next follows Agent Configuration with Lang Graph. Every agent has the tools and knowledge required to execute their functions. Control flow is managed with Lang Graph, which directs learner requests to the appropriate agent. This framework also has task history which logs progress and helps with personalizing the learning experience over time. Routing requests is automatic. For instance, a request that states: "I want to improve Graph Algorithms" would be routed to the Learning Plan Agent who will turn to the knowledge graph and determine what materials are available and devise a study plan tailored to the learner.

C. Quiz Generation Pipeline

The system then creates a dynamic Quiz Generation Pipeline. Upon request, the Quiz Agent accesses the knowledge graph to cross-reference the learner's active challenges. The agent issues instructions to the LLM using prompt templates, asking for partially filled in multiple-choice questions for each topic (generally 3-5 questions) including reasons for the correct responses and feedback, and level of difficulty. After taking the quiz, the student's submitted answers are checked and the outcomes are updated in the knowledge graph.

D. Learning Plan Recommendation

To assist a student's independent learning, the Learning Plan Recommendation process is triggered first. Using Learning Analytics, an individualized teaching plan is created by the Learning Plan Agent, who orders the topics according to their degree of difficulty, priority, and other interdependencies. Then, an LLM is applied to transform the student's learning plan into an explanatory text that preserves the outline and structures but places emphasis on natural language. The plans can include readings, videos, solving exercises, and various milestones that allow qualitative self-assessment of progress.

E. Feedback Loop and Graph Updates

Most importantly, there is the critical Feedback Loop and Graph Update mechanism. After every evaluation, the system updates the learner's knowledge graph, specifically the "Attempt" and "Result" nodes. To fine-tune the estimated topic difficulty rating, the goals set for each topic, and better the future recommendations, these estimates are recalibrated in line with the quiz scoring. This cycle guarantees that the relevant, precise, and meaningful learning framework is appropriately set and continues adapting to the learner's changing requirements. When a student does not perform well on a topic, the system is able to alter the plan automatically, which aids in adjusting to these inadequacies (see Figure 3).

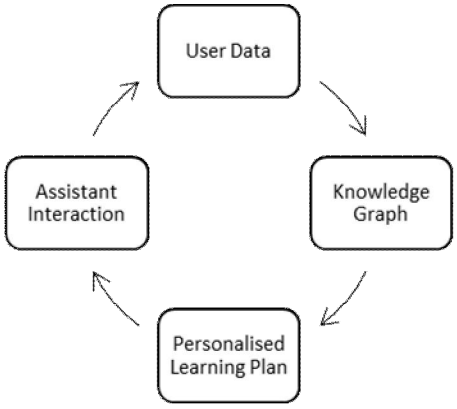


Fig. 4. Personalized Learning Loop

In summary, the integrated approaches of goal-driven alignment and constant adaptation create a strong learning ecosystem that harnesses knowledge graphs with autonomous intelligent agents, resulting in hyper-personalized learning tailored to every student's path.

V. RESULTS

This section presents a comparative analysis of the performance of the proposed personalized e-learning assistant. The evaluation has been conducted with a focus on backend performance, personalized content alignment, and relative improvements over traditional learning platforms.

A. Evaluating Quiz Relevance to User Context

Rather than only measuring quiz accuracy, this system evaluates how well each generated quiz aligns with the student's personalized graph profile. Relevance scores are calculated by combining topic alignment, difficulty matching, goal relevance, and resource preference. The results of this evaluation are presented in Table 3 and visually depicted in Figure 4.

TABLE III. Backend Comparison Using Quiz Relevance to Context

Backend Configuration	*Quiz Relevance to User Context (%)*
ChatGPT + Graph + Agents (Proposed)	*94.2%*
Gemini + Graph + Agents	*89.6%*
RAG without Graph Context	*75.4%*
Standard LLM without Personalization	*61.2%*

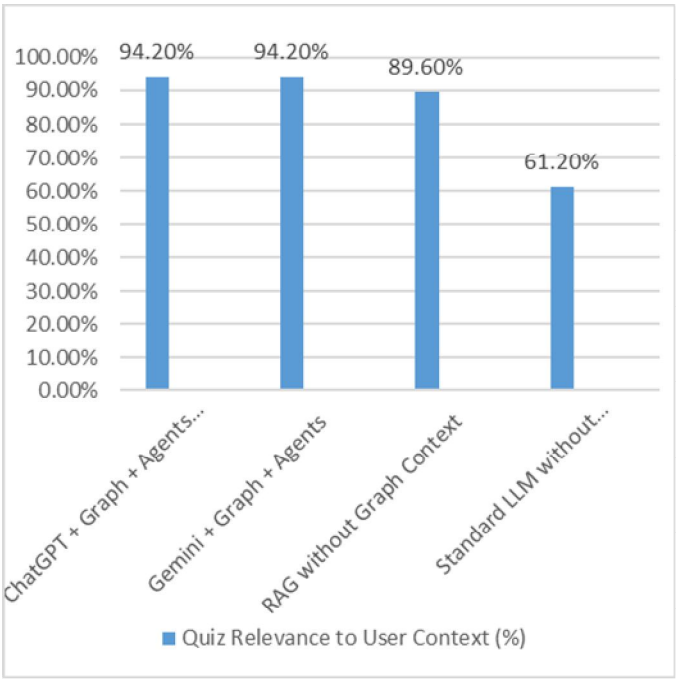


Fig. 5. Quiz Relevance to User Context accrosss Backends

B. Backend Performance Metrics

To understand how each LLM backend performed under the same agent and graph setup, two metrics were recorded: Relevancy Score and Personalization Score. These were quantitatively measured over the same batch of system interactions with a standard learner profile. The results are shown in Table 4 and visualized in Figure 5.

TABLE IV. Relevancy and Personalization Scores Across Backends

Backend Configuration	Relevancy Score (/10)	Personalization Score (/10)
ChatGPT + Graph + Agents (Proposed)	9.4	9.1
Gemini + Graph + Agents	8.8	8.5
RAG without Graph Context	7.6	6.9
Standard LLM without Personalization	6.1	4.3

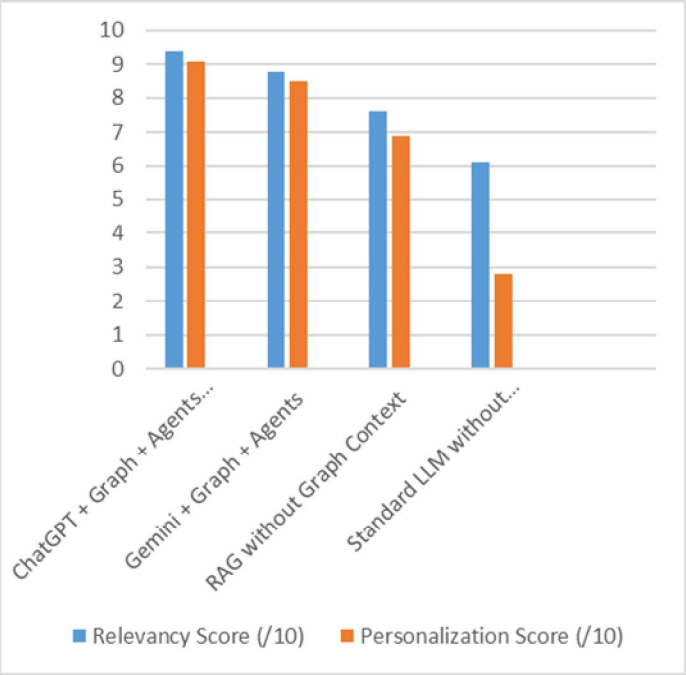


Fig. 6. Visual Comparison of Relevancy and Personalization Scores

C. Comparison with Generic Learning Platforms

To gauge the real-world value of this system, its capabilities were compared against popular e-learning platforms that offer static curriculum paths. The personalized assistant was consistently able to:

1) Provide learning plans tailored to a user's background, goals, and motivational inputs.
2) Trigger intelligent quiz generation based on learner struggles and strengths.
3) Adaptively modify recommendations after every quiz attempt using a knowledge graph feedback loop.
4) Maintain agent memory and task routing, making the interaction feel coherent and ongoing.

Generic platforms, while rich in high-quality content, often lack:

1) Contextual understanding of the learner's weak areas.
2) Dynamic adaptation to performance over time.
3) Agent-style interactivity where one task informs another.

This positions the proposed assistant as a hyper-personalized alternative that not only delivers content but also reasons about learner progress continuously.

VI. CONCLUSION

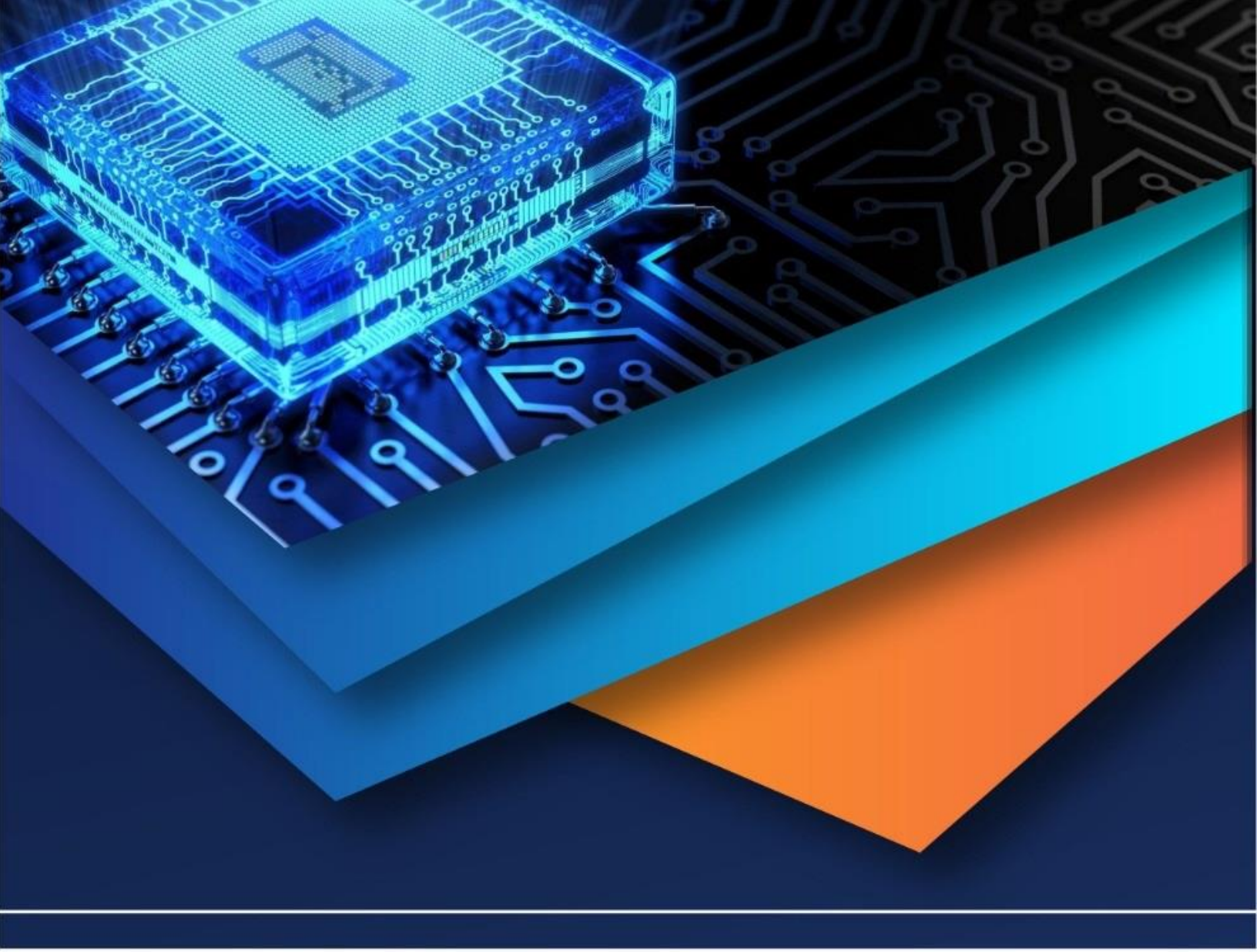
The presented system demonstrates the efficiency of combining structured knowledge representation through a semantic knowledge graph with modular agent-based orchestration to achieve deep personalization in e-learning environments. The proposed approach extends beyond traditional content delivery by continuously adapting learning pathways based on student profiles, preferences, performance, and contextual feedback. By enabling dynamic generation and refinement of learning resources, the system addresses key limitations found in conventional educational platforms, particularly in terms of engagement, relevance, and learner autonomy. The incorporation of participatory and human-centred design principles further ensures that the framework remains responsive and inclusive, reversing the one-size-fits-all paradigm commonly observed in legacy systems.

Future enhancements are envisioned to include interoperability with external educational platforms such as MOOCs and content-sharing networks, the development of real-time progress tracking interfaces for mentors and guardians, and the inclusion of behavioral and affective signals for enriched learner modeling. Through these developments, a step forward is taken toward realizing intelligent tutoring systems that are seamlessly personalized, minimally intrusive, and capable of supporting diverse learning needs at scale.

REFERENCES

[1] R. Kumar and S. Patel, "LangGraph-Orchestrated Multi-Agent Systems for Personalized Course Recommendations," unpublished manuscript.
[2] S. Bhuvanesh et al., "Knowledge Graph Based Medical Chatbot," in Proc. IEEE Global Conference for Advancement in Technology (GCAT), 2023.
[3] H. Aberbach et al., "A Personalized Learning Approach based on Learning Speed," Journal of Computer Science, vol. 17, no. 2, pp. 112–120, 2021.
[4] S. Zhao et al., "Advancements and Transformations in Educational System Reforms," SHS Web of Conferences, vol. 165, 2024.
[5] LangChain Documentation. [Online]. Available: https://docs.langchain.com
[6] LangGraph GitHub Repository. [Online]. Available: https://github.com/langchain-ai/langgraph
[7] S. Zerhoudi and M. Granitzer, "PersonaRAG: Enhancing Retrieval-Augmented Generation Systems with User-Centric Agents," arXiv preprint arXiv:2407.09394, 2024.
[8] Y. Shi et al., "ERAGent: Enhancing Retrieval-Augmented Language Models with Improved Accuracy, Efficiency, and Personalization," arXiv preprint arXiv:2405.06683, 2024.
[9] M. Thway et al., "Battling Botpoop: RAG Chatbot for Higher Education," arXiv preprint arXiv:2406.07796, 2024.
[10] D. Zhou et al., "Evaluating Knowledge Graph Based Retrieval-Augmented Generation Methods under Knowledge Incompleteness," arXiv preprint arXiv:2504.05163, 2024.
[11] N. Niyozov et al., "The Pedagogical Principles and Effectiveness of Utilizing ChatGPT for Language Learning," E3S Web of Conferences, vol. 385, 2023.
[12] A. Bijanov et al., "Blended Learning and Personalized Feedback in Modern Education," Proc. RSES Conference, 2023.
[13] H. Wan, B. Che, H. Luo, and X. Luo, "Learning Path Recommendation Based on Knowledge Tracing and Reinforcement Learning," unpublished manuscript.
[14] Y. Zhang et al., "Dynamic Knowledge Graph Construction for Adaptive E-Learning Systems," unpublished manuscript.
[15] A. Joshi et al., "Scalable Multi-Agent Frameworks for Real-Time Learning Analytics," unpublished manuscript.

[16] P. Gupta et al., "TAGCN-Based Knowledge Graphs for Learner Preference Modeling," unpublished manuscript.
[17] X. Wang et al., "Federated Knowledge Graphs for Privacy-Preserving E-Learning," unpublished manuscript.
[18] Z. Li et al., "CrewAI-Driven Task Decomposition for Collaborative Learning Agents," unpublished manuscript.
[19] M. Singh et al., "Quantum-Inspired Optimization for Knowledge Graph-Based Tutoring Systems," unpublished manuscript.
[20] D. Fernandez et al., "Streaming Knowledge Graphs for Real-Time Learner Feedback," unpublished manuscript.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY