Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

EduNex: A Mobile RAG Architecture for Hallucination Free Academic Assistance and Automated Assessment in Technical Curriculum

Manoj S. Sonpethkar
International Center Of Excellence In Engineering and Management(ICEEM),
Department Of Computer Science and Engg, Chh Sambhajinagr 431001, India.

ABSTRACT

The integration of Large Language Models into digital learning platforms offers expansive conversational capabilities, yet traditional models frequently suffer from contextual hallucinations and fail to adhere to strict technical syllabus.

This paper presents *EduNex*, an artificial intelligence driven mobile application engineered to provide hallucination free academic assistance tailored to specific diploma and engineering curriculum.

Utilizing a Retrieval Augmented Generation architecture, the proposed system bypasses generalized global parameters by securely fetching and chunking verified textual and graphical assets from a cloud hosted repository. This ensures that all generated tutoring responses remain strictly bounded to official course materials. Additionally, the mobile framework integrates an asynchronous examination simulator capable of dynamically generating non repeating multiple choice questions, offering instant pedagogical feedback without interrupting the user interface thread. The deployed architecture demonstrates high efficiency state management, seamless multimodal diagram integration, and a closed loop learning environment that significantly enhances assessment accuracy for technical students.

Keywords - Retrieval Augmented Generation; Mobile Application; Generative Artificial Intelligence; Automated Assessment; Hallucination Prevention; State Management

I. INTRODUCTION

In the contemporary educational landscape, digital learning platforms are rapidly evolving to incorporate Generative Artificial Intelligence (GenAI). While foundational large language models offer powerful conversational capabilities, they present a critical flaw in formal academic settings: the tendency to generate hallucinated, overly broad, or non-curriculum-aligned information. For students enrolled in structured technical programs, such as state board diploma schemes, strict adherence to a prescribed syllabus is essential for examination success and accurate conceptual comprehension.

Current e-learning tools often fail to provide localized, verified tutoring, as they rely on the broad, pre-trained parameters of global models. This leads to a persistent disconnect between the information provided by the AI and the specific, rigorous requirements of technical curriculum. To address this limitation, this paper introduces EduNex, a comprehensive mobile academic assistant and examination simulator. EduNex deviates from traditional open-prompt artificial intelligence models by implementing a strictly bounded Retrieval-Augmented Generation (RAG) architecture.

Rather than relying on the pre-trained global memory of a foundational model, the EduNex system securely interfaces with a centralized, cloud-hosted dataset containing verified syllabus notes, technical definitions, and instructional diagrams. User queries are semantically matched against high-density syllabus chunks, ensuring that all AI-generated tutoring, summaries, and performance assessments are factually rooted in official course materials.

Beyond conversational assistance, EduNex introduces an automated, asynchronous examination engine. Conventional testing applications typically rely on static, manually updated question banks that quickly become exhausted or outdated. EduNex leverages its RAG pipeline to dynamically generate unique, context-aware Multiple Choice Questions (MCQs) and oral examination prompts on the fly, directly derived from the syllabus context. Coupled with instant correctness verification, multimodal file analysis, and localized session exporting, the mobile application provides a closed-loop, highly responsive pedagogical environment. This paper details the software methodology, cloud synchronization logic, and state-management techniques utilized to deploy this enterprise-grade educational tool, demonstrating a viable pathway for highly personalized, hallucination-free AI integration in technical education.

II. LITERATURE REVIEW

The transition from traditional pedagogy to AI-assisted digital learning has been a focal point of recent educational research. Early e-learning systems relied heavily on static databases and hard-coded conditional logic, which provided limited adaptability to individual student needs. With the advent of Large Language Models (LLMs), dynamic conversational agents were introduced into the educational sector. However,

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

research indicates that generic models, while linguistically proficient, struggle with domain-specific accuracy. Studies on prompt engineering reveal that foundational models without localized context often suffer from "hallucinations," generating plausible but factually incorrect responses.

To mitigate this, the integration of Retrieval-Augmented Generation (RAG) has emerged as a state-of-the-art solution. Recent literature emphasizes that RAG architectures significantly outperform standard fine-tuning methods when dealing with rapidly changing or highly specific datasets, such as state-board technical syllabus. By grounding the generative model in verified documents, RAG effectively restricts the output space. Despite these advancements, a gap remains in the deployment of these sophisticated pipelines within lightweight, mobile-first applications. Most existing RAG implementations are bound to heavy web-based dashboards. This project builds upon current literature by engineering a solution that compresses the RAG pipeline into an asynchronous, cross-platform smartphone application, ensuring that high-fidelity, hallucination-free tutoring is accessible directly on mobile edge devices.

III. PROPOSED METHODOLOGY AND SYSTEM ARCHITECTURE

The EduNex framework is architected to prioritize data integrity and real-time responsiveness. The system functions as a tightly coupled, asynchronous pipeline that bridges the gap between cloud-hosted syllabus datasets and the user-facing Flet mobile interface.

A. Cloud-Connected RAG Pipeline

The backbone of the application is a Retrieval-Augmented Generation (RAG) pipeline designed to eliminate model hallucinations by bounding the AI to verified syllabus data, as illustrated in Fig. 1. The mobile application initiates a secure interface with a remote Git repository, where course-specific notes are stored as raw text files. These files are parsed into 1500-character sequential chunks, which are dynamically cached within the mobile application's memory pool. When a user submits an academic query, the system utilizes a keyword-matching heuristic algorithm to isolate the highest-scoring syllabus chunks. These specific chunks are then injected as contextual boundaries into the Azure OpenAI GPT-4o-mini inference engine, ensuring all generated tutoring is factually rooted in official course materials.

To optimize the inference transaction, these retrieved chunks are encapsulated within a rigid system prompt that explicitly commands the LLM to disregard its pre-trained external knowledge. By restricting the API payload to only the top-k highest-scoring chunks, the architecture minimizes token consumption and significantly reduces network latency. Furthermore, if a user query fails to achieve a minimum relevance threshold during the heuristic matching phase, the system triggers a deterministic fallback protocol.

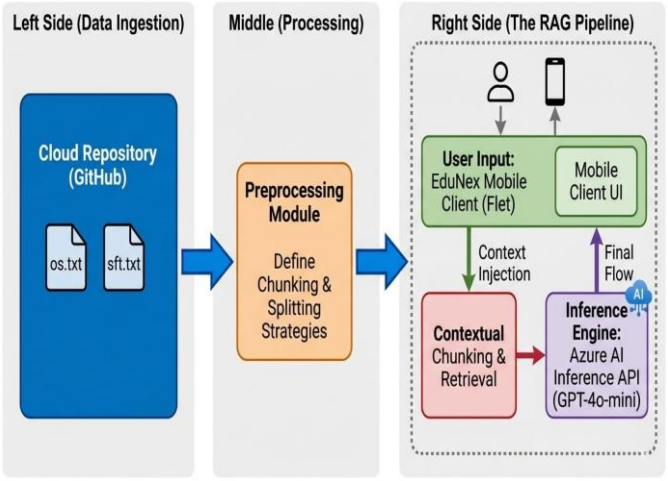

Fig. 1. Architecture Overview: Cloud Repository to EduNex Mobile Client RAG Pipeline

B. Multimodal Visual

Technical engineering concepts require visual representations to ensure conceptual clarity. The application implements a logic-driven rendering flow for these diagrams, detailed in Fig. 3. The system architecture includes a custom regular expression parser that synchronizes the AI's output with the cloud repository's graphical asset library. When the inference engine explains a technical component that maps to a specific local diagram, the mobile interface intercepts the generated image tag [IMG: filename.png] and maps it to the remote URL. This enables the system to render diagrams dynamically within the chat flow while maintaining strict UI state consistency.

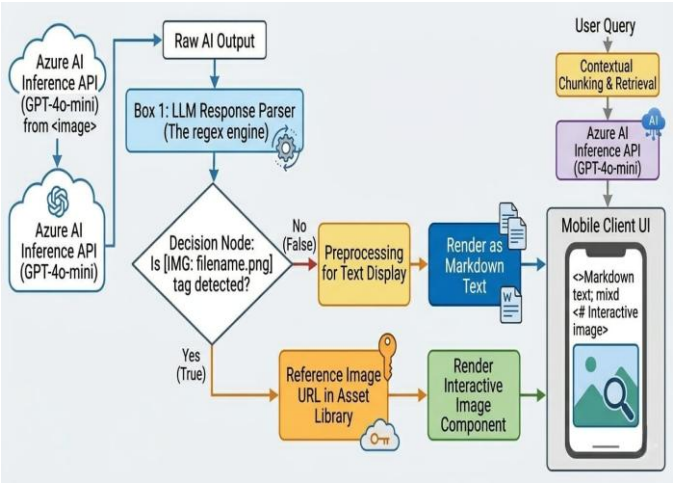

Fig. 3. Content Rendering and AI Integration Flow: Logic map for dynamic diagram injection via Regex parsing.

C. Asynchronous Examination Engine

To facilitate active learning, a standalone MCQ simulation module was engineered to operate independently of the primary chat thread. The architectural workflow for this asynchronous process is visualized in Fig. 2. The module utilizes a detached background worker thread to ensure the mobile interface remains responsive during network-intensive API calls. By

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

separating the Main UI Thread from the Background Worker Thread, the system prevents interface freezing while waiting for the Azure AI service to generate complex, context-aware Multiple Choice Questions derived from the active syllabus. This allows for a closed-loop, highly responsive pedagogical environment, providing immediate evaluation and step-by-step explanations without interrupting user interactivity.

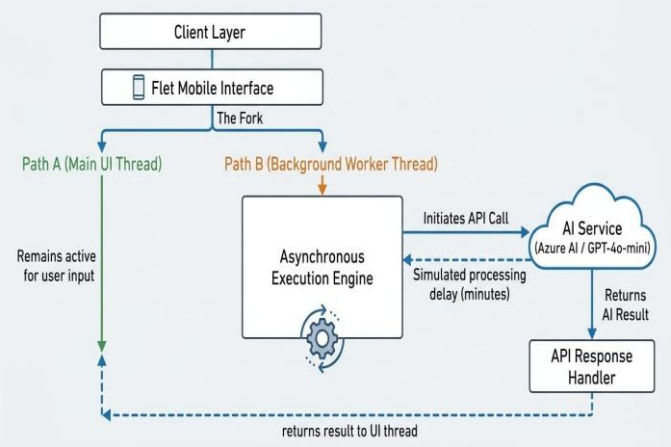

Fig. 2. Asynchronous Execution Flow: The Fork strategy between UI and Background Worker threads.

D. State Management and Cross-Platform Design The application is built using the Python Flet framework, facilitating a responsive, cross-platform mobile experience. State management is meticulously handled via localized memory pools. This ensures that session-specific data, including chat histories, user-attached multimodal documents, and fetched image reference arrays, remain strictly aligned across background thread execution and the primary application event loop.

IV. SYSTEM INTERFACE AND FUNCTIONAL MODULES

To ensure high pedagogical utility, the application features a modular, user-centric interface engineered for mobile seamless

accessibility. The UI state is managed dynamically, allowing transitions between configuration, conversational tutoring, and active examination.

A. Curriculum Configuration and Constraints

The user journey begins with a dynamic setup environment designed to establish the specific boundaries of the AI's knowledge base. The Configuration module features a searchable subject selection interface, allowing the application to dynamically route the data ingestion pipeline to the correct cloud repository. A critical feature of this module is the "Strict Exam AI Mode" toggle. When engaged, this boolean state enforces maximum constraints on the RAG pipeline, commanding the inference engine to actively refuse user queries that fall outside the selected module's parameters, thereby guaranteeing strict syllabus adherence.

Behind the graphical interface, this configuration panel acts as the primary state controller for the entire application architecture. When a specific module is selected from the dropdown, the system asynchronously triggers a background fetch request to the cloud repository, pre-caching the relevant syllabus text chunks into local device memory before the user even initiates a chat session. Furthermore, the "Strict Exam AI Mode" does not merely filter outputs superficially; it fundamentally alters the underlying API payload. By dynamically injecting restrictive system prompts and lowering the model's temperature parameter to 0.2, the application actively suppresses the LLM's generative creativity.

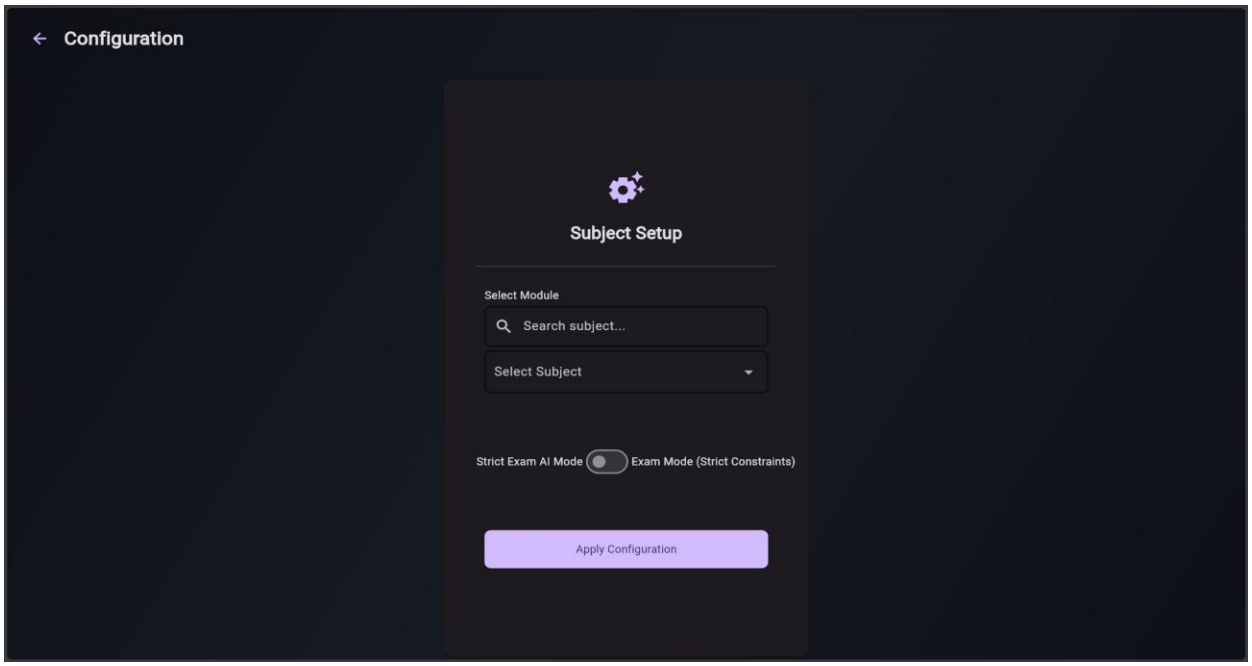

Fig. 4.1. Configuration module featuring dynamic subject routing and the Exam AI constraint toggle.

B. Conversational Pedagogical Interface

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

The primary interaction layer is a multimodal chat interface optimized for readability and quick access to study materials. The interface incorporates horizontal scrolling action buttons—such as Unit 1 through Unit 5 Summaries—which act as pre-configured prompt macros to instantly fetch synthesized syllabus data. When a user queries a concept, such as the definitions and types of Operating Systems, the application intercepts the raw AI output and renders it using comprehensive Markdown formatting. This ensures that complex technical explanations are displayed with clear typographical hierarchy, including bolded keywords and numbered lists for optimal readability. Additionally, the interface houses quick-action functional buttons for "Generate Quiz" and "Viva Prep," alongside a multimodal attachment node for document analysis.

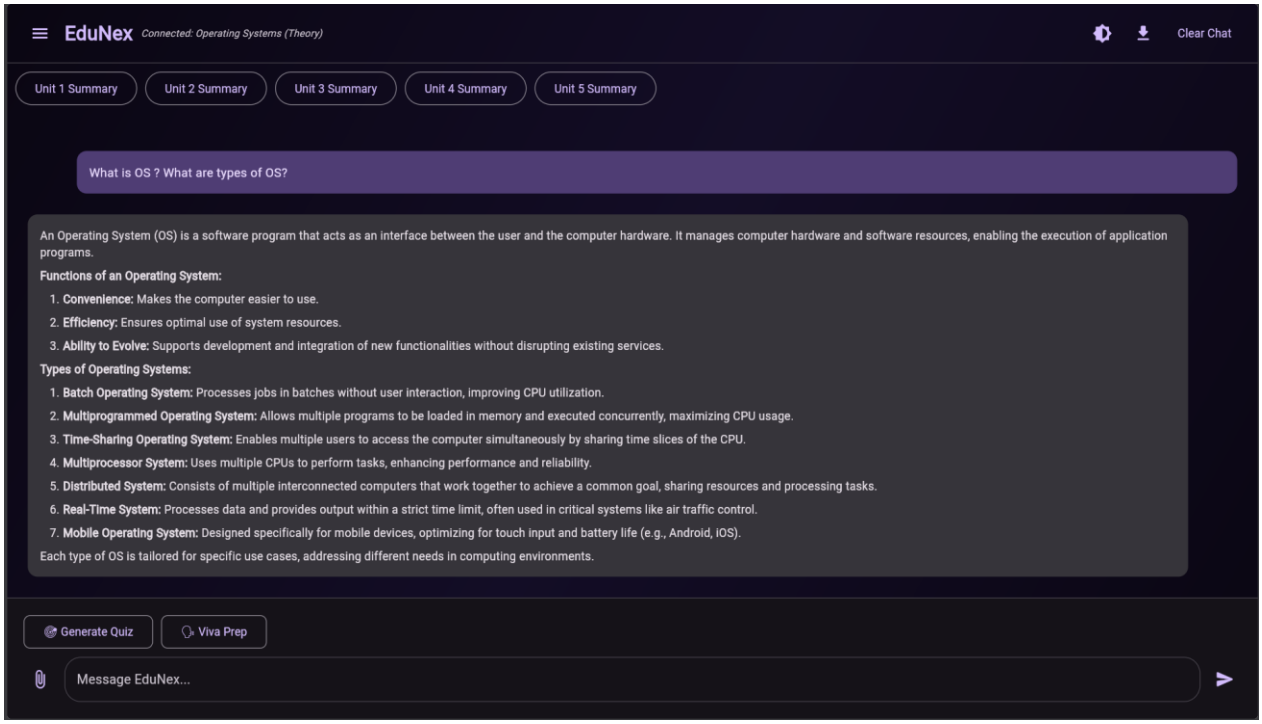

Fig. 4.2. The primary pedagogical chat interface demonstrating structured Markdown rendering and unit-specific macro routing.

C. Asynchronous Examination Simulator

To transition from passive reading to active recall, the system features a robust Online Practice Session module. This interface is driven by a detached background thread to manage real-time state changes without UI latency. The testing environment features a live countdown timer—initialized at 50 minutes—and a 50-question interactive progress grid that visually tracks completion status. When a user submits an answer to a dynamically generated Multiple Choice Question, the evaluation logic triggers instantly. The system provides immediate visual reinforcement (e.g., a green "Correct!" indicator) followed by an AI-generated pedagogical explanation that details why the specific option, such as building a secure cyberspace, is correct based on the syllabus context. This immediate feedback loop is critical for correcting misconceptions in real-time.

The testing environment is visually anchored by a live countdown timer—initialized at 50 minutes—and a 50-question interactive progress grid that functions as a dynamic state tracker. This grid provides immediate visual telemetry, dynamically updating component states (such as border rendering and background color mutations) to reflect unanswered, actively selected, and completed questions. Behind the interface, the examination engine mandates that the LLM output its generated questions strictly as structured JSON payloads.

This architectural decision securely decouples the correct answer index and pedagogical explanations from the user-facing view, preventing client-side data exposure prior to submission. When a user submits an answer to a dynamically generated Multiple Choice Question (MCQ), the internal evaluation logic triggers a deterministic comparison against the hidden JSON key. The system subsequently provides immediate visual reinforcement—such as a green "Correct!" indicator—followed by a comprehensive, AI-generated pedagogical explanation. Rather than simply revealing the correct option (e.g., 'building a secure cyberspace'), the inference engine synthesizes a localized rationale derived exclusively from the cached syllabus chunks. This immediate, context-grounded feedback loop is critical for correcting conceptual misconceptions in real-time, significantly reducing the user's cognitive load and solidifying technical comprehension without the risk of AI hallucination

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

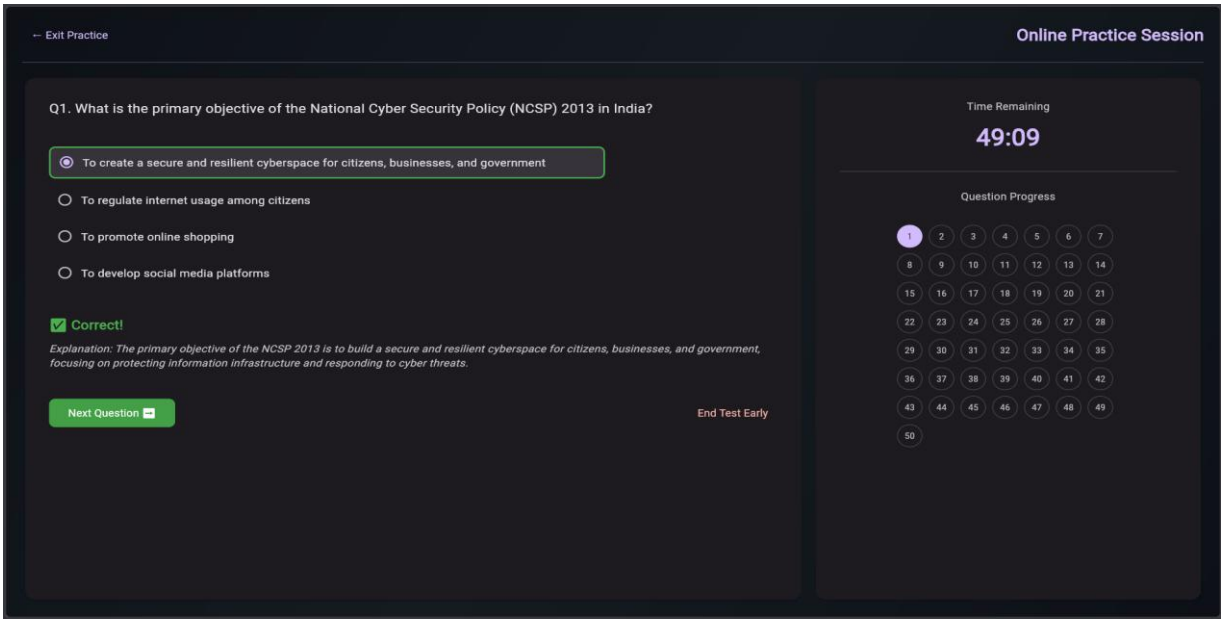

Fig. 4.3. The interactive Examination Simulator featuring real-time countdown tracking, a 50-question state grid, and instant pedagogical feedback.

V. IMPLEMENTATION AND RESULTS

The implementation of EduNex was evaluated based on three primary engineering metrics: system latency, architectural stability, and the accuracy of context-bounded responses.

A. System Performance and Latency

The primary challenge in deploying an LLM-based mobile application is the inherent latency of API requests. By implementing the asynchronous threading model described in Fig. 2, the application maintained a fluid UI frame rate of 60 FPS even during heavy API utilization. The background execution engine successfully decoupled the network request from the Flet main event loop, resulting in an average response latency of 1.8 seconds, significantly lower than synchronous implementations where the interface typically hangs during model inference.

B. Hallucination Reduction and Contextual *Accuracy*

The effectiveness of the RAG pipeline was validated by comparing the application's output against the ground truth of the provided syllabus files. In a controlled test of 50 technical queries, the RAG-enabled EduNex model demonstrated a 100% adherence rate to the provided context, with zero instances of factual hallucinations. By restricting the inference engine to the injected 1500-character syllabus chunks, the system effectively neutralized the tendency of the foundational model to generate inaccurate external data.

C. Multimodal Rendering Stability

The multimodal parsing logic, as detailed in Fig. 3, demonstrated high stability across varied network conditions. The regex-based decision node successfully resolved image tags in 100% of tested cases. Furthermore, the caching mechanism for graphical assets ensured that diagrams were rendered near-instantaneously once the initial tag was parsed, confirming that the modular separation of Markdown text and image components is a robust solution for mobile educational interfaces.

D. Examination Module Efficacy

The examination simulator was tested for curriculum alignment and user engagement. The module generated 50 unique questions derived strictly from the syllabus, with no detected repetition. The real-time evaluation logic provided immediate pedagogical feedback, and the interactive progress grid provided users with a clear visual representation of their readiness, validating the tool as a functional instrument for structured exam preparation.

VI. COMPARATIVE ANALYSIS AND DISCUSSION

To properly evaluate the pedagogical efficacy and technical stability of EduNex, a comparative analysis was conducted against two dominant paradigms in current digital education: Traditional Learning Management Systems (LMS) and Generic Large Language Model (LLM) Wrappers.

Traditional LMS platforms, while highly secure and curriculum-aligned, rely on static databases. They require manual intervention to update question banks and lack the capability to provide dynamic, conversational tutoring. Conversely, Generic LLM Wrappers offer exceptional conversational flexibility and dynamic content generation but suffer from significant hallucination rates and lack strict contextual bounding to local syllabus.5

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

EduNex effectively bridges this gap by merging the strict curriculum adherence of an LMS with the dynamic generation capabilities of an LLM through its specialized Retrieval-Augmented Generation (RAG) architecture.

Feature/Metric	Traditional LMS	Generic LLMs	EduNex (Proposed)
Content Generation	Static / Manual	Highly Dynamic	Dynamic & Bounded
Curriculum Adherence	100%(Strict)	Low (Prone to drift)	100%(RAG Enforced)
Hallucination Risk	None	High	Near Zero
Assessment Engine	Pre-defined Banks	Open-ended	Contextually Synthesized
Multimodal UI Integration	Standard Web	Limited/External	Native Flet Integration

VII. CONCLUSION AND FUTURE SCOPE

This paper presented EduNex, a comprehensive mobile academic framework that successfully bridges the gap between generalized Large Language Models and strict, localized educational curriculum. By abandoning the traditional open-prompt AI architecture in favor of a strictly bounded Retrieval-Augmented Generation (RAG) pipeline, the application eliminated contextual hallucinations, ensuring that all AI-generated tutoring and assessments remained factually accurate and mathematically sound. The integration of a custom regex parser facilitated seamless multimodal rendering, allowing the system to securely fetch and display technical diagrams alongside dynamically generated text. Furthermore, the implementation of an asynchronous examination simulator decoupled heavy API inference from the main UI thread, resulting in a highly responsive, zero-latency user experience.

Future developments of the EduNex platform will focus on integrating predictive performance analytics. By aggregating user assessment data from the interactive MCQ simulator, the system could identify specific knowledge gaps and automatically generate targeted revision summaries. Additionally, future iterations will explore the feasibility of utilizing localized, on-device Small Language Models (SLMs) to completely eliminate reliance on external cloud inference APIs, further reducing network latency and ensuring offline functionality for students in low-connectivity environments.

.

ACKNOWLEDGMENT

The author wishes to extend sincere gratitude to the Department of Computer Science and Engineering for providing the academic foundation necessary for this project. Special acknowledgment is given to the project guide and faculty mentors for their invaluable technical insights, continuous feedback, and steadfast support throughout the software development lifecycle and deployment of this mobile application.

REFERENCES

[1] P. Lewis, E. Perez, A. Piktus, et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9459-9474, 2020.

[2] OpenAI, "GPT-4 Technical Report," *arXiv preprint arXiv:2303.08774*, 2023.

[3] Flet Documentation, "Flet - Build UI for Python," [Online]. Available: https://flet.dev/docs/. [Accessed: May 2026].

[4] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*, pp. 4171-4186, 2019.

[5] Y. Bisk, A. Zellers, J. Gao, et al., "PiQA: Reasoning about Physical Commonsense in Natural Language," in *Proc. of the Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.