Computer Science NEA

REBORN — An adaptive Habit Tracker

REDACTED

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

Centre Name: **REDACTED**

Qualification Code: 7517

January 2, 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
clear2018atomic

In recent years, human attention has turned into a commodified resource. Algorithms running platforms such as TikTok and Instagram are built to maximise engagement through *variable reward schedules***variable-reward-schedules**, at the cost of user agency. As a student taking four A-Levels, I face a constant mental load just to plan and maintain positive habits. Nearly all existing habit tracking tools only allow for static tracking or depend on manual organisation and do not support predictive/adaptive behaviour management.

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 and adaptation, 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. Technology is not only the largest source of procrastination, but also doesn't provide intelligent ways to reduce this problem. 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 the organisation.

The system also addresses a larger audience, such as:

- Professionals who need structured routines and work-life balance.
- People maintaining a regular fitness routine.
- Individuals interested in self-improvement and habit formation.

All of these users encounter self-regulation failure, where manual habit tracking often results in cognitive overload and burnout. By moving planning and predictive analysis to software, REBORN makes habit formation more accessible and responsive to individual circumstances.

2.3 Secondary Research

Current habit-tracking and productivity applications vary significantly in the amount of computation they perform. Some are manual-input database systems, others are game-like behavioural frameworks. A technical analysis 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 **buyalskaya2023what**. 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. 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 behaviours like regularly going to the gym **buyalskaya2023what**.

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.

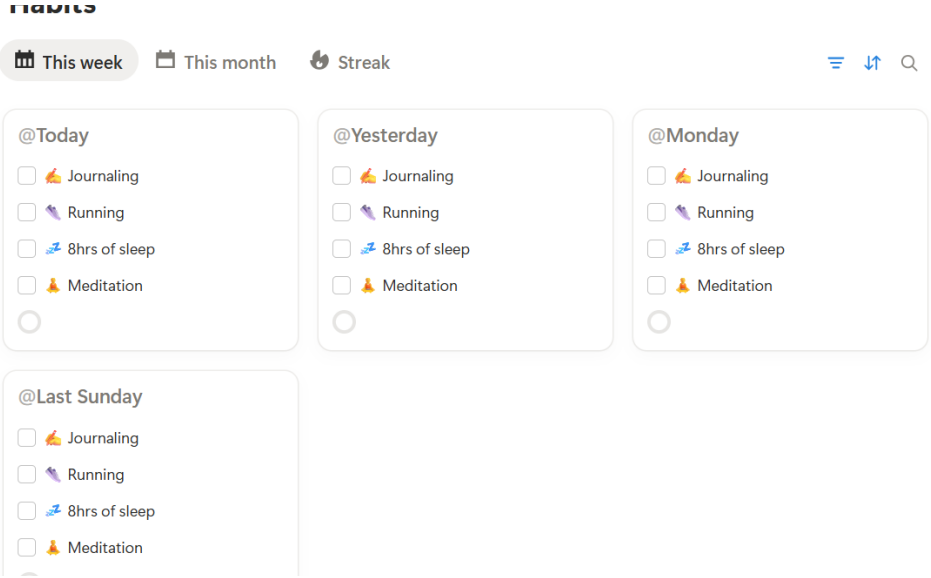

Figure 1: A Notion Habit Tracker showing the manual checkbox system.

Habitica

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

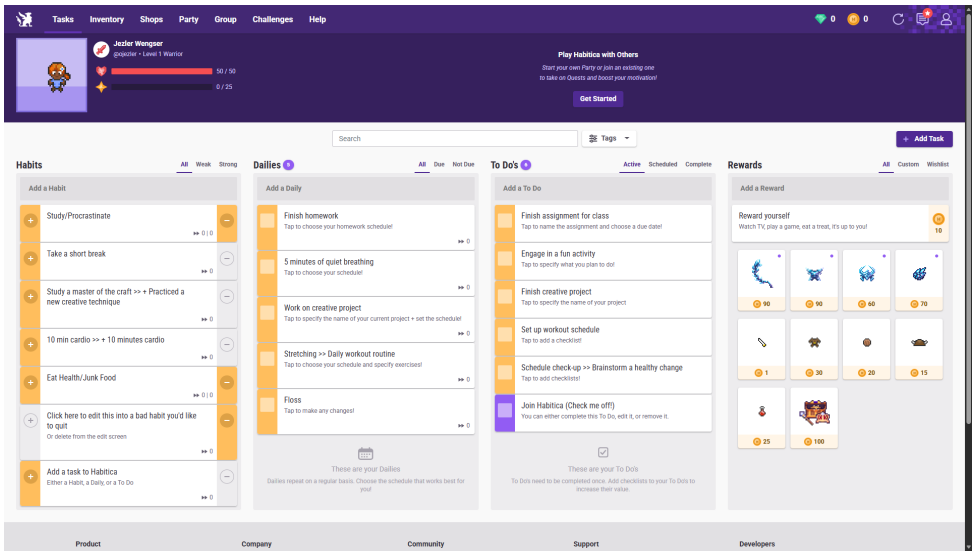

Figure 2: The Habitica dashboard showing the RPG avatar and health bars.

Streaks

Streaks is a minimalist habit tracker deeply integrated into the Apple ecosystem. Unlike database tools, it focuses entirely on the *Don't Break the Chain* psychological method **clear2018atomic**. Technically, it is notable for its API integration with Apple HealthKit **apple2025healthkit**, allowing it to automatically mark habits (e.g., "Walk 5,000 steps") as complete without user input.

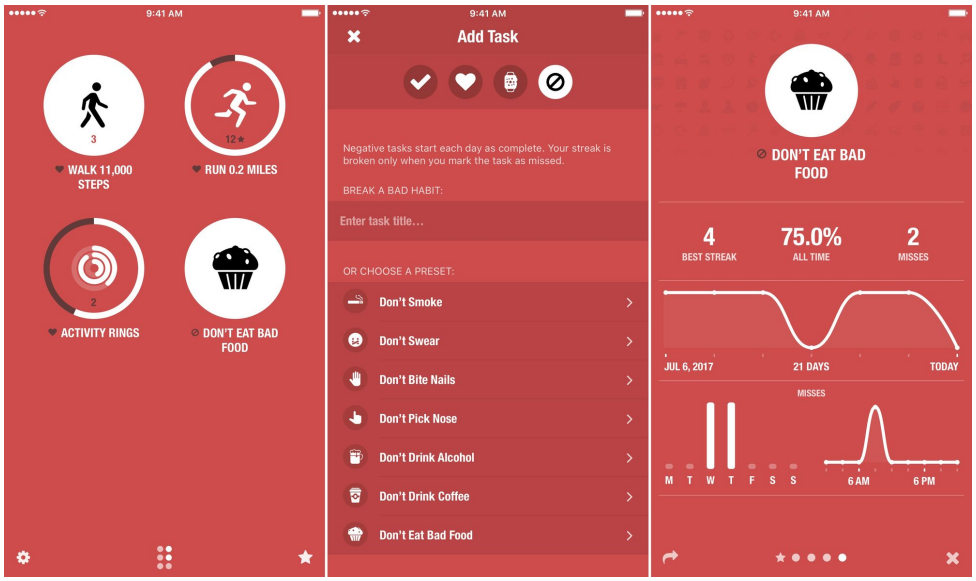

Figure 3: The Streaks interface showing its main features.

Todoist

Todoist is a task management system primarily 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 discontinued natively because of accuracy issues **todoist2020smartschedule**, similar behaviour is achievable through third-party integrations like Trevor AI, which apply heuristic time-blocking and NLP to parse tasks and insert them into available calendar gaps.

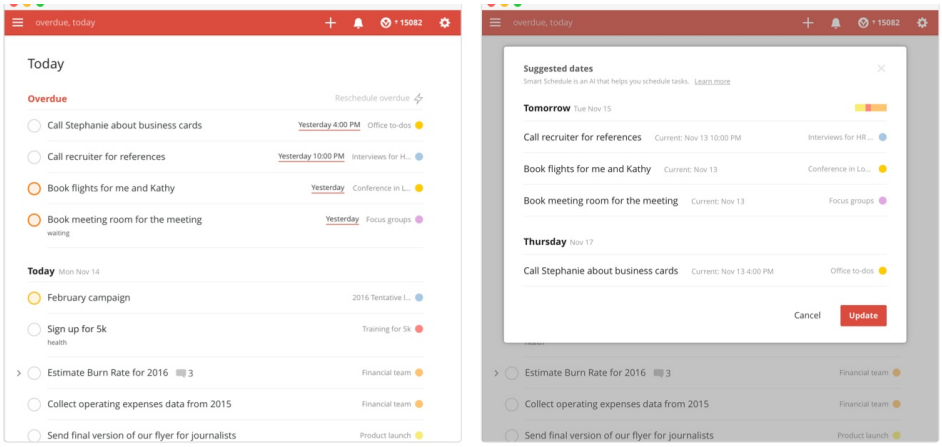

Figure 4: Todoist's "Smart Schedule" interface suggesting times for tasks **macstories2023todoist**.

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
Highly flexible data modeling allowing for complex relationship definitions.	**The *Blank Slate* problem:** Users must build the system themselves; there is no built-in logic.
Visual customisation reduces initial friction for design-oriented users.	**No constraint satisfaction:** It cannot detect schedule conflicts or optimise time slots automatically **russell2020ai**.
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.	**The *All-or-Nothing* flaw:** It relies on strict binary logic. If a user misses one day, the streak resets to zero. This often causes the *What-the-Hell Effect* **polivy1975perceived**, where users abandon the app after a failure.
Visual Feedback: The circular UI provides immediate visual status of daily progress, leveraging *loss aversion* effectively **kahneman2011thinking**.	**No Scheduling Logic:** The app tracks *completion*, but offers no assistance with *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.	Lack of adaptive difficulty; the target remains static regardless of user burnout.

Habitica

Advantages	Disadvantages
Gamification (XP/Gold) uses dopamine loops **eyal2014hooked** to increase initial engagement.	**Static Recurrence:** It relies on fixed rules (e.g., "Every Mon/Tue") rather than 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.
	No awareness of external schedules or calendar conflicts.

Todoist

Advantages	Disadvantages
NLP (Natural Language Processing) enables rapid task entry using natural language **trevorai2025**.	**Task vs. Habit:** The system focuses on one-off tasks and lacks a risk or failure model suitable 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.
	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 critical 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 leads to burnout.

- **I will include:** A strong relational database backend (PostgreSQL) similar to Notion's architecture to ensure 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, rather than punishing them *after*.

From Streaks:
Visualising progress is vital, 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 rather than resetting to zero, which is mathematically more forgiving and encourages recovery.

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 the **SM-2 algorithm** (Spaced Repetition) **wozniak1990sm2** to dynamically reduce habit frequency when users struggle, preventing overload and burnout.
- **I will avoid:** Rigid heuristic scheduling by integrating adaptive control and predictive modelling to maintain long-term adherence.

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 via backtracking or constraint propagation to guarantee conflict-free allocation **russell2020ai**.

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 **wozniak1990sm2**) 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 **buyalskaya2023what** led mathematically with prediction. No system integrates these into a single application.

This presents a clear opportunity to close the gap by moving from passive data recording to active, algorithmic planning, treating habit formation as a computational problem instead of just a psychological one.

REBORN — Computer Science NEA