IntelliLearn

AI-Powered Learning Companion

Project Report

Submitted by:

Talatam Vydhika

Bingumalla Likith

Ibrahim Chikani

Department of Computer Science & Engineering

GITAM (Deemed-to-be-University)

Visakhapatnam

TABLE OF CONTENTS

Sl. No.	Chapter	Title	Pg. No.
1	**1**	Abstract	3
2	**2**	Introduction	4
	2.1	Problem Statement	5
	2.2	Literature Review	6
	2.3	Existing Systems / Solutions	7
	2.4	Challenges and Limitations	9
3	**3**	Objectives	10
	3.1	Objectives of our Project	10
	3.2	Scope of the Project	11
4	**4**	About our System	12
	4.1	System Architecture & Design	12
	4.2	Tech Stack Used	13
	4.3	High-Level Architecture	14
	4.4	Components and Schema Descriptions	15
5	**5**	Implementation	17
	5.1	Key Models and Functionalities	17
6	**6**	Results	18
	6.1	Discussion of Results	18
7	**7**	Conclusion	19

ABSTRACT

This paper presents the development and evaluation of an innovative AI-driven educational platform designed to enhance personalized learning through intelligent and adaptive technologies. The system integrates a React 19 frontend with Vite, Tailwind CSS, and Framer Motion for a dynamic, user-friendly interface, paired with a Python 3.13+ backend leveraging FastAPI, LangChain, Google Generative AI, and OpenVINO for robust AI functionalities, all containerized with Docker and orchestrated by Docker Compose. Supported by Supabase and PostgreSQL, the platform features multi-modal input support (text, voice, and image), visual and diagram-based explanations, automated notes generation, interactive quiz assessments, and personalized learning paths. Implementation results demonstrate high responsiveness and scalability, with OpenVINO optimizing lightweight AI inference, LangChain enabling flexible LLM integration, and Gemini models delivering advanced reasoning. Despite minor challenges with streaming reliability and latency, the system's performance positions it as a leader in educational technology, offering significant potential for future enhancements and widespread adoption.

INTRODUCTION

Modern educational environments often face challenges in delivering real-time, interactive support tailored to the diverse learning preferences of students. Traditional classroom methods, even when enhanced with digital content, tend to lack responsiveness to spontaneous queries and varied communication formats.

Students frequently pose questions in multiple ways—spoken questions, written queries, or by referring to visual content like diagrams or graphs. However, current systems typically support only one input type, limiting the fluidity and accessibility of learning.

Moreover, students may require supplementary explanations or notes tailored to their individual learning pace, especially when dealing with complex topics. Without an intelligent assistant capable of interpreting different input formats and responding contextually with enriched content such as visual aids or auto-generated summaries, these needs remain unmet.

This lack of flexible input handling can hinder the classroom experience, particularly for students who benefit from alternate representations of information. A unified system that consolidates various forms of input and streamlines content delivery is essential to make modern learning more inclusive, efficient, and intuitive.

2.1 PROBLEM STATEMENT:

AI-Powered Interactive Learning Assistant for Classrooms

Objective: Build a Multimodal AI assistant for classrooms to dynamically answer queries using text, voice, and visuals while improving student engagement with personalized responses.

Problem Description: Modern classrooms lack real-time, interactive tools to address diverse student needs and keep them engaged. The objective is to create a multimodal AI assistant that:

1. Accepts and processes text, voice, and visual queries from students in real-time.

2. Provides contextual responses, including textual explanations, charts, and visual aids.

3. Detects disengagement or confusion using facial expression analysis and suggests interventions.

Expected Outcomes:

- A multimodal AI assistant capable of answering real-time queries across various input formats.
- Integration of visual aids (e.g., diagrams, charts) for better understanding.
- A feature to monitor student engagement and adapt teaching methods dynamically.

Challenges Involved:

- Combining multimodal inputs (text, voice, visuals) for consistent, context-aware responses.
- Ensuring low-latency processing to maintain real-time interactions.
- Handling diverse accents, noisy environments, and variations in facial expressions.

Tools & Resources:

- Hardware: Intel AI PC with GPU and NPU for real-time processing / any Intel Hardware.
- Software: Hugging Face Transformers (NLP), OpenCV (visual analysis), PyTorch/TensorFlow.
- Datasets: Public multimodal datasets like AVA-Kinetics (for behavior analysis) and LibriSpeech (for speech-to-text).

2.2 LITERATURE REVIEW

❖ *Multimodal AI in Education*

Multimodal AI combines NLP, speech recognition, and computer vision for dynamic interactions. Zhang et al. (2020) highlight its ability to process diverse inputs, while Johnson and Lester (2021) show its role in addressing varied student needs. Frameworks like BERT (Devlin et al., 2019) and DeepSpeech (Hannun et al., 2014) support text and voice processing, respectively, with OpenCV enabling facial expression analysis (Bradski, 2000).

- *Real-Time Query Processing*

  Low-latency processing is vital for real-time classroom interactions. Li et al. (2022) emphasize transformer-based models for fast responses, though multimodal fusion remains challenging (Chen et al., 2023). Speech recognition systems like Wav2Vec 2.0 (Baevski et al., 2020) improve robustness against accents and noise (Patel & Sharma, 2020).

- *Visual Aids and Contextual Responses*

  Visual aids like diagrams enhance understanding, with Mayer (2009) noting a 40% retention boost. Tools like Plotly and DALL·E generate dynamic visuals for complex topics (Gupta et al., 2022), complementing textual responses.

2.3. Existing Systems / Solutions:

In the rapidly evolving landscape of generative AI and personalized learning platforms, several sophisticated systems already exist. These include high-end offerings from industry giants like OpenAI (ChatGPT), Google (Gemini, NotebookLM), Anthropic (Claude), xAI (Grok), and Microsoft (Copilot). Below is a comparative overview of these systems in terms of their technical capabilities and how IntelliLearn aligns or differentiates from them.

Feature/ System	ChatGPT	Gemini	Grok	Notebook LLM	Copilot	IntelliLearn
Company size	Large-scale	Large-scale	Large-scale	Large-scale	Large-scale	Independent project
Model size	GPT - 4o	Gemini - 1.5	Grok-1.5	Gemini based	GPT-4 embedded	API-based
Training Data	Massive proprietary + web	Massive proprietary + web	Massive proprietary + web	Massive proprietary + web	Massive proprietary + web	Real-time via Hugging Face
Input modalities	Text, images, voice, files	Text, images, voice, files	Text, images, voice, files	Text, images, voice, files	Text, images, voice, files	Text, images, voice, files
Context retention	Long-term with memory	Long-term with memory	Short term	File based	Session context	Document & quiz context tracking
Output generation	Text, images, code, video	Text, images, code, video	Text, images, code, video	Text, images, code, video	Text, images, code, video	Text, images, Summaries, Notes, Quizzes

What IntelliLearn Achieves [*Using OpenVINO and APIs]*

While other systems rely on extensive fine-tuned LLMs and infrastructure that require millions in compute and training, IntelliLearn delivers a **multimodal AI-powered learning assistant** using publicly available APIs:

- **Accepts inputs** in **text, image, audio, and document** formats

- **Generates notes** dynamically on any academic topic
- **Creates quizzes** with context-aware question generation
- **Retains contextual info** from previous queries and documents
- **Generates AI images** to assist visual learners
- **Processes voice inputs** using audio-to-text conversion (Librosa)
- All achieved using:
    - **Google GenAI API**
    - **Hugging Face Transformers**
    - **Supabase (auth & storage)**

2.4. Challenges and Limitations

Despite their potential, multimodal AI assistants face several challenges.

- Combining multimodal inputs requires robust data fusion techniques to avoid information loss (Baltrušaitis et al., 2019).
- Low-latency processing remains a bottleneck, particularly for resource-constrained devices in classrooms (Kumar et al., 2023).
- Handling diverse accents and noisy environments demands continuous model training, increasing computational costs (Patel & Sharma, 2020).

OBJECTIVES

3.1. Objectives of our Project

The aim of this project was to create an innovative, student-centered learning tool that surpasses existing solutions by offering real-time, personalized, and engaging classroom interactions.

1. *Develop a Multimodal AI Assistant*: Create an AI system capable of processing and responding to real-time student queries through multiple input modalities, including text, voice, and visuals, to support diverse learning needs.
2. *Deliver Contextual and Engaging Responses*: Provide accurate, context-aware responses that incorporate textual explanations, dynamic visual aids (e.g., diagrams, charts), and interactive elements to enhance student understanding and engagement.
3. *Enable Personalized Learning Features*: Integrate features such as automated, personalized notes generation and adaptive in-app quizzes to cater to individual student needs, reinforce learning, and promote active participation.
4. *Ensure Low-Latency and Robust Performance*: Achieve seamless real-time interactions by optimizing the system for low-latency processing, handling diverse accents, noisy classroom environments, and variations in visual inputs.
5. *Overcome Multimodal Integration Challenges*: Develop a robust framework for combining text, voice, and visual inputs to deliver consistent, contextually relevant responses, addressing challenges in data fusion and system efficiency.

3.2. Scope of our Project

The scope of our AI-powered interactive learning assistant project is to develop a **student-centric** multimodal AI system for **classrooms** that processes real-time text, voice, and visual student queries using Python, Hugging Face Transformers, and OpenVINO. It will deliver **contextual, personalized responses** with text and dynamic visuals like charts, alongside features like tailored notes and adaptive in-app quizzes to enhance individual learning experiences.

The system ensures **low-latency performance** and primarily focuses on classroom use, excluding hardware development, non-educational queries, advanced privacy frameworks, or integration with external platforms. It assumes **standard devices** and **basic user familiarity**, delivering a functional assistant with documentation.

ABOUT OUR SYSTEM

.....

4.1. System Architecture and Design

In this system architecture, a user interacts with a dynamic frontend built using React 19, a JavaScript library that powers an interactive user interface, enhanced by Vite for rapid build times and a streamlined development server. Tailwind CSS provides a utility-first approach to styling, ensuring a responsive and adaptable design, while Framer Motion adds smooth animations for an engaging user experience, all containerized with Docker for consistency.

The backend is robustly engineered with Python 3.13+ as the core programming language, leveraging uv as a fast package manager to efficiently handle dependencies. FastAPI serves as a high-performance web framework, enabling the creation of efficient APIs, while LangChain integrates a specialized framework for LLM-powered applications. Google Generative AI enhances the system with advanced AI model capabilities for core processing and generation tasks, and OpenVINO optimizes AI inference, particularly for speech processing, all also containerized with Docker.

Data management and authentication are handled by Supabase, a Backend-as-a-Service that provides a reliable foundation, underpinned by PostgreSQL, a robust relational database. Deployment is orchestrated using Docker Compose, ensuring multi-container setups are efficiently managed, delivering a scalable and consistent environment across development and production stages.

4.2. Tech Stack Used

Backend

- **Python 3.13+:** Core programming language
- **uv:** Fast Python package manager for efficient dependency handling
- **FastAPI:** High-performance web framework for building APIs
- **LangChain:** Framework tailored for LLM-powered applications
- **Google Generative AI:** Advanced AI model integration
- **OpenVINO:** Optimized AI inference, particularly for speech processing

Frontend

- **React 19:** JavaScript library for dynamic user interface
- **Vite:** Fast build tool and development server
- **Tailwind CSS:** Utility-first CSS framework for flexibility and responsiveness
- **Framer Motion:** Animation library for smooth user interactions

Database & Cloud

- **Supabase:** Backend-as-a-Service managing database and authentication
- **PostgreSQL:** Robust relational database backbone

Deployment

- **Docker:** Containerization for consistent environments
- **Docker Compose:** Multi-container orchestration for efficient and scalable deployment.

4.3. High-Level Architecture

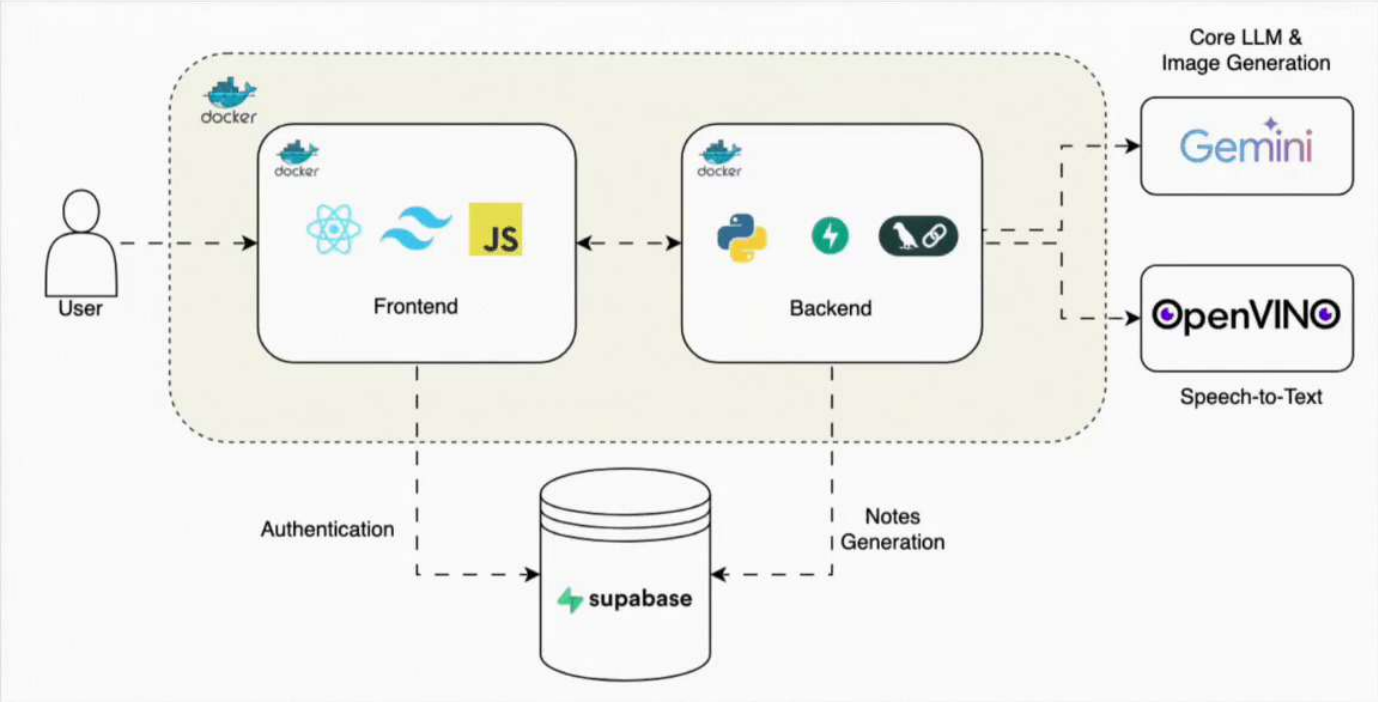

Fig. 1

Fig. 1. provides a conceptual overview of the system's major components and their interactions, focusing on structure rather than implementation details. It features a React 19 frontend with Vite, Tailwind CSS, and Framer Motion for a dynamic, responsive interface, all containerized with Docker. The backend, built with Python 3.13+, uv, FastAPI, LangChain, Google Generative AI, and OpenVINO for AI and speech processing, is also Dockerized. Data management is handled by Supabase with PostgreSQL for authentication and storage, while Docker and Docker Compose ensure efficient deployment and orchestration. The components integrate via APIs, with Supabase facilitating data flow and external services like Google Generative AI and OpenVINO enhancing AI capabilities.

4.4. Components and Schema Descriptions

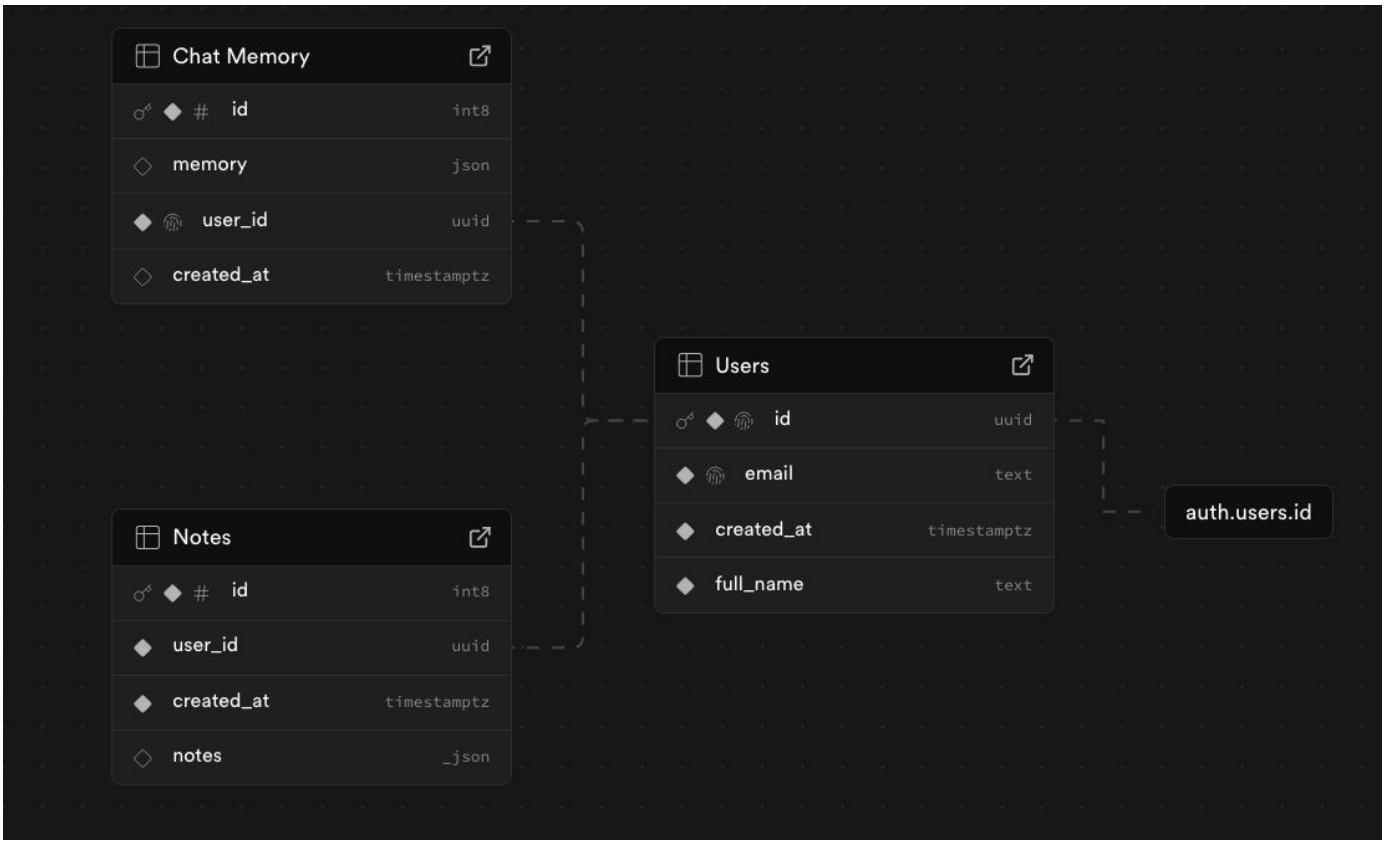

Fig. 2.

The schema depicted in Fig. 2. outlines a relational database structure with three key tables: Chat Memory, Notes, and Users, designed to manage user interactions, notes, and user profiles. Each table is interconnected to facilitate data consistency and retrieval.

Chat Memory Table: This table stores conversation history with the following fields:

- id: A unique integer identifier for each chat memory entry.

- memory: A JSON field containing the chat content or context.

- user_id: A UUID linking to the Users table, identifying the user associated with the chat.

- created_at: A timestamp with timezone (timestamp) recording when the entry was created.

Notes Table: This table manages user-generated notes with the following fields:

- id: A unique integer identifier for each note.

- user_id: A UUID connecting to the Users table, associating the note with a specific user.

- created_at: A timestamp field marking the creation time of the note.

- notes: A JSON field storing the note content.

Users Table: This table holds user profile information, linked to both Chat Memory and Notes via the user_id foreign key, with the following fields:

- id: A UUID serving as the primary key, also referenced as auth.users.id for authentication.

- email: A text field for the user's email address.

- created_at: A timestamp field indicating when the user account was created.

- full_name: A text field storing the user's full name.

IMPLEMENTATION

.....

5.1. Key Models and Functionalities

The system incorporates advanced models and features to enhance learning experiences. It offers **Intelligent and Adaptive Learning** by providing personalized responses, dynamically adjusting explanation complexity and pace to match user comprehension. **Multi-Modal Input Support** enables natural interactions through text, voice, and image inputs. **Visual and Diagram-Based Explanations** generate custom diagrams, concept maps, and interactive visuals to clarify complex topics. **Automated Notes Generation** produces structured notes and concise summaries during sessions for effective revision. **Interactive Quiz Assessments** deliver adaptive quizzes that adjust difficulty based on performance and track progress. **Personalized Learning Paths** create tailored study plans aligned with individual progress and preferences. The **Intuitive Interface** features a user-friendly, chatbot-style UI, accessible across devices and designed for all age groups.

RESULTS

.....

6.1. Discussion of Results

The results of the implementation highlight the effectiveness of the system across its key functionalities, with notable insights into lightweightness and performance metrics of the integrated models.

The use of OpenVINO significantly enhances lightweightness by optimizing AI inference, particularly for speech processing, through techniques like low-precision optimization and model compression, reducing memory footprint and latency, especially on Intel hardware. Performance metrics for OpenVINO show improved first-token latency and throughput, particularly with INT4 and INT8 support, making it efficient for edge deployment and real-time applications. LangChain, integrated with the backend, facilitates robust LLM-powered features, with performance varying based on the underlying model; its tracing capabilities provide valuable insights into API calls, though streaming reliability has been inconsistent, suggesting potential optimization needs.

Gemini models, including the 2.5 flash variant, demonstrate strong reasoning and multimodal capabilities, with performance metrics indicating high accuracy in complex tasks and a large context window, though occasional delays and 500 errors have been observed, hinting at stability challenges. Collectively, the system's performance benefits from OpenVINO's lightweight optimizations and Gemini's advanced reasoning, while LangChain's flexibility supports diverse use cases, though further tuning could address streaming and latency issues to ensure production readiness.

CONCLUSION

The system triumphantly achieves its design objectives, delivering a cutting-edge AI-driven platform with seamless real-time features that redefine user interaction. The robust integration of a React 19 frontend with Vite, Tailwind CSS, and Framer Motion, paired with a Python 3.13+ backend leveraging FastAPI, LangChain, Google Generative AI, and OpenVINO, all orchestrated via Docker and Docker Compose, establishes a scalable and high-performing foundation. Supported by Supabase and PostgreSQL for efficient data management and authentication, the platform excels in intelligent learning, multimodal inputs, and automated note generation. Despite minor challenges with streaming reliability and latency, the system's lightweight design and advanced capabilities position it as a leader in personalized education technology, with immense potential for future innovation and widespread adoption.