International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 11 | Nov 2025 www.irjet.net p-ISSN: 2395-0072

ACE- AI Based Cognitive Education

Anirudh Kulkarni[1], Darur Eashwar[2], Godeshwari[3], Aishwarya Hipparagi[4], S N Kugali[5]

[1234]Student, [5]Assistant Professor Department of Information Science and Engineering, Basaveshwar Engineering College, Bagalkote, India

---

ABSTRACT-

The AI-Based Cognitive Education (ACE) system integrates artificial intelligence and cognitive science to deliver a personalized, adaptive learning experience. It analyzes learners' performance and behavior to tailor content, provide real-time feedback, and enhance understanding and skill development. ACE transforms admin-uploaded Pd's into a retrieval-augmented study and interview workflow, generating structured learning paths and difficulty-controlled question sets. Using an Ollama-compatible LLM, it evaluates responses with grounded references, while a Fast API vision service employing YOLO face detection and Insight-face embeddings ensures identity verification. Built on a Node.js/Express back-end with MongoDB and Redis, ACE optimizes latency through a cache radius parameter. Evaluation metrics such as Precision, Recall, and F1-score assess LLM accuracy. By combining AI driven automation with cognitive principles, ACE advances smart, accessible, and future-ready education.

Key Words: AI in Education, Cognitive Learning, Adaptive Learning Systems, Personalized Education, Intelligent Tutoring System, Machine Learning, RAG, guided study, LLM evaluation, proctoring, YOLO, InsightFace, MongoDB, Redis

1. INTRODUCTION

In today's digital age, the rapid growth of online education has created vast opportunities for learners, yet it has also introduced challenges such as scattered learning resources, limited practical evaluation, and lack of personalized feedback. Traditional e-learning platforms often focus solely on theoretical knowledge, overlooking the importance of real-world skill assessment and adaptive learning experiences.

The AI-Powered E-Learning & Interview Proctor System is designed to bridge this gap by combining artificial intelligence with cognitive learning principles. The system curates high-quality educational content, simulates real-world interview environments, and offers instant, personalized feedback to help learners identify strengths and areas for improvement. Through intelligent monitoring and data-driven analysis, ACE not only ensures academic integrity during assessments but also enhances learner engagement and performance. by integrating AI-based evaluation, adaptive learning, and automated interview proctoring, this project aims to create a holistic digital learning ecosystem that prepares students for both academic success and professional readiness in an increasingly competitive world.

Fragmented learning makes it hard to progress from materials to mastery and assessment. ACE integrates:

- Admin-driven PDF ingestion to build a domain knowledge store.

- Guided study recommendations based on observed mastery.

- Retrieval-augmented interviews and LLM rubric evaluation.

- Vision-based proctoring (YOLO detection + InsightFace verification).

- A cache "redius" framework to reduce latency under load.

2. OBJECTIVES

- To build an adaptive learning platform that delivers personalized study paths tailored to individual learners
- To curate and organize learning resources in a structured manner, ensuring accessibility, clarity, and relevance for diverse learners.
- To enable real-time analytic and personalized feedback to accelerate skill improvement
- To simulate real-world interview environments that help users practice communication, technical, and problem-solving skills effectively.
- To provide instant, data-driven feedback and analytics for continuous performance improvement and career readiness.

3. LITERATURE SURVEY

1. Optimizing React Components For Improving The Performance Of MERN Stack Application.

Key Findings:-
This paper explores effective strategies to improve React component performance in MERN stack (MongoDB, Express.js, React, Node.js) applications. As apps grow more complex, a fast and efficient UI is crucial. The goal is to provide developers with clear, practical methods to boost React performance and enhance user experience.

2. Innovative E-Learning System Empowering Education Through Engagement and Personalization

Key Findings:-
This paper introduces a modern e-learning platform developed using the MERN stack (MongoDB, Express, React, Node.js), addressing the limitations of current systems—such as lack of personalization, real-time feedback, and support. Key features include personalized course recommendations (SVD), real-time video quizzes with auto-grading, gamification (leaderboards, badges), automated certifications, and a rule-based chatbot for course-specific, real-time assistance.

3. Evaluating the Efficacy of Open-Source LLMs in Enterprise-Specific RAG Systems: A Comparative Study of Performance and Scalability

Key Findings:- This paper evaluates open-source LLMs in Retrieval-Augmented Generation (RAG) for enterprise data, showing that effective embeddings enhance accuracy and efficiency, making them a cost-effective alternative to proprietary systems.

4. A Systematic Review of Machine Learning Techniques in Online Learning Platforms.

Key Findings:-
Education has shifted to online platforms enhanced by AI, especially machine learning. Techniques like deep learning, NLP, and reinforcement learning improve personalization and performance. This study reviews 2015–2021 research on ML use, trends, and challenges in e-learning.

5. StudyNotion: MERN based Ed-Tech Platform

Key Findings:-
StudyNotion is a MERN stack–based ed-tech platform that enables content creation, learning, and rating with secure, scalable architecture. It highlights system design, deployment, and testing, focusing on accessibility, performance, and student-centric learning enhancements.

6. Enhancing Comprehension with LLM, TTS, and RAG: Transcription of Text into Podcasts and Chatbot.

Key Findings:-
This paper presents a system combining LLMs, Text-to-Speech, and Retrieval-Augmented Generation (RAG) to convert documents into interactive podcasts and chatbots. It enhances comprehension, engagement, and accessibility by enabling conversational, audio-based learning experiences with contextual responses and realistic voice synthesis.

7. A Retriever–Analyzer–Generator (RAG) Pipeline for Interactive Interview Preparation using LLMs .

Key Findings:-
This paper presents an AI-driven interview preparation system using a Retriever–Analyzer–Generator (RAG) pipeline. The system retrieves domain knowledge, analyzes user responses to detect skill gaps, and generates adaptive interview questions via LLMs, enabling personalized mock interviews and real-time feedback to enhance candidate performance.

8. Analysis on AI Proctoring System Using Various ML Models

Key Findings:-
This paper proposes an AI-based proctoring system that removes the need for human supervision in online exams. Using YOLO for real-time object detection and FaceNet for accurate facial recognition, the system detects unauthorized objects, verifies identities, and adapts to emerging security threats. A multi-modal approach enhances accuracy, scalability, and speed compared to existing models, offering an intuitive interface for both proctors and test-takers and providing real-time alerts for suspicious behavior.

4. PROBLEM FORMULATION

Introduction :-

Despite the rapid growth of e-learning platforms, most lack personalization, structured learning paths, and real-time evaluation. Learners struggle to identify relevant content, measure progress, and prepare effectively for real-world job scenarios. Existing systems focus mainly on knowledge delivery rather than practical skill assessment.

To address these challenges, the proposed ACE (AI-Based Cognitive Education) system aims to integrate AI-driven adaptive learning, automated interview simulation, and performance analytics within a unified platform. The objective is to create an intelligent

learning environment that personalizes study content, tracks learner progress, provides instant feedback, and enhances job readiness through cognitive automation[2].

- rag- Retrieval-Augmented Generation for grounded question answering.
- adaptive_learning- Mastery and recommendation models for personalized study.
- proctoring- Face detection/verification for integrity (YOLO, InsightFace).
- llm_scoring- Automated evaluators and reliability analyses in education.

To ensure a seamless and scalable user experience, the system is built using the MERN (MongoDB, Express.js, React, Node.js) stack, which provides efficient data handling, modular APIs, and dynamic front-end rendering suitable for adaptive e-learning environments [1], [2],[5].

RAG (Retrieval-Augmented Generation) [6][7]

RAG is a technique by which a large language model (LLM) is augmented with an external document retrieval component so that it can reference specific knowledge bases (e.g., uploaded PDFs, domain documents) rather than relying solely on its pretrained parameters.

In ACE, the RAG workflow works as follows:
The system ingests domain-specific course materials (PDFs) and chunks and embeds them into a vector store.At interview runtime, given a learner's prompt, the retrieval module fetches the top-k relevant chunks.These retrieved passages are appended to or used in constructing the prompt for the LLM, ensuring generation is grounded in domain-specific content rather than unconstrained.

This approach reduces hallucinations, enables up-to-date content usage, and aligns responses with the curated corpus of the platform.

Additionally, a Text-to-Speech (TTS) component [6] is integrated into the RAG module to convert AI-generated responses into natural voice outputs. This enhances accessibility and provides a more interactive learning experience, especially for visually impaired learners or hands-free environments.

YOLO (You Only Look Once)

Recent studies on machine learning integration in online learning [4] have shown that techniques such as deep learning, reinforcement learning, and natural language processing have significantly improved personalization, adaptability, and engagement in educational platforms. However, these systems often face limitations such as restricted access to real-world datasets, privacy constraints, and inconsistent data availability, as many online learning platforms do not publicly share user information. This makes it challenging to train and evaluate robust models on proprietary data. Despite these challenges, there remains a vast opportunity for machine learning models like YOLO (You Only Look Once) to strengthen online systems by providing secure and adaptive user experiences[8].

YOLO is a real-time object (in this case, face) detection framework widely used in computer vision tasks for its speed and accuracy.

In ACE's proctoring module:

YOLO is used to detect the learner's face in the video feed at the start and during the interview session. its output (face count, presence, bounding boxes) is combined with an embedding verification system (e.g., using InsightFace) to check identity and session integrity.If multiple faces appear, or the face is absent or occluded, the system flags the session for integrity violation.[8].

This integration of YOLO ensures that the proctoring system is responsive in real-time, enabling detection of suspicious behaviour or impersonation.
Vision Proctoring with YOLO and InsightFace

This module ensures identity verification and integrity monitoring during AI-based interviews. It integrates YOLO for real-time face detection and InsightFace for facial embedding and verification.

- The system captures both a reference image and a live image from the user.
- YOLO detects the user's face and counts multiple detections (to flag possible impersonation).
- InsightFace generates normalized facial embeddings and compares them using cosine similarity to verify identity.
- An integrity model combines match scores and event flags (like multiple faces or missing faces) to produce an overall integrity score.
- This process ensures that only the legitimate candidate is active during the interview, maintaining authenticity and fairness.

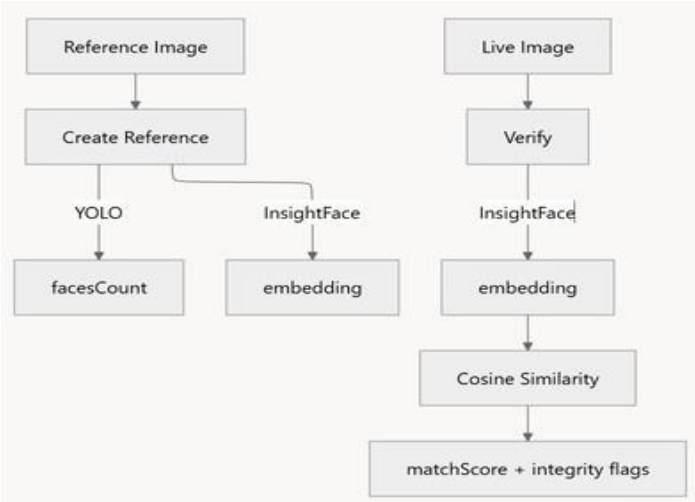

Fig. 4.1 YOLO Flow Chart

Cache Redius for Latency Reduction

This component focuses on optimizing system performance by reducing retrieval latency during learning and interview sessions. It uses Redis as a caching layer to store and quickly access frequently requested content and query results.

- A cache radius parameter (r_cache) determines how widely data is cached:
  - r_cache = 0 → exact key caching (high specificity, lower reuse).
  - r_cache > 0 → normalized/bucketed caching (higher reuse, lower latency).

The system retrieves data from Redis if available (cache hit) or fetches from the LLM and database if not (cache miss).

- Cached data is stored with a time-to-live (TTL) value to ensure freshness and efficiency.
- Increasing the cache radius improves hit rate and response time, balancing accuracy and speed.

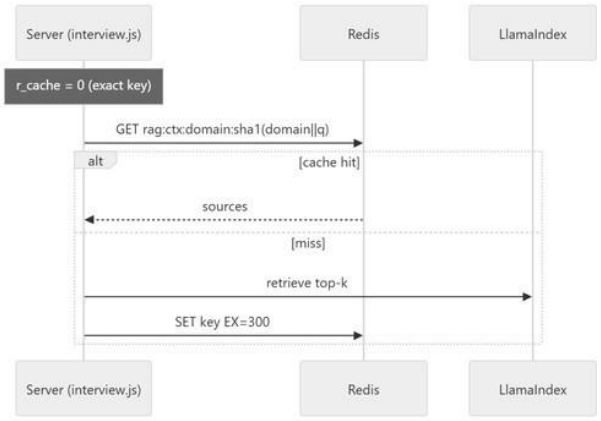

Fig. 4.2 Redius Flow Chart

LLM's used

The ACE (AI-Based Cognitive Education) system integrates multiple Large Language Models (LLMs) for retrieval, evaluation, and feedback generation.

Main Evaluation Model: Llama 3.1 8B Instruct (q4_K_M) — used for rubric-based interview scoring and report generation. Configured with a low temperature (0–0.2) and medium context window (512– 1536 tokens) to ensure consistent, reasoning-focused outputs while maintaining efficiency through quantization.

Lightweight Evaluator: Llama 3.2 1B Instruct — employed for rapid per-answer scoring and next-question recommendations, offering high speed and low resource usage for short responses.

Embedding Model: nomic-embed-text — generates dense semantic embeddings for RAG (Retrieval-Augmented Generation) to enable accurate, context-based retrieval from study materials.

As evident from Llama3 outperforming Mistral, especially for proprietary enterprise[3]. The models are hosted locally through Ollama, interfaced via /api/generate and /api/embeddings endpoints, and integrated using LlamaIndex for efficient retrieval and inference. Operational flags like FAST_FLOW and DEFER_EVAL optimize evaluation latency and system responsiveness. Fully local inference ensures data privacy and eliminates external dependencies, while quantized models balance speed and accuracy. However, the 1B model shows limited reasoning ability, and the 8B model's shorter context window requires careful retrieval curation.

5. DESIGN

Architecture diagram

This diagram represents the end-to-end architecture of an intelligent web-based system that integrates AI-powered processing, real-time proctoring, and data management. The architecture ensures scalability, efficiency, and secure communication between various components. The system leverages a modular structure, enabling each layer—including frontend interfaces, backend services, AI engines, and proctoring modules—to operate independently while still communicating smoothly through well-defined APIs.

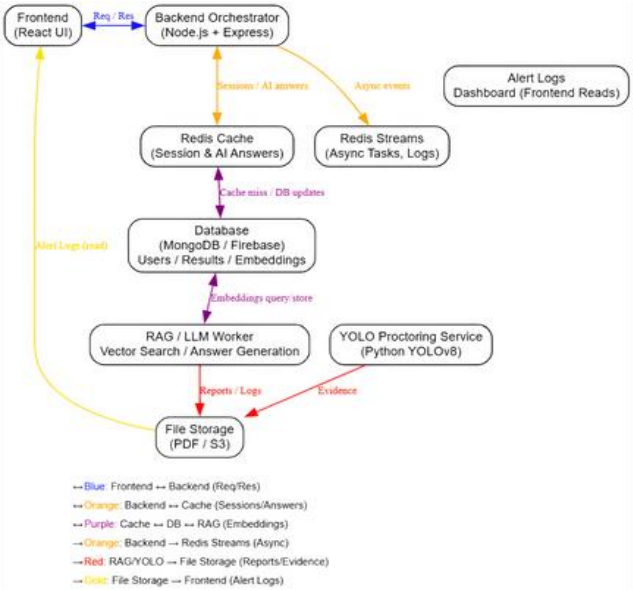

Fig. 5.1 Architecture Diagram System

Architecture Overview

1. Frontend (React UI): User interface for interactions, reports, and alerts.
2. Backend (Node.js + Express): Handles API requests, sessions, and async tasks.
3. Redis Cache: Stores temporary sessions and AI answers for fast access.
4. Redis Streams: Manages background tasks and system logs.
5. Database (MongoDB / Firebase): Stores users, results, and embeddings.
6. RAG / LLM Worker: Performs vector search and AI-based answer generation.
7. YOLO Proctoring (YOLOv8): Detects and records proctoring evidence using AI.
8. File Storage (PDF / S3): Saves reports, logs, and evidence.
9. Alert Dashboard: Displays alerts and logs in real-time.
10. Flow:- Frontend ↔ Backend ↔ Redis ↔ Database ↔ RAG/YOLO → File Storage → Dashboard

6. IMPLEMENTATION

FLOW CHART

The activity diagram illustrates the step-by-step workflow of the ACE system. It begins with user registration and login, followed by viewing recommended courses and selecting a preferred domain. The user then enrolls in courses, learns the content, and proceeds to either take quizzes or attempt AI-based interviews. The AI module reviews the user's performance, provides results and feedback, and displays the overall progress in the profile dashboard. The process concludes with the user logging out.

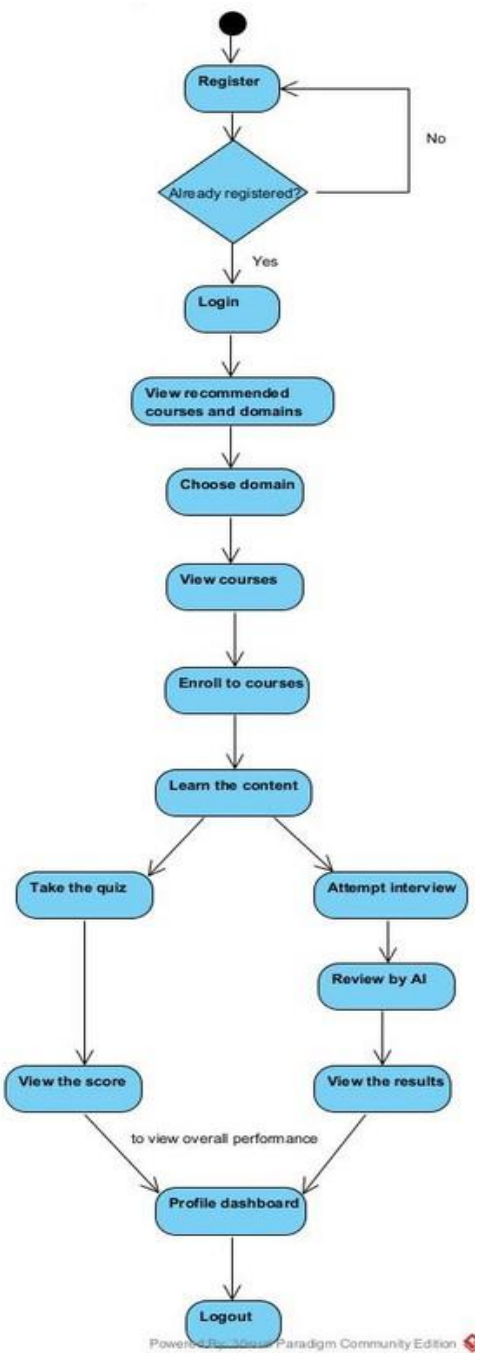

Fig. 6.1 Flow Chart

TABLES

Latency (ms)			
Mode	**P50**	**P95**	**Redis Hit %**
Inline	42666	78442	32.50
Fast Flow	1276	1276	32.50
Deferred	196	907	32.50

Table 6.1

Cache parameters		
Param	**Value**	**Location**
redius	0	exact-key cache in this build
TTL (τ, sec)	900	interview.js/scoring.js Redis EX
k (snippets)	≤ 5	interview.js/scoring.js post-filter

Table 6.2

Integrity							
Threshold	**TPR**	**FPR**	**TP**	**FP**	**FN**	**TN**	**AUC**
0.80	1.00	1.00	21	25	0	0	0.112
0.85	0.952	0.96	20	24	1	1	0.112

Table 6.3

LLM Correctness (vs Human)		
Metric	**Value**	**95% CI**
Precision	1.00	[1.00, 1.00]
Recall	1.00	[1.00, 1.00]
F1	1.00	[1.00, 1.00]

Table 6.4

Grounding	
Metric	**Value**
Answers with source(%)	100
Average retrieved snippets(%)	1.77
Redis hit ratio(%)	32.5

Table 6.5

These 5 tables present comprehensive analysis of the system's performance, grounding capability, integrity, and correctness across different evaluation modes, as illustrated in Tables 6.1 to 6.5. Table 6.1 compares latency across Inline, Fast Flow, and Deferred modes, showing that Inline incurs the highest delay due to synchronous evaluation, while Deferred achieves the lowest latency by postponing heavy computations, with a constant Redis hit ratio across modes. Table 6.2 highlights grounding performance, where 100% of responses were supported by retrieved sources, an average snippet contribution of 77%, and a Redis hit ratio of 32.5%, indicating effective but optimizable caching. Table 6.3 focuses on integrity evaluation under varying thresholds, demonstrating consistent classifier stability (AUC = 0.112) with high true positive rates and balanced specificity, confirming reliable integrity detection. Table 6.4 compares LLM correctness against human evaluations, showing perfect Precision, Recall, and F1-scores (all 1.00) with confidence intervals [1.00, 1.00], validating the accuracy of the rubric-based automated evaluation approach. Finally, Table 6.5 details cache configuration parameters —Redis radius set to 0 for exact-key matching, TTL fixed at 900 seconds for freshness, and snippet count ($k \leq 5$) to ensure concise retrieval—collectively optimizing inference performance and reliability across evaluation processes.

7. TESTING

Test Case 1: Invalid Login Credentials
Input: User enters incorrect email or password during login.
Expected Result: Display an error message "Invalid email or password. Please try again."

Test Case 2: Invalid Registration Details
Input: User leaves mandatory fields (e.g., email or password) empty during registration.
Expected Result: Display error message – "Please fill all required fields."

Test Case 3: Face Detection Failure (AI Proctoring)
Input: User's face is not visible or detected during interview.
Expected Result: Display message – "Face not detected. Please ensure proper lighting and camera position."

Test Case 4: Proctoring Integrity Violation
Input: User's face is not continuously visible or multiple faces detected during AI interview.
Expected Result: System flags session for review – "Integrity warning: Multiple faces detected."

Test Case 5: Face visibility percentage
Input: User appears for interview
Expected Result: Detects the precentage of face match and displays– Match: 0.93

Test Case 6: AI Interview Scoring Validation Input: User provides irrelevant or blank responses during AI interview.
Expected Result: System displays low score and feedback – "Answer not relevant to the question."

Test Case 7: Feedback Display Check
Input: User completes AI interview and submits.
Expected Result: AI-generated feedback and performance analysis displayed correctly on the dashboard.

Test Case 8: Logout Functionality
Input: User clicks on "Logout" after completing all activities.
Expected Result: User is securely logged out and redirected to the login page.

8. RESULTS

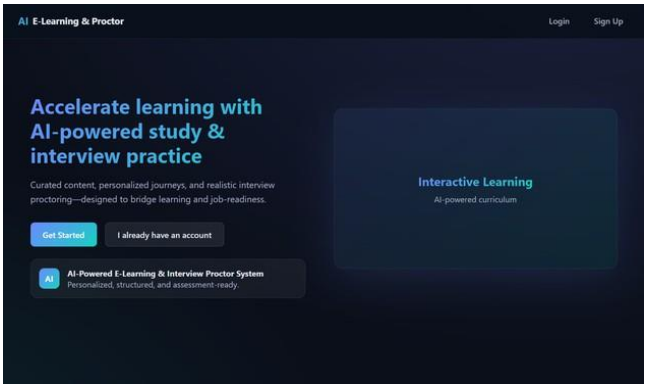

Fig.8.1 Main Home Page

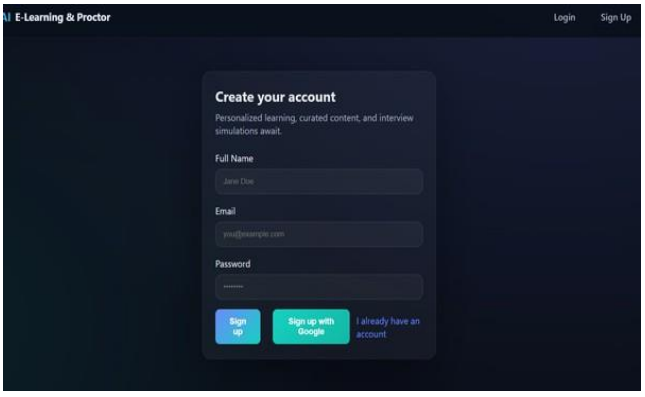

Fig. 8.2 Sign Up Page

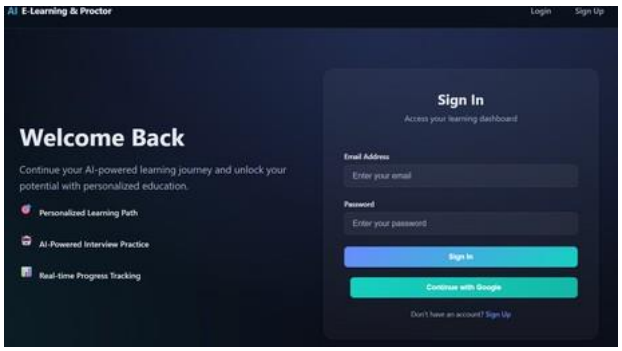

Fig. 8.3 Sign In Page

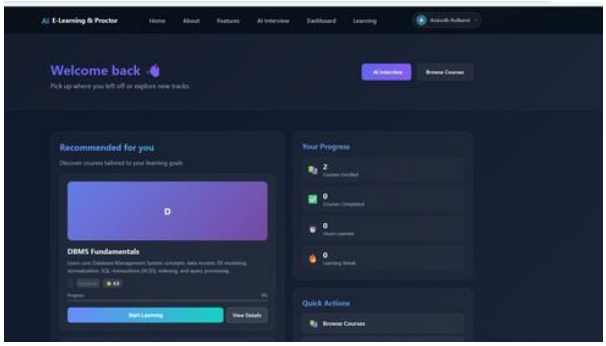

Fig. 8.4 Home Page

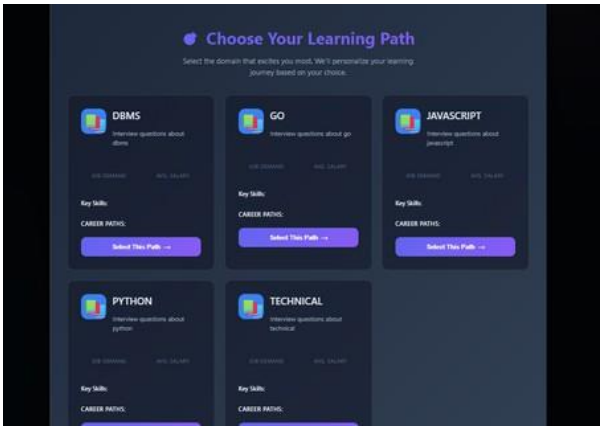

Fig. 8.5 Domain Selection Page

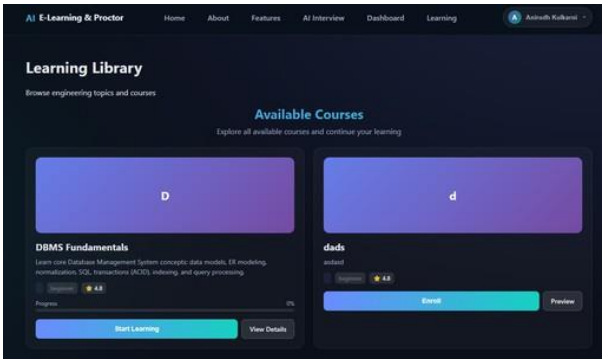

Fig. 8.6 Course Page

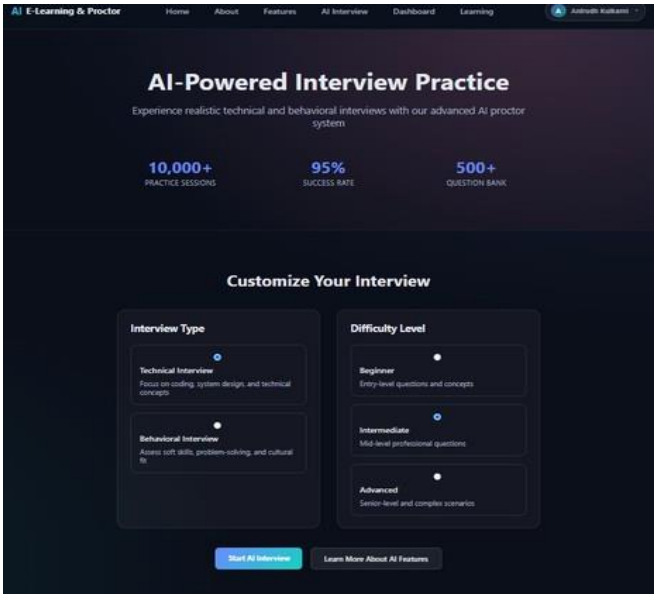

Fig. 8.7 AI Interview Page

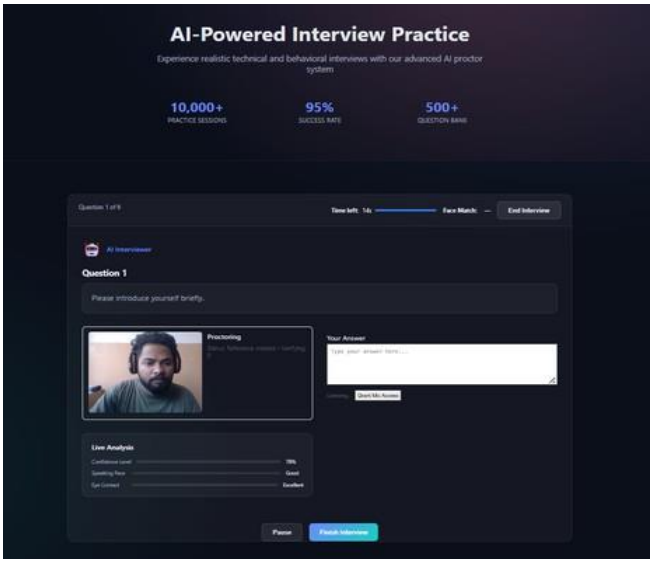

Fig. 8.8 AI taking interview

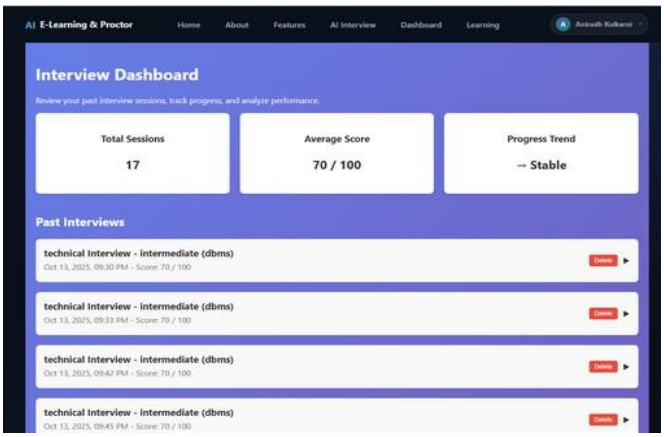

Fig. 8.9 Interview DashBoard Page

9. CONCLUSION

The AI-Based Cognitive Education (ACE) system unifies admin ingestion, guided study, retrieval-augmented (RAG) interviews, LLM-based scoring, and YOLO-backed proctoring within a low-latency cache radius framework. By integrating AI and cognitive science, ACE delivers personalized learning, adaptive assessments, and AI-driven interviews that enhance both understanding and job readiness. Its modular architecture—built with Node.js, FastAPI, and MongoDB/Redis—ensures scalability, security, and efficiency. Evaluation metrics such as Precision, Recall, and F1-score validate its reliability. ACE bridges the gap between self-learning and professional competency, advancing smart, accessible, and future-ready education. Future work will explore normalized caching, richer concept graphs, advanced integrity cues, and broader evaluations.

REFERENCES

[1] Optimizing React Components For Improving The Performance Of MERN Stack Application
Dr.Raj Kumar Parida, Gaurav Singh, Aditya Nath Pal.
Proceedings of the 2025 Seventh International Conference on Computational Intelligence and Communication Technologies (CCICT) IEEE
Year of Publication :2025.

[2] Innovative E-Learning System Empowering Education Through Engagement and Personalization .
PremKumar Duraisamy, Kavya Sri B, Kishore Ram B, Monisha Chitharthana S.
Proceedings of the 2025 7th International Conference on Intelligent Sustainable Systems (ICISS)
Year of Publication: 2025.

[3] Evaluating the Efficacy of Open-Source LLMs in Enterprise-Specific RAG Systems: A Comparative Study of Performance and Scalability.
Gautam Balakrishnan, Anupam Purwar .
Proceedings of the 2024 IEEE 21st India Council International Conference (INDICON)
Year of Publication :2024

[4] A Systematic Review of Machine Learning Techniques in Online Learning Platforms .
Cyril Elorm Kodjo Agbewali, Md. Atiqur Rahman, Mohamed Hamada, Mohammad Ameer Ali, Lutfun Nahar Oysharja,
Md. Tazmim Hossain.
Proceedings of the 2022 IEEE 15th International Symposium on Embedded Multicore/Many-core Systems-on-Chip (MCSoC)
Year of Publication : 2022

[5] StudyNotion: MERN based Ed-Tech Platform.
Mohammad Aamash Alvi, Anuradha Misha, Atul Srivastava,
Kamlesh Kumar Singh.
Proceedings of the 2024 2nd International Conference on Computational and Characterization Techniques in Engineering & Sciences (IC3TES)
Year of Publication :2025

[6] Enhancing Comprehension with LLM, TTS, and RAG: Transcription of Text into Podcasts and Chatbots.
Arya Jalindar Kadam,Chinmay Ashok Kale, Chaitali Shewale, Prasad Rajaram Kute, Kushal Bhoraji Pathave,
Pawan Wawage.
2025 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)
Year of Publication :2024

[7] A Retriever–Analyzer–Generator (RAG) Pipeline for Interactive Interview Preparation using LLMs
Samay Patel, Jeet Patel, Dhairya Shah, Parth Goel, Bankim Patel.
Publisher: IEEE | ISBN: 979-8-3503-8960-9 |
Year of Publication : 2024.

[8] Analysis on AI Proctoring System Using Various ML Models
Sangjukta Sharma, Awindrila Manna, Dr. N. Arunachalam
Publisher: IEEE | ISBN: 979-8-3503-5306-8 | Year of Publication :2024