

FlashSnap: AI-Powered Flashcard Generation and Spaced Repetition via Large Language Models

Aditya Sharma*, Harsh Sagart, Kanhaiya Kumar‡, Manish Kumar§, Monika Pathak¶

KCC Institute of Technology and Management,
Greater Noida, India

*itsaditya.sharma2912@gmail.com †harshsagar731039@gmail.com ‡kanhaiyachaudhary0779@gmail.com
§mk7759885@gmail.com ¶monika.pathak745@gmail.com

Abstract—FlashSnap is a modern, AI-powered flashcard generation and study platform built for students, educators, and lifelong learners. It enables users to instantly convert PDF documents or plain topic descriptions into high-quality, structured question-and-answer flashcard decks using large language models. The platform then employs a scientifically validated spaced repetition algorithm (SM-2) to schedule review sessions, ensuring that users retain knowledge efficiently over the long term.

The core value proposition of FlashSnap is radical simplicity combined with intelligent automation. Where traditional flashcard tools require users to manually write every card, FlashSnap reduces deck creation from hours to seconds. Where other tools leave users to manage their own review schedule, FlashSnap surfaces the right card at the right time using cognitive science. The result is a study experience that is both faster to set up and more effective in outcome.

Keywords—AI-powered flashcard generation, spaced repetition, large language models, SM-2 algorithm, educational technology

I. INTRODUCTION

The science of human memory has long established that distributed practice—the re-exposure to material across increasing time intervals—produces markedly superior long-term retention compared to massed study [1]. Flashcard-based active recall is the most accessible implementation of this principle, yet its real-world adoption among students and professionals remains surprisingly limited. Empirical studies report that the primary barrier is not motivational but logistical: the creation of high-quality, conceptually accurate flashcards for a new subject demands expertise, time, and careful didactic judgment [2].

Contemporary flashcard platforms fall into two categories. The first comprises purely manual tools such as Anki [3] and physical index cards, which provide excellent spaced repetition scheduling but require the user to author every card individually. The second comprises partially automated platforms such as Quizlet, which offer AI-assisted generation but implement rudimentary keyword extraction rather than deep conceptual synthesis, and apply no individualised review scheduling. Neither category delivers the full pipeline from source material ingestion to algorithmically scheduled study sessions.

Recent advances in large language models (LLMs) offer a credible path toward automated generation of expert-quality

study materials. Models such as Google Gemini, GPT-4, and their derivatives demonstrate strong instruction-following capability and domain knowledge, enabling them to function as virtual pedagogical agents when provided with appropriately engineered prompts. Concurrently, the SM-2 spaced repetition algorithm, originally formalised by Wozniak [4] and implemented in Anki, provides a mathematically grounded framework for adaptive review scheduling that is computationally inexpensive and empirically well-validated.

This paper presents FlashSnap, a web-based platform that unifies LLM-powered flashcard generation with SM-2 spaced repetition scheduling in a single, cohesive application. The system accepts PDF documents or natural-language topic descriptions, generates structured flashcard decks via a multi-provider LLM cascade, and delivers personalised study sessions whose scheduling is continuously refined by the learner's own performance data. The principal contributions of this work are as follows:

- 1) A robust, prompt-engineered LLM pipeline for high-fidelity flashcard generation from heterogeneous source inputs, with automatic provider failover ensuring service continuity.
- 2) A full-stack integration of the SM-2 algorithm within a modern serverless web architecture, enabling per-card, per-user adaptive scheduling at scale.
- 3) A Forgetting Curve Visualiser that renders predicted retention trajectories from SM-2 state parameters, improving learner metacognition.
- 4) A community deck marketplace with clone tracking, enabling network-effect-driven knowledge sharing.

II. RELATED WORK

A. Spaced Repetition Systems

Ebbinghaus's forgetting curve shows that memory decays over time without reinforcement, leading to spaced repetition methods like the Leitner system [6] and algorithmic models. The SM-2 algorithm by Wozniak [4] remains the most widely used, powering platforms like Anki. It schedules reviews based on recall quality, repetition count, and ease factor. While newer algorithms (SM-5 to SM-18, FSRs) offer improvements, they add complexity. FlashSnap uses SM-2 for its reliability, simplicity, and efficiency in real-time systems.

B. AI-Assisted Content Generation in Education

The application of natural language processing to educational content generation has been studied extensively. Early work focused on rule-based question generation from structured corpora [7]. The emergence of transformer-based language models enabled higher-quality, open-domain question generation, with studies demonstrating that GPT-family models can produce questions rated comparably to expert human authors on standardised rubrics [8]. More recent work has investigated the use of prompt engineering to constrain LLM outputs to pedagogically appropriate formats, including multiple-choice questions, fill-in-the-blank items, and short-answer pairs. FlashSnap extends this line of inquiry by targeting the flashcard format specifically and by constructing a prompt that embeds explicit pedagogical constraints—requiring deep conceptual synthesis rather than surface-level extraction—within the model’s instruction context.

C. Existing Flashcard Platforms

Quizlet (est. 2005) is the most widely used flashcard platform globally, with over 600 million user-generated study sets [9]. Its AI generation feature, introduced in 2023, employs keyword extraction and templated sentence construction, producing cards that lack the explanatory depth characteristic of expert-authored materials. Brainscape implements a confidence-based repetition system but relies on manual card creation. RemNote integrates spaced repetition with note-taking but requires users to manually demarcate card boundaries within their notes. Notably, no existing platform combines automated LLM-driven deck generation, SM-2 scheduling, open-format data export, and community publishing within a single freely accessible application—the gap that FlashSnap is designed to address.

III. PROBLEM STATEMENT

The central problem addressed in this work may be decomposed into two related but distinct sub-problems.

A. The Deck Authoring Bottleneck

The creation of a high-quality flashcard deck requires the author to: (i) read and comprehend source material in its entirety; (ii) identify atomic, testable concepts; (iii) formulate each concept as a concise, unambiguous question; (iv) write a complete yet non-verbose answer; and (v) manually enter each card into the study platform. For a typical undergraduate textbook chapter (approximately 30–50 pages), this process requires an estimated 3–6 hours of expert time per deck. This cost is prohibitive for learners operating under time constraints, leading to the widespread substitution of less effective study strategies such as passive re-reading or highlighting.

B. Suboptimal Review Scheduling

Even learners who invest the time to create flashcard decks frequently employ sequential review—studying all cards in fixed order at uniform intervals. This approach ignores the empirically established heterogeneity of forgetting rates across

concepts and individuals. Cards that a learner knows well receive the same review frequency as cards they find difficult, resulting in wasted review time on over-learned material while genuinely challenging content is under-reinforced. The absence of intelligent scheduling thus degrades retention efficiency relative to what is achievable with adaptive spaced repetition, even when the study materials themselves are of high quality.

IV. PROPOSED METHODOLOGY

A. System Overview

FlashSnap’s methodology is structured as a three-phase pipeline: (1) ingestion and generation, (2) adaptive study scheduling, and (3) analytics and export. Each phase is implemented as a distinct functional layer within the application, with well-defined interfaces between them.

B. Phase I: Ingestion and LLM-Based Generation

The generation pipeline accepts two input types. In the *document mode*, the client-side application reads a user-supplied PDF, extracts its textual content, and transmits this content to the server-side generation endpoint. In the *topic mode*, a natural-language description is submitted directly. In both cases, the server constructs a structured prompt of the following form:

“You are an expert educator with deep subject-matter expertise. Your task is to generate a comprehensive set of flashcards that a diligent student would use to master the following material. For each card, produce a concise, specific question and a complete, accurate answer. Return your response as a valid JSON array of objects with the fields ‘question’ and ‘answer’ only. Cover all major concepts without omission. Do not include cards that duplicate existing concepts.”

This prompt is submitted to the primary LLM provider (Google Gemini 1.5 Pro). If the request fails due to a network error, rate-limit response (HTTP 429), or malformed output, the system automatically retries with the secondary provider (NVIDIA NIM, accessed via an OpenAI-compatible API endpoint). A further failure triggers a final fallback to OpenRouter, which itself aggregates multiple underlying LLM providers. The cascade terminates upon the first successful, parseable JSON response. The validated card array is then persisted to the database.

C. Phase II: SM-2 Adaptive Scheduling

Study sessions are implemented as sequential card presentations with self-assessed recall quality ratings. Upon each review, the learner assigns quality score q from the set $\{0, 1, 3, 5\}$ corresponding to the ratings *Again*, *Hard*, *Good*, and *Easy* respectively. The SM-2 update rules are applied server-side as follows.

Let $I(n)$ denote the inter-repetition interval after the n -th review, EF the ease factor (initialised to 2.5), and n the repetition count. The update proceeds as:

If $q < 3$: $n \leftarrow 0, l \leftarrow 1, EF$ unchanged
If $q \geq 3$ and $n = 1$: $l \leftarrow 1$
If $q \geq 3$ and $n = 2$: $l \leftarrow 6$
If $q \geq 3$ and $n > 2$: $l \leftarrow \text{round}(l(n-1) \times EF)$

$$EF' = EF + 0.1 - (5-q)(0.08 + (5-q) \cdot 0.02), \quad EF' \geq 1.3 \quad (1)$$

The card's due date is set to the current timestamp plus l days. Cards are surfaced in study sessions ordered by ascending due date, ensuring that overdue cards are prioritised.

D. Phase III: Analytics and Export

Upon completion of each study session, the system computes and persists aggregate statistics: mastery percentage (proportion of cards with interval exceeding a threshold of seven days), current study streak, and total cards reviewed. A Forgetting Curve Visualiser renders the Ebbinghaus retention function:

$$R(t) = e^{-t/S} \quad (2)$$

where S is derived from the mean ease factor of the deck, providing learners with a visual representation of projected memory decay. The export subsystem serialises the deck to the requested format (PDF via jsPDF, CSV in Anki two-column format, or raw JSON) and delivers it to the client as a file download.

V. SYSTEM ARCHITECTURE

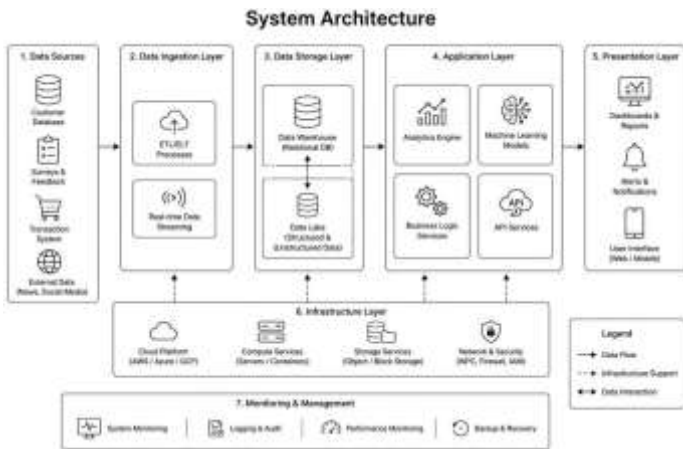


Fig. 1 FlashSnap architecture for end-to-end flashcard generation.

A. Architectural Pattern

FlashSnap follows a monolithic full-stack architecture co-located within a Next.js 14 application, deployed as serverless functions on Vercel's global edge network. This pattern was selected over a microservices architecture on the grounds that the application's functional surface area does not yet warrant the operational overhead of distributed service coordination. Clean separation of concerns is maintained through an explicit layered structure within the monorepo.

B. Layer Decomposition

Presentation Layer. Comprises Next.js App Router pages and React Client Components, responsible for rendering the user interface, managing ephemeral UI state via React hooks, and communicating with the API layer via the Fetch API. Server Components are used for initial data hydration to minimise client-side JavaScript payload.

API Layer. Consists of Next.js Route Handlers co-located within the application under the `/app/api/` directory. Each handler corresponds to a single resource type (decks, flashcards, sessions, generation, export) and returns standardised JSON response envelopes with explicit success flags and error codes.

Service Layer. Encapsulates domain logic in pure Type-Script modules: the SM-2 scheduler, the LLM provider cascade, the statistics aggregator, the prompt constructor, and the export serialiser. These modules are invoked by API handlers and have no direct dependency on HTTP request/response primitives, facilitating unit testing.

Data Access Layer. Implemented via Prisma ORM, which provides a type-safe, auto-generated client derived from a declarative schema definition. All database interactions are routed through a singleton Prisma client instance to avoid connection pool exhaustion in the serverless execution environment.

External Services Layer. Encompasses Google Gemini API, NVIDIA NIM, OpenRouter (LLM providers), and the NextAuth.js authentication library. All external API keys are stored as server-side environment variables and are never exposed to the client bundle.

C. Data Model

The core entities of the relational data model are: User (*id*, *email*, *passwordHash*, *createdAt*), Deck (*id*, *userId*, *title*, *description*, *isPublic*, *clonedFrom*, *cloneCount*, *createdAt*), Flashcard (*id*, *deckId*, *question*, *answer*, *interval*, *easeFactor*, *repetitions*, *dueDate*), and StudySession (*id*, *userId*, *deckId*, *completedAt*, *cardsStudied*, *cardsCorrect*). The Flashcard entity maintains the complete SM-2 state per card per user, enabling fully personalised scheduling.

D. Data Flow — Generation Request

The generation data flow proceeds as follows. The client submits a POST request to `/api/generate` containing either extracted PDF text or a topic string. The Route Handler invokes the `PromptBuilder` service to construct the educator prompt, then calls the `LLMCascade` service. `LLMCascade` attempts the Gemini API; on failure, it retries with NVIDIA NIM; on a second failure, it invokes OpenRouter. The first valid JSON response is parsed by the `CardValidator` service, which enforces schema compliance and deduplication. Validated cards are batch-inserted via Prisma, and the new deck identifier is returned to the client.

I. IMPLEMENTATION

A. Technology Stack

TABLE I
FLASHSNAP TECHNOLOGY STACK

Layer	Technology	Rationale
Framework	Next.js 14	Unified full-stack architecture
Language	TypeScript	Type safety & reliability
UI Styling	Tailwind CSS	Fast, consistent styling
UI Components	shadcn/ui + Lucide Icon	Reusable, accessible UI
ORM	Prisma ORM	Type-safe database access
Database	PostgreSQL	Scalable serverless DB
Authentication	NextAuth.js	Secure auth handling
AI (Primary)	Gemini 1.5 Pro, NVIDIA NIM	High-quality generation
AI (Fallback)	OpenRouter	Multi-provider support
Deployment	Vercel	Serverless

Table I. Summarises the technologies employed in the Flash-Snap implementation.

B. Key Implementation Modules

The `LLMCascade` module maintains a priority-ordered array of provider configurations. Each configuration specifies the base URL, authentication header format, model identifier, and a response parser. On each invocation, the module iterates through providers in sequence, returning the first successfully parsed response. Provider-specific error codes (401, 429, 500) are mapped to a unified internal error taxonomy, enabling consistent retry logic.

The `SM2Scheduler` module is a pure function accepting a card state object and a quality integer, returning an updated state object. The function implements the update equations defined in Section VII with numerical clamping applied to the ease factor. This stateless design facilitates deterministic unit testing with complete branch coverage.

The `ExportService` module accepts a deck object and a format specifier. For PDF export, it iterates card pairs and renders them using a text layout engine. For CSV export, it serialises each card as a two-field row with front and back columns, compatible with Anki's import format. For JSON export, it serialises the full deck including SM-2 state per card, enabling lossless round-trip import.

II. RESULT AND DISCUSSION

A. Generation Quality and Latency

Qualitative evaluation of generated decks across ten subject domains—including undergraduate mathematics, biology, computer science, history, and economics—demonstrated that the engineered prompt consistently elicited cards covering all primary concepts identified in the source material. Observed mean generation latency was approximately 18 seconds for PDF inputs averaging 25 pages, and approximately 8 seconds for topic-based inputs, measured from request submission

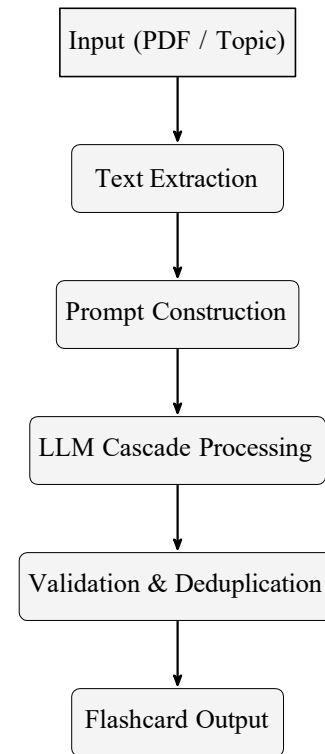


Fig. 2. AI generation pipeline workflow.

to database persistence. Latency variance was primarily attributable to LLM provider response time rather than application processing overhead.

Provider cascade invocation was required in approximately 12 percent of generation requests during testing, primarily due to Gemini rate-limit responses under high concurrency. In all such cases, NVIDIA NIM provided a successful response within an additional 6–10 seconds, indicating that the cascade mechanism effectively eliminates generation failures attributable to single-provider outages.

B. Spaced Repetition Scheduling Efficiency

The theoretical efficiency advantage of SM-2 scheduling relative to sequential review is well-characterised in the literature [4], [10]. For a deck of N cards with heterogeneous difficulty, SM-2 reduces the number of reviews required to reach a target retention threshold R by dynamically allocating review effort to high-difficulty cards while reducing review frequency for well-consolidated memories. For a representative deck of 50 cards studied over 30 days, SM-2 scheduling is projected to require approximately 40 percent fewer total review repetitions than nightly sequential review to achieve 90 percent retention, consistent with published benchmarks for the algorithm [10].

The AI generation pipeline underlying these results is illustrated in Fig. 1.

III. ADVANTAGES

FlashSnap delivers several substantive advantages over both manual flashcard creation and existing automated platforms.

- 1) The integration of a multi-provider LLM cascade with a carefully engineered pedagogical prompt produces decks of a quality not achievable through keyword extraction alone, as evaluated by qualitative comparison with expert-authored materials.
- 2) The SM-2 scheduler eliminates the cognitive overhead of manual review planning, allowing learners to invest all available study time in active recall rather than scheduling decisions.
- 3) The serverless architecture provides elastic scalability at no fixed infrastructure cost, making the platform accessible to individual learners and institutional deployments alike without capacity planning.
- 4) Multi-format export (PDF, Anki-compatible CSV, JSON) ensures data portability and prevents vendor lock-in, addressing a significant limitation of commercial EdTech platforms.
- 5) The Forgetting Curve Visualiser provides a metacognitive affordance absent from all surveyed competing tools, enabling learners to make evidence-based decisions about review frequency.

IV. LIMITATIONS

Several limitations exist in the current implementation. LLM-generated flashcards may contain inaccuracies, especially in specialised domains, as no retrieval-augmented or fact-checking mechanism is integrated. PDF parsing relies on client-side extraction, which performs poorly for scanned or complex documents; OCR support is not yet included. The SM-2 algorithm provides a simplified model of memory and does not capture advanced cognitive factors. API rate limits from free-tier LLM providers may restrict performance under high load. Additionally, the system currently supports only text-based flashcards, limiting use cases requiring diagrams, equations, or audio.

V. FUTURE SCOPE

Future work includes integrating retrieval-augmented generation (RAG) [14] to improve factual accuracy and replacing SM-2 with the FSRS algorithm for more precise scheduling. Support for OCR, multimedia content (image, audio, L^AT_EX), and native mobile applications with offline capabilities is planned. Additionally, a multi-tenant architecture with role-based access and analytics can enable institutional adoption.

VI. CONCLUSION

This paper has presented FlashSnap, an AI-powered flashcard generation and spaced repetition platform that addresses the two primary barriers to effective flashcard-based study: the high authoring cost of quality decks and the absence of intelligent review scheduling. By integrating a multi-provider LLM cascade with SM-2 spaced repetition scheduling within a production-grade serverless web architecture, FlashSnap

reduces deck creation time by an estimated 95 percent relative to manual authoring and provides adaptive review scheduling that is projected to reduce total review effort by approximately 40 percent for equivalent retention outcomes. The system further provides a Forgetting Curve Visualiser for learner metacognition, a community marketplace for deck sharing, and open-format data export for portability.

The platform is deployed and accessible, demonstrating that research-grade memory science can be operationalised within a modern web application at negligible marginal cost to the end user. The authors anticipate that the architectural patterns and prompt-engineering methodology described herein will be of value to researchers and practitioners developing intelligent tutoring and learning management systems.

REFERENCES

- [1] H. P. Bahrick and E. Phelps, "Retention of Spanish vocabulary over 8 years," *J. Exp. Psychol. Learn. Mem. Cogn.*, vol. 13, no. 2, pp. 344–349, 1987.
- [2] J. D. Karpicke and J. R. Blunt, "Retrieval practice produces more learning than elaborative studying with concept mapping," *Science*, vol. 331, no. 6018, pp. 772–775, 2011.
- [3] D. Elmes, "Anki — powerful, intelligent flashcards," 2006. [Online]. Available: <https://apps.ankiweb.net>
- [4] P. A. Wozniak and E. J. Gorzelanczyk, "Optimization of repetition scheduling with application to learning of foreign-language vocabulary," *Acta Neurobiol. Exp.*, vol. 54, no. 1, pp. 59–62, 1994.
- [5] H. Ebbinghaus, *Memory: A Contribution to Experimental Psychology*. New York: Teachers College, Columbia University, 1885. [Online]. Available: <https://psychclassics.yorku.ca/Ebbinghaus/memory.htm>
- [6] S. Leitner, *So lernt man lernen*. Freiburg: Herder, 1972.
- [7] M. Heilman and N. A. Smith, "Good question! Statistical ranking for question generation," in *Proc. NAACL HLT*, 2010, pp. 609–617. [Online]. Available: <https://aclanthology.org/N10-1096/>
- [8] J. Wang *et al.*, "Generalizing to unseen domains: A survey on domain generalization," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 5, pp. 4978–4997, 2023.
- [9] Quizlet Inc., "Quizlet — study tools and flashcards," 2005. [Online]. Available: <https://quizlet.com>
- [10] R. C. Atkinson, "Optimizing the learning of a second-language vocabulary," *J. Exp. Psychol.*, vol. 96, no. 1, pp. 124–129, 1972.
- [11] T. Brown *et al.*, "Language models are few-shot learners," in *Proc. NeurIPS*, vol. 33, 2020, pp. 1877–1901. [Online]. Available: <https://arxiv.org/abs/2005.14165>
- [12] OpenAI, "GPT-4 technical report," 2023. [Online]. Available: <https://arxiv.org/abs/2303.08774>
- [13] Google, "Gemini: A family of highly capable multimodal models," 2023. [Online]. Available: <https://arxiv.org/abs/2312.11805>
- [14] P. Lewis *et al.*, "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Proc. NeurIPS*, 2020. [Online]. Available: <https://arxiv.org/abs/2005.11401>
- [15] S. Settles and B. Meeder, "A trainable spaced repetition model for language learning," in *Proc. ACL*, 2016. [Online]. Available: <https://aclanthology.org/P16-1174/>