Computer Science NEA

REBORN — An adaptive Habit Tracker

REDACTED

Candidate Number: **XXXX** | Centre Number: **XXXXX**

Centre Name: **REDACTED**

Qualification Code: 7517

January 21, 2026

"Civilization advances by extending the number of operations we can perform without thinking about them." — A. North Whitehead

---

Contents

List of Figures

List of Tables

1 Introduction

"Make it so easy you can't say no."

James Clear, Atomic Habits [1]

Modern digital systems increasingly compete for human attention through algorithmic optimisation, yet provide little computational support for helping users regulate behaviour and build sustainable habits. Many widely used platforms prioritise engagement over user agency, contributing to inconsistent routines and behavioural instability.

This project, **REBORN**, aims to build an intelligent habit-tracking system that automates scheduling, analyses behavioural risk, and dynamically adapts difficulty. When software handles the heavy planning, users can focus their energy on execution.

2 Analysis

2.1 Problem Definition

Many people experience *self-regulation failure* - they want to improve, but the cognitive effort needed to plan and continuously maintain habits often leads to burnout [2], [3]. Technology is not only the largest source of procrastination, but also doesn't provide intelligent ways to reduce this problem [4]. Habit formation, therefore, is both a computational challenge and a psychological one. Tools that can automate planning, predict high-risk periods, and adapt interventions in real time are therefore necessary.

2.2 Users

The primary users of REBORN are students, especially those studying multiple A-Levels and struggling with procrastination and the mental load of planning habits. For this group, automation and predictive intervention allow them to focus on doing the habits instead of organisation.

The system is also intended for a larger audience, such as:

- Professionals who need structured routines and work-life balance.
- Individuals maintaining regular fitness or health-related habits.
- Users interested in general self-improvement and long-term behaviour change.

All of these user groups experience forms of self-regulation failure, where manual habit tracking can lead to cognitive overload and burnout.

2.3 Secondary Research

Current habit-tracking and productivity applications differ widely in how much computation they perform. Some are manual-input database systems, others are game-like behavioural frameworks. An analysis of these tools shows that most apps use psychological principles (loss aversion/variable reward schedules) instead of algorithmic reasoning.

To justify the development of REBORN, I will contrast existing tools with a theoretical "gold standard" of computational habit formation, and then find specific technical gaps in scheduling, prediction, and how systems adapt to user context over time.

2.3.1 PCS Framework

The strongest computational framework for habit formation is the **Predicting Context Sensitivity (PCS)** method [5]. Unlike the *21-day myth*, this machine-learning methodology models habit formation as a continuous measure of context-sensitive predictability.

Using **LASSO** (Least Absolute Shrinkage and Selection Operator) regression—a regularised machine-learning method that identifies the most predictive contextual variables while reducing overfitting—on datasets with 40 million+ observations, Buyalskaya et al.[5] showed that habit formation timelines are domain-specific. The estimated times range from a few weeks for simple hygiene tasks to 2–3 months for more complex behaviour like regularly going to the gym.

Most importantly, no current app makes use of this kind of context data (e.g., time lags, or day-of-week streaks) to predict user behaviour.

2.3.2 Existing Systems

This section analyses the four most popular habit tracking and task management apps: Notion, Habitica, Streaks, and Todoist. While other applications exist, these four represent the distinct architectural archetypes currently dominating the market.

2.3.2.1 Description and Functionality

Notion

Notion is a workspace that acts like a relational database wrapper. The user defines schemas (tables) and views (Kanban, Calendar) manually. The backend uses a SQL-like structure which is then served to a React frontend.

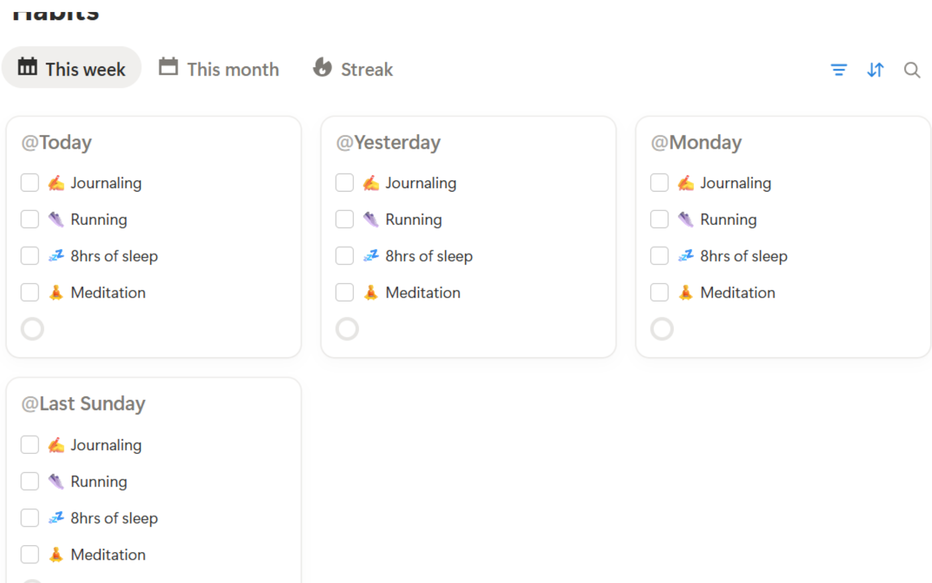

Figure 1: A Notion Habit Tracker showing the manual checkbox system.

Habitica

Habitica is a gamified task manager that uses operant conditioning [6]. It wraps a CRUD system in RPG mechanics, where doing tasks gives *Gold/XP* and missing them causes *Health Loss*.

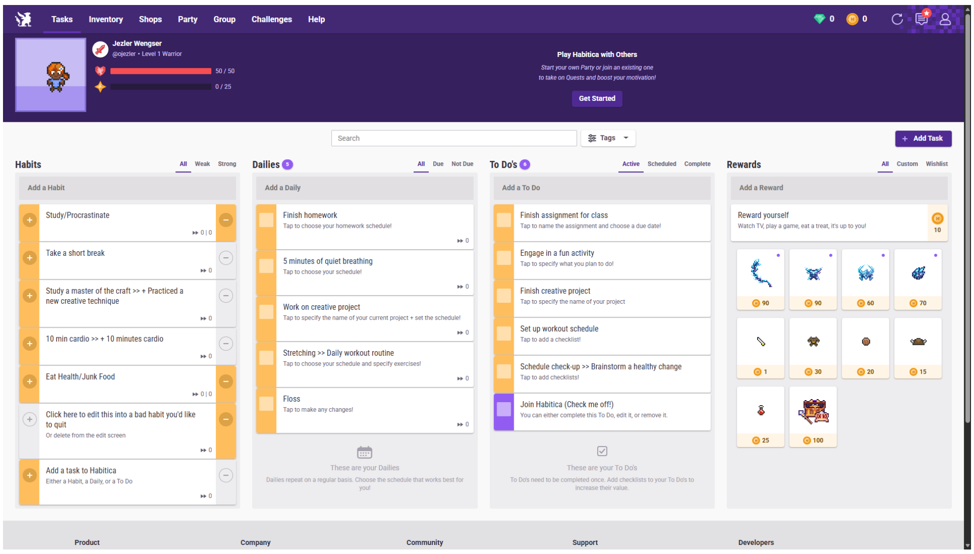

Figure 2: The Habitica dashboard showing the RPG avatar and health bars.

Streaks

Streaks is a minimalist habit tracker integrated into the Apple ecosystem. Unlike database tools, it focuses entirely on the *Don't Break the Chain* psychological method [1]. Technically, it is notable for its API integration with Apple HealthKit [7], allowing it to automatically mark habits (e.g., "Walk 5,000 steps") as complete without user input.

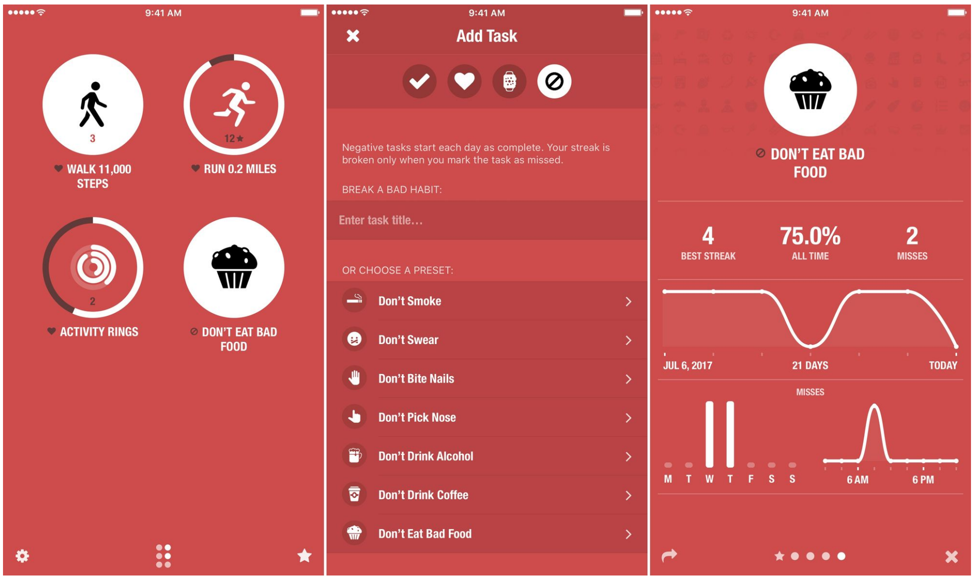

Figure 3: The Streaks interface showing its main features.

Todoist

Todoist is a task management system, designed for one-off tasks. In earlier versions, Todoist experimented with automated rescheduling through its *Smart Schedule* feature, which attempted to suggest new due dates for overdue tasks. Although it was discontinued because of accuracy issues [8], third-party integrations like Trevor AI [9] can provide similar functionality, which apply heuristic time-blocking and NLP to parse tasks and suggest times for them based on available calendar gaps.

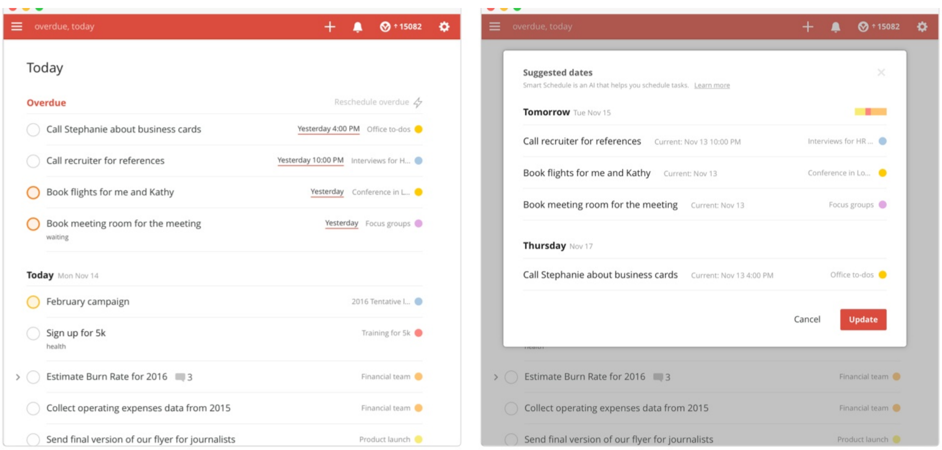

Figure 4: Todoist's "Smart Schedule" interface suggesting times for tasks [10].

2.3.2.2 Analysis of Advantages and Disadvantages

The following tables breakdown the strengths and weaknesses of each system's architecture and user experience model.

Notion

Advantages	Disadvantages
Flexible data modeling that allows complex relationship definitions.	**The *Blank Slate* problem:** Users have to build the system themselves; there is no built-in logic. (without templates)
Visual customisation reduces initial friction for design-oriented users.	**No constraint satisfaction:** It cannot detect schedule conflicts or optimise time slots automatically [11].
Excellent database management for historical record keeping.	Entirely manual input leads to high administrative load (cognitive friction).

Streaks

Advantages	Disadvantages
API Automation: Integration with Apple HealthKit removes *administrative friction* by tracking physical habits automatically.	***All-or-Nothing*:** It uses binary logic. If the user misses one day, the streak resets to zero. This often causes the *What-the-Hell Effect* [12], where users abandon the app after a failure.
Visual Feedback: The circular UI provides visual status of daily progress, using *loss aversion* for engagement [13].	**No Scheduling Logic:** The app tracks *completion*, but not *allocation*. It cannot help the user find time in a busy schedule to actually perform the habit.
Minimalist design reduces cognitive load compared to complex tools like Notion.	**No adaptive difficulty:** The target remains static regardless of user burnout.

Habitica

Advantages	**Disadvantages**
Gamification (XP/Gold) uses dopamine loops [14] to increase engagement.	**Static Recurrence:** It relies on fixed rules (e.g., “Every Mon/Tue”) instead of adaptive scheduling.
Social accountability features (parties/guilds) help with external motivation.	**Reactive Punishment:** It creates a *death spiral*. If a user is busy and misses tasks, the game punishes them, demotivating them further.
Visualisation of progress from avatar customisation and item unlocks shows long-term habit progress.	No awareness of external schedules or calendar conflicts.

Todoist

Advantages	**Disadvantages**
NLP (Natural Language Processing) allows for quick task entry using natural language [15].	**Task vs. Habit:** The system focuses on one-off tasks and does not have a risk or failure model for long-term habit formation.
Heuristic time-blocking reduces the cognitive load of manual scheduling.	Scheduling behaviour is static; repeated failure does not reduce workload or adjust task frequency.
Deep ecosystem integrations (email, calendar, browser, automation tools) allow Todoist to function as a centralised task capture hub across contexts and devices.	Closed-source algorithms prevent inspection or tuning for individual habit difficulty curves.

2.3.2.3 Implications for Proposed System

By investigating these systems, I have identified specific features to adopt and flaws to improve upon in the proposed solution.

From Notion:
While a relational database is important for storing habit history, relying on the user to manually schedule every event will likely lead to burnout.

- **I will include:** A strong relational database backend similar to Notion's architecture for data integrity.
- **I will improve:** Instead of manual entry, I will implement a Constraint Satisfaction Scheduler to automate the data entry process, solving the *Blank Slate* problem.

From Habitica:
Feedback mechanisms are effective, but "punishment" (losing HP) is flawed if the user is busy or overwhelmed.

- **I will include:** A feedback loop system, but instead of RPG stats, I will use a **Simulation** to visualise habit stability.
- **I will improve:** The system will be predictive rather than reactive. By using **Logistic Regression**, my system will warn the user of risk *before* they fail, instead of punishing them *after*.

From Streaks:
Visualising progress is very important, but the *streak counter* model is too fragile for students with variable workloads.

- **I will include:** A strong visual representation of habit health, similar to the *rings* in Streaks, to provide immediate feedback.
- **I will improve:** Instead of a binary *streak* that breaks (0 or 1), I will use my **simulation**. This treats habit consistency as a continuous variable (stability). If a user misses a day, the stability decays instead of resetting to zero, which is mathematically more forgiving and encourages the user to recover.

From Todoist:
Algorithmic scheduling is desirable, but fixed heuristics fail when psychological factors and user failure are not modelled.

- **I will include:** Automated time-blocking to ensure habits and tasks are placed within realistic daily constraints.
- **I will improve:** The use of a *Spaced Repetition* algorithm to dynamically reduce habit frequency when users struggle, preventing overload and burnout.
- **I will avoid:** Set heuristic scheduling, by integrating adaptive control and predictive modelling to maintain long-term habit.

2.3.3 Identification of Technical Gaps

The comparison of existing tools shows three algorithmic gaps in the current market that **REBORN** aims to solve:

1. **Constraint Satisfaction Solvers (CSP):** Current apps rely on manual time-picking. No system treats the daily schedule as a combinatorial search problem that can be solved using backtracking or constraint propagation to guarantee conflict-free allocation [11].

2. **Predictive Logistic Regression:** Applications are reactive; they show historical graphs instead of forecasting. None use the PCS framework's approach (Logistic Regression) to calculate a probability $P(fail)$ for the upcoming day to trigger preventative interventions.

3. **Static Frequency vs. Adaptive Control:** Habit trackers use static schedules (e.g., "Daily"). They do not implement control loops (like the SM-2 spaced repetition algorithm [16]) to adjust frequency based on user performance, for example, by scaling back load to prevent burnout.

System	Scheduling Logic	Predictive Model	Adaptivity	Primary Domain
Notion	Manual (User defined)	None	None	Database
Habitica	Fixed Recurrence	None	None	Gamification
Streaks	Fixed Recurrence	None	None	Health Tracking
Todoist + Trevor	Heuristic Time-blocking	Duration Estimation	None	Task Management
REBORN	**Constraint Solver (DFS)**	**Logistic Regression**	**SM-2 Algorithm**	**Habit Optimisation**

Table 1: Comparison of existing solutions against the proposed system.

2.3.4 Conclusion

The existing market is saturated with tools that record data but can't compute solutions. Todoist already shows how algorithmic scheduling can help with one-off tasks and academic papers on the PCS framework [5] led mathematically with prediction. No system integrates these into a single application.

2.4 Primary Research

2.4.1 Methodology

I conducted a "Whiteboard Focus Group" within my Computer Science class ($n = 9$). The participants represent the core demographic: students with high technical literacy but heavy academic workloads.

I asked the binary question:

> "Have you ever abandoned or failed to create a planner/habit app because making/updating it was too much effort?"

2.4.2 Results

The raw photographic evidence is available in **Appendix ??**. The results were unanimous.

Response	**Visual Tally**	**Count**	**Percentage**
Yes (Too much effort)	卌 IIII*	**9**	**100%**
No (Manageable)	*(No votes recorded)*	0	0%

Table 2: Results showing unanimous dissatisfaction with manual tracking.

* One participant drew an infinity symbol (∞) instead of a tally mark. This was counted as a single 'Yes' vote.

2.4.3 Analysis and Technical Justification

Quantitative Analysis: The fact that 100% of the focus group ($n = 9$) voted "Yes" suggests that *administrative friction* is a universal problem for this demographic. It confirms that relying on user discipline to maintain a database (like Notion) leads to failure. This makes the **Constraint Satisfaction Scheduler (CSP)** a requirement, as the system must take over the planning process.

Qualitative Observation: The participant who used an infinity symbol (∞) instead of a tally is important. It suggests that manual habit tracking feels like a *never-ending* cycle of work without reward.

This specific observation justifies the need for the **SM-2 adaptive algorithm**. The system needs to be able to detect this feeling of *infinite load* (from missed check-ins) and mathematically lower the habit frequency to break the cycle of burnout.

2.5 Primary Research Online

After the initial focus group, a larger quantitative survey was created using Google Forms to test the hypothesis across a larger sample size. The direct screenshots of the responses are available in **Appendix ??**.

2.5.1 Methodology

The survey collected **37 responses**. The distribution was focused on academic communities (A-Level and University), as these groups have high scheduling pressure.

- **Demographics:** 51.3% A-Level Students, 37.8% University Students, 10.9% Professionals/Other.
- **Relevance:** The majority of respondents fit the "Student/Academic" user profile defined in the Analysis section, so the data is representative of the target audience.

2.5.2 Theme 1: The "Administrative Burden" & Scheduling

The findings support the main claim that manual scheduling is a failure point.

- **Friction is Fatal: 83.8%** of respondents abandoned a productivity app because keeping it updated was "too tedious."
- **Decision Fatigue: 75.7%** agreed that the act of planning their day contributes to decision fatigue.
- **Demand for Automation:** When asked if an intelligent system that automatically rearranges schedules around fixed events would be helpful, **94.6%** responded positively (scoring 3 or 4 out of 4).

Technical Justification: These statistics show that a static database (like Notion) is not enough for maintaining habits. The high rate of abandonment from "tedium" (83.8%) justifies the use of a **Constraint Satisfaction Solver (CSP)**. The system must take over the work of "time-slotting" away from the user to reduce the decision fatigue reported by 75.7% of the sample.

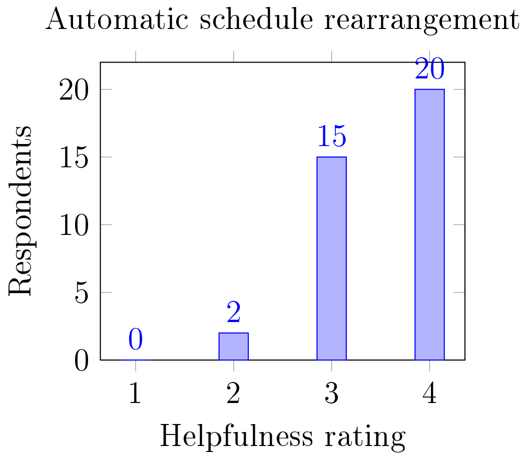

Figure 5: Automatic schedule helpfulness (n=37)

2.5.3 Theme 2: The Failure of "Streaks" & Need for Adaptation

The survey shows a flaw in current market leaders (e.g., Duolingo, Streaks app) in how they handle consistency.

- **Streak Fragility: 91.3%** of users felt that traditional streaks do not accurately represent their momentum and described them as "too fragile." In addition, **64.9%** found streak counters actively stressful.
- **Burnout: 78.4%** agreed that rigid schedules (e.g., "Must do daily at 9 AM") lead to burnout.
- **Want for Adaptation: 89.2%** of users wanted a system that "detects when you are struggling and automatically lowers difficulty," instead of one that just keeps them accountable.

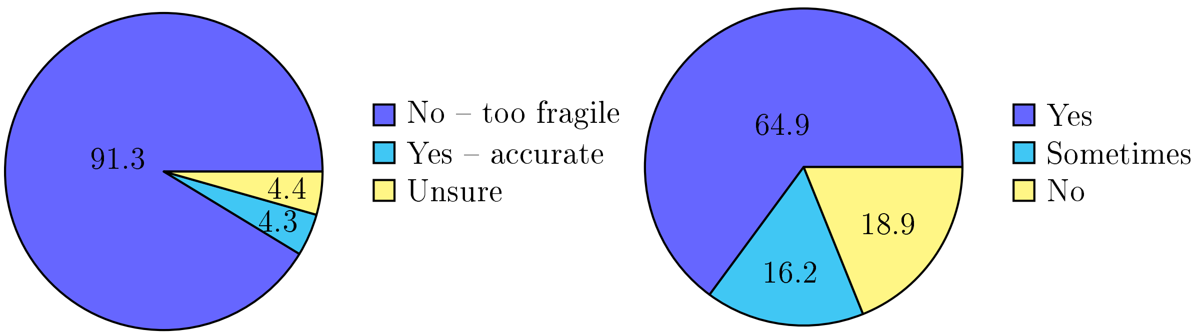

Figure 6: User perception of traditional streak mechanisms: perceived fragility (left, $n = 23$) and stress induction (right, $n = 37$)

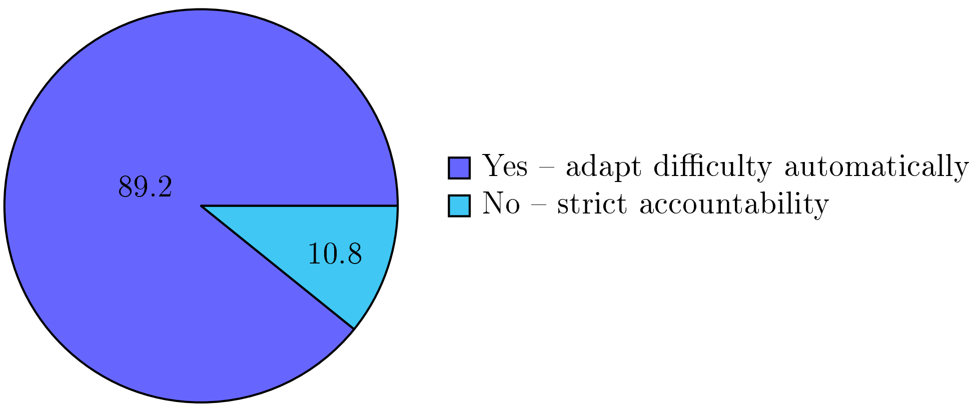

Figure 7: User preference for adaptive difficulty over rigid accountability ($n = 37$)

Technical Justification: Figures 6 and 7 show that binary streak counters are do not represent habit stability well. The view of fragility and stress suggests that a fixed-recurrence model is likely to result in demotivation instead of resilience.

The strong preference for adaptive difficulty (89.2%) supports the implementation of the **SM-2 spaced repetition algorithm** with **logistic regression**. This would allow the system to widen intervals when a higher failure risk is detected. Behavioural momentum is kept going instead of being wiped out by a full reset of progress.

2.5.4 Theme 3: Visualisation and Risk Perception

The survey also looked at how users perceive their habit stability and what kind of feedback they respond to.

- **Invisible Risk: 87%** of users said it is hard to tell the difference between a habit that is "slightly inconsistent" and one "at high risk of total relapse" in current apps.
- **Visualising Decay: 83.7%** said that a dynamic visualisation of *habit decay* would help them prioritise tasks.
- **The Feedback Split:** When asked to choose between "Numerical Data" and "Abstract Visuals," the results were essentially an even split (**51.4% Numerical** vs **48.6% Abstract**).

Technical Justification: The 87% difficulty in identifying risk supports the use of the **physics-inspired simulation** to make *decay* visible. However, the nearly 50/50 split on feedback suggests that using a single visual style (either just graphs or just the simulation) would alienate half of the user base.

Design Consequence: This necessitates a **Dual-Mode Interface**. The system cannot rely solely on a visual *Simulation*. It must implement a *toggleable dashboard* architecture where the user can switch views:

1. **Simulation Mode:** A physics-based *Black Hole* view for users who prefer abstract, intuitive risk assessment (48.6%).
2. **Analytics Mode:** A data-rich dashboard with precision graphs and streak counters for users motivated by raw numbers (51.4%).

2.5.5 Qualitative Feedback Analysis

Users were asked for features that do not currently exist. Several responses directly aligned with the proposed complex algorithms:

> *"To have check-ins that actually do something, some habit trackers I use ask me how I'm doing but... it just logs that and doesn't do anything / change anything."*

Analysis: This participant is describing the frustration of *passive data logging*. In systems engineering terms, they are criticising open-loop systems where output does not affect input. This explicitly validates the need for a **closed-loop control system**. It confirms that the *User Interface* must not just act as a database form, but as a sensor for the **SM-2 algorithm**. Qualitative input (e.g., a difficulty rating) must mathematically alter the system state (future frequency), transforming the application from a passive tracker into an active agent.

> *"A comprehensive FSRS algorithm in order to perfectly time when to complete tasks."*

Analysis: The specific request for FSRS (Free Spaced Repetition Scheduler) suggests a target audience that wants mathematical optimisation over simple tracking. FSRS is a modern stochastic model [17] based on the DSR (Difficulty, Stability, Retrievability) framework, much more complex than standard calendars. While FSRS is designed specifically for memory decay, this feedback is best interpreted as a strong user requirement for **adaptive efficiency**. It explicitly validates the decision to implement the **SM-2 algorithm**, satisfying the user's desire for dynamic scheduling while avoiding the complexity overhead of a machine-learning-based memory model for simple habits.

> *"Integration with calendar systems and smart conflict detection for competing commitments."*

Analysis: This request identifies *Time* as a finite resource that requires management rather than just display. The user's specific terminology—*conflict detection*—implies that the system must handle mutually exclusive events. This elevates the project requirements beyond simple CRUD (Create, Read, Update, Delete) operations and justifies the use of a **Constraint Satisfaction Problem (CSP)** solver. The system must verify that $\text{Time}(Habit_A) \cap \text{Time}(Commitment_B) = \emptyset$, a logical condition that requires the recursive backtracking algorithm proposed in the solution.

> *"I would like the ability to toggle all the different features (streaks, visuals, etc) so that i can choose what i want / what i like."*

Analysis: This qualitative request reinforces the quantitative finding of the "50/50 split" regarding feedback preferences (Theme 3). It indicates that users have distinct mental models: some require strict data accountability, while others require abstract visual motivation. Technically, this validates a **Decoupled View Architecture**. The application must separate the *State* (habit performance data) from the *Representation* (UI). This justifies the use of a component-based framework like **React**, allowing the system to hot-swap between a "Data Dashboard" and the *simulation engine* without altering the underlying data structure, ensuring the system caters to both user personas simultaneously.

REBORN — Computer Science NEA

References

[1] J. Clear, *Atomic Habits: An Easy & Proven Way to Build Good Habits & Break Bad Ones*, English. New York, NY, USA: Avery, an imprint of Penguin Random House, 2018, p. 320, First published October 16, 2018; Hardcover edition; Bestseller on habit formation, ISBN: 9780735211292 (cit. on pp. 1, 5).

[2] R. F. Baumeister and T. F. Heatherton, "Self-regulation failure: An overview," *Psychological Inquiry*, vol. 7, no. 1, pp. 1–15, 1996, Foundational theoretical review defining mechanisms of self-regulation failure across domains. DOI: 10.1207/s15327965pli0701_1 (cit. on p. 2).

[3] R. F. Baumeister et al., "Ego depletion: Is the active self a limited resource?" *Journal of Personality and Social Psychology*, vol. 74, no. 5, pp. 1252–1265, 1998, Classic empirical paper proposing limited self-control resources; central to burnout and sustained effort arguments. DOI: 10.1037/0022-3514.74.5.1252 (cit. on p. 2).

[4] F. M. Sirois, M. L. Melia-Gordon, and T. A. Pychyl, "Procrastination, stress, and chronic health conditions: A temporal perspective," in *Procrastination, Health, and Well-Being*, F. M. Sirois and T. A. Pychyl, Eds., Frames procrastination as a self-regulation failure with downstream health consequences, San Diego, CA: Academic Press, 2016, pp. 67–92. DOI: 10.1016/B978-0-12-802862-9.00004-9 (cit. on p. 2).

[5] A. Buyalskaya et al., "What can machine learning teach us about habit formation? evidence from exercise and hygiene," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 120, no. 17, e2216115120, 2023, Published 17 April 2023; indexed in PNAS; evidence on habit formation curve using large panel datasets. DOI: 10.1073/pnas.2216115120 Accessed: Dec. 18, 2025. [Online]. Available: https://www.pnas.org/doi/10.1073/pnas.2216115120 (cit. on pp. 3, 9).

[6] B. F. Skinner, *The Behavior of Organisms: An Experimental Analysis.* New York, NY: D. Appleton-Century Company, 1938, pp. ix + 456, Foundational work on operant conditioning; original edition predates ISBNs (cit. on p. 4).

[7] Apple Inc. "Healthkit documentation," Accessed: Dec. 15, 2025. [Online]. Available: https://developer.apple.com/documentation/healthkit (cit. on p. 5).

[8] Todoist. "Reply regarding smart schedule accuracy." Official response about Smart Schedule algorithm, Accessed: Dec. 12, 2025. [Online]. Available: https://x.com/todoist/status/1260030543556603904 (cit. on p. 5).

[9] Trevor AI. "Trevor ai: Plan your tasks with todoist." Integration documentation for Trevor AI and Todoist task planning, Accessed: Dec. 12, 2025. [Online]. Available: https://www.trevorai.com/integrations/todoist (cit. on p. 5).

[10] F. Viticci. "Todoist launches smart schedule, an ai-based feature to reschedule overdue tasks." Image showing Todoist's Smart Schedule feature, reproduced from the article, Accessed: Dec. 18, 2025. [Online]. Available: https://www.macstories.net/news/todoist-launches-smart-schedule-an-ai-based-feature-to-reschedule-overdue-tasks/ (cit. on p. 5).

[11] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, English, 4th ed. Harlow, England / London, UK (Global Edition): Pearson, 2020, Approx. 1, 152 (varies by edition), Global Edition, comprehensive textbook on AI covering search, CSPs, reasoning, planning, and machine learning, ISBN: 9781292401133 (cit. on pp. 6, 9).

[12] J. Polivy and C. P. Herman, "Anxiety, restraint, and eating behavior," *Journal of Abnormal Psychology*, vol. 84, no. 6, pp. 666–672, 1975, Early experimental demonstration of counter-regulatory overeating in restrained eaters, foundational to the later "what-the-hell effect" concept in this literature. DOI: 10.1037/0021-843X.84.6.666 (cit. on p. 6).

[13] D. Kahneman, *Thinking, Fast and Slow*, English. New York, NY, USA: Farrar, Straus and Giroux, 2011, p. 498, Explores dual-system theory of cognition and decision making, ISBN: 9780374275631 (cit. on p. 6).

[14] N. Eyal, *Hooked: How to Build Habit-Forming Products*, English. New York, NY, USA: Portfolio (Penguin Random House), 2014, p. 256, Guide to the Hook Model for designing engaging, habit-forming products, ISBN: 9781591847786 (cit. on p. 7).

[15] Doist. "Task quick add." Todoist Help Centre documentation describing natural-language task entry, Accessed: Jan. 16, 2026. [Online]. Available: https://www.todoist.com/help/articles/use-task-quick-add-in-todoist-va4Lhpzz (cit. on p. 7).

[16] P. A. Wozniak. "Optimization of learning: The sm-2 algorithm." Original description of the SM-2 spaced repetition algorithm, Accessed: Dec. 14, 2025. [Online]. Available: https://www.supermemo.com/en/archives1990-2015/english/ol/sm2 (cit. on p. 9).

[17] J. Ye, J. Su, and Y. Cao, "A stochastic shortest path algorithm for optimizing spaced repetition scheduling," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, ACM, 2022. DOI: 10.1145/3534678.3539081 (cit. on p. 14).