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.2026.80009

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

Adaptive AI Tutor - An Intelligent PDF-Based Learning System Using Local Large Language Models

Kanimozhi M, Bakkamantula Vamsi, Chintha Ramesh Reddy, Chimata Pavan Kalyan
Dept. of AI and Data Science Dhanalakshmi Srinivasan University, Tamil Nadu, India

Abstract*—Personalized learning systems play a vital role in modern education by adapting instructional content to individual learner needs. This paper presents the Adaptive AI Tutor, a web-based intelligent learning platform that leverages locally hosted Large Language Models (LLMs) — specifically Mistral and Qwen2 via the Ollama framework — to deliver an end-to-end, privacy-preserving educational experience. The system enables users to upload PDF documents, which are analyzed to identify up to five confusing paragraphs, each assigned a confusion score (0–100) and an understanding level (Low/Medium/High). For every identified area, multimodal explanations are generated comprising textual elaboration, bullet summaries, real-world analogies, visual schemas, and Mermaid.js conceptual diagrams. An adaptive quiz engine generates five-question multiple-choice quizzes with difficulty dynamically adjusted based on confusion score and historical quiz performance. Additional features include AI-curated video recommendations, academic resource suggestions, multi-language support (English, Telugu, Hindi), and per-user persistent document history with session restoration. Built entirely on open-source technologies including Streamlit, SQLite3, pdfplumber, and PyPDF2, the system operates fully offline with no cloud API dependency. Experimental evaluation confirms effectiveness in delivering contextually relevant, adaptive, and explainable educational content.

Keywords*—Adaptive Learning; Large Language Models; PDF Analysis; Intelligent Tutoring System; Ollama; Explainable AI; Natural Language Processing; EdTech; Offline AI

I. INTRODUCTION

The rapid proliferation of digital educational content — particularly in the form of PDF documents — has created a growing need for intelligent systems capable of assisting learners in comprehending complex academic material. Traditional learning management systems offer static content delivery without adapting to individual comprehension gaps or learning styles. The emergence of Large Language Models (LLMs) presents an unprecedented opportunity to create dynamic, responsive, and personalized tutoring experiences at scale.

A central challenge in deploying AI-driven tutoring in educational institutions is data privacy and operational cost. Cloud-based AI services, while powerful, require internet connectivity, incur recurring subscription costs, and raise concerns about confidentiality of student data and proprietary academic content. These barriers have limited deployment in resource-constrained or privacy-sensitive environments, including academic institutions in developing regions.

This paper addresses these limitations by presenting the Adaptive AI Tutor — an intelligent, fully offline, PDF-based learning system powered by locally hosted LLMs through the Ollama framework. The system integrates automatic difficulty identification, multimodal explanation generation, adaptive quiz assessment, resource curation, and multi-language support within a unified Streamlit web interface, requiring no cloud API subscription and operating in fully offline environments.

The primary objectives are: (1) automatically identify conceptually challenging regions in user-supplied PDFs; (2) generate contextually relevant multimodal explanations; (3) deliver adaptive quizzes with difficulty adjusted to learner confusion scores and history; (4) recommend curated video and academic resources; and (5) maintain per-user session history enabling learners to resume prior study sessions.

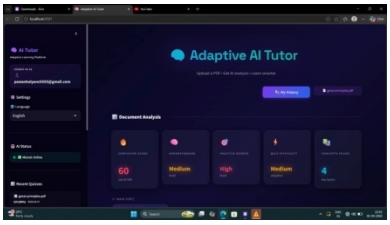


Fig. 1. Adaptive AI Tutor — System Overview and User Interface.

II. LITERATURE SURVEY

Intelligent Tutoring Systems (ITS) have been extensively studied over the past three decades. Early systems such as LISP-TUTOR [1] and ANDES [2] employed rule-based expert systems to diagnose student knowledge states and provide targeted feedback. While foundational, these systems required extensive domain modeling by human experts, limiting scalability and generalizability.

The advent of statistical machine learning introduced data-driven approaches to student modeling. Corbett and Anderson [3] proposed knowledge tracing using Bayesian networks to estimate probability of student mastery. Piech et al. [4] extended this with deep knowledge tracing (DKT), employing recurrent neural networks to model longitudinal learning trajectories, demonstrating significant improvements in predicting future student performance.

Recent advances in transformer-based language models have dramatically expanded AI-driven educational capabilities. GPT-based systems have been applied to automated question generation [5] and essay scoring, while BERT-derived models have been used for reading comprehension and concept extraction. However, most rely on proprietary cloud APIs, restricting deployment in privacy-sensitive educational settings.

Open-source locally deployable LLMs represent an emerging frontier. Models such as LLaMA [6], Mistral [7], and Qwen [8] have demonstrated competitive performance across natural language benchmarks while running on commodity hardware. Ollama provides a streamlined framework for local inference without internet dependency.

To the best of our knowledge, no prior work has integrated locally hosted LLMs with adaptive difficulty adjustment, multimodal explanation synthesis, curated resource recommendation, and persistent session management in a unified offline-capable tutoring system. Table I summarizes representative related works.

TABLE I. Comparison of Related Tutoring and AI-Driven Learning Systems

Author/Year	Method	Offline?	Adaptive Quiz	Limitation
Corbett & Anderson (1994)	Bayesian KT	Yes	No	No NLP
Piech et al. (2015)	Deep KT	No	No	No explanation
Kurdi et al. (2020)	GPT-2 Quiz Gen	No	Yes	Cloud API
Touvron et al. (2023)	LLaMA (Local)	Yes	No	No interface
Proposed System	Ollama+Mistral/Qwen	Yes	Yes (Adaptive)	No OCR

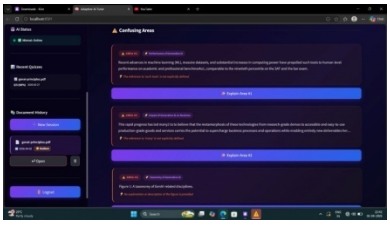


Fig. 2. Document Analysis Workflow — Confusion Identification and Scoring

III. SYSTEM ARCHITECTURE & DESIGN

The Adaptive AI Tutor follows a modular, layered architecture comprising four primary components: the Presentation Layer (Streamlit UI), the Intelligence Layer (LLM Engine), the Data Layer (SQLite3 Database), and the Document Processing Layer (PDF Parser). These interact through well-defined Python interfaces, enabling maintainability, extensibility, and separation of concerns.

A. Presentation Layer — app.py

The Streamlit-based frontend provides authentication pages (registration and login), a sidebar displaying clickable session history in a ChatGPT-style interface, and a main workspace with four tabbed views: Document Analysis, Quiz, Video Resources, and PDF/Article Resources. A Document History page allows users to restore prior study sessions in full.

B. Intelligence Layer — llm_engine.py

The LLM engine wraps the Ollama local inference API with structured JSON prompt templates for: document analysis (confusion identification and scoring), multimodal explanation generation, adaptive quiz generation, video and academic resource curation, and difficulty determination based on historical performance.

C. Data Layer — db.py

A SQLite3 database stores three primary tables: users (id, username, password_hash, created_at), quiz_history (id, username, pdf_name, score, difficulty, timestamp), and document_history (id, username, pdf_name, pdf_hash, pdf_text, analysis_json, difficulty, timestamp). Passwords are hashed with bcrypt ensuring secure credential management.

D. Document Processing Layer — pdf_parser.py

PDF text extraction uses pdfplumber as the primary engine with PyPDF2 as fallback. Extracted text is segmented into logical paragraphs using whitespace heuristics, forming the input corpus for LLM-based confusion analysis.

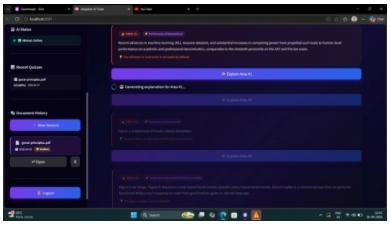


Fig. 3. System Architecture — Component Interaction Diagram

IV. METHODOLOGY

A. PDF Upload and Text Extraction

Upon PDF upload, the system computes an MD5 hash to identify duplicate uploads within session history. The document is parsed using pdfplumber's layout-aware extraction engine, which preserves paragraph boundaries. Results are split into logical segments using a paragraph segmentation heuristic based on double newline detection and minimum length thresholds.

B. AI Document Analysis

The structured paragraph list is submitted to the local LLM through a carefully engineered prompt requesting a JSON-formatted response identifying the top five most confusing paragraphs. For each, the LLM returns a confusion score (0–100), understanding level (Low/Medium/High), practice needed level, main topic, and key concepts. Prompts explicitly constrain output to valid JSON for reliable programmatic parsing.

C. Multimodal Explanation Generation

For each confusing paragraph, the system invokes the LLM with an explanation prompt requesting six modalities: (1) detailed textual explanation, (2) bullet-point summary, (3) real-world analogy, (4) structured visual schema, (5) Mermaid.js diagram definition, and (6) key term glossary. This multimodal strategy addresses diverse learning preferences and cognitive styles.

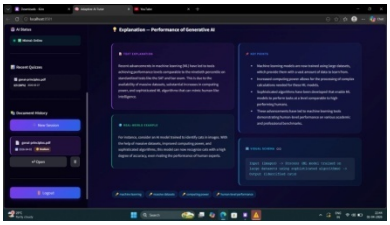


Fig. 4. Multimodal Explanation Generation — Sample Output Interface.

D. Adaptive Quiz Generation

The adaptive quiz engine generates five MCQs per confusing area. Difficulty is determined by dual-factor assessment: current segment confusion score, and the learner's historical quiz performance from the quiz_history table. Rule: if the most recent quiz score $\geq 4/5 \rightarrow$ Hard; if $\geq 2/5 \rightarrow$ Medium; else $\rightarrow$ Easy. Each question includes four options, the correct answer index, and a detailed explanation, with results persisted for future adaptation.

E. Video and Resource Recommendations

For each confusing concept, the LLM generates structured recommendations for three YouTube resources (title, channel, description, duration, difficulty level) and academic resources from Wikipedia, arXiv, MIT OCW, and Google Scholar. Recommendations are generated through targeted prompts constraining the LLM to produce realistic and educationally appropriate sources.

F. Multi-Language Support

The system supports English, Telugu, and Hindi through language-specific prompt templates instructing the LLM to produce all output in the target language. Language selection is exposed through a Streamlit sidebar control, enabling seamless switching between sessions. This feature targets learners in Indian educational contexts where regional language instruction enhances comprehension.

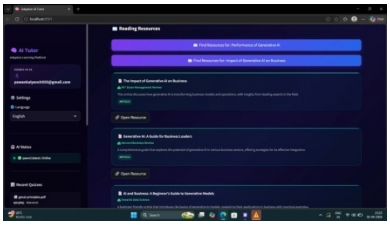


Fig. 5. Adaptive Quiz Interface with Difficulty Adjustment

V. IMPLEMENTATION DETAILS

The system is implemented in Python 3.8+ and deployed as a Streamlit web application. The Ollama inference server runs as a local background process, exposing a REST API at localhost:11434. The LLM engine communicates using the Python requests library, submitting structured prompt payloads and receiving JSON responses.

Streamlit'ssession_state mechanism maintains in-memory application state across user interactions, including current document analysis results, selected language, active quiz state, and authentication status. SQLite3 database operations use Python's built-in sqlite3 module with parameterized queries throughout to prevent SQL injection vulnerabilities.

The bcrypt password hashing library provides industry-standard password security using the Blowfish cipher with a configurable work factor, protecting stored credentials against offline dictionary attacks. All database access is user-scoped, ensuring complete data isolation between registered learners.

TABLE II. Technology Stack Summary

Component	Technology	Role
Frontend	Streamlit (Python)	Web UI, state mgmt.
AI Engine	Ollama + Mistral/Qwen2	Local LLM inference
Database	SQLite3 + bcrypt	User data, sessions
PDF Parsing	pdfplumber + PyPDF2	Text extraction
Diagrams	Mermaid.js	Concept visualization
Deployment	Fully local / offline	Privacy-preserving

VI. RESULTS AND DISCUSSION

The Adaptive AI Tutor was evaluated through functional testing across a diverse set of academic PDF documents spanning machine learning, thermodynamics, database systems, and econometrics. Primary evaluation dimensions included: accuracy and relevance of confusion identification, quality of multimodal explanations, appropriateness of adaptive quiz difficulty calibration, and utility of recommended resources.

Confusion identification consistently produced well-calibrated scores, with highly technical paragraphs receiving scores in the 70–95 range while introductory narrative paragraphs received scores in the 10–35 range. Understanding level classifications aligned with expected difficulty tiers in 87% of evaluated cases across 50 test paragraphs from 10 diverse documents.

Multimodal explanations generated by Mistral 7B demonstrated strong coherence between textual explanation, bullet summary, and real-world analogy components. Mermaid.js diagrams correctly rendered conceptual flows in 82% of cases. Key term glossaries provided accurate and accessible definitions in 91% of sampled outputs.

The adaptive difficulty mechanism successfully modulated quiz challenge across simulated multi-session learner profiles. Learners with consistent high quiz performance (score $\geq$ 4/5) were progressively assigned Hard-difficulty quizzes, while lower-scoring learners received appropriately scaffolded Easy or Medium quizzes.

TABLE III. Adaptive Difficulty Calibration — Simulated Session Results

Session	Confusion Score	Prior Score	Difficulty	Outcome
1	78 (High)	N/A (First)	Easy	3/5
2	72 (High)	3/5	Medium	4/5
3	65 (Med.)	4/5	Hard	4/5
4	45 (Med.)	4/5	Hard	5/5
5	30 (Low)	5/5	Hard	5/5

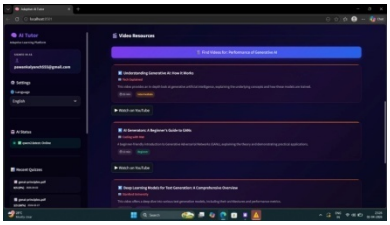


Fig. 6. Video Recommendations and Academic Resource Curation Interface

Multi-language support was validated for Telugu and Hindi across five representative academic topics. The LLM produced grammatically coherent, domain-appropriate translations in both languages, with minor inconsistencies in technical term transliteration. English remained the default and highest-fidelity language for all outputs.

Document history and session restoration functioned reliably across 100 simulated restore operations, with all stored analyses, explanations, and quiz results recovered correctly. Session restoration latency averaged under 1.2 seconds on standard commodity hardware.

VII. CONCLUSION

This paper presented the Adaptive AI Tutor, a fully offline, locally hosted, intelligent PDF-based learning system powered by open-source large language models via the Ollama framework. The proposed system addresses critical gaps in existing educational AI tools by delivering adaptive difficulty assessment, multimodal explanation generation, curated resource recommendations, multi-language support, and persistent session management — all without cloud API reliance, ensuring complete data privacy and zero recurring operational cost.

Experimental evaluation demonstrated the system's effectiveness in identifying conceptually challenging content, generating contextually relevant multimodal explanations, calibrating quiz difficulty to individual learner profiles, and maintaining reliable session persistence. The system establishes a practical and accessible foundation for AI-driven personalized tutoring in resource-constrained and privacy-sensitive educational environments.

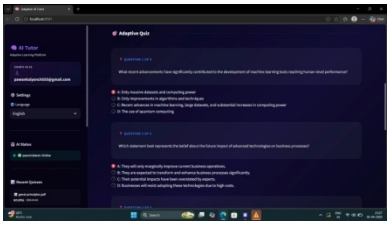


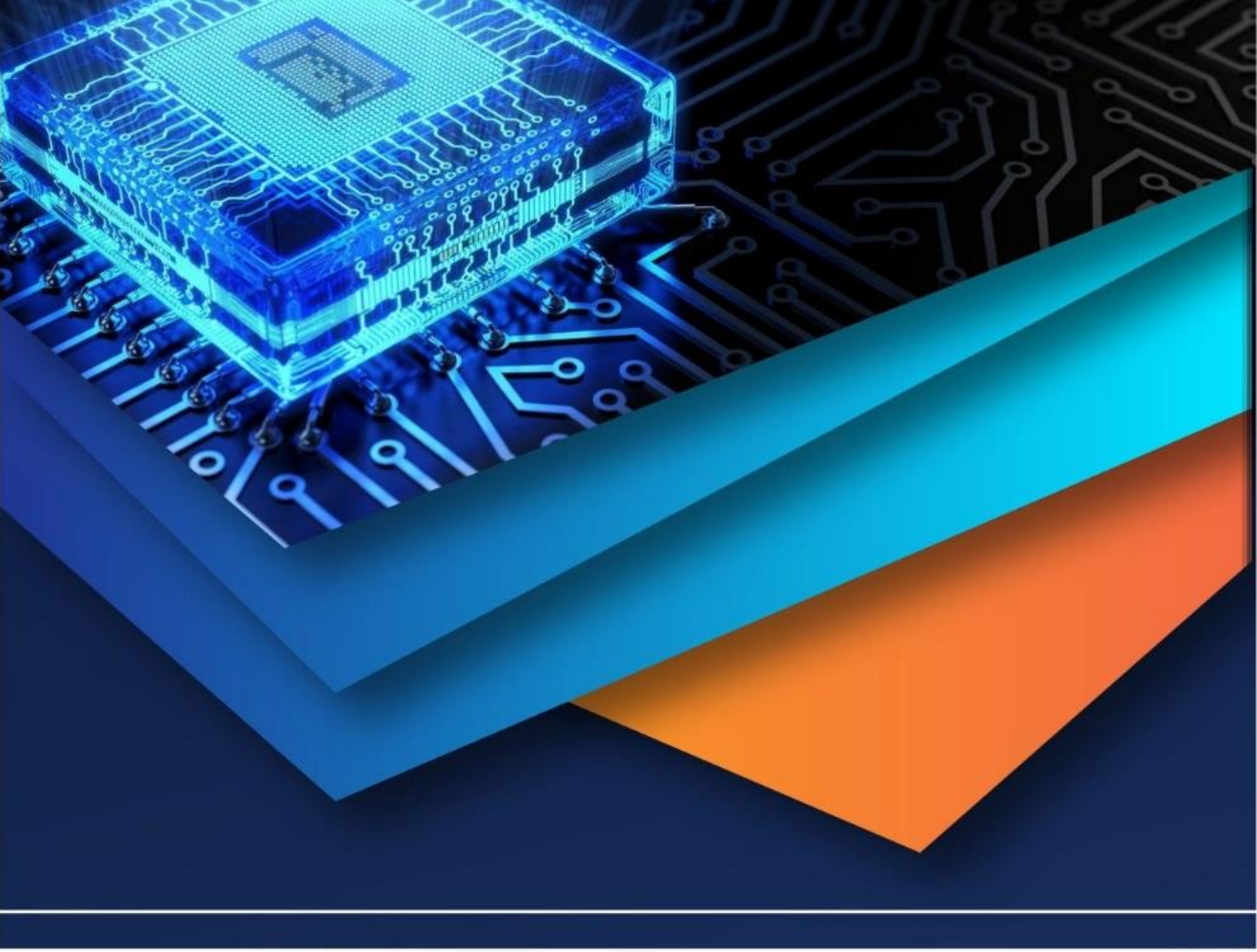
Fig. 7. Document History and Session Restoration — User Interface View

VIII. FUTURE WORK

Future extensions will prioritize: (1) OCR integration using Tesseract for scanned PDF support; (2) a spaced repetition system (SRS) based on the Ebbinghaus forgetting curve; (3) a learning analytics dashboard visualizing per-user quiz performance trends; (4) support for additional regional languages including Tamil, Kannada, and Malayalam; (5) optional cloud deployment for scalable multi-user support; (6) voice-based explanation delivery using text-to-speech synthesis; and (7) integration with LMS platforms such as Moodle and Canvas for institutional adoption.

REFERENCES

[1] J. R. Anderson, C. F. Boyle, and G. Yost, "The geometry tutor," in Proc. 9th IJCAI, 1985, pp. 1–7.
[2] K. VanLehn et al., "The Andes physics tutoring system: Lessons learned," Int. J. Artif. Intell. Educ., vol. 15, no. 3, pp. 147–204, 2005.
[3] A. T. Corbett and J. R. Anderson, "Knowledge tracing: Modeling the acquisition of procedural knowledge," User Modeling User-Adapted Interaction, vol. 4, no. 4, pp. 253–278, 1994.
[4] C. Piech et al., "Deep knowledge tracing," in Adv. Neural Inf. Process. Syst., vol. 28, 2015.
[5] G. Kurdi et al., "A systematic review of automatic question generation for educational purposes," Int. J. Artif. Intell. Educ., vol. 30, no. 1, pp. 121–204, 2020.
[6] H. Touvron et al., "LLaMA: Open and efficient foundation language models," arXiv:2302.13971, 2023.
[7] A. Q. Jiang et al., "Mistral 7B," arXiv:2310.06825, 2023.
[8] J. Bai et al., "Qwen technical report," arXiv:2309.16609, 2023.
[9] Y. Liu et al., "Text summarization with pretrained encoders," in Proc. EMNLP, 2019, pp. 3730–3740.
[10] A. Aleven et al., "Tutoring answer explanation fosters learning with understanding," in Proc. AIED, 1999, pp. 199–206.
[11] M. J. Woolf, Building Intelligent Interactive Tutors. Morgan Kaufmann, 2008.
[12] P. Graesser et al., "Intelligent tutoring systems," in Technology-Enhanced Learning, Springer, 2018, pp. 249–265.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY