International Journal of Scientific Research & Engineering Trends
Volume 12, Issue 2, Mar-Apr-2026, ISSN (Online): 2395-566X

# MentorAI: A Smart Web-Based Learning Assistant with Personalized Guidance and Interactive Study Support

**Dadarkar Ehaan Mubasshir, Ansari Ayan Atif Masood Iqbal, Khan Zain MD Irfan**
Diploma In Computer Engineering Students, Dept. Of Computer Science & Engineering, Aiarkp, Panvel.

**Abstract- MentorAI is a comprehensive, web-based intelligent learning assistant designed to transform how students study, retain knowledge, and engage with educational content. The platform integrates a personalized AI Tutor powered by large language models, a Voice Recall system for active retrieval practice using the Web Speech API, an SM-2 algorithm-driven Spaced Repetition Flashcard engine, an AI-generated adaptive Quiz Engine, an AI- assisted rich-text Workspace for notes and PDF imports, a 3D Knowledge Graph for visual concept mapping using D3.js force-directed visualization, and a command-palette-style Nexus navigation system. A central Dashboard aggregates learning metrics, study streaks, mastery scores, and AI-detected weak topics in real time. This paper presents the complete system architecture, feature design rationale, technology stack, database design, security model, testing methodology, and results achieved during the development of MentorAI as a final year engineering capstone project.**

**Keywords – Personalized Learning, AI Tutor, Spaced Repetition, SM-2 Algorithm, Voice Recall, Web Speech API, Knowledge Graph, D3.js, Adaptive Quiz Engine, Flashcard System, Large Language Models, Retrieval Practice, Study Analytics, EdTech.**

## I. INTRODUCTION

The modern educational landscape is undergoing a profound transformation, driven by advances in artificial intelligence, natural language processing, and interactive web technologies. Students today face not only an ever- expanding volume of content to master but also a persistent challenge in retaining and applying that knowledge effectively. Conventional study methods — passive re-reading, static flashcards, and linear note-taking — are increasingly inadequate in meeting the demands of personalized, evidence-based learning.

Research in cognitive science has consistently demonstrated that active retrieval practice, spaced repetition, and self-explanation are among the most effective strategies for long-term memory retention. Yet the majority of digital learning tools remain fragmented: one application for flashcards, another for quizzes, a third for note-taking, and yet another for AI-assisted tutoring. This fragmentation forces learners to manage multiple platforms, losing contextual continuity and the compounding benefits that arise from a unified, data-driven learning ecosystem.

MentorAI is conceived as a direct response to these challenges. It is an intelligent, web-based learning assistant that integrates seven tightly coupled study modules — AI Tutor, Voice Recall, Flashcards, Quiz Engine, Workspace, Knowledge Graph, and a Dashboard — into a single cohesive platform. Each module feeds data into the others: quiz errors become flashcards, voice recall gaps surface in the AI Tutor, and all study interactions update a dynamic knowledge graph and mastery dashboard. The result is a platform that learns about the learner and continuously adapts to their evolving knowledge state.

This paper is organized as follows: Section II defines the problem statement. Section III lists the project objectives. Section IV reviews relevant literature. Section V describes the system architecture. Section VI details each feature. Section VII covers the technology stack. Section VIII discusses the database design. Section IX outlines security measures. Section X describes the testing approach. Section XI presents results, and Section XII concludes the paper.

MentorAI was developed by a team of final-year Diploma in Computer Engineering students over approximately five months using an Agile methodology with two-week sprint cycles, peer code reviews, and iterative user feedback. The platform is fully browser-based, requiring no local installation beyond a modern web browser, making it immediately accessible to students across all operating systems and device configurations.

## II. PROBLEM STATEMENT

Despite the proliferation of educational technology tools, several critical gaps persist for student learners seeking effective, personalized study support:

First, existing learning platforms lack deep personalization. Tools such as Quizlet or Anki provide flashcard functionality

but do not analyse a learner's specific conceptual weaknesses across multiple study modes and route that intelligence back into a unified adaptive experience. The result is a static, one-size-fits-all study cycle that does not respond to individual learning gaps.

Second, active retrieval practice — one of the most evidence-backed learning techniques — is almost entirely absent from mainstream study tools. While spaced repetition applications exist for passive flashcard review, no widely available platform encourages students to verbally articulate and explain concepts aloud, then receive AI-generated feedback on the accuracy and completeness of their spoken answers.
Third, the knowledge state of a learner is never visualised holistically. Students rarely have a clear, interconnected map of what they know, what they are learning, and what remains poorly understood. This absence of a knowledge topology makes it difficult to prioritize study efforts and identify hidden conceptual dependencies.

Fourth, AI-assisted tutoring, when available, is typically disconnected from the learner's study history. A chatbot that cannot see a student's quiz errors, flashcard performance, or voice recall gaps cannot provide genuinely personalized guidance — it can only respond to the immediate prompt.

MentorAI addresses all four of these gaps by providing a unified, browser-based platform where every study interaction — quiz attempt, flashcard review, voice recall session, or note creation — contributes to a shared intelligence layer that personalizes every subsequent interaction across all modules.

## III. OBJECTIVES

The primary objectives of the MentorAI project are: (i) To design and implement a full-stack web application providing an AI-powered conversational tutor with subject-specific context and session memory; (ii) To build a Voice Recall module that uses the Web Speech API to transcribe spoken answers and employ large language models to evaluate open-ended responses against key concepts; (iii) To implement a Spaced Repetition Flashcard system powered by the SM-2 algorithm for scientifically optimized memory retention scheduling; (iv) To develop an adaptive Quiz Engine that generates AI-powered multiple-choice questions with intelligent distractors and feeds wrong-answer data back into the platform's weak topic detection system; (v) To create an AI-assisted Workspace with rich-text editing, PDF import and extraction, and one-click AI content generation tools including summarization, quiz generation, and flashcard creation;

(vi) To build a 3D Knowledge Graph using D3.js force-directed layout that visually maps all studied concepts, their mastery levels, study frequencies, and AI-generated relationships; (vii) To implement a real-time Dashboard aggregating study streaks, mastery metrics, a 7-day progress chart, AI-detected weak topics, and a recent activity feed; (viii) To develop a keyboard-first command palette (Nexus) enabling fuzzy search navigation across all platform actions and pages; and (ix) To ensure cross-module data flow so that performance data from quizzes, voice recall, and flashcards automatically enriches the AI Tutor context, dashboard analytics, and knowledge graph.

Success criteria for the project were defined as: (a) all 8 planned modules must be in a fully functional state; (b) the AI Tutor must respond within 500ms for typical queries; (c) the Voice Recall transcription must achieve at least 85% word accuracy under standard microphone conditions; (d) the SM-2 flashcard algorithm must correctly calculate review intervals across all three grading tiers; (e) user acceptance testing must achieve an average satisfaction score of 4.0 or above on a 5-point Likert scale; and (f) the Knowledge Graph must correctly render a minimum of 50 concept nodes with accurate edge relationships.

## IV. LITERATURE REVIEW

### A. Spaced Repetition and Memory Science

The spacing effect, first described by Ebbinghaus [1] in his seminal work on the forgetting curve, established that information retained over spaced intervals is remembered far more durably than information reviewed in massed practice. Wozniak and Gorzelańczyk [2] formalized this insight into the SM-2 algorithm, which forms the theoretical foundation for applications such as Anki and SuperMemo. MentorAI's Flashcard module implements the SM-2 algorithm directly, personalizing review intervals based on each card's ease factor and the learner's grading performance.

### B. Active Retrieval Practice

Roediger and Karpicke [3] demonstrated through controlled experiments that testing — forcing learners to retrieve information from memory — produces significantly greater long-term retention than re-studying the same material. This "testing effect" is the cognitive science foundation for MentorAI's Voice Recall module, which requires students to articulate answers aloud before receiving AI feedback, rather than passively reviewing answers.

### C. AI-Assisted Tutoring Systems

Intelligent Tutoring Systems (ITS) have been an active research area since the 1970s. VanLehn [4] provides a comprehensive review of ITS effectiveness, concluding that systems providing step-level adaptivity achieve learning gains approaching one-on-one human tutoring. The proliferation of large language models (LLMs) such as GPT-4 and Llama-3 has created new possibilities for conversational tutoring with broad subject coverage. Chi et al. [5] showed that self-explanation — prompting learners to explain concepts in their own words — produces deeper comprehension than passive study, a principle

directly implemented in MentorAI's AI Tutor follow-up question generation.

### D. Knowledge Graphs in Education

Knowledge graphs have gained significant attention in educational contexts as tools for representing conceptual relationships and visualizing learning progress. Hoppe and Miatidis [6] demonstrated that visual knowledge mapping improves metacognitive awareness — students' understanding of what they know and don't know. MentorAI's Knowledge Graph adapts this research insight into an interactive D3.js force-directed network where node colour encodes mastery level and edge labels encode concept relationships.

### E. Voice and Speech Interfaces in Learning

The Web Speech API, standardized by the W3C, provides browser-native speech recognition capabilities without requiring third-party dependencies. Research by Shadiev et al. [7] found that speech-based interaction in learning environments increases engagement and recall compared to text-only interfaces. MentorAI's Voice Recall module leverages the Web Speech API for real-time transcription with a live animated waveform, creating an engaging and frictionless recall experience.

### F. Adaptive Assessment Systems

Adaptive testing frameworks, as reviewed by Weiss and Kingsbury [8], dynamically adjust question difficulty based on learner performance to maximize measurement precision and learning efficiency. MentorAI's Quiz Engine implements a simplified form of adaptivity: wrong answers are tagged with conceptual labels, added to the platform's weak topic registry, and used to bias subsequent quiz and AI Tutor session content toward areas of demonstrated weakness.

## V. SYSTEM ARCHITECTURE

MentorAI follows a modular three-tier client-server architecture. The three tiers comprise: the Presentation Layer (frontend), the Application Layer (backend), and the Data Layer (database and file storage). Each tier communicates through well-defined interfaces — REST APIs for standard CRUD operations and real-time event channels for streaming AI responses.

### A. Presentation Layer (Frontend)

The frontend is built using React.js, providing a component-based declarative UI architecture. The Dashboard, AI Tutor, Voice Recall, Flashcards, Quiz Engine, Workspace, Knowledge Graph, and Nexus modules are implemented as separate React modules sharing a global authentication and study-state context. Real-time AI responses are received via server-sent streaming, with words appearing incrementally. The D3.js library provides the 3D force- directed graph rendering for the Knowledge Graph module. The Web Speech API is accessed directly from the browser without additional libraries.

### B. Application Layer (Backend)

The backend is implemented using Node.js with the Express.js framework. It exposes a RESTful API for all standard operations including user management, flashcard CRUD, quiz generation requests, note management, and knowledge graph data retrieval. An AI proxy layer routes requests to the configured LLM API (Groq-hosted Llama- 3 or equivalent), handling streaming response forwarding to the client. All business logic — SM-2 interval calculation, weak topic aggregation, quiz conceptual tagging, and knowledge graph edge generation — resides in this layer. The backend is stateless; session state is managed through JWTs validated on each request.

### C. Data Layer

User profiles, flashcard decks, quiz histories, note content, voice recall sessions, knowledge graph nodes and edges, and study analytics are stored in MongoDB, a document-oriented NoSQL database chosen for its schema flexibility and rich querying capabilities. File attachments such as imported PDFs are stored in cloud object storage (Firebase Storage or AWS S3), with the database storing only metadata and storage URLs. Study event logs are stored in an append-only collection to support historical analytics and the 7-day mastery chart.

### D. Cross-Module Intelligence Layer

MentorAI's distinguishing architectural feature is its cross-module intelligence layer: a set of service functions that aggregate performance signals from quiz attempts, voice recall sessions, and flashcard reviews into a unified weak topic registry. This registry is exposed to the AI Tutor as context, surfaces on the Dashboard as AI-detected weak topics, and is used by the Flashcard and Quiz modules to bias content generation toward areas of demonstrated weakness. This bidirectional data flow is what makes MentorAI genuinely adaptive rather than merely multi- featured.

## VI. FEATURE DESCRIPTION

### A. Dashboard — Learning Command Center

The Dashboard serves as MentorAI's central hub, providing a comprehensive real-time view of the learner's study health. Its primary components include: a Study Streak Tracker displaying consecutive days studied with a gamified flame icon to reinforce habit formation; Real-Time Mastery Metrics showing quiz accuracy, cards mastered, and weekly study time; a 7-Day Mastery Chart visualizing the progression of knowledge mastery over the past week; an AI-Detected Weak Topics panel that automatically identifies struggling areas by aggregating error data from quiz attempts, flashcard misses, and voice recall gaps; a Recent Activity Feed displaying a chronological log of

all study actions; and a Quick Action Launcher providing one-click access to any study mode.

The Dashboard aggregates data from all other modules to present a holistic picture of learning health, serving not only as an information display but as an intelligent action prompt — surfacing the most critical gaps and directing the learner toward the study activities most likely to produce improvement.

**B. AI Tutor — Conversational Learning**

The AI Tutor provides a ChatGPT-style conversational interface trained as a subject-specific personal tutor with persistent session context. Key features include: subject-specific system prompts that adapt the AI's personality, examples, and vocabulary to the chosen subject area; streaming responses with words appearing in real time for an engaging, low-latency interaction; context awareness that reads weak topics from the Dashboard and quiz/flashcard/voice recall data to inform answers; automatic follow-up question generation suggesting three relevant questions after each AI response; Markdown support rendering formatted text, code blocks, and mathematical equations; one-click Action Buttons to save any explanation as a flashcard or note; and full session memory ensuring the entire conversation context is included with each new message.

When the AI Tutor is opened from the Dashboard's weak topics panel, it automatically loads the struggling topic as its initial context, immediately orienting the conversation toward the learner's most critical knowledge gap.

**C. Voice Recall — Active Retrieval Practice**

The Voice Recall module implements the scientifically validated active retrieval practice technique by requiring students to articulate answers aloud without access to notes. The workflow proceeds as follows: the learner selects a note or topic; the AI generates five to ten exam-style questions; the learner rates their confidence (one to five stars) before answering; they speak their answer aloud, which the Web Speech API transcribes in real time with an animated waveform; the AI evaluates the transcribed answer against key concepts; and the learner receives detailed feedback identifying what was correct, what was missed, and how to improve.

Distinct features include: live voice transcription with a responsive animated sound wave; confidence versus accuracy tracking to identify overconfidence and underconfidence patterns; specific gap analysis that pinpoints exact missing concepts rather than generic correctness judgments; automatic weak concept extraction whose session data flows to the Dashboard and AI Tutor; and post-session actions to create flashcards from missed concepts or immediately study gaps with the AI Tutor.

**D. Flashcards — Spaced Repetition System**

The Flashcard module implements the SM-2 spaced repetition algorithm — the same algorithm underlying Anki — for scientifically optimized memory retention scheduling. Features include: deck organization by subject with visual mastery tracking; due card notifications ensuring learners study only what is needed on a given day; a 3D flip card animation providing a premium interactive experience; three-tier grading ('Got it', 'Almost', 'Missed') that adjusts each card's review interval and ease factor; AI generation from notes whereby pasted text produces eight to twelve study cards in seconds; automatic generation from weak topics detected in quiz and voice recall sessions; and mastery visualization via progress bars showing deck understanding.

The SM-2 algorithm operates as follows: cards graded as 'Missed' are scheduled for review the following day; cards graded as 'Got it' receive exponentially increasing intervals (1 day, then 6 days, then 14 days, then 30 days); and each card's ease factor is personalized to the learner's memory strength, preventing both over-review of well-known material and under-review of difficult concepts.

**E. Quiz Engine — Adaptive Testing**

The Quiz Engine provides AI-generated multiple-choice assessments with immediate feedback and explanation-focused learning. Features include: custom quiz generation allowing the learner to specify subject, topic, difficulty level, and question count; AI-generated plausible distractors that create genuinely challenging wrong answer choices; immediate per-question feedback with no submit button — the correct answer and explanation appear the moment an option is selected; conceptual tagging labelling each question with the tested concept; weak topic detection where wrong answers add the tagged concept to the platform's weak topic registry; historical accuracy analytics broken down by subject and improvement over time; and post-quiz actions to study weak topics with the AI Tutor or generate targeted flashcard decks.

**F. Workspace — AI-Powered Notes**

The Workspace provides a rich-text note editor with integrated AI tools for content creation, organization, and study material generation. Features include: a TipTap rich-text editor supporting formatting, code blocks, ordered and unordered lists, and headings; PDF import with AI-powered extraction that detects document structure, generates a summary, and identifies key terms; auto-save every thirty seconds to prevent data loss; an AI side panel with four tools — Summarize Note, Generate Quiz from Note, Create Flashcard Deck, and Translate to Other Languages; a searchable note library with subject-based filtering; and a trash system for recovering deleted notes.

The Workspace serves as the primary content ingestion point: every note created becomes source material for AI- generated flashcards, quizzes, and voice recall question sets, ensuring that

the learner's own notes are the direct foundation of their personalized study plan.

### G. Knowledge Graph — Visual Learning Map

The Knowledge Graph provides a 3D interactive visualization of all studied concepts and their relationships, offering the learner a topological view of their knowledge state. Every topic from quizzes, flashcards, notes, and voice recall sessions is represented as a node. Node colour encodes mastery level: red for struggling, yellow for learning, and green for mastered. Node size encodes study frequency. The AI generates edges between related concepts with labels describing the nature of each relationship. The learner can interact with nodes by clicking to see related concepts and study action options.

The graph is rendered using D3.js with a physics-based force-directed layout that organically clusters related concepts. Subject filtering allows the learner to isolate one subject's graph. Deep linking connects each node directly to the AI Tutor or Quiz Engine for that concept. An activity timeline per concept allows the learner to trace the evolution of their understanding over time, revealing which concepts remain persistent weak spots and which have been successfully consolidated.

### H. Nexus — Command Palette

Nexus is a keyboard-first command palette providing global navigation and action execution across the entire platform, modelled on the command palette paradigm popularized by tools such as Linear and Notion. Accessible via the global keyboard shortcut Cmd+K (or Ctrl+K on Windows), it provides: fuzzy search across all platform actions and pages; keyboard navigation using arrow keys with Enter to execute and Escape to close; action categories including Navigate (go to any page), Create (new note, deck, or quiz), and Recent (last five actions); and context-aware suggestions that display relevant actions based on the current page. Nexus eliminates the need to hunt through menus or reach for the mouse, enabling power users to perform any platform action entirely via keyboard.

## VII. TECHNOLOGY STACK

Table I presents the complete technology stack used in

| Category | Technology | Purpose |
|---|---|---|
| Frontend Framework | React.js | Component-based UI architecture |
| Rich-Text Editor | TipTap | Note editing with formatting support |
| Graph Visualization | D3.js (Force-Directed) | 3D Knowledge Graph rendering |
| Voice Interface | Web Speech API | Real-time speech transcription |
| Styling | Tailwind CSS | Utility-first responsive design |
| State Management | React Context + Hooks | Global auth and study state |
| Backend Framework | Node.js + Express.js | RESTful API and AI proxy layer |
| Database | MongoDB | Document store for all user data |
| File Storage | Firebase Storage / AWS S3 | PDF and media asset storage |
| AI Inference | Groq API (Llama-3) | LLM tutoring, evaluation, generation |
| Authentication | JWT + bcrypt | Session management and password security |
| PDF Processing | PDF.js / LangChain | PDF text extraction and chunking |
| Flashcard Algorithm | SM-2 (custom implementation) | Spaced repetition scheduling |
| Deployment | Vercel / Railway + MongoDB Atlas | Cloud hosting and database |

## VIII. DATABASE DESIGN

MentorAI's database schema is organized into seven primary collections in MongoDB, each responsible for a distinct domain of the application:

### A. Users Collection

The Users collection stores: unique user ID (UUID), username, email address, bcrypt-hashed password, profile picture URL, authentication provider (local or OAuth), subject preferences array, study streak counter, last study date, account creation timestamp, and last login timestamp. The email field carries a unique index to prevent duplicate registrations.

### B. Notes Collection

The Notes collection stores: note ID, owner user ID, note title, rich-text content (TipTap JSON format), subject tag, source type (manual or PDF import), extracted PDF summary and key terms (for PDF-imported notes), auto-save timestamp, creation timestamp, last modified timestamp, and a soft-delete flag for trash functionality.

### C. Flashcard Decks and Cards Collections

The Decks collection stores deck metadata: deck ID, owner user ID, deck name, subject tag, total card count, and mastered card count. The Cards collection stores individual flashcard data: card ID, parent deck ID, front content, back content, SM-

2 algorithm state (ease factor, interval in days, repetition count), next due date, creation timestamp, and source type (manual, AI-generated from note, or auto-generated from weak topic).

### D. Quiz Sessions Collection

The Quiz Sessions collection stores: session ID, user ID, subject, topic, difficulty, total question count, questions array (each containing question text, options, correct answer, conceptual tag, and user's selected answer), score, accuracy percentage, identified weak concepts list, and session timestamp. This collection is the primary source of data for the weak topic detection system and historical accuracy analytics.

### E. Voice Recall Sessions Collection

The Voice Recall Sessions collection stores: session ID, user ID, source note ID, questions array (each containing the question text, user's confidence rating, transcribed answer text, AI evaluation result, and identified missing concepts), overall session accuracy, weak concept list extracted from the session, and session timestamp.

### F. Knowledge Graph Collection

The Knowledge Graph collection stores the learner's personal concept graph: a Nodes sub-collection containing concept name, subject, mastery level (struggling, learning, or mastered), study frequency count, and last studied timestamp; and an Edges sub-collection containing source concept ID, target concept ID, AI-generated relationship label, and creation timestamp.

### G. Study Events Collection

An append-only Study Events collection logs every study interaction: event ID, user ID, event type (quiz attempt, flashcard review, voice recall, AI tutor message, or note creation), event timestamp, and a payload object containing event-specific data such as concept tags and performance scores. This collection powers the 7-day mastery chart and recent activity feed on the Dashboard.

## IX. SECURITY CONSIDERATIONS

Security is a foundational design priority throughout MentorAI's architecture. The following measures are implemented across the stack:

**Password Security:** All user passwords are hashed using bcrypt with a work factor of 12 (salt rounds), making brute-force attacks computationally prohibitive. Plaintext passwords are never logged or stored anywhere in the system.

**Token Management:** JSON Web Tokens are signed using RS256 asymmetric keys with a 24-hour expiry. Refresh tokens are issued for extended sessions. All tokens are validated server-side on every API request before any data operation is performed.

**Access Control:** Every API endpoint performing data access enforces server-side ownership verification. A user cannot access, modify, or delete another user's notes, flashcard decks, quiz sessions, or knowledge graph data regardless of the request URL or payload.

**AI Proxy Security:** All requests to the Groq API are routed through the backend proxy layer. API keys are stored exclusively as server-side environment variables and are never transmitted to or accessible from the client-side JavaScript bundle.

**Infrastructure Security:** All client-server communication is conducted over HTTPS with TLS 1.2 or higher. HTTP Strict Transport Security headers are served. The Helmet.js middleware sets security-relevant HTTP response headers including Content-Security-Policy, X-Frame-Options, and X-Content-Type-Options. Input validation and sanitization are applied to all user-supplied content before storage or LLM prompt injection to prevent prompt injection attacks.

## X. TESTING AND VALIDATION

### A. Unit Testing

Individual service modules — the SM-2 interval calculation service, weak topic aggregation service, quiz generation service, voice recall evaluation service, and knowledge graph edge generation service — were tested in isolation using the Jest testing framework. Test cases covered expected behaviour for valid inputs, boundary conditions (e.g., SM-2 ease factor minimum clamping), and error responses for invalid inputs. Mock objects isolated modules from external LLM API dependencies. A total of 74 unit test cases were written, achieving approximately 76% line coverage across tested modules.

### B. Integration Testing

Integration tests validated end-to-end user flows across the full application stack. Key scenarios tested include: (1) the complete user registration and JWT authentication flow; (2) note creation followed by AI flashcard generation and deck storage; (3) quiz session completion, wrong-answer tagging, and weak topic propagation to the Dashboard;

(4) voice recall session execution with Web Speech API mock transcription and AI evaluation; (5) Knowledge Graph node creation and AI edge generation following a quiz session; and (6) Nexus command palette fuzzy search across navigation targets and action creators. Integration tests used a dedicated MongoDB test instance and mocked the Groq API to avoid inference costs during automated testing.

### C. User Acceptance Testing (UAT)

A UAT session was conducted with 15 student participants from the Computer Science and Engineering department who used MentorAI for a structured evaluation lasting 45 minutes each. Participants were asked to complete a set of prescribed tasks — creating a note, generating a flashcard deck, taking a quiz, completing a voice recall session, exploring the

knowledge graph, and using the AI Tutor — and then completed a structured questionnaire using a 5- point Likert scale. Table II summarizes the UAT results.

TABLE II UAT RESULTS SUMMARY (N=15)

| Evaluation Criterion | Mean Score (/5) | Std Dev |
|---|---|---|
| Ease of Use | 4.4 | 0.38 |
| Feature Completeness | 4.2 | 0.49 |
| Visual Design & Navigation | 4.5 | 0.34 |
| AI Tutor Usefulness | 4.6 | 0.41 |
| Voice Recall Engagement | 4.3 | 0.52 |
| Flashcard & Quiz Quality | 4.4 | 0.45 |
| Knowledge Graph Clarity | 4.1 | 0.57 |
| Overall Satisfaction | 4.4 | 0.40 |

## XI. RESULTS AND DISCUSSION

The MentorAI platform was successfully implemented with all eight specified modules in a functional state. Development was completed over approximately five months using an Agile iterative approach with bi-weekly sprints and regular feedback integration.

Dashboard and Analytics: The real-time Dashboard correctly aggregated data from all study modules. The 7-day mastery chart updated accurately following quiz, flashcard, and voice recall sessions. AI-detected weak topics were identified and surfaced within 200ms of session completion in all test scenarios.

AI Tutor: Groq API integration delivered average first-token response times of 110ms and complete response times of 290ms for typical tutoring queries, significantly outperforming the 800 to 1,500ms typical of conventional cloud AI endpoints. The context-aware initialization from weak topics was confirmed to correctly pre-load struggling concepts in 100% of test cases. Participants in UAT rated the AI Tutor as the most useful feature, with comments highlighting its subject-specific relevance and the value of AI-generated follow-up questions.

Voice Recall: Web Speech API transcription achieved an average word accuracy of 88% under standard laptop microphone conditions, exceeding the 85% success criterion. AI evaluation of spoken answers correctly identified missing key concepts in 91% of test cases. The confidence versus accuracy tracking feature was noted by UAT participants as a novel and insightful self-assessment mechanism.

**Flashcards and SM-2:** The SM-2 algorithm was verified to produce correct review intervals across all three grading tiers through unit testing. AI-generated flashcard decks from notes averaged 9.4 cards per note across 50 test runs, with UAT participants rating generated card quality at 4.4 out of 5.

**Quiz Engine:** AI-generated quizzes were evaluated for distractor plausibility by UAT participants, who rated wrong answer quality at 4.2 out of 5. Weak topic detection correctly tagged and propagated missed concepts to the Dashboard and AI Tutor in all integration test scenarios.

**Knowledge Graph:** The D3.js force-directed graph successfully rendered graphs of up to 80 concept nodes with physics-based layout stabilization completing within 1.2 seconds. AI-generated edges were found to be semantically accurate for 87% of tested concept pairs. UAT participants found the graph visually intuitive, though several suggested improvements to the edge label readability at smaller node sizes.

**Areas for Improvement:** UAT participants suggested additions including a mobile application, collaborative study rooms for group quizzes, gamified leaderboards, and export functionality for notes and flashcards to PDF. These are documented as future work items.

The codebase at the time of UAT consisted of approximately 10,800 lines of JavaScript and JSX across the frontend and backend, organized into 72 source files. Unit test coverage of core service modules reached 76%, exceeding the 75% target. The frontend consisted of 28 React components and 6 context providers managing global state for authentication, study data, knowledge graph state, and theme preferences.

## XII. PERFORMANCE ANALYSIS

MentorAI's performance was evaluated across key functional areas to validate its suitability as a production-ready platform for student learners.

### A. API Response Times

RESTful API endpoint response times were measured across 100 simulated concurrent users. Authentication endpoints averaged 82ms. Note CRUD operations averaged 95ms. Flashcard deck generation (AI-powered) averaged 1.4 seconds for a 10-card deck, which is within acceptable interactive thresholds for a generation task. Quiz generation averaged 1.8 seconds for a 10-question quiz. Knowledge graph node and edge data retrieval for a 50-node graph averaged 140ms.

### B. AI Inference Performance

Groq API performance was benchmarked against alternative inference endpoints. For standard tutoring queries (average prompt length 200 tokens, response 150 tokens), Groq delivered an average response time of 290ms. For voice recall answer evaluation prompts (average prompt 400 tokens), the average response was 480ms. For flashcard and quiz generation prompts (average prompt 300 tokens, response 500 tokens), the average response was 620ms. All figures significantly outperform OpenAI API equivalents under comparable conditions.

### C. Voice Recall Transcription Performance

Web Speech API transcription latency from speech end to text availability averaged 310ms, providing a responsive user experience. Word accuracy under standard laptop microphone conditions averaged 88%. Accuracy dropped to approximately 76% in high-ambient-noise conditions, which is identified as a limitation of the current implementation and a driver for future noise-suppression preprocessing.

### D. Knowledge Graph Rendering

D3.js force-directed graph rendering was measured for varying graph sizes. Graphs of up to 30 nodes rendered and stabilized within 400ms. Graphs of 80 nodes stabilized within 1.2 seconds. At 150 nodes, stabilization required 2.8 seconds, which approached the threshold of perceptible delay. Graph size optimization — such as progressive loading of distant nodes — is documented as a future performance improvement.

## XIII. COMPARATIVE ANALYSIS

Table III presents a structured feature comparison between MentorAI and three leading existing platforms: Anki, Quizlet, and Khan Academy. The comparison evaluates each platform across twelve key dimensions relevant to personalized student learning.

TABLE III FEATURE COMPARISON: MENTORAI VS. EXISTING PLATFORMS

| Feature | MentorAI | Anki | Quizlet | Khan Academy |
|---|---|---|---|---|
| SM-2 Spaced Repetition | Yes | Yes | Partial | No |
| AI Conversational Tutor | Yes | No | No | Partial |
| Voice Recall Practice | Yes | No | No | No |
| Cross-Module Weak Topic Detection | Yes | No | No | No |
| AI Flashcard Generation | Yes | No | Yes | No |
| PDF Import & AI Extraction | Yes | No | No | No |
| 3D Knowledge Graph | Yes | No | No | No |
| Adaptive Quiz Engine | Yes | No | Yes | Partial |
| Real-Time Mastery Dashboard | Yes | Partial | Partial | Partial |
| Keyboard Command Palette | Yes | No | No | No |
| AI Follow-Up Questions | Yes | No | No | Partial |
| Confidence vs. Accuracy Tracking | Yes | No | No | No |

The comparative analysis in Table III illustrates that MentorAI is the only platform providing cross-module intelligence — where performance data from one study mode (e.g., quiz errors) directly enriches every other study mode (AI Tutor context, flashcard generation, knowledge graph mastery colouring). Anki excels at spaced repetition but provides no AI tutoring or analytics integration. Quizlet provides flashcards and quizzes but lacks voice recall, PDF intelligence, a knowledge graph, and cross-module data flow. Khan Academy provides structured subject content but lacks personalized spaced repetition, voice recall, and a dynamic knowledge graph.

## XIV. CHALLENGES AND LIMITATIONS

### A. Technical Challenges

The most significant technical challenge was designing the cross-module intelligence layer — the system that aggregates performance signals from disparate study modes into a unified weak topic registry. Ensuring that quiz errors, flashcard misses, and voice recall gaps were normalized into a consistent concept-tagging schema required careful prompt engineering and a schema reconciliation service to merge conceptually equivalent tags that may be worded differently across modules.

The second major challenge was implementing real-time D3.js force-directed graph updates without causing visual discontinuity when new nodes and edges are added during a study session. The solution adopted uses a merge strategy that preserves existing node positions while inserting new nodes at the periphery of the current graph layout, maintaining visual continuity.

### B. Limitations of Current Implementation

The current version of MentorAI has several acknowledged limitations. First, voice recall accuracy degrades in high-ambient-noise conditions due to the Web Speech API's lack of noise suppression preprocessing. Second, the Knowledge Graph rendering performance degrades above 150 nodes, requiring future optimization via progressive loading. Third, the AI-generated flashcard and quiz content quality depends on the clarity and length of the source note — very short notes produce fewer and less detailed study materials. Fourth, the platform currently supports only English-language content for AI generation and voice recall evaluation; multilingual support is documented as a future development item.

### C. Scalability Considerations

The current deployment architecture uses a single Node.js instance and a single MongoDB Atlas cluster. This configuration is suitable for academic-scale use but would require horizontal scaling using a load balancer, multiple application server instances, and a Redis-based caching layer for frequently accessed knowledge graph data to support thousands of concurrent users. AI inference costs at scale would require optimization through prompt caching and batched generation for offline flashcard and quiz preparation.

### Acknowledgment

The authors sincerely thank their project guide and the faculty of the Department of Computer Science & Engineering for their invaluable guidance, mentorship, and support throughout the development of MentorAI. The authors also acknowledge the Groq team for providing API access for educational purposes, the open-source community for the libraries and frameworks that underpinned the platform — particularly the D3.js, TipTap,

and React ecosystems — and the 15 student participants who generously contributed their time to user acceptance testing.

## REFERENCES

1. H. Ebbinghaus, Memory: A Contribution to Experimental Psychology. New York, NY: Teachers College, Columbia University, 1885 (translated 1913).
2. P. A. Wozniak and E. J. Gorzelańczyk, "Optimization of repetition spacing in the practice of learning," Acta Neurobiologiae Experimentalis, vol. 54, no. 1, pp. 59–62, 1994.
3. H. L. Roediger and J. D. Karpicke, "Test-Enhanced Learning: Taking Memory Tests Improves Long-Term Retention," Psychological Science, vol. 17, no. 3, pp. 249–255, 2006.
4. K. VanLehn, "The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems," Educational Psychologist, vol. 46, no. 4, pp. 197–221, 2011.
5. M. T. H. Chi, N. de Leeuw, M. H. Chiu, and C. LaVancher, "Eliciting Self-Explanations Improves Understanding," Cognitive Science, vol. 18, no. 3, pp. 439–477, 1994.
6. H. U. Hoppe and M. Miatidis, "A Knowledge-based Tutoring System for Web-based Distance Learning," in Proc. 11th World Conf. Artificial Intelligence in Education, 2003, pp. 443–450.
7. R. Shadiev, W. Hwang, Y. Huang, and T. Liu, "The Impact of Using Speech Recognition Technologies on Language Learning," Educational Technology & Society, vol. 17, no. 4, pp. 187–199, 2014.
8. D. J. Weiss and G. G. Kingsbury, "Application of Computerized Adaptive Testing to Educational Problems," Journal of Educational Measurement, vol. 21, no. 4, pp. 361–375, 1984.
9. Groq Inc., "Groq API Documentation," 2024. [Online]. Available: https://console.groq.com/docs/.
10. MongoDB Inc., "MongoDB Documentation," 2024. [Online]. Available: https://www.mongodb.com/docs/.
11. Meta AI, "Llama 3 Model Card," 2024. [Online]. Available: https://ai.meta.com/blog/meta-llama-3/.