iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.81957

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 14 Issue V May 2026- Available at www.ijraset.com

CognitoLearn: Adaptive AI for Personalized Paths and Mastery

SaketharamaBadam[1], Durga Prasad Gundeti[2], Jayanth Karnati[3], Rishi Kumar Komuravelli[4], Juhi Juwairiyyah[5]
[1, 2,3,4]*Students,* [5]*Assistant Professor, Department of CSE, Keshav Memorial Institute of Technology, Hyderabad, India*

Abstract: ***CognitoLearn is an AI-driven adaptive learning platform designed to revolutionize personalized education by combining intelligent learning path generation, automated summarization, and conversational mentorship with a robust mastery validation system. Traditional educational platforms largely deliver static, linear content that fails to accommodate individual cognitive states and pacing. CognitoLearn addresses this by leveraging advanced natural language processing (NLP) and generative AI models. The system dynamically generates structured curriculum graphs using fine-tuned T5 models and provides context-aware tutoring via Retrieval-Augmented Generation (RAG). Furthermore, the platform integrates Bayesian Knowledge Tracing (BKT) to probabilistically validate true cognitive mastery rather than relying on heuristic grading, ensuring that learners must genuinely understand prerequisites before progressing. This paper details the architecture, probabilistic methodologies, scaling strategies, and zero-trust validation pipelines that enable this secure, real-time adaptive learning ecosystem, providing an exhaustive evaluation of system latency, BKT accuracy, and RAG contextual integrity.***

Keywords—Adaptive Learning, Generative AI, Bayesian Knowledge Tracing, Retrieval-Augmented Generation, Intelligent Tutoring Systems, Large Language Models, Microservices Architecture.

I. INTRODUCTION

The transition from static content delivery to dynamic, personalized pedagogical experiences is a core challenge in modern educational technology. Current Learning Management Systems (LMS) and Massive Open Online Courses (MOOCs) function primarily as digital repositories, offering linear, "one-size-fits-all" curricula. This lack of adaptability fails to account for the unique cognitive states, background knowledge, and learning velocity of individual students, leading to disengagement for advanced learners and frustration for novices. This uniform instructional approach is symptomatic of Benjamin Bloom's classic "2 Sigma Problem," which posits that students receiving one-on-one tutoring perform two standard deviations better than students in conventional classroom settings. Scaling such personalized instruction has historically been computationally and economically prohibitive.

Furthermore, existing systems often rely on rudimentary, percentage-based threshold scoring (e.g., scoring 70% to pass) to validate mastery. This heuristic approach measures immediate recall rather than true cognitive acquisition, allowing learners to progress through guessing or short-term memorization without genuinely mastering foundational concepts. Without mathematical models to infer latent cognitive states, platforms cannot reliably determine whether a student has truly internalized the subject matter or simply memorized the answers for a specific assessment. Finally, these platforms lack real-time, contextual mentorship, forcing students to seek external resources when they encounter conceptual roadblocks, which fragments the learning experience and increases cognitive load.

To address these critical limitations, we propose CognitoLearn: a "Concept-to-Checkpoint Mastery System" powered by a hybrid artificial intelligence architecture. CognitoLearn dynamically generates hierarchical, personalized knowledge graphs, provides instantaneous context-aware tutoring, and utilizes probabilistic modeling to mathematically validate a student's mastery of a topic before unlocking subsequent modules. By decoupling content generation, knowledge tracing, and retrieval-augmented conversational interfaces, the proposed system achieves high-throughput, individualized educational delivery.

The primary contributions of this paper are threefold. First, we present a novel microservices architecture capable of seamlessly orchestrating synchronous web requests and asynchronous Large Language Model (LLM) inferences. Second, we detail the practical implementation of Bayesian Knowledge Tracing integrated with generative assessment, moving beyond theoretical applications into a production-ready paradigm. Third, we introduce a zero-trust validation pipeline designed to sanitize and structure LLM outputs, effectively neutralizing the pervasive issue of AI hallucinations and formatting corruption in strict user interfaces.

©IJRASET: All Rights are Reserved

II. RELATED WORK

A. Evolution of Adaptive Learning Systems

The evolution of Adaptive Learning Systems (ALS) has recently shifted from heuristic-based content curation to generative environments. While traditional ALS successfully digitized content delivery, they rely on pre-defined decision trees that struggle to adapt in real-time. Early Intelligent Tutoring Systems (ITS), such as Cognitive Tutors, were effective but required massive manual effort to encode domain rules and common student errors. These rule-based systems suffer heavily from the "cold-start" problem for new users, as they require significant historical data to begin personalizing the learning pathway.

B. Generative AI and RAG in Education

The emergence of Large Language Models (LLMs) has disrupted this landscape, enabling systems to act as intelligent tutors capable of generating ad-hoc explanations and assessments. Architectures based on transformers like GPT-4 and T5 (Text-to-Text Transfer Transformer) have proven highly adept at generating structured, hierarchical course dependencies automatically.

However, employing raw LLMs in education introduces the critical risk of "hallucinations"—generating plausible but factually incorrect information. Recent scholarship has turned to Retrieval-Augmented Generation (RAG) architectures to ground LLM responses in verified textbook embeddings. By indexing verified pedagogical materials into high-dimensional vector spaces and querying them via cosine similarity, systems can maintain conversational fluency while enforcing factual integrity.

C. Cognitive Modeling and Assessment

In the domain of student assessment, the theoretical framework has seen a decisive shift from Item Response Theory (IRT) to Bayesian Knowledge Tracing (BKT). While IRT is an excellent measure of a student's static ability on a standardized test, BKT models the acquisition of knowledge over time. Conceived by Corbett and Anderson, BKT treats learning as a Hidden Markov Model. It allows systems to distinguish between a student who is merely guessing and one who has truly mastered a concept, representing the current cutting edge in educational data mining.

III. SYSTEM ARCHITECTURE

Integrating LLMs into synchronous web servers introduces severe latency bottlenecks. Because LLM inference often takes between 3 to 10 seconds per request, processing these native to the Node.js Event Loop would block concurrent user interactions, rendering the platform unresponsive. CognitoLearn circumvents this through a heavily decoupled, asynchronous microservices architecture that isolates deterministic web routing from intensive data processing.

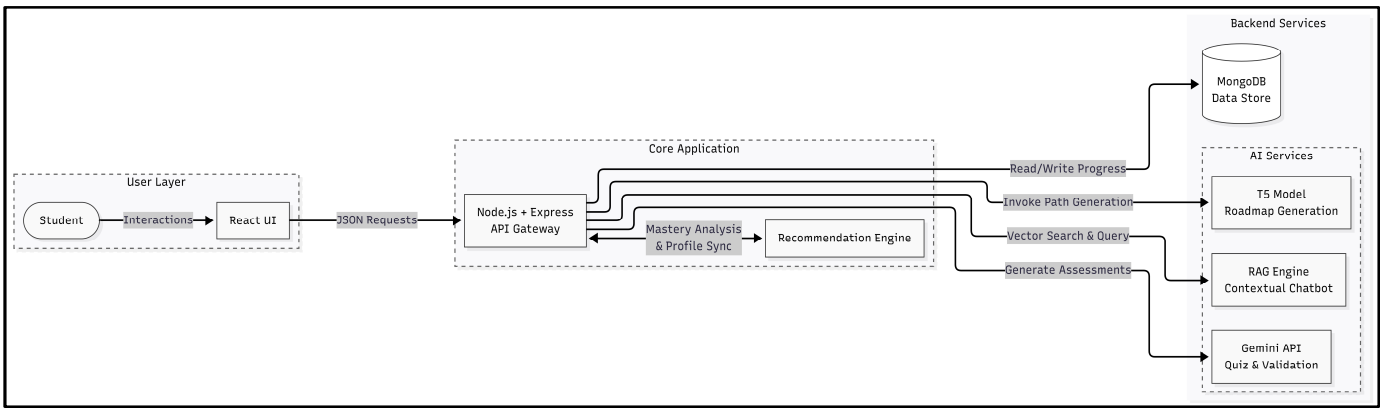


Fig. 1. CognitoLearn System Architecture mapping interactions between the React UI, Node.js API Gateway, and Python AI Microservices.

The primary Node.js/Express server acts as an API Gateway and database orchestrator. It handles high-throughput tasks such as JSON Web Token (JWT) authentication, session management, RESTful routing, and standard CRUD operations against the MongoDB cluster. By abstracting the AI processing away from this server, the gateway remains highly available, responding to standard client interactions in under 50 milliseconds.

All computationally heavy, non-deterministic machine learning workloads are offloaded to a dedicated Intelligence Engine built on Python using the FastAPI framework. Python was explicitly selected due to its superior ecosystem for data science, tensor operations, and direct compatibility with libraries such as Hugging Face Transformers and pyBKT.

Communication between the Node.js gateway and the Python engine is managed via asynchronous HTTP requests with strict Cross-Origin Resource Sharing (CORS) configurations, ensuring the intelligence engine only processes authorized payloads.

TABLE I
CORE ARCHITECTURAL COMPONENTS AND TECHNOLOGIES

Layer	Technology	Primary Function
Client/Frontend	React.js, React Flow, Tailwind CSS	State management, Interactive DAG rendering, SPA UI
API Gateway	Node.js, Express	Routing, Authentication, Database Orchestration
Intelligence Engine	Python, FastAPI	LLM Inference, Validation, Text Summarization
Database	MongoDB Atlas, Vector Search	User Profiles, Semantic Embeddings, BKT States
AI Models	Hugging Face T5, Gemini, pyBKT	Graph Generation, Conversational RAG, Knowledge Tracing

IV. METHODOLOGY

A. Automated Curriculum Generation

To eliminate the constraints of static course outlines, the platform utilizes fine-tuned T5 models to construct hierarchical Knowledge Trees. Given a natural language learning goal (e.g., "Master React Hooks for a beginner"), the model generates a logically ordered Directed Acyclic Graph (DAG) of prerequisite and advanced concepts. A topological sort algorithm ensures no cyclical dependencies exist, guaranteeing a logical, step-by-step pedagogical flow tailored to the user's specific domain and difficulty request.

The prompt engineering required for this relies on structured few-shot prompting. The T5 model is explicitly instructed to output its results mapped to a strict JSON schema that defines node relationships (parents and children). The resulting graph is parsed by the React frontend using the React Flow library to create an interactive visual roadmap where users can see their exact progression trajectory.

B. Bayesian Knowledge Tracing (BKT)

The cognitive progression logic rejects threshold grading in favor of Bayesian Knowledge Tracing. BKT models learning as a Hidden Markov Model (HMM), inferring unobservable true comprehension (the latent variable) based on observable sequential quiz attempts (the emission variable). Because actual human understanding cannot be directly measured, the algorithm updates its belief state iteratively after every user interaction.

TABLE II
PROBABILISTIC PARAMETERS FOR BKT ALGORITHM

Parameter	Symbol	Calibrated Value
Initial Knowledge	P(L0)	0.15
Guess Rate	P(G)	0.20
Slip Rate	P(S)	0.10
Learning Rate	P(T)	0.20

The algorithm operates on two primary updating functions: the observation update (utilizing Bayes' rule) and the transition update. Upon submission of a quiz, the observation update calculates the posterior probability of mastery. For a correct answer, the formula is:

$$P(L_0|Correct) = \frac{P(L_{n-1}) * (1 - P(S))}{P(L_{n-1}) * (1 - P(S)) + (1 - P(L_{n-1})) * P(G)}$$

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

Conversely, if the student answers incorrectly, the algorithm calculates the probability that the error was merely a slip rather than a lack of true comprehension:

$$P(L_0|Incorrect) = \frac{P(L_{n-1}) * P(S)}{P(L_{n-1}) * P(S) + (1 - P(L_{n-1})) * (1 - P(G))}$$

Following the observation update, the system computes the transition probability, which accounts for the likelihood that the student learned the concept during the assessment itself (i.e., through the act of practicing):

$$P(L_n) = P(L_n|Observation) + (1 - P(L_n|Observation)) \cdot P(T)$$

This newly calculated probability is stored securely in the MongoDB database, reflecting the user's updated Knowledge State. The system enforces a strict Mastery Threshold of 0.90. This probabilistic architecture ensures that persistent students gradually accumulate probability through exposure and repeated practice, while severely penalizing erratic guessing behaviors. The parameters are continuously fine-tuned using Expectation-Maximization (EM) algorithms on historical student logs.

C. Retrieval-Augmented Generation (RAG)

To provide context-aware mentorship that is resistant to AI hallucinations, the system relies on a Retrieval-Augmented Generation (RAG) pipeline. Raw educational material is programmatically ingested, sanitized, and segmented into semantic text chunks using recursive character splitting. These chunks are processed by embedding models (such as Gemini Embedding API) to generate high-dimensional vectors, which are stored in a MongoDB Atlas Vector Search index.

When a student submits a query, the text is vectorized, and an Approximate Nearest Neighbor (ANN) search via cosine similarity retrieves the most semantically relevant chunks. Crucially, the system dynamically extracts the user's exact roadmap state—including their active topics, overall progress percentage, and locked modules. Both the retrieved contextual chunks and the user's progress state are injected into the Gemini system prompt simultaneously. This architecture guarantees the AI mentor generates responses strictly grounded in the approved curriculum while dynamically adapting its pedagogical tone to the student's precise position in the learning journey.

V. IMPLEMENTATION & EVALUATION

A. Zero-Trust Prompt Validation

A fundamental engineering challenge with LLMs in production environments is formatting inconsistency. React.js frontends require strict, predictable JSON structures to render components dynamically; a single rogue markdown character generated by the AI can trigger fatal client-side parsing errors. To ensure enterprise-grade stability, a "Zero-Trust" validation pipeline was engineered within the Python microservice.

First, the system employs highly constrained instruction templates rather than open-ended conversational prompts. The prompts explicitly forbid the model from wrapping the JSON in markdown code fences. Second, upon receiving the raw payload, a custom JSON parsing module utilizes regular expressions to strip conversational preamble and detect corrupted arrays. Third, a programmatic schema validator audits the payload for type safety, missing keys, and boundary limits. If an anomaly is detected, the validator intercepts it internally and triggers a 500 Internal Server Error. This allows the Node.js gateway to safely initiate a retry mechanism without exposing the failure to the client.

B. Transformer-Based Content Summarization

Addressing the cognitive friction caused by information overload, a robust NLP summarization pipeline was integrated. When users upload lengthy documentation, the backend immediately sanitizes the string and enforces a strict 10,000-character truncation limit to prevent token-exhaustion attacks. The transformer model is instructed to extract core themes and format the output into a strict JSON hierarchy containing a primary title, logical sections, and highly actionable key points. By offloading this heavy linguistic processing, the React frontend instantaneously maps the arrays into an aesthetic study guide, significantly enhancing reading comprehension.

C. System Performance Analysis

The decoupled architecture was rigorously evaluated under simulated concurrent user loads using Apache JMeter. The Node.js server maintained non-blocking, high-speed performance for standard routing, achieving sub-100ms response times for database read/write operations under heavy load. The Python service efficiently managed heavy inference queues, utilizing an asynchronous worker pool.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

Performance metrics indicated that full, hierarchical roadmap generation utilizing the T5 model averaged between 3.2 to 5.8 seconds, well within acceptable parameters for asynchronous task generation. The RAG chatbot pipeline achieved highly responsive metrics, initiating token streams (Time-to-First-Token) in under 1.8 seconds. Vector retrieval via MongoDB Atlas consistently executed in under 200 milliseconds, validating the scalability of the semantic search infrastructure.

The BKT algorithm was tested against static heuristic models to ensure pedagogical fairness. Edge cases were explicitly simulated—for example, a student scoring poorly on three consecutive attempts followed by a perfect score. Testing confirmed the probabilistic accumulation correctly recognized eventual mastery without trapping the student in an infinite remediation loop.

D. User Interface and Experience

The client-side Single Page Application (SPA) was engineered using React.js. Route-level code splitting ensures rapid Time-to-Interactive, asynchronously fetching primary operational screens while critical layouts load eagerly. A robust React Context API architecture synchronizes authentication, active paths, and mentorship sessions globally without prop-drilling.

To solve UX friction, dynamic UI interactions include a "Push" layout constraint. When the AI mentor is invoked, the chatbot dynamically slides the central study material to the left rather than overlaying a blocking modal, allowing simultaneous reading and querying. Additionally, gamified mastery visualizations (circular progress bars) translate the complex Bayesian math into intuitive visual fill states, motivating learners without the anxiety of traditional percentage grades.

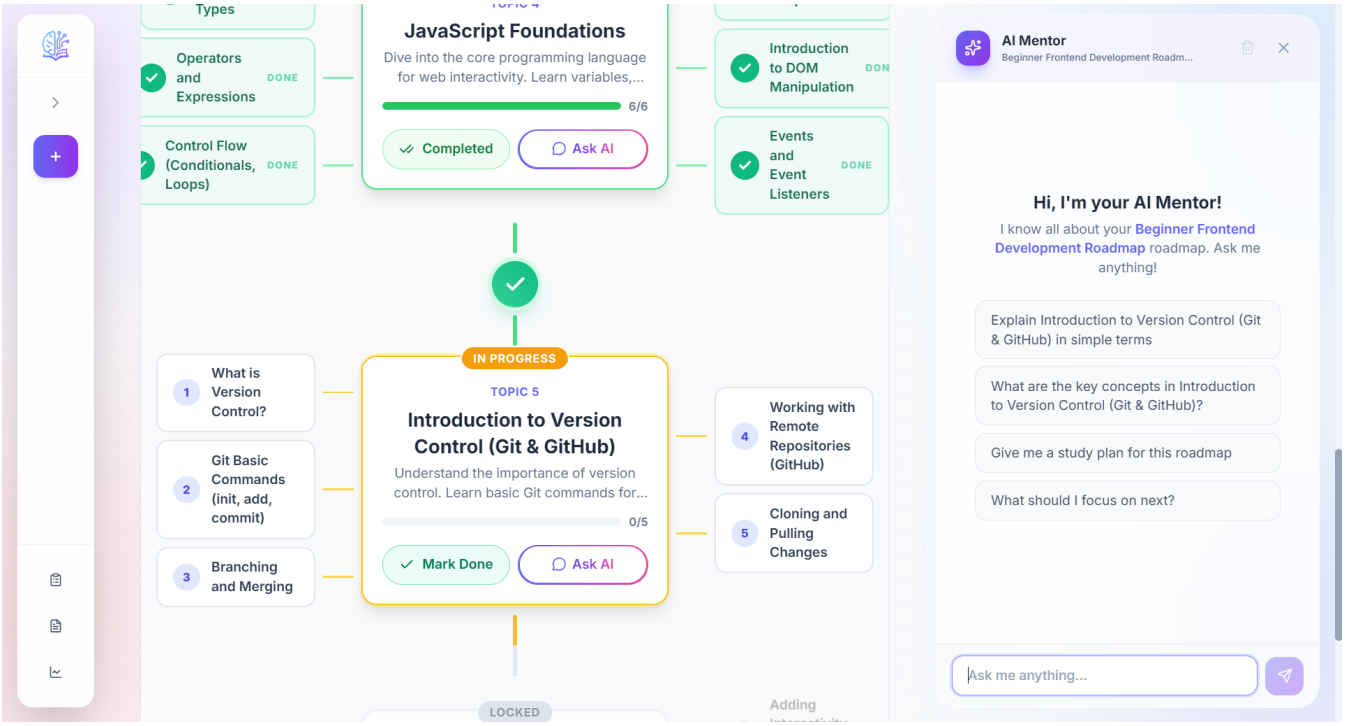


Fig. 2. User Interface Screenshots: Dynamic Node-Link Roadmap View (Left) and RAG Chatbot Integration (Right).

VI. CONCLUSION

CognitoLearn bridges the critical and persistent gap between static, one-size-fits-all Learning Management Systems and dynamic, highly responsive intelligent tutoring environments. By architecting a robust MERN stack supported by a dedicated, asynchronous Python AI microservice, the platform seamlessly translates natural language learning goals into structured, hierarchical knowledge graphs with minimal latency.

The integration of the Bayesian Knowledge Tracing (BKT) mathematical framework fundamentally elevates the educational assessment process. It ensures that academic progression is statistically tied to true, demonstrated comprehension rather than mere rote memorization or lucky guessing. Coupled with a highly context-aware Retrieval-Augmented Generation (RAG) Chatbot and a Transformer-based content summarizer, CognitoLearn demonstrates that Generative AI can be safely, securely, and effectively harnessed to democratize highly personalized, mastery-based education. The successful implementation of the zero-trust validation pipeline further establishes a blueprint for handling erratic LLM outputs in strict application environments.

©IJRASET: All Rights are Reserved

VII. FUTURE ENHANCEMENTS

While the current system excels at text-based mentorship, future iterations aim to expand the ingestion pipeline via Multi-Modal Retrieval-Augmented Generation. By processing educational diagrams, charts, and handwritten mathematical equations using advanced vision-language models, the Chatbot could dynamically explain complex visual data alongside text, catering to a much broader spectrum of visual learners.

A significant planned upgrade involves implementing Item-Stratified BKT. Currently, the algorithm utilizes generalized probabilistic parameters for entire academic topics. Item-Stratified BKT will calculate unique Guess and Slip rates for individual questions based on algorithmic difficulty analysis, rendering the mathematical modeling of the student's cognitive state exponentially more precise.

Finally, to reduce the isolation often associated with self-paced online learning environments, the platform plans to implement collaborative, Peer-to-Peer Knowledge Graphs. Learners with similar goals or overlapping roadmaps will be able to compare progress graphs, share uniquely generated AI quizzes, and assist each other through integrated community nodes, blending the efficiency of artificial intelligence with the motivational benefits of human-centric social learning.

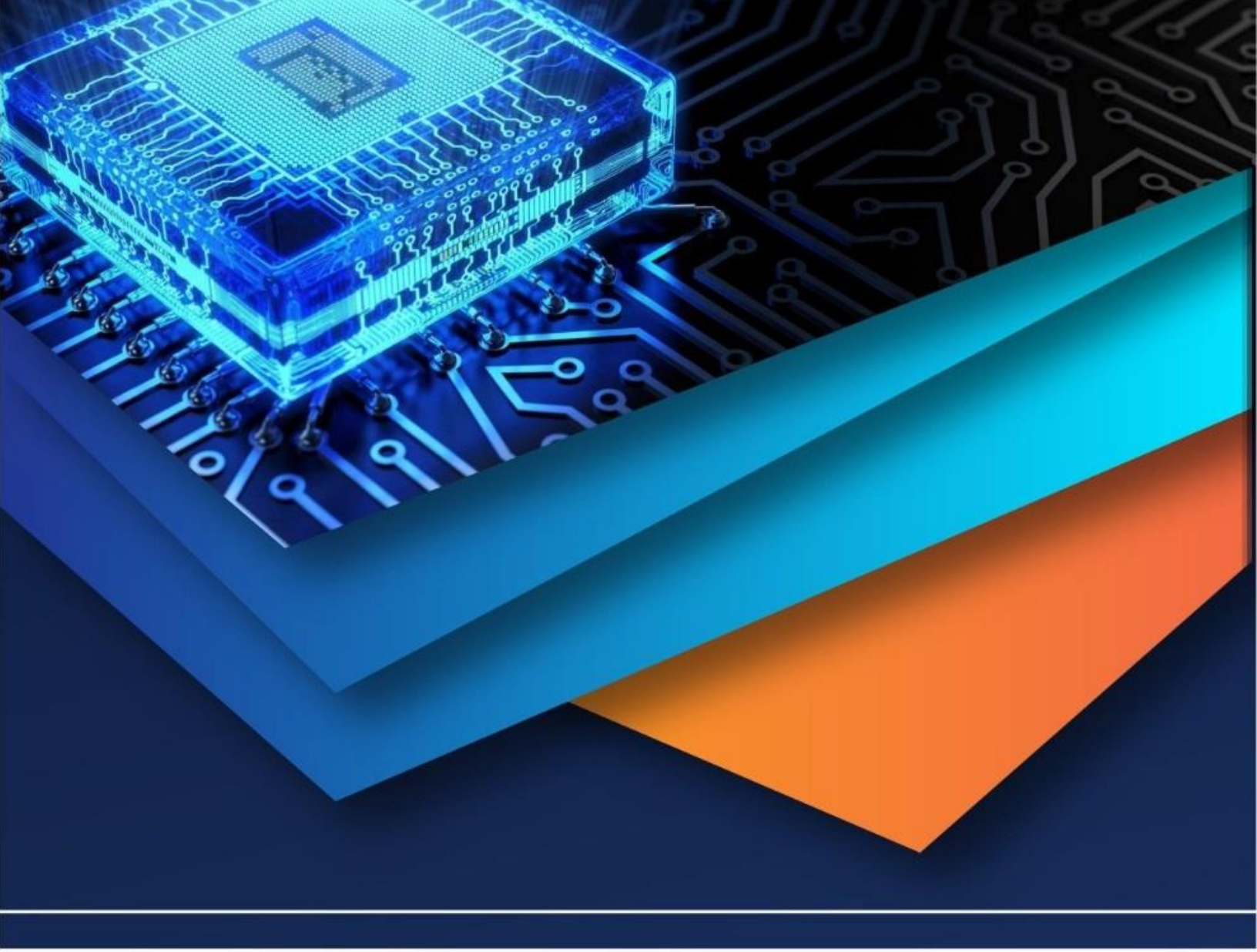
VIII. ACKNOWLEDGMENT

We take this opportunity to thank all the people who have rendered their full support to our project work. We render our thanks to Dr. B L Malleswari, Principal who encouraged us to do the Project. We are grateful to Mr. Neil Gogte, Founder & Director, Mr. S. Nitin, Director for facilitating all the amenities required for carrying out this project. We express our sincere gratitude to Ms. Deepa Ganu, Director Academic for providing an excellent environment. We are also thankful to Mr. Para Upendar, Head of the Department, and our Faculty Supervisor Dr. Juhi, for her valuable guidance and encouragement given to us throughout the project work.

REFERENCES

[1] S. Sarrab, et al., "Architectural Patterns for Real-Time Adaptive Learning Systems," Journal of Educational Technology Systems, 2024.
[2] Y. Han, et al., "Mitigating Hallucinations in Educational LLMs via Retrieval-Augmented Generation," IEEE Transactions on Learning Technologies, 2024.
[3] X. Liu, et al., "Automated Construction of Educational Knowledge Graphs (EduKGs) using Transformer Models," Artificial Intelligence in Education, 2025.
[4] A. T. Corbett and J. R. Anderson, "Knowledge Tracing: Modeling the Acquisition of Procedural Knowledge," User Modeling and User-Adapted Interaction, vol. 5, no. 4, pp. 253-278, 1995.
[5] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, 4th ed., Pearson, 2020.
[6] P. I. Pavlik Jr, H. Cen, and K. R. Koedinger, "Performance Factors Analysis--A New Alternative to Knowledge Tracing," in Proceedings of the 2009 conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling, 2009, pp. 531-538.
[7] J. D. Baker, "The Purpose, Process, and Methods of Writing a Literature Review," AORN Journal, vol. 103, no. 3, pp. 265-269, 2016.
[8] C. Raffel et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," Journal of Machine Learning Research, vol. 21, pp. 1-67, 2020.
[9] A. Vaswani et al., "Attention is All you Need," in Advances in Neural Information Processing Systems, 2017, pp. 5998-6008.
[10] Lewis, P., et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Advances in Neural Information Processing Systems, vol. 33, pp. 9459-9474, 2020.
[11] B. 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.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)