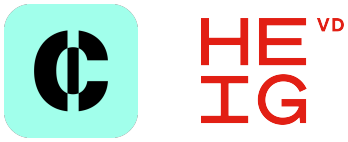

Software Requirements Specification

commit.

PDG - HEIG-VD

Leonard Cseres Tristan Gerber Aladin Iseni David Schildböck

September 4, 2025

Contents

1 Introduction

- **Purpose**:
  - Primary users: Private adult individuals who want to take action in their lives and seek stronger follow-through on commitments.
  - Core problem: Providing extra motivation to get things done by introducing a financial stake and social accountability.
  - Top objectives:
    1. Increase goal adherence by letting users stake money
    2. Enable social accountability via groups and shared challenges with pooled stakes.
    3. Provide automated financial consequences when goals are missed.
  - Success metrics:
    * Successfully transfer money from one user to another
    * Cross platform mobile application (IOS and Android)
- **Scope**:
  - Platforms: iOS and Android via Expo; web landing page (marketing/information only).
  - Geography at launch: Switzerland.
  - User types: Standard users, reviewer, and admin.
  - MVP goal types: checkin (time-based), photo (manual review), movement (timer). Location method exists in the model but no server action endpoint in MVP.
  - Money flow: Stripe card payments (PaymentIntents) and Customer Balance credits for winners; no TWINT in MVP.
  - Group challenges: Invite-only, private groups via invite codes (no expiration in MVP).
  - Out of scope (MVP): App Store/Play Store deployment, tablet support, advanced analytics, notifications, appeals/dispute handling, dedicated location-action endpoint, goal editing, invite expiration, fallback destination when all fail.
  - Languages: English only at launch.
  - Accessibility & compliance: Out of scope for MVP.

2 Project Description

- **Objective**:
  - Product vision: Commit turns intentions into actions by combining simple goal tracking with real financial stakes and social accountability.
  - Primary use cases:
    1. Daily wake-up challenge at a set time
    2. Location-based activity (arrive and stay at a gym/park)
    3. Focused session for a specified duration (no-phone-use)
  - Differentiators: Ease of use, pooled stakes for group challenges.
  - Constraints/guiding principles: Minimize operational fees with Stripe; avoid bank payouts in MVP.

3 System Overview

- High-level description of the app
  - Solo flow: A user creates a goal, due date or recurrence, selects a verification method (check-in/photo/movement), sets a stake amount and a destination for funds. If the goal is completed and verified within the rules, no transfer occurs; if not, the staked amount is transferred to the configured destination.
  - Group flow: A user creates a private, invite-only group challenge with a defined goal, due date or recurrence, selects a verification method (check-in/photo/movement) and stake amount. Invitees accept the stake. Upon completion, successful participants receive the pooled stakes from members who failed. If winners=0 or losers=0, no redistribution occurs in MVP.
- Mobile: iOS and Android (phones only; no tablet support. Expo + React Native)
- Permissions/capabilities: Camera; movement timer in-app. Location method exists in the data model but no server action endpoint in MVP.
- Web: Single static marketing landing page (Astro)
- Target audience
  - Adults 18+ seeking productivity, fitness improvements, and habit-building.

4 Functional Requirements

4.1 Roles

- **User:**
  - Standard user role
- **Reviewer:**
  - Has all the capabilities of a standard user.
  - Can access and review user-submitted pictures for goal verification.
  - Can approve or reject verification evidence as part of the manual review process.
- **Admin:**
  - Role exists in the system but currently has no special functions defined for MVP.
  - Future admin capabilities will be specified as the project evolves.

4.2 User Management

- Users can register
- Users can manage their profile

4.2.1 Authentication

- Sign-in methods: Google, Apple via Better-Auth/Expo (no email/password at MVP)
- Sessions: handled by Better-Auth on Cloudflare Workers
- Dev/preview: header `X-Commit-Dev-Auto-Auth: <email>` may impersonate seeded users
- Pre-stake verification: No email/phone verification required before staking
- KYC/identity checks: Not required at MVP for credits via Stripe Customer Balance

4.2.2 Payments & Account Receivers (MVP decisions)

- Charging model: Stakes are defined at creation but only charged if a user fails at settlement; if achieved, no funds are captured.
- Currency: CHF only at launch.
- Recipients and settlement:
  - Solo challenge: on failure, the owner is charged; the money goes to the developers (if possible in MVP, to charities).
  - Group challenge: losers are charged; winners are credited evenly via Stripe Customer Balance. If winners=0 or losers=0, no redistribution occurs.
- Platform fees: No platform commission (Stripe fees apply).
- Payment methods: Stripe cards; users save a card via SetupIntent. Group join does not require a saved card upfront in MVP.
- Payout timing: Customer Balance credits for winners; no bank payouts in MVP.

4.3 Core Features

- Goal lifecycle: create, delete, view;
- Group lifecycle: create private group, invite via code, join/leave, view results
- Verification capture: check-in (time), in-app photo (manual review), movement timer (start/stop). Location verification not available in MVP.
- Money: charge losers, credit winners via Stripe Customer Balance; no fallback destination in MVP

4.4 Goal Creation

- Required fields: name, stake, start date, and a scheduling mode: either a single window or a weekly recurrence using a bitmask of the days of a week between the start and endDate (not required if not a recurrence). Description is optional.
- Method: one of checkin, photo, movement. Movement requires a duration. There is an optional grace time that applies after the window ends. Location is implemented only if there is time for it in MVP.
- Constraints: must have coherent dates (starting date $<$ end date) and timestamps (start time $<$ end time).
- Verification behavior:
  - Check-in: immediate approval when performed within the active window (or grace).
  - Photo: user uploads during the window; approval is manual by reviewers.
  - Movement: user starts/stops a timer; must satisfy the duration within the allowed window.
- Failure: missing or late verification leads to failure; no retries beyond grace in MVP.

4.5 Group Challenges

- Stake uniformity: all participants share the same goal and stake.
- Join flow: invite via code. Joiners must register/sign in.

- Schedule: groups reference a single shared goal; all participants follow that goal's schedule.
- Distribution: on resolution, winners split losers' captured stakes evenly. If winners=0 or losers=0, no redistribution occurs in MVP.
- Failure to verify: not providing required verification within the window is an automatic failure.
- Cancellation: deleting the group (and its goal) before settlement results in no captures.

4.6 Settings

- Update payment details

4.7 Optional Functions (Future Enhancements)

- **Community Challenges:** Public or open group challenges where any user can join and compete, with shared stakes and leaderboards.
- **AI Image Check:** Automated verification of user-submitted photos using AI to reduce manual reviewer workload and improve scalability.
- **Additional Payment Methods:** Support for more payment options beyond cards (e.g., Twint, Apple Pay, bank transfers, PayPal).
- **Charity Donations:** Users can choose a charity to donate to when selecting the destination for a goal.

5 Non-Functional Requirements

5.1 Security

- Data encryption in transit and at rest
- Authentication: Better-Auth sessions (managed by Workers) with Google/Apple social sign-in. No email/password in MVP. Dev/preview may use a trusted header to impersonate seeded users.
- Photo storage: stored as objects in Cloudflare R2; access is reviewer-only via a protected serve endpoint. Retention policy: out of scope for MVP.
- Secrets: Stripe keys and Cloudflare credentials managed via Cloudflare project settings; no secrets in the repository.

5.2 Privacy

- Photos are stored for verification purposes only; access is restricted to the account owner and authorized reviewers.
- Compliance posture: out of scope for MVP.

5.3 Usability

- Intuitive navigation
- Consistent UI across platforms

5.4 Reliability & Availability

- Graceful error handling

5.5 Compatibility

- Responsive design for different iPhone/android screen sizes

5.6 Offline Behavior

- Online-only MVP: goal creation and verification require connectivity

6 Preliminary Architecture Description

- Presentation layer: React Native mobile app
- Application layer: Expo/React Native (frontend); Cloudflare Workers backend using Hono with chanfana (OpenAPI) and Better-Auth; scheduled Worker runs settlements.
- Data layer: Cloudflare D1 (SQLite-based relational DB)
- Infrastructure: Cloudflare Workers + R2 + D1 (global CDN/edge)

7 Mockups / Landing Page

- Figma designs
- Paper/whiteboard sketches

8 Technical Choices

- Programming languages & frameworks: Expo + React Native (TypeScript)
- Database: Cloudflare D1 (SQLite)
- Backend/services: Cloudflare (Workers, D1, R2) with Hono + chanfana
- Payments: Stripe cards only (SetupIntents, PaymentIntents); winners credited via Customer Balance;
- Third-party libraries & APIs: Stripe SDK, Better-Auth, Expo Camera
- Hosting: Cloudflare (backend, db, workers); Cloudflare Workers (Astro landing page)

9 Work Process

9.1 Agile Methodology: SCRUM

- **Team roles:** Roles (Product Owner, Scrum Master, Developers) are rotating among team members throughout the project.
- **Sprint length:** Each sprint lasts 3 days, reflecting the short 3-week project timeline.
- **Ceremonies:** The team holds a daily standup and a sprint review at the end of each sprint. Other SCRUM ceremonies (planning, retrospective) are adapted or combined as needed.

- **Backlog management:** All tasks and user stories are tracked as GitHub Issues, organized and prioritized in a GitHub Project Kanban board.
- **Definition of Done:** A task is considered done when the code is merged, all tests pass, and the feature works as intended.
- **Sprint goal:** Each sprint has a defined goal or deliverable to be implemented.
- **Estimation:** Tasks are estimated in terms of expected time to complete.
- **Review and acceptance:** The team collectively reviews and accepts completed work at the end of each sprint.
- **Adaptations:** Roles may be combined and ceremonies adapted based on project advancement and team needs, in line with the university context.
- **Process**: lightweight Kanban for MVP

9.2 Git Flow

- The `main` branch contains production code.
- Every feature/fix/task is discussed via an issue and a branch is created with the issue.
- All changes are integrated via pull requests with code review and CI checks

9.3 Devops

- Continuous delivery to production when changes pass CI and review

10 Development Tools Setup

- Issue tracker: GitHub Issues
- Code review: GitHub pull requests
- Documentation: repository README and SRS in `docs/`
- Code style: Prettier + ESLint with TypeScript rules
- Testing approach: Vitest for API Workers and React Native tests where applicable
- Secrets/config: Cloudflare and Stripe keys via env files with secure storage

11 Deployment Environment

- Mobile app backend/API: Cloudflare Workers (API + business logic + auth), D1 (relational DB), R2 (image storage).
- Web landing page: Astro static site deployed on Cloudflare Workers (global CDN).
- Mobile app: Expo EAS builds with two channels: development and production. App Store/Play Store distribution is out of scope for MVP; internal distribution only (TestFlight/Android internal testing).
- Environments: development, preview, and production with separate Cloudflare projects or namespaces and isolated resources.
- Secrets/configuration: managed via Cloudflare project settings, and EAS secrets. Stripe runs in Test mode for development and Live mode for production.

12 CI/CD Pipeline

- CI on PRs: lint and tests
- CI on `main`: build Expo APK and compile docs
- CD on `main`: deploy static website
- CD `manually`: release mobile (from build APK)

13 Constraints & Assumptions

- Budget & time constraints: 70K CHF budget; 3-week MVP timeline
- Regulatory compliance: out of scope for MVP

A Appendix

A.1 Glossary

Stake	The amount of money a user commits that may be captured if the goal is not achieved.
Capture	Charging the authorized stake when a goal is marked as failed.
Goal window	The scheduled time period during which the user must complete and verify the goal.
Verification window	The allowed buffer around the scheduled time for submitting verification.
Geofence	A virtual radius around a location used to verify presence.
Dwell time	The minimum time a user must remain inside a geofence.
Destination	The recipient configured to receive funds when a goal fails (person, charity, or platform donation).
Winner pool	The set of participants in a group challenge who achieved the goal and split the captured stakes.
Group challenge	A private, invite-only challenge with a uniform stake and shared rules created by a user.