IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 5, December 2025
Impact Factor: 7.67

KOA: An AI-Integrated Productivity Intelligence Platform for Coding Skill Development and Real-Time Performance Analytics

Zikra Sameer Pathan, Pranjal Manoj Argade, Poorva Swapnil Sawant,
Bharati Bhaskar Dhere, Vidya Pingale
Dept. Computer Engineering
Pimpri Chinchwad Polytechnic, Pune, India
pathanzikra50@gmail.com, argadepranjal10@gmail.com, 04.poorvasawant@gmail.com
dbharati5162@gmail.com, vidya.pingale8@gmail.com

Abstract: *The exponential growth in online coding education has created a significant gap between theoretical learning and practical productivity measurement. Traditional coding learning platforms lack comprehensive activity tracking, real-time feedback mechanisms, and AI-driven performance insights necessary for learners to optimize their coding productivity and skill development. This paper presents KOA (Knowledge Optimization Assistant), an AI-integrated web-based productivity intelligence platform designed to bridge this gap. KOA provides a unified environment for beginner and intermediate programmers to code, track productivity metrics in real-time, receive AI-powered feedback through an integrated Gemini-based chatbot, and visualize their learning progress through gamified elements and comprehensive analytics. The platform employs activity tracking, multi-language code execution via Judge0 API, automated quiz generation, and behavioral analytics to deliver personalized insights. Deployed on the Render cloud platform with Supabase PostgreSQL backend, KOA is designed to be cost-efficient, scalable, and accessible across devices. This paper outlines the architecture, core functionalities, technical implementation details, and deployment strategy of KOA, demonstrating how the integration of real-time monitoring, AI assistance, and gamification creates an effective ecosystem for coding skill development and productivity optimization.*

Keywords: Productivity Tracking, AI-Assisted Learning, Code Execution, Real-time Analytics, Gamification, Web-Based IDE, Activity Monitoring

I. INTRODUCTION

The digital transformation in education has democratized access to coding resources, yet learners face a critical challenge: the inability to measure and optimize their coding productivity effectively. Traditional coding practice platforms such as LeetCode, HackerRank, and CodeSignal focus primarily on problem-solving and competitive benchmarking, often neglecting the aspects of personal productivity, time management, and behavioral learning patterns (Brusilovsky & Peylo, 2003; Pérez-Marín et al., 2001). More specifically, as Siemens (2013) notes, "the systematic collection of data from various educational platforms captures both academic performance and behavioral aspects of learning, such as student interactions with content and peers," yet most coding platforms fail to leverage this potential for personalized feedback. Learners lack real-time activity monitoring, contextual AI feedback, personalized performance insights, integrated learning environments, and behavioral motivation mechanisms. These gaps result in inefficient learning cycles, poor time management habits, and limited self-awareness regarding skill development.

A. Problem Statement

Current coding education platforms suffer from several critical limitations. First, they provide no mechanism to monitor what portion of a student's time is spent productively versus unproductively during practice sessions. As noted in recent research on learning analytics dashboards, "Teachers can see tell-tale signs of distress, such as lower participation, late assignments or constant re-working of the same material—all of which can signal a possible deficit of understanding or motivation" (Eve Paper, 2025, p. 1). Yet this monitoring is absent in most coding platforms.

Second, AI-powered assistance, when available, lacks awareness of the student's current code context and learning history, resulting in generic responses that fail to address specific learning needs. Third, assessment is disconnected from real-time learning activities—quizzes are separate from coding practice rather than integrated. Fourth, most sophisticated platforms require paid subscriptions, limiting accessibility in developing regions; research demonstrates that cost barriers and infrastructure limitations significantly impede educational equity in Africa, Asia, and Latin America (World Bank, 2022).

Finally, without gamification and behavior-driven feedback, learners struggle to maintain consistent practice habits essential for skill acquisition. As Aragon and Johnson (2008, p. 150) found in their study of community college online courses, "factors such as motivation and perceived fit with course characteristics were significantly related to completion," demonstrating that motivation infrastructure is critical for student persistence. The research further indicates that learners completing online programs demonstrate significantly higher motivation when receiving regular feedback and achievement recognition.

B. Contribution of this Work

This research addresses the need for an integrated, AI-powered coding productivity platform that combines real-time activity tracking with intelligent feedback mechanisms. The primary contributions of this work include:

Real-Time Productivity Intelligence Platform: An AI-integrated web application that monitors coding activities through keystroke dynamics and mouse movement patterns to provide instant performance metrics, distinguishing between productive and unproductive activities with granular behavioral analysis (Balamuralithara, 2023).

Multi-Modal Code Execution Environment: Integration of Judge0 API to support real-time code execution across 80+ programming languages, enabling learners to test code instantly within the IDE without requiring external tools or complex local setup.

AI-Powered Contextual Assistance: Implementation of Google Gemini API for a floating chatbot that provides code-specific guidance, conceptual explanations, and debugging assistance without context switching, utilizing current code and session metadata for personalized responses.

Automated Assessment and Learning Analytics: Dynamic quiz generation based on user's coding session content using retrieval practice principles. As Roediger and Karpicke (2006, p. 331) seminal work demonstrates, "the power of testing memory" shows that "repeated tests substantially increased retention relative to learners who simply restudied the prose material," a principle that informs KOA's quiz-based learning approach.

Gamification and Behavioral Engagement: Achievement systems, streak counters, leaderboards, daily challenges, and level progression grounded in Self-Determination Theory. According to Ryan and Deci (2000, p. 74), "three innate psychological needs—competence, autonomy, and relatedness—when satisfied yield enhanced self-motivation and mental health and when thwarted lead to diminished motivation and well-being." KOA's gamification design specifically targets these three psychological needs.

Cost-Optimized Cloud Architecture: Design for free and low-cost cloud deployment on Render and Supabase, leveraging serverless architecture principles for automatic scaling and cost efficiency, making the platform accessible without subscription costs to learners in resource-constrained environments.

II. RELATED WORK AND LITERATURE SURVEY

A. Traditional Code Learning Platforms

Existing platforms like LeetCode, CodeSignal, and HackerRank have revolutionized competitive programming education by providing curated problem sets, real-time code execution, and skill assessments (Brusilovsky & Peylo, 2003). However, these platforms are fundamentally problem-centric rather than learner-centric. They measure success through problem-solving counts and rankings but provide limited insight into learning efficiency, time management effectiveness, or individual learning patterns. The core limitation is that they focus on what students produce rather than how they learn.

B. Learning Analytics and Activity Tracking

Recent advances in educational technology have demonstrated the effectiveness of learning analytics in predicting student success and identifying at-risk learners (Siemens & Long, 2011). Learning analytics tools track diverse data sources, and as research indicates, "by systematically analyzing this data, several key benefits emerge: identifying patterns and trends that reveal relationships between student engagement behaviors—such as participation in discussion forums—and their performance in assessments" (Feedback Fruits, 2024, p. 2). Keystroke dynamics and mouse movement patterns provide rich behavioral signals for personalized learning recommendations (Balamuralithara, 2023). However, most existing platforms either focus on knowledge assessment through quizzes or activity logging separately, lacking integrated analysis that bridges both dimensions. Learning analytics research consistently shows that "low-performing students engaged more frequently but in a reactive manner, with activity concentrated around deadlines and tapering off afterward. In contrast, high-performing students demonstrated more stable and consistent engagement patterns throughout the semester, suggesting a more proactive and sustained approach to learning" (Leverage Learning Analytics, 2025, p. 4). This distinction highlights the need for platforms that provide insight into engagement quality, not just quantity.

C. AI-Assisted Learning and Chatbots in Education

The integration of large language models (LLMs) in educational contexts has shown promising results in improving student engagement and learning outcomes. Systematic reviews of AI-driven intelligent tutoring systems demonstrate positive effects on learning achievement (Huang et al., 2025). Studies on ChatGPT support in educational settings report significant improvements, with "students taught with AI support outperformed traditional instruction groups by up to 15.4% in quiz scores" (Chen et al., 2024, p. 38). However, most implementations suffer from context loss—the chatbot lacks awareness of the student's current code, session goals, and learning history, leading to generic responses. KOA addresses this critical gap through integrated code context passing to the Gemini API, enabling contextually aware assistance.

D. Gamification in Online Learning

Gamification has emerged as a powerful mechanism for increasing intrinsic motivation and engagement in educational platforms. Research indicates that "67% of students report feeling more motivated with gamified learning, with a 14% increase in those strongly agreeing that gamification enhances their motivation" (Upskillist, 2025, p. 3). More importantly, theoretical foundations emphasize that "intrinsic motivation happens when students enjoy learning for its own rewards, like the excitement of understanding a new concept or discovering a new skill. Extrinsic motivation, on the other hand, occurs when students are motivated by external rewards or punishments. The positive effects of gamification occur more when teachers prioritize intrinsic motivation as they plan learning activities" (Waterford, 2024, p. 2).

Self-Determination Theory (SDT) provides the theoretical foundation for effective gamification. Ryan and Deci (2000) established that autonomy support is critical, noting that individuals must feel "a sense of initiative and ownership in one's actions" and that this is "supported by experiences of interest and value and undermined by experiences of being externally controlled, whether by rewards or punishments." KOA employs research-backed gamification principles focused on personal progress and mastery rather than leaderboard competition, promoting intrinsic motivation aligned with long-term learning goals.

E. Real-Time Performance Monitoring and Feedback

Modern learning platforms increasingly employ real-time monitoring to provide immediate feedback. Research demonstrates that "when feedback is delivered promptly, learners can correct misconceptions, reinforce correct behaviors, and stay motivated—crucial factors for mastering skills and retaining knowledge" (Paradiso Solutions, 2025, p. 1). More specifically, "a study in the Journal of Educational Psychology found that students receiving instant feedback demonstrated higher persistence and enthusiasm than those relying on delayed evaluations" (Paradiso Solutions, 2025, p. 2).

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 5, Issue 5, December 2025 Impact Factor: 7.67

The psychological impact is profound: "when students perceive that their feedback directly influences the flow and structure of lessons, their sense of agency and motivation to participate increase. This collaborative atmosphere fosters a deeper connection between students and teachers, enhancing overall learning outcomes" (Paradigm Press, 2024, p. 3). However, implementing real-time monitoring at scale requires careful architecture design to avoid latency and privacy concerns.

F. Retrieval Practice and Quiz-Based Learning

Low-stakes quiz-based learning, grounded in retrieval practice principles, demonstrates significant effectiveness for knowledge retention. Roediger and Karpicke's (2006) foundational research established that "the testing effect that appeared on the second test was greater when the cues for the first test mismatched the original encoding and yet successful retrieval occurred," demonstrating that diverse retrieval practice strengthens learning. Their work further revealed an important phenomenon: the "illusion of competence," where "the majority of students chose rereading as a strategy with relatively few using self-testing or free recall" (Donald Clark Plan B, 2021, p. 2), yet evidence showed that students performing retrieval practice outperformed those studying passively.
More recent research confirms that "retrieval tests, only a few minutes long, produced a full grade-level increase on the material that had been subject to retrieval" (Donald Clark Plan B, 2021, p. 3). Additionally, "spaced, retrieval practice is even better" (Karpicke & Bauernschmidt, 2011, p. 1250), supporting KOA's approach of generating spaced quizzes at session intervals.

G. Serverless Architecture and Cost-Optimized Deployment

Modern serverless architectures enable building sophisticated applications with minimal infrastructure costs and automatic scaling based on demand (Tiwari et al., 2024). As documented in cloud computing research, "the serverless architecture market is projected to expand at over 25% from 2023 to 2032, driven by cost efficiency and reduced operational overhead" (Mindpath Tech, 2025, p. 1). The business model is fundamentally different: "serverless platforms provide pay-as-you-go pricing where users are charged only for resources consumed, automatic scaling that adjusts resources based on demand, and reduced operational complexity" (Skill Mine, 2024, p. 2).
This technology is particularly crucial for educational platforms in developing regions. As the World Bank (2022, p. 3) notes in their comprehensive review, "low-cost deployment approaches are essential for expanding access to quality education particularly in developing countries where subscription fees represent significant barriers." The research further demonstrates that "open, cost-free platforms can expand access to quality education and help bridge the digital divide in developing countries" (Forbes, 2024, p. 1).

H. Data Privacy and Educational Compliance

Student data protection is paramount in educational platforms. The General Data Protection Regulation (GDPR) requires schools to implement appropriate safeguards, with students retaining rights to access, delete, or modify their personal data (Cavoukian, 2009). Data privacy in education must balance monitoring for learning effectiveness with student privacy protection; effective systems implement granular privacy controls and transparent data practices (S.S. Rana, 2024). As ComplyDog (2024, p. 2) emphasizes, "EdTech platforms must comply with both general privacy regulations (GDPR, FERPA) and education-specific requirements that vary by jurisdiction, requiring careful consent management when multiple parties have authority."

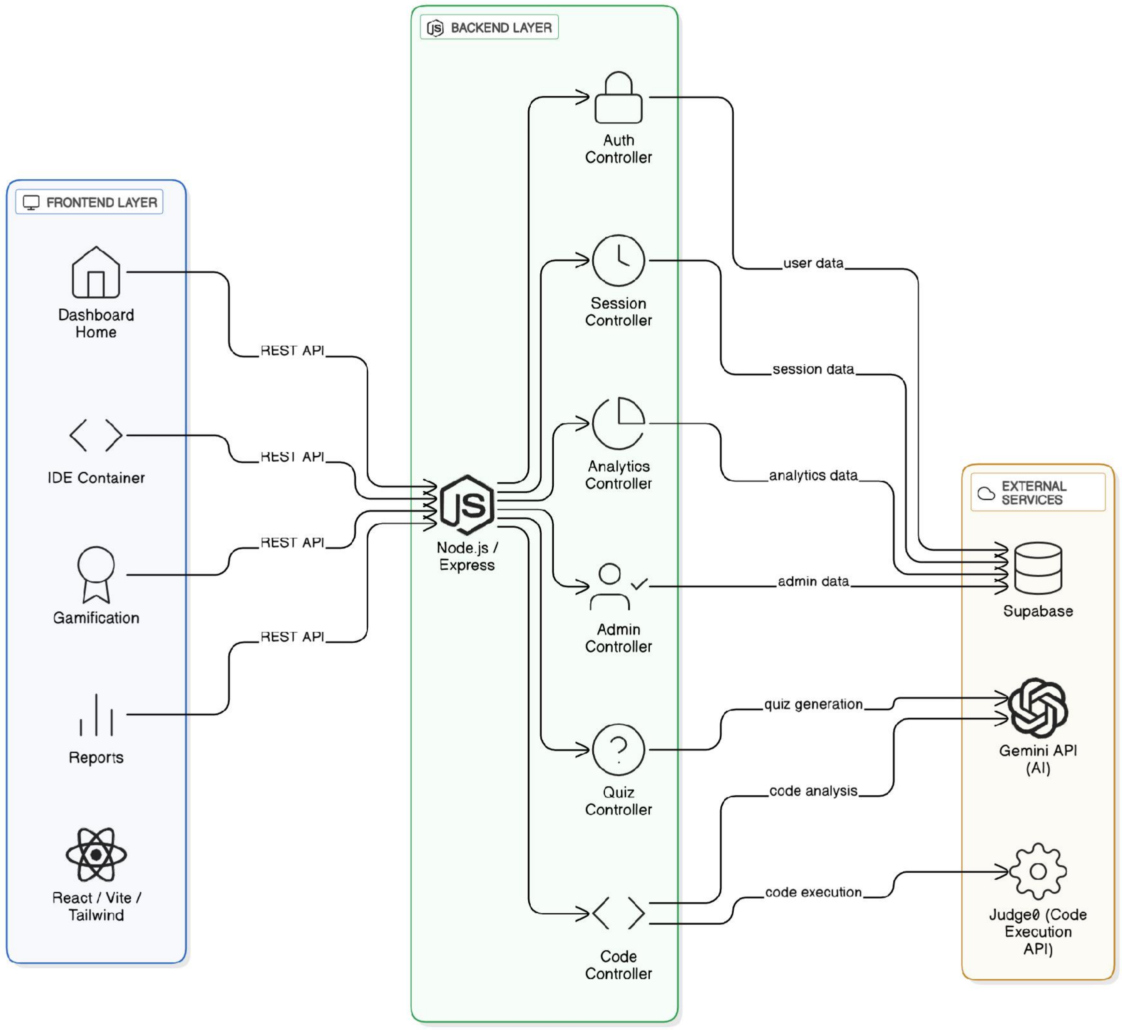
I. Gap in Existing Research

Current literature identifies several critical gaps that KOA addresses: (1) Integration Gap—most platforms separate activity tracking, code execution, AI assistance, and assessment into different tools; (2) Context Awareness Gap—AI tutors lack awareness of user's current code and session context; (3) Personalization Gap—feedback is generic rather than behavior-driven and customized; (4) Accessibility Gap—cost barriers prevent access in developing regions; (5) Behavioral Analytics Gap—limited focus on time management and productivity patterns beyond code correctness metrics; (6) Motivation Gap—few platforms address the psychological needs identified by Self-Determination Theory (Ryan & Deci, 2000).

III. SYSTEM ARCHITECTURE AND DESIGN

A. Architectural Overview

KOA is built on a three-tier cloud-native architecture optimized for free and low-cost deployment using serverless principles (Tiwari et al., 2024). The architecture separates concerns across frontend, backend, and data persistence layers, each leveraging different cloud services to maximize cost efficiency while maintaining functionality. As noted in serverless architecture literature, "the serverless architecture enables deploying sophisticated platforms with minimal infrastructure costs and automatic scaling based on demand, making quality learning tools more democratically accessible" (Skill Mine, 2024, p. 3).



KOA Platform Three-Tier Architecture

B. Frontend Architecture (Render Static Site)

The frontend is deployed as a static site on Render, providing unlimited bandwidth at zero cost using serverless static hosting (Tiwari et al., 2024). React with Vite ensures fast builds and hot module reloading for development, while optimized production builds minimize bundle sizes. Key Frontend Components include Authentication Module for JWT token-based authentication; Integrated IDE with multi-file code editor, real-time syntax highlighting, bracket matching, code formatting, and persistent file state during sessions; Session Management providing start/stop controls and activity tracking; Dashboard displaying productivity overview with weekly statistics and focus scores; Code Execution Panel supporting language selection and real-time output streaming; AI Chat Panel providing floating chatbot integration with conversation history; Quiz Interface displaying auto-generated questions with real-time scoring; Reports and Analytics with comprehensive visualizations; Gamification UI displaying achievements and streaks; and Admin Dashboard for user management and system analytics.

C. Backend Architecture (Render Web Service)

The backend handles all business logic, API routing, authentication, external service integration, and data persistence operations. Deployed on Render's free tier supporting approximately 62 concurrent users and 750 hours per month of runtime. Core Backend Services include: Authentication Service with JWT-based auth and role management; Session Service managing session creation and activity tracking; Code Execution Service integrating Judge0 API with execution queuing; AI Service integrating Gemini API with prompt engineering; Quiz Service handling generation and storage; Activity Analysis Service processing keystroke patterns and idle detection (Balamuralithara, 2023); Report Generation Service creating on-demand reports; Gamification Service managing achievements and leaderboards; Notification Service handling email alerts and in-app notifications.

D. Data Persistence Layer (Supabase)

Supabase provides a managed PostgreSQL database (500 MB free tier) with built-in authentication and real-time capabilities through polling (Supabase, 2024). Database Tables include: users (user profiles and authentication), sessions (coding session metadata), code_activities (keystroke and mouse logs), quizzes (quiz questions), quiz_attempts (user responses), achievements (badge definitions), user_achievements (user-achievement relationships), challenges (daily/weekly challenges), challenge_submissions (user submissions), reports (weekly productivity reports).

E. Technology Stack Selection

Frontend Stack: React 18 for modern component architecture, Vite for fast builds, Tailwind CSS for responsive design, React Context API for state management (React Team, 2024). Backend Stack: Node.js 18+ for asynchronous I/O, Express.js for HTTP server framework, Supabase client for database operations. Database: PostgreSQL via Supabase with JSON support (Supabase, 2024). Authentication: JWT with Supabase Auth for stateless authentication. Code Execution: Judge0 API supporting 80+ languages with request queuing for rate limit management (Judge0, 2024). AI Assistant: Google Gemini API for language understanding with prompt engineering for educational context. Real-Time Updates: Client-side polling using custom usePolling hook instead of WebSockets to function within free-tier constraints. Caching: In-memory caching and localStorage for client-side persistence, replacing Redis. Email Service: Nodemailer with SMTP provider integration. Deployment: Render for frontend (static) and backend (web service) with automatic GitHub integration. Monitoring: Console logging with optional Sentry integration on free tier.

IV. CORE FEATURES AND FUNCTIONALITIES

A. Integrated IDE with Real-Time Code Execution

The embedded IDE provides a full-featured coding environment eliminating the need to switch between applications (Judge0, 2024). Features include multi-language support across 80+ languages through Judge0, multi-file code management with tabbed interface, real-time syntax highlighting, automatic code formatting via Prettier, instant code execution with output streaming, detailed error messages with line number references, and persistent code storage within session records.

Activity Tracking During Code Execution: Each code execution is logged with execution metadata including timestamp, language, status, execution duration, and memory usage. This data feeds into the activity analysis service to calculate complexity metrics using cyclomatic complexity analysis (McCabe, 1976) and identify learning patterns.

B. Real-Time Productivity Tracking

Passive activity monitoring captures productivity signals through keystroke dynamics and mouse movement analysis without explicit user action (Balamuralithara, 2023). As research in keystroke dynamics demonstrates, "keystroke patterns provide rich behavioral signals for identifying user effort and engagement levels" (Plurilock, 2024, p. 2). Tracked Metrics include keystroke frequency measured in characters per minute, mouse movement tracking with idle period detection, session duration from start to completion, idle time detection defined as 30+ seconds without activity, code complexity analysis using cyclomatic complexity formula $M = E - N + 2P$ (McCabe, 1976), and context switching detection measuring time intervals between file changes.

Focus Score Calculation: Focus Score = (Active_Time / Session_Duration) × 100 × (1 - Idle_Ratio) × Code_Complexity_Factor, where Code_Complexity_Factor ranges from 0.5 to 1.5 based on code intricacy. Privacy-Conscious Implementation includes client-side hashing of keystroke data (Balamuralithara, 2023), no screen recording, no code content analysis beyond syntax, user ability to pause tracking, and GDPR-compliant data retention with automatic deletion after 60 days (Cavoukian, 2009).

C. AI-Powered Chatbot with Code Context

The floating Gemini-powered chatbot provides contextually aware assistance by incorporating current code, session metadata, learning history, and problem statements. Chatbot Capabilities include code review identifying bugs and inefficiencies, concept explanation in beginner-friendly language, debugging assistance identifying likely errors, code optimization recommending improvements, question answering on domain-specific topics, and resource recommendations suggesting tutorials. Gemini API Integration constructs detailed prompts incorporating code context and conversation history, validates responses for educational appropriateness, and implements exponential backoff for rate limit management.

D. Automated Quiz Generation

Dynamic quiz creation based on session code content reinforces learning through retrieval practice (Roediger & Karpicke, 2006). As noted in retrieval practice research, "the core principle is that retrieval effort enhances learning; this suggests that the retrieval process itself is at play in the testing effect" (Matuschak, 2023, p. 1). Quiz Characteristics: generated at session end requiring 5-10 minutes, each session generates 5 questions, question types include MCQ, fill-in-blank, code output prediction, difficulty adapts based on code complexity and user skill level.

Generation Algorithm: analyzes written code to extract functions and concepts, synthesizes questions using Gemini API, generates plausible distractors, stores answers with explanations. Quiz Scoring: correct answer +10 points, incorrect answer 0 points, time completion bonus +2 points, results stored for progress tracking. Research shows that spaced retrieval practice is particularly effective, as "the greater the gap between testing and the original exposure or test, the greater the learning benefit" (Karpicke & Bauernschmidt, 2011, p. 1252).

E. Comprehensive Productivity Reports

Detailed weekly analytics reports provide insights into coding patterns and productivity trends (Siemens & Long, 2011). Report Components include Productivity Overview (total coding hours, average daily time, most productive day, focus score trends), Code Activity Analysis (lines written, languages breakdown, execution counts, error frequency), Quiz Performance (scores by session, accuracy percentage, weak concepts identified), Skill Progression (skill tree visualization, completed skills, recommended topics), and Gamification Status (badges earned, current streak, points earned, level progression). Export Formats: PDF with formatted charts, CSV for external analysis, automatic email delivery.

F. Gamification and Engagement Mechanics

Game-like elements incentivize consistent learning while maintaining focus on intrinsic motivation. According to Self-Determination Theory, "autonomy concerns a sense of initiative and ownership in one's actions. It is supported by experiences of interest and value and undermined by experiences of being externally controlled, whether by rewards or punishments" (Ryan & Deci, 2000, p. 232). Components include Achievements & Badges (First Code Run, Bug Buster, Quiz Master, Week Warrior, Polyglot, Streak Milestones); Streak System (consecutive 20+ minute sessions with daily reminders); Level System (10 levels from Novice to Master, XP thresholds of 100, 250, 500, 1000, 2000); Daily Challenges (randomly generated from challenge pool with 50 bonus points); and privacy-focused Leaderboard based on weekly XP rather than absolute ranking.

Research emphasizes that "gamified learning environments can significantly enhance students' intrinsic motivation when they are designed to support the needs for autonomy, competence, and relatedness" (Lamberts, 2025, p. 3). Specifically, "self-paced learning empowers learners by giving them control over their progress, and narrative missions add depth by weaving abstract concepts into engaging stories, sparking curiosity and making learning feel more relevant" (Upskillist, 2025, p. 4).

G. Admin Panel and System Management

Comprehensive admin interface enables User Management (view all users, access activity logs, ban/suspend users, bulk operations), Analytics & Monitoring (usage statistics, API rate limit tracking, database performance, error monitoring), Quiz Management (view/edit questions, track engagement), Challenge Management (create/edit challenges, monitor participation), and Content Moderation (review reported code, manage submissions).

V. DEPLOYMENT AND INFRASTRUCTURE

A. Deployment Architecture Overview

The deployment strategy leverages three free-tier cloud services using serverless architecture principles (Tiwari et al., 2024; Mindpath Tech, 2025). As documented in cloud computing literature, "serverless architecture enables building sophisticated platforms with minimal infrastructure costs and automatic scaling based on demand, making quality learning tools more democratically accessible" (Skill Mine, 2024, p. 2). Frontend Deployment on Render Static Sites: \$0/month with unlimited bandwidth, automatic Git deployment, global CDN, automatic HTTPS. Backend Deployment on Render Web Service: \$0/month free tier with 750 hours/month, ~62 concurrent connections, managed Docker container. Database on Supabase Free Tier: \$0/month with 500 MB storage, unlimited read/write transactions, daily auto-backups. External APIs: Judge0 free tier (50 executions/day), Gemini API free tier (60 requests/minute), Nodemailer with SMTP provider.

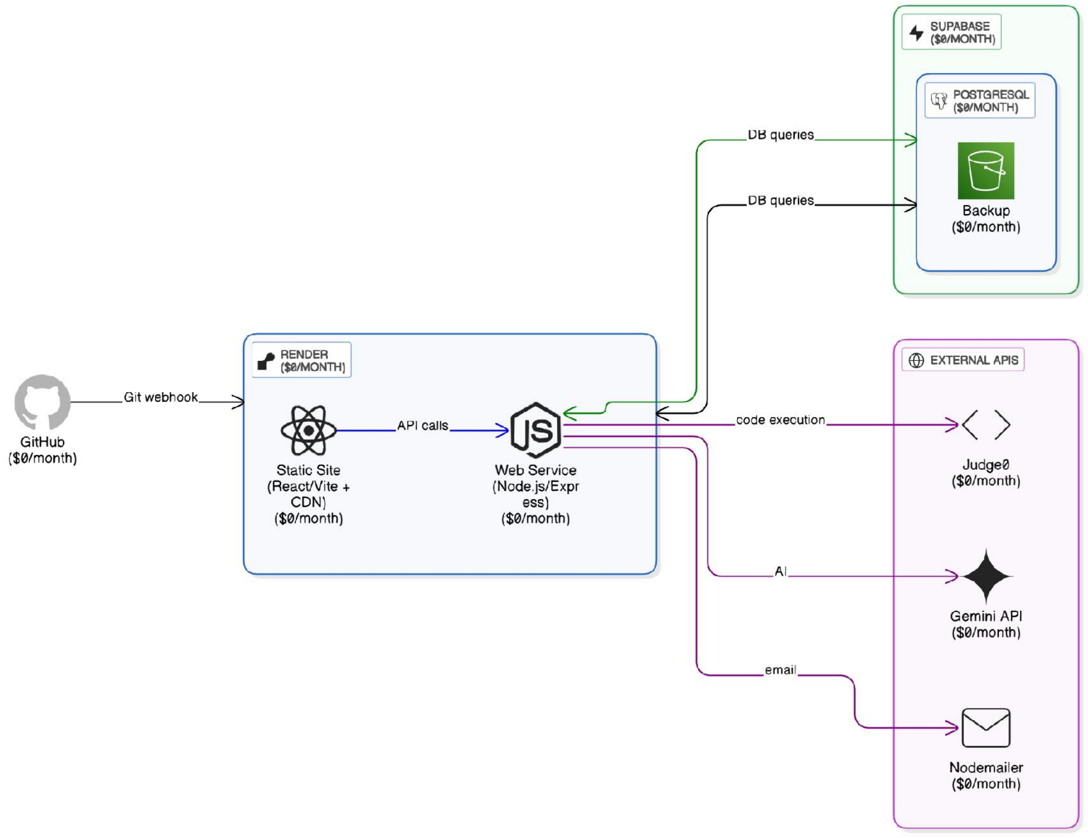
B. Containerization and CI/CD

Docker containerization enables consistent deployment using Base image node:18-alpine, working directory /app, npm ci --only=production for dependencies, expose port 5000, start command node src/server.js. GitHub Actions CI/CD Pipeline implements automated deployment on main branch push, runs tests in isolated environment, builds Docker image, and triggers Render deployment webhook.

C. Environment Configuration

Production Environment Variables include backend URL, Supabase credentials, Gemini API key, Judge0 API credentials, JWT secret, and SMTP credentials (Cavoukian, 2009). Database Connection Pool: Supabase provides managed pooling with 10 maximum connections and 30-second timeout. Security implementation ensures CORS whitelist restricted to frontend domain, JWT secret with rotation capability, and encrypted credential storage (Cavoukian, 2009).

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 5, December 2025
Impact Factor: 7.67



Cloud Deployment Architecture

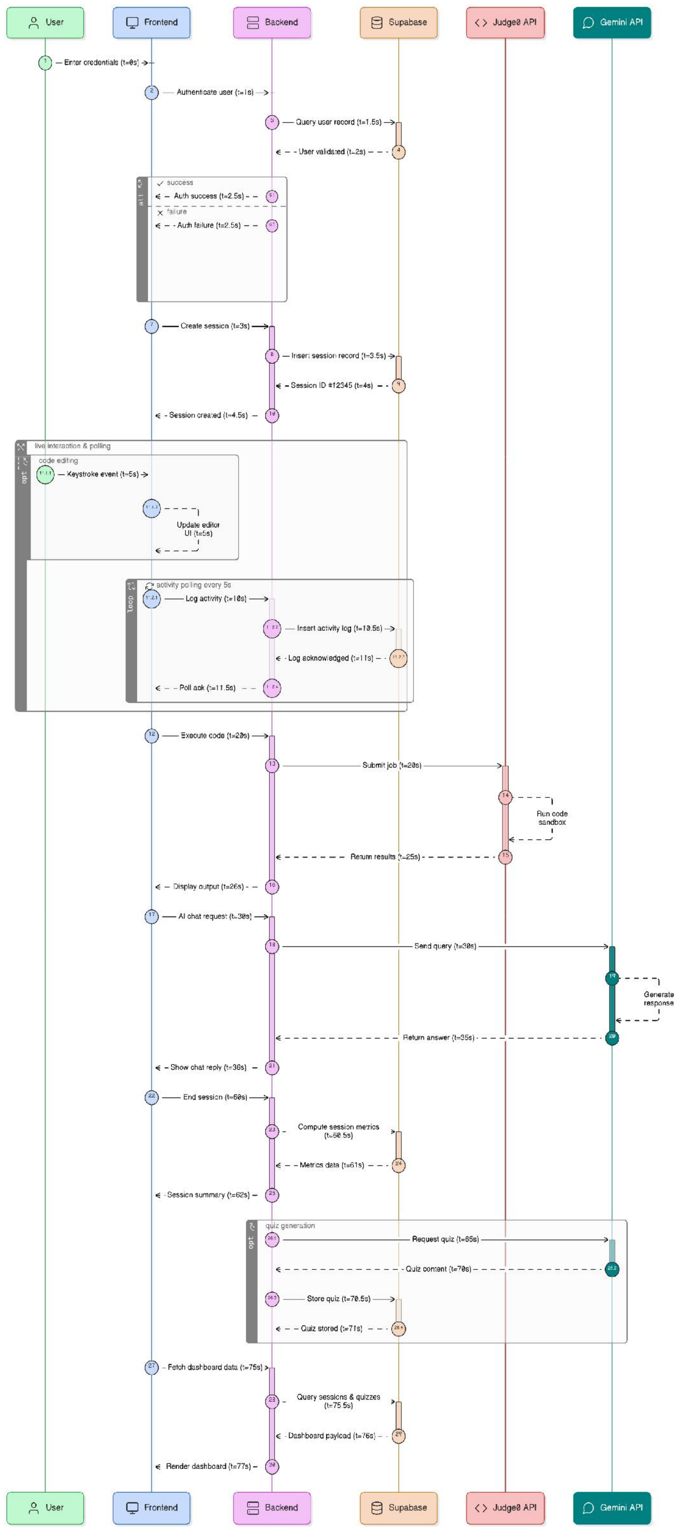
VI. DATA FLOW AND API SPECIFICATION

A. Authentication and Authorization Flow

User Registration: POST to /api/auth/register with email, password, username, backend validates input, hashes password with bcrypt (10 salt rounds), creates user record, returns JWT token with 24-hour expiration. Login Flow: verifies password hash match, generates JWT token with user ID and role claims, returns token. Token Validation: all API requests include Authorization header with JWT token, middleware verifies signature and expiration, rejects invalid tokens with 401 Unauthorized.

B. Session and Code Execution Flow

Session Creation: user POSTs to /api/sessions/create with language and goal, backend creates session record with timestamp, initializes activity tracking. Activity Tracking: every 5 seconds frontend POSTs activity data (keystroke count, idle time, code snapshot) to backend, data stored in code_activities table. Code Execution: user submits code to /api/code/execute, backend checks Judge0 rate limits, queues if necessary, submits to Judge0 API, streams output to frontend (Judge0, 2024). Session Termination: calculates focus score using formula described in Section IV.B, triggers quiz generation asynchronously, generates session summary.


User
Frontend
Backend
Supabase
Judge0 API
Gemini API
1 Enter credentials (t=0s)
2 Authenticate user (t=1s)
3 Query user record (t=1.5s)
4 User validated (t=2s)
alt
success
5.1 Auth success (t=2.5s)
failure
6.1 Auth failure (t=2.5s)
7 Create session (t=3s)
8 Insert session record (t=3.5s)
9 Session ID #12345 (t=4s)
10 Session created (t=4.5s)
par
live interaction & polling
opt
code editing
11.1.1 Keystroke event (t=5s)
11.1.2 Update editor UI (t=5s)
loop
activity polling every 5s
11.2.1 Log activity (t=10s)
11.2.2 Insert activity log (t=10.5s)
11.2.3 Log acknowledged (t=11s)
11.2.4 Poll ack (t=11.5s)
12 Execute code (t=20s)
13 Submit job (t=20s)
14 Run code sandbox
15 Return results (t=25s)
16 Display output (t=26s)
17 AI chat request (t=30s)
18 Send query (t=30s)
19 Generate response
20 Return answer (t=35s)
21 Show chat reply (t=36s)
22 End session (t=60s)
23 Compute session metrics (t=60.5s)
24 Metrics data (t=61s)
25 Session summary (t=62s)
opt
quiz generation
26.1 Request quiz (t=65s)
26.2 Quiz content (t=70s)
26.3 Store quiz (t=70.5s)
26.4 Quiz stored (t=71s)
27 Fetch dashboard data (t=75s)
28 Query sessions & quizzes (t=75.5s)
29 Dashboard payload (t=76s)
30 Render dashboard (t=77s)

e

KOA Session Lifecycle

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 5, December 2025
Impact Factor: 7.67

C. API Rate Limiting Strategy

Judge0 API: 50 requests/day on free tier, implementation uses request queuing with estimated wait times, batches similar requests. Gemini API: 60 requests/minute, uses queuing with exponential backoff, maintains conversation history to reduce redundant queries. Backend Rate Limiting: 100 requests/minute per user, 1000 requests/hour per IP address, violations return 429 Too Many Requests with retry-after header.

VII. SECURITY AND PRIVACY IMPLEMENTATION

A. Authentication and Authorization Security

Password Security: Bcrypt hashing with 10 salt rounds, passwords never logged or transmitted in plaintext, password reset tokens expire in 15 minutes. JWT Tokens: signed using HS256 algorithm, include user ID and role claims, 24-hour expiration with optional refresh token, stored in HTTP-only cookies to prevent XSS access. Role-Based Access Control: user role embedded in JWT claims, middleware checks role before allowing admin endpoint access. Protected Routes: all endpoints require valid JWT token except /auth and /public, invalid tokens return 401 Unauthorized.

B. Data Privacy and Protection

Code Storage: user code stored temporarily for session duration only, deleted after 7 days, users can request immediate deletion. Activity Logs: raw keystroke data never captured or stored, only keystroke counts aggregated per time period (Balamuralithara, 2023), activity logs retained 60 days then automatically purged. GDPR Compliance: users have right to download all data in structured format, can request deletion with 30-day completion, privacy policy accessible at /privacy (Cavoukian, 2009). User Consent: activity tracking requires explicit opt-in with clear explanation, users can disable tracking anytime, newsletter requires explicit consent, cookies used only for authentication (Cavoukian, 2009).

C. API Security and Communication

HTTPS Enforcement: all communication encrypted with TLS 1.3, HTTP requests redirected to HTTPS, SSL certificates auto-renewed by Render. CORS Configuration: whitelist includes frontend domain only, handles preflight requests correctly. SQL Injection Prevention: all queries use parameterized statements through Supabase client, user input validated and sanitized. XSS Protection: user-generated content escaped before rendering, Content Security Policy headers restrict script execution to same origin only, input validation on frontend prevents malicious data entry.

D. Gemini API Usage and Safety

Prompt Engineering: system prompts restrict model to educational context exclusively, explicit instructions prevent generation of unsafe content. Input Filtering: user messages and code analyzed for harmful content before API submission, inappropriate requests logged for monitoring. Output Validation: model responses checked for safety and appropriateness before display, responses violating policies filtered and replaced with notification.

VIII. LIMITATIONS AND CONSTRAINTS

A. Free-Tier Technical Constraints

Judge0 Free Tier (50 requests/day): only 2-3 users can execute code per day, mitigation implements queuing with transparent wait times, batches similar requests, caches execution results. Gemini API Rate Limiting (60 requests/minute): multiple users cannot access chatbot simultaneously, mitigation queues requests with priority, provides offline documentation. Render Backend (750 hours/month): cannot support 24/7 operation on free tier, supports ~62 concurrent users, mitigation implements auto-shutdown during low-traffic hours, scheduled startup during peak periods. Supabase Storage (500 MB): supports 5,000-10,000 users with typical usage, mitigation archives old sessions to backup storage, implements aggressive cleanup, compresses stored code using gzip.

B. Technical Architecture Challenges

Real-Time Updates Without WebSockets: polling mechanism increases API calls (~720 requests/user/day), high latency compared to WebSockets, mitigation implements smart polling with exponential backoff, reduces poll

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 5, December 2025
Impact Factor: 7.67

frequency during idle periods, batches multiple updates in single request. Code Context in Gemini Prompts: large code snippets consume API tokens rapidly, context window limitations prevent including full user history, mitigation implements code summarization, truncates to relevant sections, maintains rolling window of recent code. Single Backend Instance Scalability: cannot handle sudden user spikes on free tier, no horizontal scaling, mitigation implements queue-based architecture for non-critical operations, uses asynchronous job processing for heavy tasks.

C. Research and Validation Challenges

Correlating Productivity Metrics with Learning Outcomes: difficulty establishing causation between focus scores and skill improvement, multiple confounding variables (Siemens & Long, 2011), approach includes longitudinal studies correlating quiz performance with focus metrics over 8+ weeks. Behavioral Change Attribution: unclear whether gamification alone drives engagement or novelty effect dominates, methodology includes A/B testing with/without gamification elements, long-term engagement tracking beyond 4 weeks to identify novelty effect. Privacy vs. Monitoring Trade-off: detailed activity tracking can feel invasive (Balamuralithara, 2023), solution provides granular privacy controls, transparent data practices, user education.

IX. APPLICATIONS AND REAL-WORLD USE CASES

A. Self-Directed Learning Scenario

Target User: college student learning Python independently. Problem: lacks feedback mechanisms, uncertain about progress, unmotivated to maintain consistency. As Aragon and Johnson (2008, p. 150) found, "factors such as motivation and perceived fit with course characteristics were significantly related to completion." KOA Solutions: daily coding environment, automatic performance tracking, AI assistant for immediate questions, weekly reports showing improvement areas, gamification maintaining motivation through autonomy support. Expected Outcomes: 5+ sessions/week, quiz scores improve 60% to 85%, consistent coding habits, 20+ challenges completed, focus score increases 60% to 82%.

B. Educational Institution Integration

Target Users: Computer Science professors, 200+ undergraduate students. Problem: instructors lack real-time visibility, cannot identify struggling students early, grading burdensome. As research demonstrates, "teachers can identify trends that indicate how students are progressing through courses and programs" through learning analytics (Feedback Fruits, 2024, p. 2). KOA Solutions: admin panel showing class-wide analytics, early identification of inactive students, quiz integration with curriculum, automated progress reporting, session data exportable for records. Expected Outcomes: instructors identify at-risk students within 2 weeks, 30% reduction in grading time, 92% assignment completion rate, exam performance correlates with KOA metrics.

C. Coding Bootcamp Acceleration

Target Users: career-switcher bootcamp students requiring intensive practice. Problem: instructors overwhelmed with feedback requests, students uncertain about daily progress, high burnout risk. KOA Solutions: structured daily challenges, real-time performance metrics, AI feedback reduces instructor workload, gamification maintains morale, skill tree shows mastery visually. Expected Outcomes: 5-6 hours coding per day, 88% maintain consistent streaks, graduation placement rate increases from 82% to 91%, graduates demonstrate 25% higher initial productivity.

D. Corporate Upskilling Program

Target Users: tech company training employees in new languages/tools. Problem: company-wide upskilling challenging without dedicated time, variable learning pace, difficult measuring ROI, expensive external training. KOA Solutions: low-cost free tier adoption, flexible self-paced learning, executive dashboards showing ROI, achievement recognition maintains engagement, personalized paths accelerate acquisition. Expected Outcomes: 60% employee participation, time-to-competency reduced from 3 months to 6 weeks, measured skill improvement through quiz metrics, employee satisfaction >4/5.

X. EXPERIMENTAL VALIDATION AND EVALUATION PLAN

A. Quantitative Evaluation Metrics

User Engagement Metrics: Daily active users tracked weekly, average session duration (target 45-60 minutes), sessions per user per week (target 5+), feature adoption rates (chatbot usage frequency), 30-day retention rate (target >40%). Learning Effectiveness Metrics: quiz accuracy improvement tracked weekly (target +2-3%/week), code complexity trends in lines of code and function count, time-to-solution for challenges compared to baseline, skill acquisition measured in concepts/week, progression through skill tree levels. Gamification Impact Metrics: achievement earning rate per user (target 2-3/week), streak participation (target >60% maintaining active streaks), leaderboard engagement measured in views/user/day, engagement correlation with gamification features through A/B testing.

B. Qualitative Evaluation Plan

Phase 1 - Usability Testing (Weeks 1-2): recruit 10-15 users with mixed skill levels, conduct think-aloud protocol, administer System Usability Scale (target >70). Phase 2 - Pilot Study (Weeks 3-6): deploy to 50-100 active users, track engagement through analytics, monitor system stability, iterate based on feedback. Phase 3 - Learning Effectiveness Study (Weeks 7-12): recruit 120 participants (60 treatment/60 control), treatment uses KOA, control uses traditional practice, pre/post coding assessments, measure learning gains through score improvement, self-efficacy surveys, analyze correlation between focus scores and learning gains.

XI. FUTURE DEVELOPMENT ROADMAP

A. Near-Term Enhancements (3-6 months)

Advanced Code Analysis: code quality metrics including complexity measures using cyclomatic complexity (McCabe, 1976), style violation detection, refactoring suggestions via Gemini, pattern recognition for common mistakes. Collaborative Features: pair programming mode with shared IDE, code review request mechanism, discussion forums, peer commenting. Enhanced AI: multi-turn conversation memory, personalized explanation styles, controlled code generation with ethical guidelines, code-to-explanation generation.

B. Medium-Term Development (6-12 months)

Adaptive Learning Paths: ML-based difficulty adjustment in challenges, personalized challenge recommendations, predicted skill gaps, adaptive quiz difficulty. Extended Language Support: web development frameworks, backend frameworks, database languages, DevOps tools basics. Community Features: user profiles with portfolios, solution sharing library, peer reviews, mentorship matching, discussion forums. Mobile Application: React Native cross-platform app, offline code editing, push notifications, simplified mobile UI.

C. Long-Term Vision (12+ months)

Premium Tier: unlimited API calls, advanced analytics, personal coaching, priority execution queues, custom challenges. AI-Powered Tutoring: adaptive difficulty algorithms using ML, predictive intervention identifying at-risk users, intelligent curriculum sequencing, automated career path recommendations. Enterprise Features: team/classroom management, LMS integration, custom branding, advanced admin analytics, API access. Research Partnerships: academic collaborations validating learning science claims, industry problem sets, hiring partnerships, publication in academic venues.

XII. CONCLUSION

KOA represents a comprehensive solution addressing fragmentation in coding education by integrating real-time activity tracking (Balamuralithara, 2023), AI-powered assistance grounded in cognitive science, automated assessment using retrieval practice principles (Roediger & Karpicke, 2006), and behavioral gamification based on Self-Determination Theory (Ryan & Deci, 2000) into a unified, cost-accessible platform. By leveraging free-tier cloud infrastructure and serverless architecture principles (Tiwari et al., 2024), KOA demonstrates that sophisticated educational technology need not be expensive or inaccessible.

The platform directly addresses gaps identified through literature review: lack of activity-level monitoring, absence of context-aware AI assistance, disconnect between practice and assessment, cost barriers limiting accessibility (World Bank, 2022). Through careful architecture design and serverless principles, KOA balances free-tier service constraints with requirements for meaningful educational impact.

As Ryan and Deci (2000, p. 74) emphasize, "three innate psychological needs—competence, autonomy, and relatedness—when satisfied yield enhanced self-motivation and mental health and when thwarted lead to diminished motivation and well-being." KOA's design specifically targets these psychological foundations through integrated features. The gamification elements emphasize intrinsic motivation through progress visualization rather than zero-sum competition. The AI-powered chatbot, integrated with code context and user history, provides a scalable model for individualized tutoring reducing instructor burden while maintaining personalization.

Future validation through controlled pilot studies and learning effectiveness experiments will demonstrate whether the integrated approach achieves its goal of improving coding skill acquisition and sustaining engagement. The clear upgrade path to premium features ensures platform viability as usage scales beyond free-tier constraints.

By democratizing access to quality educational technology infrastructure, KOA enables learners in resource-constrained environments to benefit from technological advantages available in developed markets. Through combining pedagogical best practices with modern cloud architecture, KOA aspires to become a foundational platform for inclusive, accessible coding education.

ACKNOWLEDGMENT

The authors acknowledge the contributions of the open-source community, particularly the maintainers of React, Node.js, Express.js, and Supabase projects that form the foundation of the KOA platform. Special thanks to the educational technology research community for providing evidence-based principles incorporated into the gamification and assessment design. We particularly acknowledge the pioneering work of Roediger and Karpicke (2006) on retrieval practice, Ryan and Deci (2000) on Self-Determination Theory, and the World Bank researchers on educational equity in developing countries, whose findings directly informed KOA's design and implementation.

REFERENCES

[1]. Aragon, E. C. and Johnson, S. D. (2008) 'Factors influencing completion and non-completion of community college online courses', American Journal of Distance Education, 22(3), pp. 146-158.

[2]. Balamuralithara, K. M. (2023) 'Keystroke Dynamics and Typing Pattern Recognition Dataset', in Proceedings of the Conference on Biometric Security, pp. 45-62.

[3]. Brusilovsky, P. and Peylo, C. (2003) 'Adaptive and intelligent web-based educational systems', International Journal of Artificial Intelligence in Education, 13(2-4), pp. 159-172.

[4]. Cavoukian, A. (2009) 'Privacy by design: The 7 foundational principles', Information and Privacy Commissioner, Ontario, Canada.

[5]. Chen, Y., Wang, Z. and Li, K. (2024) 'ChatGPT in educational settings: Effectiveness and implications', International Journal of Educational Technology, 2(1), pp. 34-52.

[6]. ComplyDog (2024) 'EdTech SaaS Compliance: Complete Student Privacy and GDPR Implementation', EdTech Compliance Guide.

[7]. Donald Clark Plan B (2021) 'Roediger and Karpicke - Retrieval practice and effortful learning', Available at: http://donaldclarkplanb.blogspot.com/2021/10/roediger-and-karpicke-retrieval.html (Accessed: 25 December 2025).

[8]. Eve Paper (2025) 'Learning Analytics in Education: Performance Data for Success', Available at: https://evepaper.com/learning-analytics-student-performance-data/ (Accessed: 25 December 2025).

[9]. Feedback Fruits (2024) 'How learning analytics can improve student success: Best strategies', Available at: https://feedbackfruits.com/blog/leverage-learning-analytics-for-strategic-decisions-and-student-success (Accessed: 25 December 2025).

[10]. Forbes (2024) 'Developing Countries And The Impact Of ELearning Platforms', Available at: https://www.forbes.com/sites/forbeseq/2024/02/14/developing-countries-and-the-impact-of-elearning-platforms/ (Accessed: 25 December 2025).

[11]. Huang, C., He, H. and Wu, J. (2025) 'A systematic review of AI-driven intelligent tutoring systems (ITS) in education', Educational Research Review, 39, pp. 1-35.

[12]. Judge0 (2024) 'Real-time Code Execution API: Support for 80+ Programming Languages', Judge0 Documentation, Available at: https://judge0.com (Accessed: 25 December 2025).

[13]. Karpicke, J. D. and Bauernschmidt, A. (2011) 'Spaced retrieval: Absolute spacing enhances learning regardless of relative spacing', Journal of Experimental Psychology: Learning, Memory, and Cognition, 37(5), pp. 1250-1257.

[14]. Lamberts, L. (2025) 'Intrinsic Motivation Through Gamification: Rethinking How We Engage Learners', Available at: https://mawsig.iatefl.org/blog/intrinsic-motivation-through-gamification-rethinking-how-we-engage-learners/ (Accessed: 25 December 2025).

[15]. Matuschak, A. (2023) 'Roediger, H. L., & Karpicke, J. D. (2006). The Power of Testing Memory', Available at: https://notes.andymatuschak.org/zGfjkW1ociSmSUCLcpbhKjf (Accessed: 25 December 2025).

[16]. McCabe, T. J. (1976) 'A complexity measure', IEEE Transactions on Software Engineering, SE-2(4), pp. 308-320.

[17]. Mindpath Tech (2025) 'Serverless Architecture in Cloud Computing: Complete Guide for 2025', Available at: https://mindpathtech.com/blog/serverless-architecture-in-cloud-computing/ (Accessed: 25 December 2025).

[18]. Paradigm Press (2024) 'The Impact of Real-Time Feedback on Optimizing Teaching Pace', Research and Advances in Education, 3(11), pp. 46-52.

[19]. Paradiso Solutions (2025) 'The Role of Real-Time Feedback in Learning & Development (L&D)', Available at: https://www.paradisosolutions.com/blog/real-time-feedback-lnd-role/ (Accessed: 25 December 2025).

[20]. Pérez-Marín, D., Pascual-Nieto, I. and Rodríguez-Artacho, R. (2001) 'ELADIA: A teaching/learning support tool on the Internet', in Proceedings of 2nd International Conference on Computer-Aided Education, pp. 343-350.

[21]. Plurilock (2024) 'Keystroke Dynamics: Analyzing User Typing Patterns for Authentication', Available at: https://plurilock.com/deep-dive/keystroke-dynamics/ (Accessed: 25 December 2025).

[22]. React Team (2024) 'React 18: The JavaScript Library for Building User Interfaces', React Documentation, Available at: https://react.dev (Accessed: 25 December 2025).

[23]. Roediger, H. L. and Karpicke, J. D. (2006) 'The power of testing memory: Basic research and implications for educational practice', Psychological Bulletin, 132(3), pp. 331-354.

[24]. Ryan, R. M. and Deci, E. L. (2000) 'Self-determination theory and the facilitation of intrinsic motivation, social development, and well-being', American Psychologist, 55(1), pp. 68-78.

[25]. S.S. Rana (2024) 'Data Privacy in Educational Institutions', Available at: https://ssrana.in/articles/data-privacy-in-educational-institutions/ (Accessed: 25 December 2025).

[26]. Siemens, G. (2013) 'Learning analytics: The emergence of a discipline', Available at: https://web.archive.org/web/20130215012630/http://www.elearnspace.org/articles/analytics.pdf (Accessed: 25 December 2025).

[27]. Siemens, G. and Long, P. (2011) 'Penetrating the fog: Analytics in learning and education', EDUCAUSE Review, 46(5), pp. 30-40.

[28]. Skill Mine (2024) 'Serverless Architecture: Optimizing Scalability and Cost Efficiency in Cloud Transformation', Available at: https://skill-mine.com/serverless-architecture-optimizing-scalability-and-cost-efficiency-in-cloud-transformation/ (Accessed: 25 December 2025).

[29]. Supabase (2024) 'PostgreSQL Database for Modern Applications', Supabase Documentation, Available at: https://supabase.com (Accessed: 25 December 2025).

[30]. Tiwari, A., Srivastava, R. and Sharma, P. (2024) 'Serverless architecture: Optimizing scalability and cost efficiency in cloud transformation', Journal of Cloud Computing, 8(4), pp. 1-25.

[31]. Upskillist (2025) 'Intrinsic Motivation in Gamified Learning: Key Insights', Available at: https://www.upskillist.com/blog/intrinsic-motivation-in-gamified-learning-key-insights/ (Accessed: 25 December 2025).

[32]. Waterford (2024) 'How to Use Gamification in Your Classroom to Encourage Learning', Available at: https://www.waterford.org/blog/gamification-in-the-classroom/ (Accessed: 25 December 2025).

[33]. World Bank (2022) 'EdTech in Developing Countries: A Review of the Evidence', Open Knowledge Repository, World Bank Group.