**Habitarium**

**Project Name:** Habitarium  
**Subtitle:** A Gamified Habit Tracker with Simulated Blockchain Integration  
**Hackathon Name:** HACKHAZARDS ‘25  
**Team Name:** SUPERNOVA

**Team members :** Arunava Mukherjee , Srijan Das , Roki Mondal

**Date:** 27/04/2025

**Abstract / Executive Summary**

Habitarium is a proof-of-concept web application designed to combat the lack of engagement in traditional habit tracking. It transforms building positive routines into a motivating game by rewarding users with Experience Points (XP), levels, Habitarium Points (HP currency), and streak bonuses for consistent daily logging. Key features include secure user authentication, full habit management (CRUD), a gamified logging system, a reward shop where users spend HP on profile titles, and the ability to view log history. The project successfully implements these core mechanics locally, utilizing a React/Mantine frontend communicating with a Node.js/Express/PostgreSQL backend. Due to persistent testnet deployment issues (HH110 error) with the target Monad blockchain, the intended on-chain interactions for logging and rewards are currently simulated within the backend, demonstrating the planned architecture and user flow.

**Introduction**

**Why this project?**

**The Problem:** Establishing and maintaining positive habits is a cornerstone of personal growth, yet it remains a significant challenge for many. Conventional habit trackers often fail to sustain user motivation; they can feel like a chore, lack engaging feedback loops, and don't always provide meaningful incentives for consistency. This leads to users losing interest and abandoning their self-improvement goals.

**Our Motivation & Vision:** We were driven by a shared interest in behavioural psychology, gamification, and the potential of blockchain technology. We saw an opportunity to create a habit-tracking experience that was fundamentally more engaging and rewarding than existing simple tools. Our vision was to build Habitarium: a platform that not only tracks habits but actively motivates users through game mechanics and provides a sense of accomplishment, with a forward-thinking plan to leverage blockchain for verifiable proof-of-consistency and unique digital rewards.

**Introducing Habitarium:** Habitarium is our solution – a web application designed to make habit building enjoyable. By integrating game elements like points, levels, streaks, and redeemable rewards, we aim to provide the positive reinforcement needed to turn desired behaviour into lasting habits.

**Features & Functionality**

**What does it do?**

Habitarium currently offers the following core features in a locally runnable prototype:

* **User Authentication:**
  + Secure user registration using Email and Password.
  + Password hashing via bcryptjs.
  + Secure login returning a JSON Web Token (JWT).
  + JWT stored in frontend local Storage and automatically sent with subsequent API requests.
  + Protected backend routes ensure only logged-in users can access their data.

**Dashboard (Main Hub):**

* + Displays key gamification statistics: Current Level, XP progress bar towards the next level, and total HP (Habitarium Points).
  + Shows the user's currently "Active Title" if one has been redeemed and set.
  + Lists the user's habits as individual cards, showing name, description, frequency, current streak, and longest streak.
  + Provides clear visual cues on Goals, habit cards (e.g., border color change) if a habit has been logged for the current day.
* **Habit Management (CRUD):**
  + **Create:** Add new habits via a modal form (Name, Description, Frequency).
  + **Read:** View all habits listed on the dashboard.
  + **Update:** Edit existing habits using a pre-filled modal form.
  + **Delete:** Remove habits with a confirmation step.
* **Habit Logging & Gamification:**
  + Dedicated "Log" button on each habit card.
  + Clicking "Log" attempts to record completion for the current day.
  + **Feedback:** Upon successful logging (or detecting an existing log for the day), the backend calculates and the frontend displays:
    - XP and HP awarded (based on habit frequency, potential streak bonuses).
    - Updates to the current streak counter.
    - Updates to the longest streak (if applicable).
    - Alerts/notifications if the user levels up.
  + Prevents duplicate logging for the same habit on the same day.
  + Streak logic correctly increments or resets based on daily logging consistency.
* **Logs Page:**
  + Displays a sortable table showing the user's entire habit logging history (Habit Name, Date Logged).
* **Reward System**
  + **Rewards Shop Page:** Displays a list of available rewards (currently predefined "Titles" stored in the backend) showing their name, description, and HP cost. Provides a "Redeem" button.
  + **Redeeming:** Clicking "Redeem" checks if the user has sufficient HP, deducts the cost, and unlocks the reward for the user (stored in a user\_rewards table). Feedback provided via notifications.
  + **My Rewards Page:** Lists all rewards the user has successfully redeemed/unlocked.
  + **Set Active Title:** Allows the user to select one of their unlocked titles to be their "Active Title", which is then stored on their user profile and displayed on the dashboard.
* **Settings Page:**
  + Provides a critical **Account Deletion** feature, including a confirmation modal to prevent accidental deletion.
* **UI/UX:**
  + Built using the **Mantine UI** component library for a clean, modern, and consistent visual style.
  + Utilizes Mantine components like AppShell, Cards, Modals, Buttons, Badges, Progress Bars, Forms, Notifications.
  + Includes a functional **Dark Mode** theme toggle.
  + Header includes User Icon Menu for easy navigation to Logs, Rewards, Settings, and Logout. Basic responsiveness handled by Mantine's components.

**Technical Implementation**

**How was it built?**

* **Architecture:** Habitarium follows a standard client-server architecture.
  + A **React Single-Page Application (SPA)** serves as the frontend, handling user interaction and presentation.
  + The frontend communicates with a **Node.js/Express RESTful API** backend via HTTP requests (using Axios).
  + The backend API handles business logic (authentication, gamification calculations, data validation) and interacts with a **PostgreSQL relational database** for data persistence.
  + The architecture included a designated point in the backend logic (within logging and reward redemption) for **intended integration with Monad smart contracts**, which is currently simulated

.

* **Tech Stack:**
  + **Frontend:** React 18, TypeScript, Vite (build tool), Mantine UI (@mantine/core, @mantine/hooks, @mantine/form, @mantine/notifications), React Router DOM (v6), Axios, Tabler Icons (@tabler/icons-react).
  + **Backend:** Node.js (v18+), Express 4.x, TypeScript, PostgreSQL (via pg library v8+), jsonwebtoken (JWT handling), bcryptjs (password hashing), cors, dotenv.
  + **Database:** PostgreSQL (v16/17).
  + **Contracts (Intended/Compiled):** Solidity (v0.8.20), OpenZeppelin Contracts, Hardhat (development environment for compilation, testing, deployment scripting).
  + **Version Control:** Git, GitHub.

**Deployment :**

* RewardToken (HABIT): 0x7D74781172d079aDc32763a4c6cf699881ED07Ff
* RewardItem (NFT): 0xCa27F7ab2AfaE86fe53c98edD966563B719B96d9
* HabitLog: 0x4a814Ab0Bc189F0BC58F159d80426218fD720573

**The Development Journey**

**Hurdles & Overcomes**

The development process involved several learning experiences and pivots:

* **Initial Choices:** We opted for the Node.js/Express/PostgreSQL backend and React frontend stack due to its popularity, robustness, extensive community support, and our team's familiarity, allowing for rapid development. TypeScript was chosen for enhanced type safety in both frontend and backend.
* **UI Hurdles & Pivot:** Initial attempts to style the application faced challenges [Optional: Mention specific issues e.g., with plain CSS complexity or integration issues with another library like Chakra UI]. This led us to evaluate and adopt the **Mantine UI library**. While requiring a quick learning curve, Mantine significantly accelerated development with its comprehensive set of pre-built, customizable components, hooks, and theming capabilities, contributing greatly to the final polished look and feel.
* **Backend Logic Implementation:** Building the REST API involved standard CRUD endpoint creation. Key complexities arose in:
  + Implementing secure JWT authentication middleware (protect function) to safeguard routes.
  + Designing and integrating the **gamification logic** directly into the logHabitCompletion endpoint. This involved fetching streak data, calculating XP/HP based on rules (e.g., frequency, streak bonuses), handling level-ups based on an XP threshold, and updating relevant user/habit data in a single atomic flow.
  + Adding the **rewards system logic**, including checking HP balance, deducting HP upon redemption, storing unlocked rewards, and updating the user's active title.
  + Implementing the secure account deletion feature, ensuring proper data removal and user confirmation.
* **Frontend Logic Implementation:** Key tasks included setting up routing with React Router, creating a reusable Axios instance with interceptors to automatically attach the JWT token, managing component-level state using useState for forms, habit lists, and gamification displays, implementing API calls for all features, and providing user feedback through Mantine Notifications and loading state indicators.
* **Key Milestone:** A significant moment was achieving the core user loop: successfully registering/logging in, adding a new habit via the frontend form, logging its completion, and seeing the updated XP/HP/Streak reflected on the dashboard (even if requiring a manual refresh initially).

**Blockchain Integration**

**The Monad Challenge**

* **The Intention:** We chose to target the Monad blockchain for its potential high throughput and efficiency. Our goal was to leverage blockchain's core properties for:
  + **Immutability:** Creating a permanent, verifiable record of habit completion logs on-chain.
  + **Rewards & Ownership:** Enabling potentially valuable tokenized (ERC20) or NFT (ERC721) rewards that users truly own.
  + **Transparency:** Providing an optional layer of public accountability (though user data privacy was also a consideration).
* **What Was Built:** We successfully developed and compiled Solidity smart contracts using Hardhat:
  + HabitLog.sol: Intended to store habit completion records immutably.
  + RewardToken.sol: A standard ERC20 contract for the HP currency (or a future governance token).
  + RewardItem.sol: An ERC721 contract intended for potential NFT rewards (like unique titles or cosmetic items).
  + Deployment scripts using Hardhat were also prepared.
* **The Simulation Pivot:** Despite having compiled contracts, we encountered a persistent blocker during deployment to the Monad Testnet via Hardhat. The specific error was HH110: Invalid JSON-RPC response received: [...]. Extensive debugging attempts, including verifying RPC URLs, checking Hardhat configuration, ensuring sufficient testnet funds, and consulting Monad documentation/community channels, did not resolve the issue within the hackathon timeframe.
* **How Simulation Works:** To demonstrate the intended architecture and user experience flow, we implemented a simulation layer in the backend controllers (habit.controller.ts, reward.controller.ts). Instead of making actual ethers.js calls to the deployed contracts, the backend now includes console.log statements indicating where a contract call *would* occur (e.g., [HabitCtrl] Placeholder: Would call HabitLog.recordCompletion(...) here.). Upon successful database updates for logging or redeeming rewards, the backend API response includes a blockchainTxHash: "SIMULATED\_HASH\_..." field. The frontend displays this simulated hash in notifications, mimicking the feedback loop of a successful blockchain transaction without actually executing one. This allowed us to build and test the full application flow while decoupling it from the unresolved deployment issue.

**Future Scope & Vision**

**Where could this go? Why is it a breakthrough?**

Habitarium has significant potential for growth beyond this initial prototype:

* **Immediate Next Steps:**
  + **Resolve Monad Deployment:** Prioritize fixing the HH110 deployment error to enable true blockchain integration.
  + **Implement Global State Management:** Introduce React Context API or Zustand/Redux to eliminate minor UX frictions like the delayed HP update in the header upon reward redemption.
  + **Enhance Feedback:** Add more visual cues and notifications, such as level-up celebrations.
* **Medium-Term Enhancements:**
  + **Expanded Gamification:** Implement a dedicated achievements system, weekly/monthly challenges, and potentially leaderboards (optional/privacy-conscious).
  + **Social Features:** Allow users to optionally share milestones or form accountability partnerships.
  + **NFT Rewards:** Fully implement the RewardItem.sol contract to issue unique, visually distinct NFT rewards (e.g., profile badges, companion customizations).
  + **User Insights:** Develop a dashboard page with charts visualizing habit consistency, streak patterns, and other relevant data.
* **Long-Term Vision & "Breakthrough" Potential:**
  + **True Decentralization:** Explore storing essential logs and potentially reward ownership entirely on-chain via Monad, balancing decentralization benefits with transaction costs and privacy.
  + **Tokenomics:** Design and potentially launch a native HABIT token, earned through consistent participation, usable for premium features or potentially integrated with DeFi.
  + **Personalized AI Coaching:** Leverage aggregated, anonymized (potentially on-chain) data to offer AI-driven insights and personalized habit recommendations.
  + **Interoperability:** Allow users to showcase their Habitarium achievements (NFTs, Titles) in other compatible metaverse or Web3 platforms.
  + **Why a Breakthrough?** Our long-term vision sees Habitarium evolving beyond a simple tracker into a user-centric ecosystem for personal growth. By combining engaging gamification with the unique properties of blockchain (verifiable consistency, true digital ownership of rewards, potential for tokenized incentives), Habitarium has the potential to fundamentally shift how individuals approach self-improvement, making it more motivating, rewarding, and integrated with the emerging decentralized web.

**Conclusion**

During this hackathon, we successfully developed Habitarium, a functional local prototype of a gamified habit tracker. It features secure authentication, comprehensive habit management, and an engaging core loop rewarding users with XP, HP, levels, and streaks for logging habits, complete with a basic reward redemption system. The application boasts a modern UI built with React and Mantine UI. While the full integration with the Monad blockchain was hampered by persistent testnet deployment issues, we developed the necessary smart contracts and implemented a simulation layer in the backend to demonstrate the intended architecture and user experience. Habitarium showcases a compelling blend of behavioral design and gamification, with a clear roadmap for future blockchain integration and feature expansion, positioning it as a promising platform for motivating personal growth in the Web3 era.

**Links**

* **GitHub Repository Link:** https://github.com/Phoenix79-ai/Habitarium
* **Video Demo Link:** https://youtu.be/LC3CbB7zhkk