FlashMind: Spaced-Repetition Flashcard Platform

Filip Łysak
University of Luxembourg
Email: filip.lysak.001@student.uni.lu

Prince Yaw Gharbin
University of Luxembourg
Email: prince.gharbin@uni.lu

Christoph Schommer
University of Luxembourg
Email: christoph.schommer@uni.lu

Abstract—**This report presents *FlashMind*, a zero-cost web platform that operationalizes spaced repetition for vocabulary and concept learning. The system pairs a modern client-first stack (React/Vite, TypeScript, Tailwind) with a free PostgreSQL backend (Supabase) to deliver**
(i) deck and card management,
(ii) an adaptive scheduler based on the SM-2 family of algorithms,
(iii) a study workflow in which users attempt recall, reveal the answer, and rate difficulty, and
(iv) transparent analytics for due cards, streaks, accuracy, and backlog. Design choices are grounded in established findings on the spacing effect, forgetting dynamics, and graduated-interval recall. We document the scheduling rule implemented, the event-sourced data model that underpins all analytics, and a reproducible assessment plan (calibration plots, efficiency curves, workload trends). The result is a transparent, maintainable system that turns well-replicated memory science into a reliable daily study routine while remaining free to run and easy to extend.

1. Introduction

This project delivers a free, browser-based spaced-repetition platform that helps learners retain vocabulary and concepts with minimal friction. Instead of cramming, the app schedules reviews at increasing intervals based on how well each item is recalled. During a session, the user sees one card at a time, attempts recall, reveals the answer, and records a 0–5 rating; the system then computes the next review date and logs the event. Over time, easy items drift further into the future while difficult ones reappear sooner, producing efficient, personalized practice.

The product focuses on three objectives:
(1) a clean study workflow that stays out of the learner's way;
(2) transparent analytics—cards due today, accuracy, streaks, backlog, weekly activity—calculated only from real review data; and
(3) portability through simple deck import/export for solo learners or classrooms.
The implementation is deliberately pragmatic and zero-cost: a React/TypeScript front end (Vite, Tailwind) talks directly to a free Postgres backend (Supabase), storing decks, cards, and review records. The scheduler adopts a documented SM-2–style baseline so its behavior is easy to explain and adjust later. The result is a practical foundation that can be deployed quickly, used immediately, and extended toward more advanced scheduling when needed.

2. Project description

2.1. Objectives and non-goals

Objectives (what we must deliver).

- A complete learning loop that feels natural: make cards and decks $\rightarrow$ study one card at a time (rate 0–5) $\rightarrow$ schedule the next review automatically $\rightarrow$ show honest, simple analytics.
- Run at zero cost: static front end on a free host, and a free PostgreSQL tier (Supabase) with no custom server to maintain.
- Keep scheduling explainable: use a documented SM-2–style rule with clear rating meanings.[1]
- Stay portable: allow export/import of decks and (optionally) review history in simple formats.[2]

Non-goals for the MVP (what is skipped intentionally for now).
No multi-user authentication or cohort dashboards; no heavy media hosting; no advanced AI deck generation. These can come later without changing the basic workflow.

2.2. Study flow

One study action creates one review event. The loop is intentionally simple:

Card $\rightarrow$ Study $\rightarrow$ Review event $\rightarrow$ Dashboard
User see a prompt, try to recall, reveal the answer, and rate how it felt (0 = miss, 5 = very easy). The app saves that rating, sets the next time this card will appear, and moves on. All dashboard numbers come from these saved events (no placeholders or seeded values).

1. SM-2 algorithm (SuperMemo)
2. JSON ; CSV

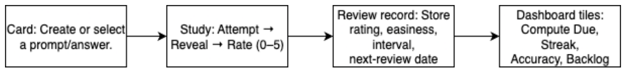

Card: Create or select a prompt/answer.
Study: Attempt → Reveal → Rate (0–5)
Review record: Store rating, easiness, interval, next-review date
Dashboard tiles: Compute Due, Streak, Accuracy, Backlog

Figure 1: Value stream: Card → Study → Review record → Analytics.

2.3. Functional scope

Deck & card management.
Create, edit, and delete decks. Add cards with `front` (prompt), `back` (answer), an optional `example`, and simple flags for media. Bulk add (paste/CSV) is optional but the data model supports it.

Study workflow.
See one card at a time, keyboard-friendly, and always rate 0–5 *after* reveal so the rating reflects actual retrieval.

Scheduling.
Use an SM-2–style rule to update each card's *easiness* and the next *interval* based on your rating, then compute the `next_review_date` and save one review event. The rule stays simple and explainable (see Section "Scheduling model" below).

Card selection (what shows next).
Order by urgency: *overdue* first (oldest first), then *due today* (earliest first), then a few *never-reviewed* items, and finally *not yet due* if the queue is short (with a little randomisation to reduce monotony).

Analytics (always from real data).
Tiles: Due Today, Streak, Cards Mastered (e.g., last three ratings ≥ 4), Accuracy (30d; ratings ≥ 4), Average Session (rolling). Chart: 7-day progress (reviews/day and first-time reviews/day).

Import/export.
JSON for full-fidelity round-trips; optional CSV for quick edits or moving content; export can include history so anyone can rebuild current schedules exactly.

Explainability.
Optional "Why now?" hints in plain language (last interval, recent ratings, and the new interval).

2.4. Scientific assumptions (brief, practical)

The design uses three well-replicated ideas:
(1) *Spacing improves retention* and the "best gap" grows with your target horizon (the lag effect).[3]
(2) *Retrieval strengthens memory* more than restudy; some effort is good.[4]
(3) *Items differ*, so each card keeps its own easiness and interval that adjust over time.[5]

3. Spacing ridgeline (Cepeda et al., 2008)
4. Testing effect review (Roediger & Karpicke, 2006)
5. Anki background: SM-2 and FSRS

2.5. Data model

We store three things, and nothing more:
(i) **decks**, (ii) **cards**, and (iii) an **append-only** list of **review events**.

- `decks(id, name, description, created_at, ...)`
- `cards(id, deck_id, front, back, example, has_image, has_audio, created_at, ...)`
- `review_records(id, card_id, rating, ease_factor, interval_days, next_review_date, created_at)`

The current schedule of a card is always read from its *latest* review record's `next_review_date`. This "event-sourced" approach keeps writes simple, analytics exact, and history honest.

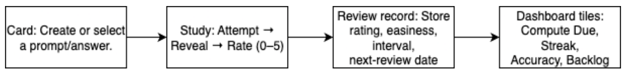

Card: Create or select a prompt/answer.
Study: Attempt → Reveal → Rate (0–5)
Review record: Store rating, easiness, interval, next-review date
Dashboard tiles: Compute Due, Streak, Accuracy, Backlog

Figure 2: Mini-ERD: one Deck has many Cards; one Card has many Review records. Key fields are highlighted.

2.6. Scheduling model (as implemented)

The idea in plain words.
If a card feels hard, bring it back soon. If it feels easy, wait longer next time. A floor on "easiness" stops very hard cards from getting stuck in tiny loops. The rule follows the SM-2 family and stays easy to explain.[6]

What your rating does.

- *Rate 0–2 (struggle).* Treat this as overshoot. Reset repetitions, set a short next gap (about one day), and nudge the card's easiness slightly down (but never below the floor, ≈ 1.3).
- *Rate 3–5 (success).* Count a success. Use two safe "seed" gaps first (about 1 day, then 6 days). After that, grow the gap by multiplying the previous gap by the card's easiness (rounded to whole days). Nudge easiness a little up or down based on the exact rating.
- *Record it.* Save exactly one review event with: the rating, the updated easiness, the chosen gap (days), and the computed `next_review_date`.

Which card shows next.
Order by urgency: *overdue* first (oldest first), then *due today* (earliest first), then a few *never-reviewed* cards, and finally *not yet due* if the queue is short (light randomisation inside ties). This keeps you near the "edge of forgetting" without feeling repetitive.

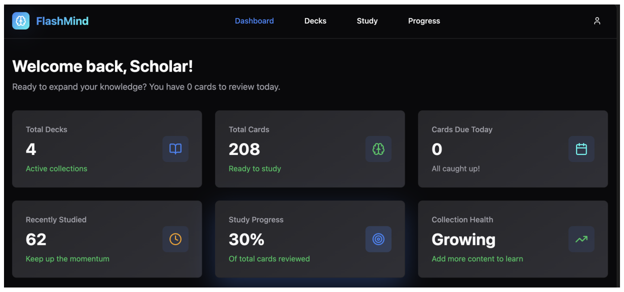
6. SM-2 algorithm (spec)

Cold start and late reviews.
New cards start with a neutral easiness and short early gaps so the system can "learn" how hard they are for you. If you see a card late and still recall it, the next gap can be longer; if you miss it, the next gap becomes short again. Both behaviors follow automatically from the same rule.[7]

2.7. Analytics: what the tiles mean and how we compute them

All analytics are computed from the same list of review events you create while studying (no simulated numbers).

Cards Due Today — cards with no prior reviews, or whose latest next-review date falls before the end of your local day.
Backlog — cards whose latest next-review date is already in the past (overdue now).
Streak (days) — how many days in a row you did at least one review (counted in your local day; if today is empty, the streak shows up to yesterday).
Accuracy (30d) — the share of ratings 4 or 5 among all ratings in the last 30 days.
Average Session (minutes) — for each day you studied: time from first to last review; we show the average over the last two weeks of active days.
Weekly Progress — a 7-day chart with (i) total reviews per day and (ii) how many cards were seen for the first time that day.



Welcome back, Scholar!
Ready to expand your knowledge? You have 0 cards to review today.

Total Decks
4
Active collections

Total Cards
208
Ready to study

Cards Due Today
0
All caught up!

Recently Studied
62
Keep up the momentum

Study Progress
30%
Of total cards reviewed

Collection Health
Growing
Add more content to learn

Figure 3: Analytics overview with formal definitions matched to on-screen tiles and chart.

2.8. Quality attributes and constraints

Cost & deployability.
Static front end on a free host; free PostgreSQL via Supabase.[8] Configuration (project URL and publishable key) is set as environment variables at deploy time.

Performance.
Card advance should feel instant ($< 200\,\text{ms}$). Use indexes for "latest review" and for due/backlog scans (e.g., `(card_id, created_at desc)` and `(next_review_date)`). Window analytics (e.g., last 6 months) to stay within free-tier limits.

Privacy & security.
The MVP stores no PII. If/when multi-user is enabled, turn on Auth and Row-Level Security (RLS) so each user only sees their rows.[9]

Accessibility.
Keyboard navigation for study (reveal, rate, next), semantic labels for buttons, and adequate colour contrast in charts.

Internationalization.
User content is UTF-8. UI strings are ready for translation. Store timestamps in UTC; display "today" and streaks in the user's local time.

Reliability.
Writes are append-only review events, which simplifies recovery. Regular JSON/CSV exports are encouraged as backups.

2.9. Risk register and mitigations

Free-tier limits.
Mitigate: paginate, throttle, and batch reads; cache the current session's queue in memory.

Clock & time zones.
Mitigate: store timestamps in UTC; compute "today" in the user's local time on the client.

Runaway backlog.
Mitigate: surface backlog early; suggest modest daily goals; optionally cap "new cards per day."

User trust.
Mitigate: show "Why now?" explanations and per-card histories so the schedule is easy to understand.

2.10. Traceability (objectives → features → metrics)

7. Anki background: SM-2 and FSRS
8. Supabase docs
9. Row-Level Security guide

TABLE 1: Traceability from objectives to features and verification metrics.

Objective	Feature	Evidence / Metric
Adaptive practice	Rating-based scheduler	Accuracy (30d); success rates by interval; interval growth patterns over time
Transparency	Dashboard + per-card history	Due Today, Backlog, Streak; per-card timelines and "Why now?" explanations
Zero cost	Browser + free DB	Deployed on free host; no server bills; free PostgreSQL tier (Supabase)
Portability	JSON/CSV import/export	Round-trip of decks and histories verified on import

3. Scientific Deliverable

3.1. Design

Aim.. The aim is to maximise long-term retention per unit time by practising near the *edge of forgetting*—neither so soon that a review is wasted nor so late that failure is likely. The design rests on three well-replicated findings:

(i) **Spacing effect:** distributed practice outperforms massed practice.[10]
(ii) **Testing effect:** active retrieval strengthens memory more than restudy.[11]
(iii) **Desirable difficulties:** effortful but successful retrieval yields deeper learning.[12]

Forgetting and horizon sensitivity.. In the absence of rehearsal, memory strength typically declines with time. A timely review refreshes the trace and slows subsequent decay. The *optimal interstudy interval* (ISI) depends on the intended *retention interval* (RI): as the target horizon increases, the recommended gap increases sub-linearly.[13][14] Consequently, schedules should be horizon-aware rather than fixed-gap.

Retrieval as measurement and intervention.. Each trial both *measures* current strength (via success and subjective difficulty) and *induces* learning (retrieval with feedback). To maintain interpretability, phases are separated: prompt $\rightarrow$ silent attempt $\rightarrow$ reveal $\rightarrow$ scalar rating (0–5). The rating acts as a sufficient statistic for scheduling toward a target recall band at the next review.

10. Cepeda et al. (2006) meta-analysis on distributed practice.
11. Roediger & Karpicke (2006) on the testing effect.
12. Bjork & Bjork (2011).
13. Ebbinghaus (1885/1913).
14. Ridgeline evidence for ISI–RI coupling.

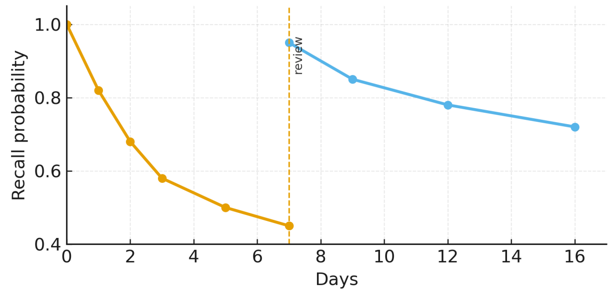


Figure 4: Forgetting curve with a review that resets recall high and slows subsequent decay.

Item-level adaptation.. Heterogeneity across items and learners is handled with per-item latent values capturing *easiness* and *stability*. After each rating: easy items migrate to longer lags; difficult items recur sooner until stabilised; successful retrieval at longer lags produces larger stability gains than trivially short lags.

Scheduling rule (baseline family).. A transparent SM-2 family rule is adopted for the baseline:[15]

(i) maintain an **easiness factor (EF)** per item with a lower bound (e.g., $\approx$1.3);
(ii) if the rating $<$ 3: reset repetitions and set a short repair gap (e.g., $\sim$1 day);
(iii) if the rating $\geq$ 3: count a success; use seed gaps (1 day, then 6 days), then multiply the previous gap by EF (rounded to whole days);
(iv) record one review event containing rating, updated EF, chosen interval, and computed next-review date.

This baseline guarantees monotonicity (better performance $\Rightarrow$ longer spacing), boundedness (EF floor), and recoverability (misses trigger short gaps).

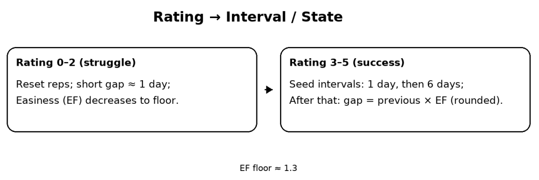

Rating → Interval / State
Rating 0–2 (struggle)
Reset reps; short gap ≈ 1 day;
Easiness (EF) decreases to floor.
Rating 3–5 (success)
Seed intervals: 1 day, then 6 days;
After that: gap = previous × EF (rounded).
EF floor ≈ 1.3

Figure 5: Mapping from rating to the next interval and state under the SM-2 family.

Selection policy (what to show next).. Card presentation is ranked by urgency to maintain practice near the edge of forgetting:

(1) **Overdue** — oldest first,
(2) **Due today** — earliest first,

15. SuperMemo: Algorithm SM-2 (official description).

(3) **Never reviewed** — introduced gradually to avoid starvation,

(4) **Not yet due** — considered only when the queue is short (light randomisation within ties).

Cold-start and late reviews.. New items initialise with neutral priors (EF near 2.5; short early gaps) and adapt rapidly over the first few observations. For overdue reviews: a correct response implies greater-than-expected stability and merits a longer next gap; a failure implies overshoot and triggers a short repair gap. Both behaviours arise directly from the rating-based updates.

Alternative models (for context).. The baseline relates to several established models:

(i) **Leitner/Pimsleur:** fixed, graduated intervals (high face validity; limited adaptivity).[16]

(ii) **Half-life regression (HLR):** recall $\approx 2^{-d/\mathrm{HL}}$ with a learned half-life, effective given sufficient data.[17]

(iii) **FSRS and related open models:** separate *difficulty*, *stability*, and *retrievability* and choose intervals by optimising retention versus workload.[18]

These approaches provide upgrade paths that reuse the same event log without altering the learner-facing loop.

Outcome framing and hypotheses.. Given a target horizon, the next interval is chosen so that predicted recall at that time lies within a healthy band (mid-to-high success). From this framing follow testable hypotheses:

(H1) Adaptive spacing surpasses fixed gaps on long-term retention per unit time.

(H2) Longer horizons shift optimal gaps outward in a sub-linear fashion.

(H3) Ratings concentrate in a mid-to-high range without ceiling or floor effects.

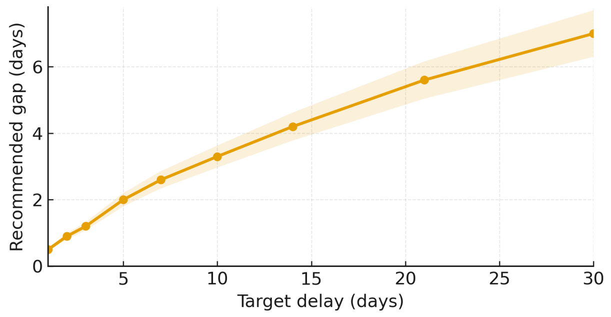


Figure 6: Ridgeline intuition: as target horizon increases, recommended interstudy gaps increase sub-linearly.

16. Testing-effect review; desirable difficulties.
17. Settles & Meeder (2016): Half-Life Regression.
18. FSRS (open spaced-repetition scheduler).

3.2. Production (scientific operationalisation)

Scope and purpose.. This subsection formalises how constructs from memory science (spacing, forgetting, retrieval practice, desirable difficulty) map to observable data, item-level state variables, and review-time decisions. Emphasis is placed on definitions, assumptions, and decision rules rather than software specifics. The description permits inspection of why a card is scheduled at a particular time and how dashboard analytics relate to underlying measures.

P1. Observables, latent states, and target performance. **Observables.** For each item i at interaction time t, the following are recorded:

(i) rating $q_{i,t} \in \{0,1,2,3,4,5\}$ given after an attempted recall and feedback (0 = miss . . . 5 = very easy),

(ii) elapsed time since the previous review, $\Delta t_{i,t}$,

(iii) wall-clock timestamp $T_{i,t}$ (stored in UTC to avoid time-zone ambiguity).

Latent state. Two per-item quantities summarise memory for scheduling:

(i) *Easiness* $EF_{i,t}$, which rises with easy successes and falls with difficult or failed recalls,

(ii) *Interval* $I_{i,t}$, the planned delay in days until the next review.

These variables reproduce the baseline scheduling behaviour and admit clear explanation.

Target performance band. The decision objective is to keep predicted recall at the next review within a band $p^{\star} \in [0.7, 0.9]$. Higher bands (e.g., 0.85–0.95) suit short-horizon, high-stakes contexts; moderate bands (e.g., 0.70–0.85) suit durable mastery, reflecting the lag–retention trade-offs in spacing research.[19]

P2. Trial structure and measurement hygiene. Ratings act as interpretable proxies for retrieval difficulty when each trial is structured into four phases:

(i) prompt exposure,

(ii) silent recall attempt,

(iii) answer reveal,

(iv) scalar rating (0–5).

Separating the attempt from the reveal avoids contaminating the difficulty signal with additional exposure, consistent with evidence that retrieval practice (testing) strengthens memory and that retrieval effort is consequential.[20]

19. Cepeda et al., 2008, "Spacing Effects in Learning".
20. Roediger & Karpicke, 2006, review of the testing effect.

P3. Update rule (SM-2 baseline; implemented). A documented SM-2–style update rule is used for transparency and broad familiarity.[21] After rating q:

$$EF' \leftarrow \max\bigl(1.3,\ EF + 0.1 - (5-q)\,\bigl(0.08 + 0.02(5-q)\bigr)\bigr),$$

with a floor at 1.3 to prevent degeneracy. Intervals follow:

$$I_1{=}1, \quad I_2{=}6, \quad I_n{=}\mathrm{round}(I_{n-1}{\cdot}EF') \quad forn \geq 3whenq \geq 3.$$

If $q < 3$, repetitions reset and the next interval is short (approximately one day). The rule satisfies:

(i) *monotonicity* — higher performance ⇒ longer spacing,
(ii) *stability* — EF bounded below,
(iii) *recoverability* — failures pull items back quickly.

This baseline generates all schedules and underpins the analytics.

P4. Selection policy (next-item priority). Next-item choice is a ranking on *urgency*:

(i) **Overdue** (past due), ordered oldest first,
(ii) **Due today**, ordered by earliest next review,
(iii) **Never reviewed**, to introduce new material steadily,
(iv) **Not yet due**, sampled only when the queue is short (light randomisation within ties).

This ordering presents material near the edge of forgetting without requiring item-wise predictive models.

P5. Cold-start and overdue cases. Cold-start items use neutral priors ($EF \approx 2.5$; short initial intervals) and adapt quickly as evidence accumulates. For late (overdue) reviews:

(i) a correct response implies higher-than-estimated stability and may warrant a longer next gap,
(ii) a failure implies overshoot and triggers a short repair gap.

Both behaviours follow directly from the update rule in P3.

P6. Horizon sensitivity and target difficulty. Spacing benefits depend on the retention horizon. The optimal interstudy interval (ISI) tends to increase with the desired retention interval, with diminishing returns at longer horizons.[22] In practice, horizon sensitivity is operationalised through the choice of $p^{\star}$: shorter horizons use higher target recall; longer horizons tolerate more ambitious gaps. The SM-2 family approximates this by lengthening intervals for items that succeed repeatedly at longer lags.

21. SuperMemo: Algorithm SM-2 (official description).
22. Cepeda et al., 2006, meta-analysis of distributed practice.

P7. Analytic definitions (derived from the event log). All analytics are defined strictly on the append-only stream of review events:

- **Cards due today:** items with no reviews or with most recent next-review date $\leq$ end of local day.
- **Backlog:** items whose most recent next-review date $<$ current time.
- **Streak (days):** consecutive local days with ≥ 1 review; if today has none, the count extends to yesterday.
- **Accuracy (30d):** proportion of ratings ≥ 4 among all ratings in the last 30 days.
- **Average session:** for each active day, $\max(T) - \min(T)$; the dashboard reports an average over the last 14 active days.
- **Weekly progress:** 7-day series of (i) total reviews per day and (ii) *new cards per day* (first-ever review).

Definitions align with the scientific objective of sustainable, horizon-aware spacing and make deviations (e.g., rising backlog) immediately visible.

P8. Validity and reliability considerations. Internal validity. The four-phase trial design (attempt $\rightarrow$ reveal $\rightarrow$ rating) preserves the meaning of ratings by avoiding extra exposures prior to judgement. Clear rating anchors (e.g., "0 = miss", "3 = correct with effort", "5 = very easy") reduce variability.
Construct validity. Ratings act as proxies for retrieval strength and effort. Healthy distributions cluster in the mid-to-high range while retaining some failures, consistent with desirable difficulty.[23] Persistent ceiling suggests gaps are too short; persistent floor suggests gaps are too long.
External validity. The cue–target abstraction is modality-agnostic (text, audio, image). The schedule adapts from outcomes, not domain assumptions.
Potential biases. Ratings may drift (e.g., generous scoring). Mitigations include per-card histories and brief "why now?" explanations that support calibration.

P9. Ethical and pedagogical posture. Transparency. Explanations such as "Reappeared after 6 days following recent 'Good/Easy' ratings" promote metacognitive awareness and trust.
Sustainability. The interface surfaces backlog early, recommends daily targets, and allows capping of "new cards per day" to avoid punitive dynamics.
Privacy. Scientific conclusions rely on event data (timestamps, item identifiers, ratings); personally identifying information is unnecessary.

P10. Evaluation (scientific checks, implementation-agnostic). Calibration. Scheduled intervals are binned (e.g., 1–2d, 3–5d, 6–10d, 11–20d, 21+d) and observed success rates (ratings ≥ 4) are plotted against a shaded target band. Well-calibrated schedules place most bins within the band.

23. Bjork & Bjork, 2011.

Counterfactual replay. Using the same event history, alternative schedulers are simulated and retention per unit time is compared under identical review budgets. Two grounded comparators are:

(i) *Half-life regression* (HLR): $p(d) = 2^{-d/\mathrm{HL}}$ with HL estimated from history; the next interval targets $p^{\star}$.[24]

(ii) *FSRS*: separates *difficulty*, *stability*, and *retrievability*; intervals are chosen by optimising isoretention under a review budget.[25]

Efficiency comparison. For baseline and comparator policies, "recalls per minute" is plotted against "reviews per day" to visualise time–benefit trade-offs.
Behavioural sustainability. Streaks, backlog trajectories, and average session duration are monitored. Stable streaks with non-increasing backlog indicate alignment between scheduling difficulty and available time.

P11. Reproducibility and reporting. Event-sourced evidence. All conclusions are reproducible from the append-only review log $(i, T, q, EF, I, nextdue)$.
Pre-specification. Prior to analysis, the target band $p^{\star}$, time windows (e.g., 30 days), and bin edges for calibration are fixed to avoid post-hoc tuning.
Open materials. An anonymised export of review events, together with exact metric definitions, enables independent verification.

Summary. Ratings and delays constitute the empirical inputs; EF and I are the succinct item-level summaries; the SM-2 rule implements a horizon-sensitive policy that aims to keep recall within a target band. Analytics derive from the same event stream, ensuring that reported progress reflects authentic behaviour.

3.3. Assessment

A. Evaluation goals. We evaluate whether the scheduling policy (i) improves long-term recall *per unit time* and (ii) maintains a sustainable daily workload (no growing backlog), while remaining explainable.

B. Outcome metrics. Table 2 defines the primary outcomes computed from the append-only review log. Figure 7 shows the intended *calibration* view (empirical success by scheduled interval, with the target band shaded). Figure 8 illustrates the *efficiency frontier* (recalls per minute vs. review volume).

C. Study designs.

C1. Offline counterfactual replay.. Re-run the existing event history under alternative policies with the *same* review budget (Baseline SM-2; Fixed-gap; optionally HLR/FSRS). Compare retention and efficiency; report bootstrap CIs.

24. Settles & Meeder, 2016, half-life regression.
25. FSRS (open-source scheduler).

Metric	Definition (window)
Retention @ delay	Share of items recalled at pre-set lags (e.g., 7/30 days); report with Wilson CIs
Efficiency	Correct recalls per minute (or per session) over the same window
Calibration	For interval bins (1–2d, 3–5d, 6–10d, ...), empirical success vs. target band (e.g., 70–90%)
Workload health	Backlog trajectory (slope of overdue count) and streak stability (consecutive days with ≥ 1 review)

TABLE 2: Primary outcomes derived from the review event log (no placeholders).

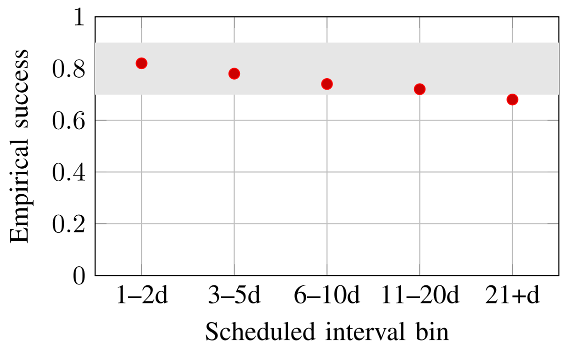


Figure 7: Calibration: empirical success rate by interval bin with target band shaded. Replace points with observed rates.

C2. Within-learner A/B.. Randomly assign each learner's cards to *Baseline* vs. *Variant* (e.g., higher target band or conservative interval cap). Interleave during normal study for 4–8 weeks. This within-person design avoids between-person confounds.

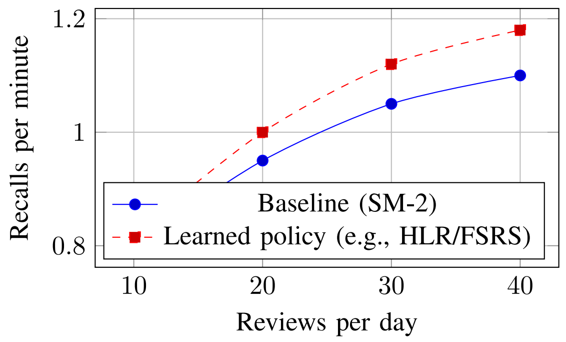


Figure 8: Efficiency frontier: retention per unit time vs. review volume. Replace curves with empirical/replay results.

D. Analysis plan.

- **Pre-specification:** Fix windows, interval bins, and target band before analysis.
- **Calibration:** Use reliability curves as in Figure 7; compute Brier/log-loss as secondary checks.
- **Efficiency:** Plot curves as in Figure 8; summarise with area under curve (AUC) over the window.

- **Workload:** Trend backlog; report fraction of days with nonzero backlog and median streak length.

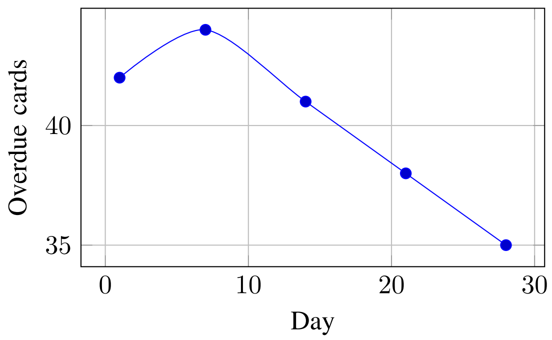


Figure 9: Workload health: backlog trend over time. Replace with actual overdue counts; slope ≤ 0 indicates sustainability.

E. Success criteria.

- Higher 30-day retention *per minute* than fixed-gap baseline.
- Calibration bins mostly inside the target band with acceptable log-loss/Brier.
- Stable streaks and non-increasing backlog (flat or negative slope).
- Replay suggests learned policies can match or exceed retention with equal/fewer reviews.

F. Reproducibility and transparency. All results are computed from the append-only review log $(\text{item}, \text{timestamp}, \text{rating}, \text{next\_due})$. Provide anonymised exports and scripts to reproduce Table 2 and Figures 7–9.

4. Technical deliverable

4.1. Design

4.1.1. A1. System architecture & stack (conceptual).

Client-first web app. A single-page application (React + TypeScript; Vite build) runs in the browser and talks directly to a managed PostgreSQL instance via Supabase's HTTPS API. No custom server is required in the MVP, which keeps costs at zero and operations simple.

Data flow at a glance. One study action produces exactly one review event. Dashboards read from the same stream. The value stream is:

(i) *Card*
(ii) *Study* (attempt, reveal, rate)
(iii) *Review record*
(iv) *Analytics*

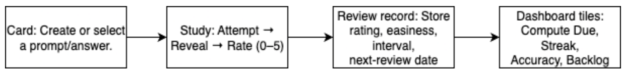

Card: Create or select a prompt/answer.
Study: Attempt → Reveal → Rate (0–5)
Review record: Store rating, easiness, interval, next-review date
Dashboard tiles: Compute Due, Streak, Accuracy, Backlog

Figure 10: Value stream: Card $\rightarrow$ Study $\rightarrow$ Review record $\rightarrow$ Analytics.

Time and locale. All stored timestamps are UTC. "Today," end-of-day cutoffs, and streaks are computed in the learner's local zone for intuitive behaviour.

4.1.2. A2. Data schema & storage model.

Event-sourced schedules. Schedules are *derived*, not stored. The current due date of a card equals the *latest* review event's next-review timestamp. Writes are append-only; analytics and dashboards reflect only recorded behaviour.

Logical entities.

- **Decks** (*id, name, description, created_at, updated_at*)
- **Cards** (*id, deck_id, front, back, example, has_image, has_audio, created_at, updated_at*)
- **Review_records** (*id, card_id, rating, ease_factor, interval_days, next_review_date, created_at*)

Indexes and integrity.

- Latest review per card: `(card_id, created_at DESC).`
- Due/backlog scans: `(next_review_date).`
- Ratings constrained to $[0, 5]$; `created_at` defaults to current UTC.

4.1.3. A3. Scheduling design (SM-2 family; implemented).

Rating anchors. 0 = miss; 1–2 = partial/slow; 3 = correct with effort; 4 = easy; 5 = very easy. Anchors stabilise judgements.

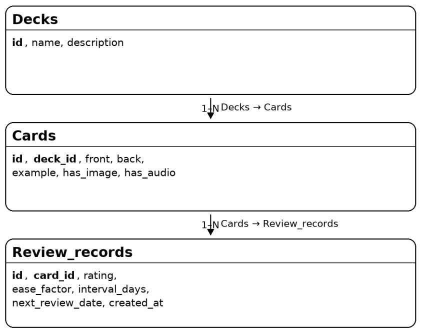

Decks
id, name, description
1..N Decks → Cards
Cards
id, deck_id, front, back,
example, has_image, has_audio
1..N Cards → Review_records
Review_records
id, card_id, rating,
ease_factor, interval_days,
next_review_date, created_at

Figure 11: Logical data model with key fields highlighted

State. Each card maintains two conceptual state variables: *easiness* factor (EF) with a lower bound (e.g., 1.3) and the planned *interval* (days).

Procedure S: per-review scheduling.

(i) Inputs: previous EF, previous interval, repetition count, rating $q \in \{0, \dots, 5\}$, and current time.

(ii) If $q < 3$: reset repetitions; set a short next interval (approx. 1 day); decrease EF gently but not below the floor.

(iii) If $q \geq 3$: first two successes use fixed seeds (1 day, then 6 days); afterwards multiply the previous interval by EF and round to whole days; nudge EF slightly based on the exact rating.

(iv) Compute *next_review_date* = now + chosen interval.

(v) Append one row to *review_records*: {card_id, rating, new EF, interval_days, next_review_date, created_at}.

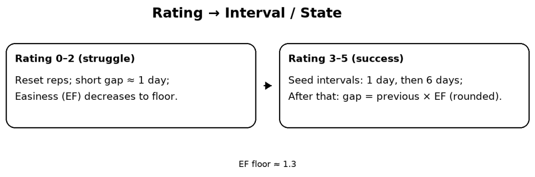

Rating → Interval / State
Rating 0–2 (struggle)
Reset reps; short gap ≈ 1 day;
Easiness (EF) decreases to floor.
Rating 3–5 (success)
Seed intervals: 1 day, then 6 days;
After that: gap = previous × EF (rounded).
EF floor ≈ 1.3

Figure 12: Rating semantics and resulting interval update in the SM-2 family.

Cold start and overdue cases. New cards initialise with neutral EF (≈ 2.5) and conservative early intervals. Overdue but correct responses support longer next gaps (evidence of stability); failures reset gaps short. Both outcomes follow directly from Procedure S.

4.1.4. A4. Next-item selection (queue design).

Goal. Prioritise the card that most needs attention to keep practice near the edge of forgetting.

Procedure Q: next-card ranking.

(i) Build four buckets for the active deck:

- (a) *Overdue* — latest next-review < now (oldest first).
- (b) *Due today* — latest next-review $\leq$ end of local day (earliest first).
- (c) *Never reviewed* — introduced gradually to avoid starvation.
- (d) *Not yet due* — used only if the queue is short; ties broken with light randomisation.

(ii) Concatenate the buckets in the order above and select the first item.

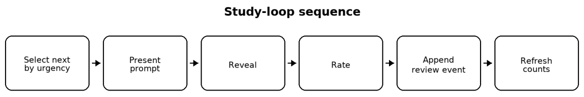


Figure 13: Sequence of operations in a single study step.

4.1.5. A5. Study flow (trial hygiene).

Four phases. Prompt $\rightarrow$ silent attempt $\rightarrow$ reveal $\rightarrow$ rating. This separation preserves the meaning of "difficulty" and aligns with retrieval-practice evidence.

Keyboard-first interaction. Space = reveal; digits 1–5 = rating; Enter = next card. Focus order is predictable; interactive elements carry accessible labels.

4.1.6. A6. Analytics (definitions and traceability).

Single source of truth. All metrics are computed from *review_records* (append-only). No placeholders are used.

Tiles and series.

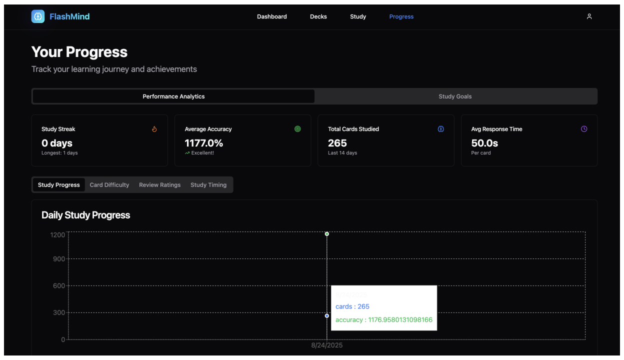
- **Cards due today** — no reviews or latest next-review $\leq$ end of local day.
- **Backlog** — latest next-review < now.
- **Streak (days)** — consecutive days with ≥ 1 review; if today is empty, the count extends only to yesterday.
- **Accuracy (30d)** — fraction of ratings ≥ 4 in the last 30 days.
- **Weekly progress** — per-day totals of reviews and first-time reviews.

4.1.7. A7. Configuration and environment.

Build-time variables (host-managed).

- `VITE_SUPABASE_URL` — Supabase project URL.
- `VITE_SUPABASE_PUBLISHABLE_KEY` — publishable anon key for client access.

Values are supplied via the hosting environment (not committed to the repository). For multi-user deployments, enable Auth and Row-Level Security and use session-scoped tokens.



Your Progress

Track your learning journey and achievements

Study Streak
0 days
Longest: 1 days

Average Accuracy
1177.0%
Excellent!

Total Cards Studied
265
Last 14 days

Avg Response Time
50.0s
Per card

Daily Study Progress

cards : 265
accuracy : 1176.9580131098166

Figure 14: Dashboard tiles and chart with definitions mapped to on-screen elements.

4.1.8. A8. Time, locale, accessibility, internationalisation.

Time. Store timestamps in UTC; compute "today" in the learner's local time zone to align streaks and due-today counts with local days.

Accessibility. Keyboard-operable study flow, visible focus indicators, WCAG-AA contrast, and descriptive text for actionable controls.

Internationalisation. UI strings separated for translation; dates/times rendered with locale-aware formatting; UTF-8 content throughout.

4.2. B. Production

4.2.1. B1. Build and configuration.

Objective. Produce a deterministic, zero-cost build of a client-first web app that connects securely to a managed PostgreSQL instance.

Configuration inputs.

- (i) Supabase project URL.
- (ii) Publishable (anon) key for client access.

Inputs are supplied via the hosting platform's environment panel and are not committed to the repository.

Procedure B1 (build).

1) Install the JavaScript toolchain (Node or Bun) and restore dependencies.
2) Inject configuration values using environment settings (Preview and Production).
3) Create a production build artifact (static bundle).
4) Validate using a local preview server (routing, assets, environment substitution).

Outputs. A static bundle suitable for a free/static host, plus a brief record of build options and environment variables for reproducibility.

4.2.2. B2. Deployment environments.

Objective. Separate Preview and Production without manual schema duplication.

Procedure B2 (environments).

1) Provision two Supabase projects (Preview, Production) with identical schemas.
2) Point the Preview build to the Preview database; Production to the Production database.
3) Restrict Preview sharing when Production holds real learner data.

Notes.

- Timestamps are stored in UTC.
- Local "end of day" is computed client-side for streaks and "due today".

4.2.3. B3. Schema migrations and versioning.

Objective. Maintain integrity and enable smooth upgrades.

Procedure B3 (migrate).

1) Keep ordered SQL migration files under version control.
2) Include a *schema_version* in exports; validate on import.
3) On structural changes, add a migration note for Decks, Cards, and Review_records (see Fig. **??**).

Integrity constraints.

- Ratings restricted to 0–5.
- Referential integrity on foreign keys.
- Default timestamps set to current UTC.

4.2.4. B4. Performance and scalability.

Indexes.

- Latest review per card: composite index on `(card_id, created_at DESC)`.
- Due/backlog queries: index on `(next_review_date)`.
- Deck scoping: index on `(deck_id)`.

Procedure B4 (query efficiency).

1) Build the study queue with minimal fields (ids, fronts, latest due markers).
2) Hydrate details lazily upon display.
3) Paginate due/backlog lists; window analytics (e.g., last six months).

Client-side practices.

- Debounce interactions that trigger writes.
- Batch reads; avoid N+1 by precomputing per-deck aggregates.

4.2.5. B5. Reliability and error handling.

Objective. Preserve study state under transient failures.

Procedure B5 (write path resilience).

1) On rating, stage the review event in a short-lived client queue.
2) Attempt insertion; on failure, show a non-blocking notice and retry with exponential backoff.

3) Use a temporary client-side identifier to deduplicate retries.
4) Flush the queue after connectivity returns; confirm success before removing staged items.

Session continuity. Queue ordering (overdue $\rightarrow$ due today $\rightarrow$ never reviewed $\rightarrow$ not yet due) is preserved across retries.

4.2.6. B6. Security and privacy. MVP posture. No personally identifying information is required. Data consists of deck/card content and review events (timestamps, ratings, derived schedule state).

Path to multi-user.

- Add a user identifier to Decks, Cards, Review_records; enable Row-Level Security (RLS).
- Define policies so each user can access only own rows.
- Replace the publishable key with session-scoped tokens via Supabase Auth.

Secrets handling. Configuration remains in the host environment. Keys are rotated when moving from MVP to shared deployments.

4.2.7. B7. Analytics computation. Objective. Present transparent metrics derived solely from review events.

Definitions (recap).

- *Cards due today:* no prior reviews or latest next-review before local end-of-day.
- *Backlog:* latest next-review earlier than current time.
- *Streak (days):* consecutive local days with at least one review; if today is empty, count extends to yesterday.
- *Accuracy (30d):* proportion of ratings ≥ 4 in the last 30 days.
- *Weekly progress:* per-day totals of reviews and first-time reviews.

Procedure B7 (refresh).

1) After each inserted event, refresh in-memory aggregates used by the current view.
2) For dashboards, recompute tiles from the event stream over the selected window.
3) Keep UI definitions identical to the scientific section to avoid drift.

4.2.8. B8. Import and export. Objective. Enable frictionless interchange and backups.

Exports.

- JSON for full fidelity (deck metadata, cards, optional review history).
- Include schema version and ISO-8601 UTC timestamps.

Imports.

- Accept JSON and lightweight CSV.
- Minimal CSV fields: *front*, *back*, *example*; events may be supplied in a second file.
- Resolve duplicates via a stable external identifier or normalized fields; report conflicts with skip/merge options.

Procedure B8 (round-trip).

1) Export: write deck, cards, and—when requested—events.
2) Import: validate schema version; upsert decks/cards; apply events.
3) Verify: reconstruct current schedules from imported events and compare sample due dates against the source.

4.2.9. B9. Testing and quality assurance. Objective. Prevent regressions in scheduling and analytics.

Scheduler tests.

- Ratings < 3 reset repetitions.
- Easiness never falls below the floor.
- Interval rounding is correct.
- Overdue success/failure paths behave as specified.

Analytics tests.

- Synthetic event logs with known outcomes for each tile (due today, backlog, streak).
- Outputs compared to gold files.

Integration tests.

- Create deck, add cards, run a short session, append events, verify dashboard tiles update.

Manual QA.

- Cross-browser rendering; keyboard-only study.
- High-contrast mode; small-screen layouts.

4.2.10. B10. Observability and monitoring. Objective. Detect problems early without collecting sensitive data.

Client telemetry (anonymous).

- Page load and interaction timings.
- Network failure counts; retried write counts.

Operational dashboards.

- Daily event volume, error rates, backlog quantiles.
- Sudden changes may indicate scheduling or query regressions.

Alarms.

- Thresholds on write failures and backlog slope over a rolling window.

4.2.11. B11. Operational runbook. Objective.

Provide concise steps for routine tasks.

Runbook items.

- *Rollout:* deploy Preview; smoke-test; promote to Production.
- *Key rotation:* update environment values; invalidate prior tokens; redeploy.
- *Backup/restore:* export JSON/CSV snapshots; test restores on a throwaway database.
- *Incident response:* if event writes fail, switch to offline queue-only mode, notify users, and restore connectivity before flushing.

4.2.12. B12. Risks and mitigations. Free-tier constraints.

- Mitigation: pagination, batching, windowed analytics.
- Scale path: denormalized "next due" cache (e.g., trigger-maintained) for faster due queries.

Backlog growth.

- Mitigation: surface backlog early; cap "new cards per day"; suggest daily goals based on recent activity.

Time zones and travel.

- Mitigation: show detected time zone with override; document daylight-saving caveats.

User trust.

- Mitigation: per-card histories and brief "Why now?" messages; every dashboard number traceable to events.

4.2.13. B13. Future technical work. Authentication and RLS.

Multi-user mode with teacher roles and cohort views; policy-based deck access.

Learned schedulers. Optional FSRS or half-life regression using the same event log; switchable per deck or per user; calibration plots for validation.

Offline-first (PWA). Cache decks; queue events; synchronise on reconnect with conflict resolution.

Media handling. Simple image/audio uploads with size limits and CDN delivery; moderation hooks for shared decks.

Performance at scale. Precompute next-review timestamps where beneficial; compact very old events into summaries (raw history still exportable).

Summary. Part B defines build, deployment, and operations: deterministic builds; clear environment separation; append-only event storage; responsive scheduling and ranking; transparent analytics; reliability safeguards; and a path to secure multi-user operation.

5. Discussion

The project translates robust principles from cognitive psychology into a practical learning workflow that remains transparent to learners and educators. At its core is a simple but defensible proposition: present each card near the *edge of forgetting*, let the learner rate the difficulty, and adapt the next interval accordingly. This decision loop—attempt, reveal, rate, schedule—preserves the signal that matters (difficulty after a delay) while keeping interaction costs low.

Two design choices are noteworthy. First, *event-sourced scheduling*. Rather than mutating a card's state in place, every study action writes one review event and the current schedule is derived from the latest event. This strategy provides an audit trail for science (calibration checks, counterfactual replays), ensures that dashboards reflect only genuine activity, and prevents subtle divergences between "state" and "history." Second, the scheduler adopts a *documented SM-2 family rule*. Although learned schedulers can outperform heuristics when ample data are available, SM-2 offers clarity: learners can understand why intervals stretch or shrink, and instructors can explain the consequences of ratings without appealing to opaque models.

The project's analytics are intentionally modest and interpretable. "Due Today," "Backlog," "Streak," "Accuracy (30d)," and a 7-day activity chart provide a high-signal, low-distraction view of progress. Importantly, each metric is defined directly on the event stream, which curbs over-fitting the interface to vanity metrics and supports reproducibility. In this sense, analytics are not merely decorative; they function as a continuous check that the system is targeting a sustainable band of difficulty and that the daily workload remains stable.

Pedagogically, the design privileges *desirable difficulty*. Ratings near the middle of the scale—"correct with effort," "good," "easy"—are desirable because they indicate effortful retrieval without persistent failure. The selection policy (overdue $\rightarrow$ due today $\rightarrow$ never reviewed $\rightarrow$ not yet due) operationalises this stance without heavy parameterisation, and aligns with horizon sensitivity: longer-term goals tolerate longer gaps; near-term goals demand higher target recall at the next review.

Finally, the system establishes a credible path forward. By retaining the same event schema, one can introduce learned schedulers (e.g., half-life regression or FSRS) without altering the learner's workflow. This separation of concerns—stable interaction loop, swappable scheduling policy—future-proofs the platform while keeping the MVP coherent and zero-cost.

6. Limitations and Future Work

Current limitations

Scheduler expressiveness.. The SM-2 family uses a single *easiness* value for both difficulty and stability. This coupling can create wide variation for very easy or very hard items, especially with irregular study. The approach is acceptable for an MVP but less precise than learned schedulers.

Rating bias and drift.. Self-reported difficulty can shift over time (for example, generous scoring during short sessions). Clear anchors and per-card history views help, but do not remove bias. Adding response time (or simple latency buckets) would strengthen the signal with minimal burden.

Free-tier constraints.. Throughput and storage are limited on free plans. The append-only design is I/O-friendly, yet very large decks or long histories may need precomputed summaries (e.g., a cached next-due per card) to keep queries fast.

Media and offline use.. Rich media and offline study (PWA) are minimal in the MVP. Both are valuable for vocabulary and accessibility, but introduce storage, sync, and moderation requirements.

Multi-user features.. Teacher workflows (shared decks, due dates, cohort analytics) are out of scope. These features require authentication, row-level security, and careful handling of learner data.

7. Ethics, Privacy, and Accessibility

Ethical posture

Transparency and autonomy. Scheduling decisions are explained ("Why now?") and per-card histories are visible. This supports metacognitive control and reduces perceived arbitrariness. Clear rating effects reduce accidental gaming.

Workload health. Backlog is surfaced early. Regular but manageable sessions are encouraged, with an option to cap "new cards per day." These safeguards limit punitive backlogs and failure streaks.

Privacy and data protection

Data minimisation. Personally identifying information is not required. The minimal dataset is deck content plus review events (timestamps, ratings, derived schedule state).

User control. Exports (JSON/CSV) are available with documentation of what is stored, for how long, and how deletion works. In multi-user mode, row-level security ensures access only to a user's own rows. Shared artefacts have explicit ownership and revocation.

Security baseline. Keys are supplied via host environments, not source code. In multi-user mode, authentication and short-lived session tokens replace the publishable key. Policies follow least-privilege.

Accessibility

Keyboard-first study.. Reveal, rate, and next actions are fully keyboard-operable with visible focus. This benefits assistive-tech users and power users alike.

Perceptual considerations. Colours meet WCAG AA contrast. Icons and charts include text alternatives or labels. Charts avoid colour-only distinctions.

Screen-reader flow. The four phases (prompt, attempt, reveal, rating) are reflected in ARIA roles and announcements so the study loop is understandable without sight.

8. Conclusion

The project delivers a coherent, zero-cost, browser-based spaced-repetition system that adheres to established memory science while remaining transparent and usable. The learning loop is intentionally simple: attempt recall, reveal the answer, rate the experience, and let the scheduler adjust the next interval. Event-sourced storage ensures that every dashboard number is traceable to actual behaviour, enabling calibration checks and counterfactual replays.

In its current form, the system balances practicality and rigour: SM-2–style scheduling is easy to explain and robust enough for broad use; analytics confirm whether practice remains in a desirable difficulty band and whether daily workload is sustainable. The architecture keeps future options open. Without altering the user experience, one can layer in learned schedulers, teacher workflows, richer media, and offline support. In short, the platform turns well-replicated scientific effects—spacing and retrieval practice—into a dependable study routine that learners can adopt immediately and institutions can understand and trust.

9. Plagiarism Statement

I declare that I am aware of the following facts:
• As a student at the University of Luxembourg I must respect the rules of intellectual honesty, in particular not to resort to plagiarism, fraud or any other method that is illegal or contrary to scientific integrity.
• My report will be checked for plagiarism and if the plagiarism check is positive, an internal procedure will be started by my tutor. I am advised to request a pre-check by my tutor to avoid any issue.
• As declared in the assessment procedure of the University of Luxembourg, plagiarism is committed whenever the source of information used in an assignment, research report, paper or otherwise published/circulated piece of work is not properly acknowledged. In other words, plagiarism is the passing off as one's own the words, ideas or work of another person, without attribution to the author. The omission of such proper acknowledgement amounts to claiming authorship for the work of another person. Plagiarism is committed regardless of the language of the original work used. Plagiarism can be deliberate or accidental. Instances of plagiarism include, but are not limited to:
1) Not putting quotation marks around a quote from another person's work
2) Pretending to paraphrase while in fact quoting
3) Citing incorrectly or incompletely

4) Failing to cite the source of a quoted or paraphrased work
5) Copying/reproducing sections of another person's work without acknowledging the source
6) Paraphrasing another person's work without acknowledging the source
7) Having another person write/author a work for oneself and submitting/publishing it (with permission, with or without compensation) in one's own name ('ghost writing')
8) Using another person's unpublished work without attribution and permission ('stealing')
9) Presenting a piece of work as one's own that contains a high proportion of quoted/copied or paraphrased text (images, graphs, etc.), even if adequately referenced

Auto- or self-plagiarism, that is the reproduction of (portions of a) text previously written by the author without citing that text, i.e. passing previously authored text as new, may be regarded as fraud if deemed sufficiently severe.

References

References

[1] P. Wozniak, “Algorithm SM-2.” Available: https://super-memory.com/english/ol/sm2.htm

[2] N. J. Cepeda, E. Vul, D. Rohrer, J. T. Wixted, and H. Pashler, “Spacing Effects in Learning: A Temporal Ridgeline of Optimal Retention,” *Psychological Science*, 2008. PDF: https://laplab.ucsd.edu/articles/Cepeda%20et%20al%202008_psychsci.pdf

[3] N. J. Cepeda, H. Pashler, E. Vul, J. T. Wixted, and D. Rohrer, “Distributed Practice in Verbal Recall Tasks: A Review and Quantitative Synthesis,” *Psychonomic Bulletin & Review*, 2006. PDF: https://augmentingcognition.com/assets/Cepeda2006.pdf

[4] H. L. Roediger and J. D. Karpicke, “The Power of Testing Memory: Basic Research and Implications for Educational Practice,” *Perspectives on Psychological Science*, 2006. PDF: https://psychnet.wustl.edu/memory/wp-content/uploads/2018/04/Roediger-Karpicke-2006_PPS.pdf

[5] R. A. Bjork and E. L. Bjork, “Making Things Hard on Yourself, but in a Good Way: Creating Desirable Difficulties to Enhance Learning,” in *Psychology and the Real World*, 2011. PDF: https://bjorklab.psych.ucla.edu/wp-content/uploads/sites/13/2016/04/EBjork_RBjork_2011.pdf

[6] H. Ebbinghaus, *Memory: A Contribution to Experimental Psychology* (1885/1913). Online: https://psychclassics.yorku.ca/Ebbinghaus/index.htm

[7] B. Settles and B. Meeder, “A Trainable Spaced Repetition Model for Language Learning,” *ACL*, 2016. PDF: https://research.duolingo.com/papers/settles.acl16.pdf

[8] Open Spaced Repetition, “Free Spaced Repetition Scheduler (FSRS),” GitHub repository. https://github.com/open-spaced-repetition/free-spaced-repetition-scheduler

[9] Anki Manual, “Background: Scheduling (SM-2/FSRS context).” https://docs.ankiweb.net/background.html

[10] Supabase, “Documentation Portal.” https://supabase.com/docs

[11] Supabase, “Row Level Security (RLS) in Postgres.” https://supabase.com/docs/guides/database/postgres/row-level-security

[12] JSON.org, “Introducing JSON.” https://www.json.org/json-en.html

[13] Wikipedia, “Comma-separated values.” https://en.wikipedia.org/wiki/Comma-separated_values

[14] P. Pimsleur, “A Memory Schedule,” *The Modern Language Journal*, 1967. Stable URL: https://www.jstor.org/stable/322091

[15] ACT-R Research Group, “ACT-R: Adaptive Control of Thought—Rational.” http://act-r.psy.cmu.edu/

Appendix

Appendix — Figures and Diagrams

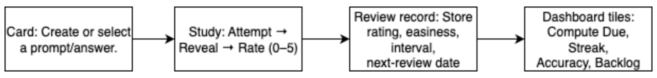

Card: Create or select a prompt/answer.
Study: Attempt → Reveal → Rate (0–5)
Review record: Store rating, easiness, interval, next-review date
Dashboard tiles: Compute Due, Streak, Accuracy, Backlog

Figure 15: Value stream: Card $\rightarrow$ Study $\rightarrow$ Review record $\rightarrow$ Analytics.

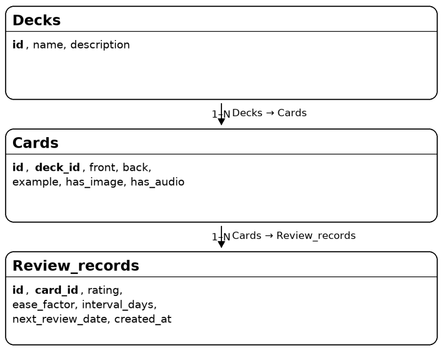

Decks
id, name, description
1..N Decks → Cards
Cards
id, deck_id, front, back, example, has_image, has_audio
1..N Cards → Review_records
Review_records
id, card_id, rating, ease_factor, interval_days, next_review_date, created_at

Figure 16: Mini-ERD: one Deck has many Cards; one Card has many Review records. Key fields in bold.

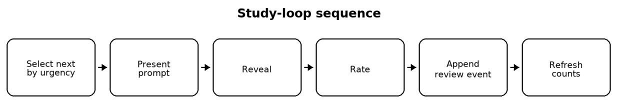


Figure 17: Study loop sequence: select next by urgency, present prompt, reveal, rate, append review event, refresh counts.

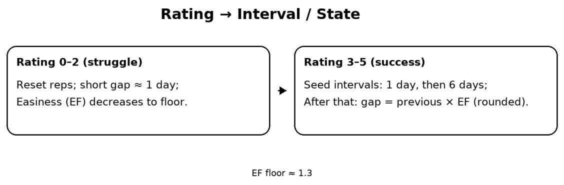

Rating → Interval / State
Rating 0–2 (struggle)
Reset reps; short gap ≈ 1 day;
Easiness (EF) decreases to floor.
Rating 3–5 (success)
Seed intervals: 1 day, then 6 days;
After that: gap = previous × EF (rounded).
EF floor ≈ 1.3

Figure 18: Rating $\rightarrow$ interval/state mapping (SM-2 family): $q < 3$ resets and short gap; $q \geq 3$ uses seeds (1d,6d) then multiplies by EF (rounded). EF floor ≈ 1.3.

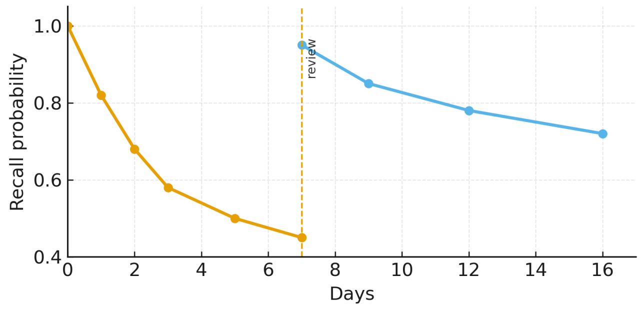


Figure 19: Forgetting with refresh: recall decays, is boosted by review, then decays more slowly (stabilisation).

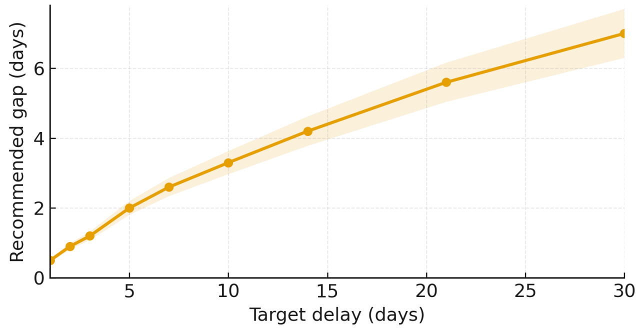
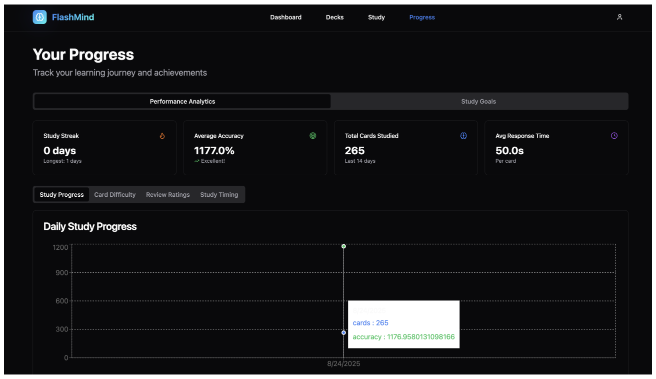


Figure 20: ISI–RI "ridgeline": recommended interstudy gap increases sub-linearly with target retention interval.


FlashMind
Dashboard Decks Study Progress

Your Progress
Track your learning journey and achievements

Performance Analytics | Study Goals

Study Streak
0 days
Longest: 1 days

Average Accuracy
1177.0%
Excellent!

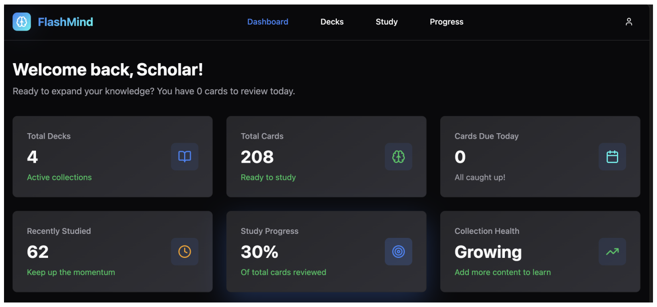
Total Cards Studied
265
Last 14 days

Avg Response Time
50.0s
Per card

Study Progress | Card Difficulty | Review Ratings | Study Timing

Daily Study Progress
cards : 265
accuracy : 1176.9580131098166

Figure 21: Dashboard annotated: Due Today, Backlog, Streak, Accuracy (30d), and 7-day progress mapped to definitions.



Welcome back, Scholar!

Ready to expand your knowledge? You have 0 cards to review today.

Total Decks
4
Active collections

Total Cards
208
Ready to study

Cards Due Today
0
All caught up!

Recently Studied
62
Keep up the momentum

Study Progress
30%
Of total cards reviewed

Collection Health
Growing
Add more content to learn

Figure 22: Analytics overview with formal definitions matched to on-screen tiles and chart.