Stellar-Journey — Master Spec Inventory

Exhaustive, enriched version — exported on 2025-09-24

Table des matières

[Metadata & Status 8](#_Toc209794013)

[Glossary 9](#_Toc209794014)

[1. Vision & Targets 11](#_Toc209794015)

[2. Architecture Overview (ARC) 12](#_Toc209794016)

[1.1. Application Architecture (ARC-APP) 12](#_Toc209794017)

[1.2. Data Management (ARC-DAT) 12](#_Toc209794018)

[1.3. Security Architecture (ARC-SEC) 12](#_Toc209794019)

[1.4. Functional Scope (ARC-FSC) 12](#_Toc209794020)

[1.5. 2.4 Back-end (ARC-BCK) 13](#_Toc209794021)

[1.6. 2.5 Online Mode (ARC-ONL) 13](#_Toc209794022)

[2. Non-Functional Requirements (NFR) 14](#_Toc209794023)

[2.1. Performance (NFR-PER) 14](#_Toc209794024)

[2.2. Usability & Accessibility (NFR-ACC) 14](#_Toc209794025)

[2.3. Security (NFR-SEC) 14](#_Toc209794026)

[3. User Profiles & Environment (USR) 15](#_Toc209794027)

[3.1. Target Users (USR-PRF) 15](#_Toc209794028)

[3.2. Supported Environments (USR-ENV) 15](#_Toc209794029)

[4. Information Architecture & Navigation 16](#_Toc209794030)

[4.1. Primary tabs (left to right in navbar) 16](#_Toc209794031)

[4.2. Default landing 16](#_Toc209794032)

[4.3. Secondary navigation 16](#_Toc209794033)

[4.4. Global navigation 16](#_Toc209794034)

[4.5. Cross-linking 17](#_Toc209794035)

[4.6. Accessibility (echo) 17](#_Toc209794036)

[5. Modules & Features 18](#_Toc209794037)

[5.1. Core modules 18](#_Toc209794038)

[5.2. Supporting modules 20](#_Toc209794039)

[6. Data & Storage 24](#_Toc209794040)

[7. UX/UI & Accessibility 25](#_Toc209794041)

[7.1. Visual Style 25](#_Toc209794042)

[7.2. Accessibility (WCAG 2.2 AA) 26](#_Toc209794043)

[7.3. Navigation & Responsive Layout 27](#_Toc209794044)

[7.4. Interactive Elements 27](#_Toc209794045)

[7.5. Animations 28](#_Toc209794046)

[7.6. Brand Voice & Tone 28](#_Toc209794047)

[7.7. Design System & Tokens 29](#_Toc209794048)

[8. Visual Identity (brand kit) 30](#_Toc209794049)

[8.1. Brand Concept 30](#_Toc209794050)

[8.2. Logo & Iconography 30](#_Toc209794051)

[8.3. Color System 30](#_Toc209794052)

[8.4. Typography 31](#_Toc209794053)

[8.5. Imagery & Backgrounds 31](#_Toc209794054)

[8.6. Effects & Surfaces 31](#_Toc209794055)

[8.7. Iconography & States 31](#_Toc209794056)

[8.8. Brand Voice & Copy Guidelines 31](#_Toc209794057)

[8.9. Theming & Extensibility 32](#_Toc209794058)

[8.10. Documentation & Assets 32](#_Toc209794059)

[9. Settings 33](#_Toc209794060)

[9.1. General Structure 33](#_Toc209794061)

[9.2. General 33](#_Toc209794062)

[9.3. Security 33](#_Toc209794063)

[9.4. Accessibility 34](#_Toc209794064)

[9.5. Habits 34](#_Toc209794065)

[9.6. Data & Storage 34](#_Toc209794066)

[9.7. Notifications 34](#_Toc209794067)

[9.8. About 35](#_Toc209794068)

[9.9. UX Considerations 35](#_Toc209794069)

[10. Security & Privacy 36](#_Toc209794070)

[10.1. Purpose 36](#_Toc209794071)

[10.2. Data Protection 36](#_Toc209794072)

[10.3. Authentication 36](#_Toc209794073)

[10.4. Import/Export Integrity 36](#_Toc209794074)

[10.5. Frontend Security 36](#_Toc209794075)

[10.6. User Safety UX 37](#_Toc209794076)

[10.7. Privacy 37](#_Toc209794077)

[10.8. Security Architecture (Local + Online) 37](#_Toc209794078)

[10.9. Auditing and Testing 39](#_Toc209794079)

[10.10. Brain Dump 39](#_Toc209794080)

[11. Performance & PWA 40](#_Toc209794081)

[11.1. Purpose 40](#_Toc209794082)

[11.2. Core Performance Targets 40](#_Toc209794083)

[11.3. Bundle and Asset Budgets 40](#_Toc209794084)

[11.4. Build and Delivery 40](#_Toc209794085)

[11.5. Service Worker (SW) 41](#_Toc209794086)

[11.6. Offline-first Data 41](#_Toc209794087)

[11.7. PWA Installability 41](#_Toc209794088)

[11.8. Resource and Battery Friendliness 41](#_Toc209794089)

[11.9. Monitoring and Diagnostics 41](#_Toc209794090)

[12. Packaging & Deployment 43](#_Toc209794091)

[12.1. Purpose 43](#_Toc209794092)

[12.2. Packaging Targets 43](#_Toc209794093)

[12.3. Deployment Environments 43](#_Toc209794094)

[12.4. Build Process 43](#_Toc209794095)

[12.5. Deployment Workflow 44](#_Toc209794096)

[12.6. Versioning & Releases 44](#_Toc209794097)

[12.7. Security Considerations 44](#_Toc209794098)

[12.8. Documentation 45](#_Toc209794099)

[13. Quality & Testing 46](#_Toc209794100)

[13.1. Purpose 46](#_Toc209794101)

[13.2. Test Strategy (Overview) 46](#_Toc209794102)

[13.3. Environments & Modes Under Test 46](#_Toc209794103)

[13.4. Test Suites & Tools 47](#_Toc209794104)

[13.5. Degraded Mode Tests (Must Pass) 48](#_Toc209794105)

[13.6. Test Data & Fixtures 48](#_Toc209794106)

[13.7. Security & Privacy Checks (Lightweight) 48](#_Toc209794107)

[13.8. Internationalisation Tests 49](#_Toc209794108)

[13.9. Accessibility (Descriptive + Checks) 49](#_Toc209794109)

[13.10. Release Smoke Checklist (Manual, 10–15 minutes) 49](#_Toc209794110)

[13.11. CI (GitHub Actions) — Solo Friendly 49](#_Toc209794111)

[13.12. Definition of Done (DoD) 50](#_Toc209794112)

[13.13. GDPR / RGPD Compliance Checks 50](#_Toc209794113)

[13.14. Brain Dump 51](#_Toc209794114)

[14. Import/Export Schemas 52](#_Toc209794115)

[14.1. Task 52](#_Toc209794116)

[14.2. Routine 52](#_Toc209794117)

[14.3. RoutineLog 53](#_Toc209794118)

[14.4. AggregatedStats 53](#_Toc209794119)

[14.5. Habit 54](#_Toc209794120)

[14.6. ScheduleEvent 54](#_Toc209794121)

[14.7. NoteLog (Planning Import) 54](#_Toc209794122)

[14.8. Import Compatibility — "Daily Note Markdown" (SJ-MD-DN v1) 55](#_Toc209794123)

[14.9. Markdown bundle mapping 59](#_Toc209794124)

[15. Internationalisation (i18n) 60](#_Toc209794125)

[15.1. Purpose 60](#_Toc209794126)

[15.2. Supported Languages 60](#_Toc209794127)

[15.3. Language Selection 60](#_Toc209794128)

[15.4. Text Management 60](#_Toc209794129)

[15.5. Date, Time, and Number Formatting 61](#_Toc209794130)

[15.6. Accessibility of Localisation 61](#_Toc209794131)

[15.7. Content to Translate 61](#_Toc209794132)

[15.8. Non-Translated Content 61](#_Toc209794133)

[15.9. Contribution & Extensibility 61](#_Toc209794134)

[15.10. Testing 62](#_Toc209794135)

[15.11. Brain Dump 62](#_Toc209794136)

[16. Notifications 63](#_Toc209794137)

[16.1. Purpose 63](#_Toc209794138)

[16.2. Notification Types 63](#_Toc209794139)

[16.3. Delivery Channels 63](#_Toc209794140)

[16.4. Timing & Snooze 64](#_Toc209794141)

[16.5. Content & Tone 64](#_Toc209794142)

[16.6. Accessibility 64](#_Toc209794143)

[16.7. User Controls 64](#_Toc209794144)

[16.8. Technical Implementation 64](#_Toc209794145)

[16.9. Privacy 65](#_Toc209794146)

[16.10. Brain Dump 65](#_Toc209794147)

[17. Statistics 66](#_Toc209794148)

[17.1. Purpose 66](#_Toc209794149)

[17.2. General Principles 66](#_Toc209794150)

[17.3. Tasks 66](#_Toc209794151)

[17.4. Routines 66](#_Toc209794152)

[17.5. Habits 67](#_Toc209794153)

[17.6. Brain Dump 67](#_Toc209794154)

[17.7. XP & Gamification Stats 67](#_Toc209794155)

[17.8. Visualisation 67](#_Toc209794156)

[17.9. Data Export 67](#_Toc209794157)

[18. Gamification, Notifications & Feedback 69](#_Toc209794158)

[18.1. Purpose 69](#_Toc209794159)

[18.2. XP System 69](#_Toc209794160)

[18.3. Feedback & Rewards 69](#_Toc209794161)

[18.4. Streaks & Milestones 70](#_Toc209794162)

[18.5. Badges 70](#_Toc209794163)

[18.6. User Controls 71](#_Toc209794164)

[18.7. 15.6 Data & Export 71](#_Toc209794165)

[18.8. Brain Dump 71](#_Toc209794166)

[19. Annex — User Stories 72](#_Toc209794167)

[19.1. Purpose 72](#_Toc209794168)

[19.2. UX/UI & Responsiveness 72](#_Toc209794169)

[19.3. Visual Identity 72](#_Toc209794170)

[19.4. Accessibility & Inclusion 73](#_Toc209794171)

[19.5. Data & Storage 74](#_Toc209794172)

[19.6. US-DS-09 – Timezone handling and UTC persistence 76](#_Toc209794173)

[19.7. Import/Export 76](#_Toc209794174)

[19.8. Tasks 78](#_Toc209794175)

[19.9. Routines 79](#_Toc209794176)

[19.10. Habits 81](#_Toc209794177)

[19.11. Brain Dump & Brain Dumps 82](#_Toc209794178)

[19.12. Schedule 84](#_Toc209794179)

[19.13. Schedule 85](#_Toc209794180)

[19.14. Gamification 87](#_Toc209794181)

[19.15. Notifications 88](#_Toc209794182)

[19.16. Statistics 90](#_Toc209794183)

[19.17. Cross-Cutting Guidance 92](#_Toc209794184)

[19.18. Developer Utilities & Maintainability 92](#_Toc209794185)

[19.19. Library (Templates) 93](#_Toc209794186)

[19.20. Internationalisation 94](#_Toc209794187)

[19.21. Testing & Reliability 95](#_Toc209794188)

[20. Developer Utilities & Maintainability 97](#_Toc209794189)

[21. Security Test Plan — Detailed 98](#_Toc209794190)

[21.1. Security Architecture Verification Checklist 98](#_Toc209794191)

[21.2. Automated & Manual Solo-Friendly Tests 98](#_Toc209794192)

[21.3. Security Headers & CSP Verification 99](#_Toc209794193)

[21.4. Import/Export Hardening 99](#_Toc209794194)

[21.5. Service Worker & PWA Security 99](#_Toc209794195)

[21.6. Authentication, Sessions & Locking 99](#_Toc209794196)

[21.7. UX & Data Lifecycle 100](#_Toc209794197)

[21.8. Supply Chain & Dependencies 100](#_Toc209794198)

[21.9. Backups & Restore (Solo-Friendly) 100](#_Toc209794199)

[21.10. UI & A11y of Security-Critical Flows 100](#_Toc209794200)

[22. Developer Utilities 101](#_Toc209794201)

[23. Tab Appearance 102](#_Toc209794202)

[23.1. Tasks Tab 102](#_Toc209794203)

[23.2. 21.2 Routines Tab 107](#_Toc209794204)

[23.3. Habits Tab 111](#_Toc209794205)

[23.4. 21.4 Schedule Tab 115](#_Toc209794206)

[23.5. Brain Dump Tab 119](#_Toc209794207)

[23.6. Library Tab 123](#_Toc209794208)

[23.7. Settings Tab 127](#_Toc209794209)

[23.8. Navbar (Global) 131](#_Toc209794210)

[23.9. Popups & Dialogs 133](#_Toc209794211)

[23.10. Statistics (UI) 135](#_Toc209794212)

[23.11. Import/Export — UX Flows 138](#_Toc209794213)

[23.12. Security Settings — UX Flows 140](#_Toc209794214)

[23.13. Static Pages Appearance 143](#_Toc209794215)

# Metadata & Status

* Doc date: 24 Sep 2025 (Europe/Paris)
* Version: 0.9 (full, enriched)
* Versioning scheme: Semantic versioning (semver) for specs • 0.x = draft / in progress • 1.0 = first public release • 1.x = minor updates (non-breaking) • 2.0 = major revision (breaking changes)
* Owner: Ayanimea (sole author & maintainer)
* Source of truth: Master document (this spec), synced to Git repository
* Change policy:
  + • All modifications committed directly by the owner
  + Version number incremented according to semver
  + Changelog maintained in /docs/CHANGELOG.md of the repo
* Distribution:
  + Editable source: Git repository
  + Official exports (Word/PDF): /dist/specs/
  + The latest version is always tagged in Git
* Licensing:
  + Project code intended to be open source
  + Under MIT Licence

# Glossary

ABT: About (static page with application info)

ACC: Accessibility

AES-256: Advanced Encryption Standard, 256-bit key

ARC: Architecture

BDP: Brain Dump (markdown notes tab)

BCK: Back-end

CON: Conflicts

DCO: Display & Colour System

DEV: Developer Utilities

ENT: Entities

ERR: Error Handling

FAV: Favicon

FBK: Feedback

FSC: Functional Scope

GAM: Gamification

HAB: Habits

IEX: Import/Export

INT: Introduction

LIB: Library

NAV: Navbar

NFR: Non-Functional Requirements

NOT: Notifications

NTR: Non-telemetry (privacy-first principle)

OFF: Offline Mode

ONL: Online Mode

OPFS: Origin Private File System

PER: Performance / Progress indicators

POP: Popups & Dialogs

PRF: User Profiles

PWA: Progressive Web App

QHS: Quiet Hours

REL: Relationships

RTN: Routines

SCH: Schedule

SEC: Security

SET: Settings

SHA-256: Secure Hash Algorithm, 256-bit

STA: Statistics

STP: Static Pages

SUX: Security UX

TSK: Tasks

UIF: User Interface Principles

USR: User Environment & Profiles

UTL: Utilities

VAL: Validation

VSH: Version History

WIP: Wipe (delete all data)

XP: Experience Points

# Vision & Targets

* Audience: Neurodivergent users (e.g., ADHD), and everyone looking for structured routines.
* Outcome: Lower cognitive load via simple capture, prioritisation, and ritualised routines. Provide reliable tools for task management, routines, and daily note-taking as core needs.
* Principles: Calm UI, minimal friction, strong accessibility, offline-first, privacy-by-default, open-source ethos, security-first, high customisability, and gamification to keep the app engaging and enjoyable.
* Sustainable pacing: The app should encourage setting small, reasonable goals and support sustainable daily organisation.
* Scientific grounding: The app should rely on the current state-of-the-art in behavioural and organisational science to support users effectively. Features, when relevant, should be documented using meta-analyses.
* Scope: The app covers: Tasks (Eisenhower matrix + backlog), Routines (step runner), Habits (streaks + milestones), Schedule (calendar), Brain Dump (markdown), Library (templates), Settings, Statistics, Import/Export, and static pages (Privacy, Legal, About).
* Non-goals: Social feed, public community sharing (no community library).

# Architecture Overview (ARC)

## Application Architecture (ARC-APP)

ARC-APP-01 The application shall be implemented as a Progressive Web App (PWA).

ARC-APP-02 The application shall be built with ReactJS, HTML5, CSS/Bootstrap, and JavaScript.

ARC-APP-03 The application shall use modular components to ensure maintainability and scalability.

ARC-APP-04 The application shall be installable as a PWA on supported browsers.

## Data Management (ARC-DAT)

ARC-DAT-01 The application shall store structured data in IndexedDB.

ARC-DAT-02 The application shall store user files and attachments in OPFS.

ARC-DAT-03 The application shall provide local-first data persistence, with no external dependency.

ARC-DAT-04 Data aggregation (for statistics) shall be precomputed and stored to avoid runtime recalculation.

## Security Architecture (ARC-SEC)

ARC-SEC-01 The application shall not include telemetry, ads, or third-party analytics.

ARC-SEC-02 All exports shall remain under user control and not be transmitted externally.

ARC-SEC-03 Exported data shall support encrypted formats (.sj.enc) using AES-256.

ARC-SEC-04 All data wipe operations shall delete IndexedDB, OPFS, and localStorage content.

ARC-SEC-05 A passphrase shall be required to access encrypted exports.

## Functional Scope (ARC-FSC)

ARC-FSC- 01 The application shall provide tabs for Tasks, Routines, Habits, Schedule, Brain Dump, Library, and Settings.

ARC-FSC-02 The application shall provide a Statistics tab showing aggregated metrics and XP.

ARC-FSC-03 The application shall support import/export workflows with validation and conflict resolution.

ARC-FSC-04 The application shall provide Security Settings for passphrase, biometric unlock, wipe, and key export.

ARC-FSC-05 The application shall provide static informational pages: Privacy Policy, Legal Notice, and About.

## 2.4 Back-end (ARC-BCK)

ARC-BCK-01 The application shall function without requiring any back-end server.

ARC-BCK-02 If a back-end service is added, it shall be optional and used only for synchronisation or backup.

ARC-BCK-03 All server communications shall be secured with HTTPS and end-to-end encryption.

ARC-BCK-04 The back-end shall not store personal user content in plain text.

ARC-BCK-05 The back-end shall not include telemetry or analytics features.

## 2.5 Online Mode (ARC-ONL)

ARC-ONL-01 The application shall provide an optional online mode for data synchronisation across devices.

ARC-ONL-02 Synchronisation shall upload and download encrypted data only.

ARC-ONL-03 The user shall remain in control of enabling or disabling synchronisation at any time.

ARC-ONL-04 Synchronisation conflicts shall be resolved locally with user prompts.

ARC-ONL-05 The application shall continue to function offline even when synchronisation is unavailable.

# Non-Functional Requirements (NFR)

## Performance (NFR-PER)

NFR-PER-01 The application shall load the main interface in under 2 seconds on average hardware.

NFR-PER-02 Statistics aggregation shall complete within 200ms for datasets up to 10,000 entities.

NFR-PER-03 Imports and exports shall stream data to avoid UI blocking.

## Usability & Accessibility (NFR-ACC)

NFR-ACC-01 All interactive elements shall be operable via keyboard navigation.

NFR-ACC-02 The UI shall include ARIA roles and aria-labels for screen readers.

NFR-ACC-03 Reduced motion mode shall replace animations (confetti, shake) with subtle feedback.

NFR-ACC-04 Font scaling (80%–200%) shall be supported via a slider in Settings.

NFR-ACC-05 High contrast and simplified interface modes shall be available.

## Security (NFR-SEC)

NFR-SEC-01 All cryptographic operations shall use standard algorithms (AES-256, SHA-256).

NFR-SEC-02 Session auto-lock shall trigger after configurable inactivity.

NFR-SEC-03 Biometric unlock shall be supported where available.

NFR-SEC-04 All destructive actions (wipe, overwrite) shall require explicit confirmation.

# User Profiles & Environment (USR)

## Target Users (USR-PRF)

USR-PRF-01 The application shall be usable by individuals with ADHD and other neurodivergent profiles.

USR-PRF-02 The application shall provide forgiving flows (undo, skip) to reduce user anxiety.

USR-PRF-03 The application shall use positive feedback instead of punitive notifications.

## Supported Environments (USR-ENV)

USR-ENV-01 The application shall run on major browsers (Chrome, Firefox, Safari, Edge).

USR-ENV-02 The application shall support offline mode for all features.

USR-ENV-03 The application shall be installable as a PWA on desktop and mobile devices.

USR-ENV-04 The application shall adapt its UI for mobile, tablet, and desktop layouts.

# Information Architecture & Navigation

## Primary tabs (left to right in navbar)

Tasks → Routines → Habits → Schedule → Brain Dump → Library → Settings

Order reflects frequency of use and logical flow:

* Tasks for immediate actions
* Routines for structured sequences
* Habits for long-term streaks
* Schedule as the central overview (default landing page)
* Brain Dump for capture and reflection
* Library for reusable templates
* Settings for configuration

## Default landing

* Schedule tab is the home page because it offers a unified view of tasks, routines, and habits across time.
* This helps users orient themselves and reduces context-switching.

## Secondary navigation

* Import/Export flows live inside relevant tabs (Tasks, Brain Dump, Library).
* Global Import/Export (Markdown, JSON, SJ.ENC) is also available from the navbar overflow menu to back up or restore the entire workspace (settings, stats, notes, tasks, routines, etc.).
* About page provides credits, licensing, and open-source acknowledgments.
* Statistics are integrated within each tab but may later become a global section.

## Global navigation

* Navbar contains the primary tabs plus:
  + Search (global entity search by title, tag, or ID)
  + Theme switcher (light/dark/custom themes)
  + Profile/settings shortcut
* On small screens, navbar collapses into a hamburger menu with tab list.

## Cross-linking

* Cross-links are governed by the Link Contract:
  + Every entity has a typed ID (prefix + UUID).
  + Forward references stored in entity fields (e.g., tasks → relatedNotes).
  + Backlinks are derived automatically and never authored directly.
  + Deletions are soft for 30 days; unresolvedRefs capture missing targets.
  + UI resolves relationships dynamically, presenting chips and previews.
  + Accessibility: references use clear type icons + text, are keyboard navigable, and SR announces type and link.

## Accessibility (echo)

* Navigation is implemented as a proper tablist with aria-selected states.
* Focus management ensures keyboard users can cycle between tabs.
* Labels and icons are redundant so no cue relies only on color.

# Modules & Features

## Core modules

### Tasks

Purpose: capture and manage individual actions with optional due dates and categories.

Features:

Eisenhower 2x2 matrix classification (Urgent/Important).

Backlog for unclassified tasks.

Subtasks nested under parent tasks; subtasks may also appear in the matrix with a backlink to the parent, so they can be worked on independently.

Drag and drop between backlog and matrix quadrants.

Metadata: title, due date, category (user-defined 10-color picker), tags, projects.

Readability: in the Eisenhower matrix, only the task title is shown; details expand on click.

Cross-links: may reference related Brain Dump or other Tasks. Forward links only; backlinks derived.

XP logic: 1 XP for non-urgent/non-important, 2 XP for important, 3 XP for urgent+important. Confetti for important tasks, double confetti for urgent+important. Haptic feedback always. (Cross-ref Gamification §4.8 and UX §21.1)

Scheduling rules: Important+Urgent tasks are auto-surfaced into Schedule on their @date or today. Important (but not urgent) tasks trigger a reminder to schedule. All others require explicit scheduling.

Import/Export: tasks appear in JSON schema §13.1 and in Daily Note Markdown importer (§13.x).

### Routines

Purpose: guide the user through multi-step sequences with timers.

Features:

Ordered steps, each with a timer. Steps may be skipped, cancelled, or reordered.

Progress bar with golden accent. Current step enlarged; n-1 faded; n+1 shown smaller (UX §21.2).

Gamification: 1 XP per step; bonus XP if on time. Routine completion = large XP, badge, confetti, sound, haptic feedback.

Tags and versioning. Automatic semantic version increment on edits. Each routine has a stable UUID distinct from filename (Data & Storage §5).

Logs: each execution generates a RoutineLog (§13.3) for later stats (§21.10).

Import/Export: routines handled in JSON schema §13.2 and Markdown bundle export.

### Habits

Purpose: track recurring daily actions and streaks.

Features:

One checkmark per day; vacation mode to pause streak.

No penalty rule: users may configure parameters x (max consecutive days skipped) and y (sequence length). Defaults: x = 2, y = 7. Constraints: y ≤ 30, x ≥ 2, x ≥ y, x ≥ 1, y ≥ 1.

At end of a sequence, app reminds the user to take a break if not already in one.

Encourage easy, quick, not-too-demanding habits, based on behavioural activation therapy (TCC activation). Include references in documentation.

Encourage definition of the habit the day before starting to increase engagement.

Ring and heatmap visualization (UX §21.3).

XP: 1 XP daily, 3 XP for new streak, 2 XP for milestones.

Milestones: 7/14/28/50/100/250/500/1000 with confetti+haptic feedback.

Categories: user-defined colors (10-color palette, shared with Tasks and Brain Dump).

Undo period (5–10s) after marking complete.

Import/Export: habits represented in JSON schema §13.5; milestones and streaks included in stats export.

### Schedule

Purpose: provide a temporal overview of Tasks, Routines, and Habits.

Features:

Week and month calendar views (desktop), agenda list (mobile).

Events created only from existing entities (Tasks, Routines, Habits, Brain Dump).

Visual cues: border color = category, urgent = red accent.

Scheduling logic: propose next free slot in day; enforce 15 min break between events (user-configurable later).

Interaction: events can be resized by dragging the bottom edge of the event card.

Drag and drop reschedule; keyboard navigation; screen reader support.

Export: schedule events serialized via JSON schema §13.6.

Import: schedule can be restored from JSON or SJ.ENC bundles; Markdown Planning rows do not create Schedule events (they are imported as NoteLogs instead).

### Brain Dump

Purpose: free-form text capture for daily notes, references, and logs.

Features:

Editor + Preview split view (desktop); toggle on mobile (UX §21.5).

Theme override: user may choose a light/dark theme specifically for the editor and viewer, independently from the global theme.

Rich Markdown flavour supported (GitHub-flavored); no raw HTML rendering.

Rich toolbar: bold, italic, link, lists, checklists, quotes, code, tables, refs, timestamps, undo/redo.

Quick Brain Dump promotable to full Brain Dump.

Cross-links: ^ref(<id>) syntax creates forward links; backlinks derived automatically.

Version history with diffs. Brain Dump can be encrypted; textarea hashing applied.

Daily Note Markdown import supported (see §13.x). Planning tables parsed into NoteLogs; checklists become Tasks.

Export: full note text preserved in Markdown and JSON bundles.

### Library

Purpose: manage reusable templates for Tasks and Routines.

Features:

Create templates from existing entities. Independent duplicates; edits do not affect originals.

Task templates: title, category, quadrant, offsets, tags, notes.

Routine templates: steps, timers, tags, XP.

Semantic versioning optional.

Search and filter by type, tag, duration, last used.

Templates exported via JSON schema §13.6 and Markdown bundle mapping.

### Settings

Purpose: control languages, security, accessibility, habits, data storage.

Features:

Languages: app language configurable; documentation always English.

Security: passphrase, biometric, auto-lock, wipe, export key (UX §21.12).

Accessibility: reduced motion, font scaling, color palettes, sounds/haptics toggles.

Habits: configure vacation days, milestones, reminders.

Data Storage: configure auto-cleanup (default 1 month), per-item overrides, encryption, autosave frequency and target (see §5 and §8.5).

Export: settings included in global export bundles.

## Supporting modules

### Statistics

Purpose: help users adjust future planning by comparing expected vs. actual durations.

Focus:

* Steps, routines, and repeatable tasks: mean duration vs expected duration.
* Highlight mismatches and suggest adjustments for more realistic future planning.

Other measures:

* Task throughput, lead time, breakdown by quadrant and category.
* Habit streaks and milestones.

Export: included in AggregatedStats schema (§13.4). Displayed in §21.10.

### Security

Applies across all modules.

See §9 Security & Privacy for detailed specifications.

### Gamification

XP system:

* 1 XP = baseline unit
  + 1 routine step = 1 XP
  + 1 non-important/non-urgent task = 1 XP
  + 1 habit daily tick = 1 XP
  + Important task = 2 XP
  + Urgent + important task = 3 XP

Habits

* New streak = +3 XP
* Milestone = +2 XP

Routines

* Bonus XP for on-time step completion
* Large XP for routine completion (extra bonus if all steps on time)

### Import / Export

Scope

* Global by default: exports and imports the entire workspace in one operation.
* Included: Settings, Accessibility and theme preferences, Data Storage preferences, Security preferences (without passphrases or raw keys), Tasks, Routines, Habits, Brain Dump (full contents), ScheduleEvents, Templates, Statistics (aggregates and logs), Link Contract references and derived indexes (backlinks recomputed on import).

Export formats

* JSON (lossless canonical form)
* SJ.ENC (encrypted bundle of JSON; recommended default for autosave)
* Markdown Bundle (human-readable, Git-friendly)
* Single Markdown file with front matter and fenced entity blocks
* Brain Dump captured verbatim as Markdown
* Other entities embedded as canonical JSON in code fences

Import formats

* JSON, SJ.ENC, Markdown Bundle

Validation:

* Typed IDs must match schemas in §13
* Reference types must be correct; missing targets become unresolvedRefs
* Backlinks are recomputed after import

Autosave to predefined file (no data loss)

* Goal: Ensure users do not lose data when closing the browser.

Mechanism:

* Primary: Origin Private File System (OPFS) persistent storage for fast, automatic snapshots
* Optional: File System Access API “Save to this file” workflow; user grants a file handle once, then the app writes incremental snapshots to the same file without re-prompting
* Frequency: Default every 60 seconds and on significant change events (create/update/delete, import)
* Format: SJ.ENC by default (encrypted with user passphrase); user can choose JSON or Markdown Bundle explicitly

Integrity:

* Each snapshot records exportTimestamp, specVersion, entityCounts, and a SHA-256 checksum of the canonical JSON
* On next load, if local DB is empty or newer snapshot exists in OPFS or chosen file, user is prompted to restore

Local persistence guarantees

* IndexedDB as durable local DB with “persistent” storage if available
* Background sync ensures pending writes are flushed before service worker sleeps
* beforeunload / visibilitychange triggers final local snapshot if safe
* PWA install recommended to increase storage quota and reliability

Git and remote workflows

* Users can export the full bundle (Markdown or JSON) and commit to GitHub
* Re-import from the website (or PWA) restores the full workspace
* SJ.ENC can be stored in private repos for privacy

Security and privacy

* SJ.ENC encrypts the canonical JSON bundle with a passphrase
* Passphrases and derived keys are never exported
* Markdown Bundle is plaintext; the UI warns before exporting unencrypted data
* Access to a file handle is revocable at any time from browser permissions

Conflict policy

* On import, entity identity is the typed ID
* Defaults: keep local if newer; user can choose replace or merge

Merge strategy:

* Simple fields (title, tags) use last-updated
* Arrays append when safe and deduplicate by typed ID or key
* Brain Dump prefer latestUpdated unless user chooses otherwise per note

Performance

* Exports use deterministic ordering for stable diffs
* Large notes and logs are chunked internally but exported as continuous text/JSON
* Progress UI for long exports/imports with cancel

# Data & Storage

Purpose Define how user data is stored, persisted, cleaned, exported, and imported, with strong guarantees of privacy, reversibility, and integrity.

Local storage Default: IndexedDB / OPFS, encrypted if passphrase set. Autosave snapshots: periodic, into OPFS or user-chosen file. On browser close: prompt to download/export data if unsaved changes exist. Cleanup: default auto-purge after 1 month, user can override globally or per item, or disable entirely. Manual cleanup: user can delete stale or specific items.

Data minimization No personal data collected server-side. App fully functional offline; all storage client-side. No third-party analytics/tracking.

Export formats Supported: .md, .html, .json, .sj.enc (encrypted). File naming: ISO 8601 UTC timestamp + slug, e.g. 2025-09-19T103211Z--brain-dump.md. User can add a chosen title/slug. Integrity: optional .sha256 file generated alongside export. Exports must be re-importable without transformation or corruption.

Import formats .md (preferred) and .html supported. Markdown: CommonMark + GFM (task lists, tables, code fences, links/images). HTML disabled at render. Front-matter YAML required on export; optional on import (auto-generated if missing). HTML import: strict sanitization (remove , on\*, style, iframe, object, embed, dangerous URLs). On import: normalize encoding to UTF-8, line endings to LF, dates to ISO 8601 UTC.

Brain dumps Special case of Brain Dump for ideation. Stored in /brain-dumps/ folder. File format: .md with YAML front-matter. Required fields: id (UUID), title, created\_at, modified\_at, app\_version, language. Optional fields: tags, energy\_level, context. Logs stored separately in /routines/logs/. Example front-matter:

id: "bd\_7f1b9b8a-9a3f-4b2d-8b4c-5c3a1a7b9f21" title: "Brain dump after product meeting" created\_at: "2025-09-19T10:32:11Z" modified\_at: "2025-09-19T10:32:11Z" app\_version: "sj-0.9.0" tags: ["brain-dump", "ideation", "ADHD-fog"] energy\_level: "low" language: "en-GB"

Statistics storage Aggregated statistics stored in separate file (/exports/stats.json) to avoid recomputation and limit exposure.

Security rules CSP enforced: no inline scripts/styles; no unsafe-eval; all external deps with SRI or self-hosted. Strict separation of modules: no logic in HTML, no inline JS/CSS. README Security document delivered with CSP policies, dependency hashes, and update policies.

UX requirements Export/Import buttons consistent across tabs (label, icon, hover/focus states). Keyboard shortcut: Ctrl/Cmd+S triggers export to .md. Live Markdown preview (safe renderer). Accessibility: labelled textarea, screen reader support, aria-live for autosave.

# UX/UI & Accessibility

Purpose

Define the user interface and experience principles for Stellar-Journey, ensuring a consistent stellar visual identity, responsive layouts, and full accessibility (WCAG 2.2 AA).

## Visual Style

Color Palette

* Base Background
  + Gradient from deep indigo (#2B1D47) to violet (#3E285C), applied to app background
* Primary Text
  + Ink (#222, --ink) for body text
  + Dim (#888, --dim) for secondary text
* Accent
  + Mint (#A3FFD6, --mint) used for highlights, buttons, confirmations
* Extended Palette (for states and alerts)
  + Celestial Coral (#FF7780, --celestial-coral) → alerts, destructive actions
  + Lunar Halo (#F5F3FF, --lunar-halo) → soft light backgrounds and contrast text
  + Stellar Indigo (#3D3A8E, --stellar-indigo) and Misty Lavender (#C5B7F3, --misty-lavender) available as theme accents

Typography

* Font family: Inter, system-ui, sans-serif
* Headings weight 600, body weight 400
* Responsive scaling: clamp(16px, 1.2vw, 18px) for body, larger clamps for headings
* Line spacing generous for legibility

Effects

* Mockup style chosen: glassmorphism with subtle blur and shadows
* Cards: transparent with backdrop blur (16px) and rounded corners (0.5rem radius)
* Appbar/Navbar: solid dark background with subtle blur
* Minimal glow effects, limited to hover/interaction emphasis

Global Progress Bar

* A fine golden line appears as a border directly below the navbar
* Used for global progress feedback (routine advancement, load state)
* Always visible when active, animated smoothly, WCAG-compliant contrast

## Accessibility (WCAG 2.2 AA)

General

* WCAG 2.2 AA compliance by default, AAA applied to critical text (alerts, destructive actions, security warnings)
* Modes: light, dark, high contrast (pure black background, pure white text, reinforced borders)
* Font scaling from 85% to 150%
* Generous line spacing for legibility

Visual

* No element coded by color alone. Urgency uses color + icon + text
* Consistent markers across the app: same icon and color scheme for identical actions
* Focus and error outlines are thick and high-contrast

Cognitive & Neurodiversity

* Minimalist UI, sections clearly separated (Tasks, Routines, Habits, Brain Dump)
* Progressive disclosure: show essentials first, advanced options via “see more”
* Simplified Mode toggle in Settings hides advanced statistics and trends
* Microcopy: clear, short, friendly; cosmic metaphors remain simple

Motor

* Full keyboard navigation (Tab, Enter, Space, Arrows for reordering)
* Touch targets at least 48px on mobile
* Drag-and-drop always has an alternative (buttons for reorder, menus for actions)
* Focus is always visible, never removed

Auditory & Multimodal

* Feedback available in visual, sound, and haptic modes (configurable separately)
* Vibration optional, respects prefers-reduced-motion
* Sounds and animations always have captions or text equivalents

Technical

* ARIA roles/labels on all buttons, menus, nav
* HTML5 landmarks (<header>, <main>, <nav>, <footer>) throughout
* Forms always have bound <label> tags
* Prefers-reduced-motion and prefers-contrast respected
* QA includes NVDA and VoiceOver testing

## Navigation & Responsive Layout

Primary Tabs: Tasks, Routines, Habits, Schedule, Brain Dump, Library, Settings

Default Landing Page: Schedule

Navbar

* Desktop: horizontal with icons and labels
* Mobile: collapsible bottom nav bar with icons
* Tablet: docked side menu in landscape, bottom nav in portrait

Global Items: search, theme switcher, account

Selected Tab: highlighted background (mockup style chosen, no underline)

Focus Effect: thick outline with subtle glow

Golden Progress Bar: integrated below navbar, remains visible during navigation

Responsive Breakpoints

* Mobile < 640px
* Tablet 640px – 1024px
* Desktop 1024px – 1280px
* Large 1280px+

Split View Layouts

* Tasks and Routines: list on left, details or matrix on right (tablet/desktop)
* Schedule: agenda list on mobile, week view on desktop, month view on large screens

Popups & Dialogs

* Full accessibility, aria-modal, focus trap, ESC closes dialog

## Interactive Elements

Buttons

* Variants: default, outline, secondary, ghost, link
* States: normal, hover, focus, active, disabled, loading, success
* Hover: translateY(-1px) with shadow enhancement
* Focus: 3px outline in Stellar Indigo
* Touch targets: min 44px, ideally 48–80px

Gestures

* Tap: complete tasks, check steps
* Long press (≥800ms): skip step with reason
* Drag & drop: reorder tasks, routines, events
* Visual feedback: animation + vibration (mobile)

Keyboard Equivalents

* Arrows move selection and reorder items
* Enter/Space trigger main actions
* Shortcuts for import/export, save, and new item creation

Feedback

* Loading: spinner
* Success: Astral Mint highlight
* Error: Celestial Coral highlight + icon
* Disabled: dimmed with reduced opacity

## Animations

* Current status: to be decided
* Spec defines fade in, slide up, page transitions, hover scale/glow
* Mockup currently implements only hover transitions (0.2s ease)
* Final animation catalogue and timing rules TBD

## Brand Voice & Tone

Current status: to be refined

Tagline: “Explore your orbit. Follow your path.”

Style: friendly, comforting, encouraging

Cosmic metaphors allowed but must remain simple and accessible

Example microcopy to be added later

## Design System & Tokens

Grid & Spacing

* Base spacing unit: 8px grid
* Margins: 16px mobile, 24px tablet, 32px desktop
* Card padding: 16px

Components

* Navbar, cards, buttons, dialogs, forms standardized
* Tokens for colors, typography, shadows, borders defined as CSS variables

Breakpoints

* Mobile, Tablet, Desktop, Large (see 6.3)

Documentation

* All tokens and component specs included in design system library

# Visual Identity (brand kit)

Purpose

Define the brand identity of Stellar-Journey, ensuring consistency across product UI, documentation, and communication.

## Brand Concept

Stellar-Journey conveys calm exploration and structured guidance.

Core metaphors: space, orbits, constellations, navigation.

Tone: encouraging, structured, not overwhelming.

## Logo & Iconography

Logo: minimalistic planet or orbit motif, circle with arc lines.

Icon set: simple line icons with rounded ends, consistent stroke weight.

Favicons: generated from logo in multiple sizes.

App icons: gradient background, central planet/orbit mark.

## Color System

Base background: gradient from deep navy (#0F172A) to slate gray (#1E293B).

Text colors: Ink (#222), Dim (#888).

Accent: Mint (#A3FFD6).

Extended palette tokens:

Celestial Coral (#FF7780) for alerts and destructive actions.

Lunar Halo (#F5F3FF) for contrast text and soft backgrounds.

Stellar Indigo (#3D3A8E) and Misty Lavender (#C5B7F3) as optional accents.

All colors defined as CSS variables, referenced in design tokens (see §6.7).

Contrast ratios must meet WCAG 2.2 AA.

## Typography

Font family: Inter, system-ui, sans-serif.

Headings: weight 600.

Body: weight 400.

Scaling responsive with clamp, body text baseline 16–18px.

Line height: 1.5–1.6.

## Imagery & Backgrounds

Background includes blue gradient sky (#0F172A → #1E293B) with stars (CSS generated).

Stars: irregular size, color, and spacing pattern for realism.

Optional planet rendered subtly in the background for thematic depth.

No photographic imagery used; all backgrounds are abstract and generated.

## Effects & Surfaces

Cards: glass effect with 16px blur, semi-transparent, rounded corners 0.5rem.

Navbar: solid dark with blur, fine golden border progress bar at bottom.

Shadows subtle; glows limited to hover emphasis.

## Iconography & States

Action icons: play, pause, skip, reorder, delete, consistent across app.

State icons always accompanied by text (no color-only coding).

Error states highlighted with Celestial Coral + warning icon.

Success states highlighted with Astral Mint + check icon.

## Brand Voice & Copy Guidelines

Voice: friendly, calm, encouraging.

Language: clear and short, avoids jargon.

Cosmic metaphors used sparingly, always easy to understand.

Example confirmation: “Routine complete! 🌟 Great work on your journey.”

Example guidance: “Choose small, steady habits — one orbit at a time.”

Tagline: “Explore your orbit. Follow your path.”

## Theming & Extensibility

Users can switch themes (light, dark, high contrast).

Theme changes do not override brand consistency: fonts and structure remain.

Developers may add new themes by extending the design tokens.

All themes must respect WCAG contrast requirements.

## Documentation & Assets

All assets (logo, icons, color tokens, typography) stored in design system library.

README Brand file to explain tokens, usage, and rules.

Figma components library includes logo, icons, backgrounds, card styles, button variants.

# Settings

Purpose

Centralize user preferences and configuration options, balancing flexibility with simplicity.

Settings must remain clear, well-organized, and accessible.

## General Structure

Settings are organized by category: General, Security, Accessibility, Habits, Data & Storage, Notifications, About.

Presented as a tabbed or accordion interface with clear headings.

All settings changes apply immediately and autosave.

## General

Language selection: English (default) and French available.

Theme selection: light, dark, high contrast.

Time zone and date/time format (24h vs 12h).

Default landing page (Tasks, Schedule, etc.).

Font size preference (small 85%, default 100%, large 115%, extra large 130%, maximum 150%).

## Security

Passphrase setup/change/reset.

Biometric unlock toggle (if device supports).

Auto-lock timeout (1, 5, 15, 30 minutes).

Export encryption key (with warnings).

Enable/disable encryption for exports (.sj.enc vs plain .md/.json).

Wipe local data option (requires confirmation).

## Accessibility

Toggle sounds, haptic feedback, and animations.

Reduced Motion Mode.

High Contrast Mode.

Simplified Mode (hides advanced statistics and trends).

Drag-and-drop alternatives enabled (arrow buttons).

Screen reader hints toggle (extended labels on controls).

## Habits

Default streak milestones (7, 14, 28, 50, 100, 250, 500, 1000).

Vacation mode configuration (pause streaks).

Sequence break reminder toggle.

Default colors for new habits.

Habit logging reminders (on/off, time of day).

## Data & Storage

Autosave frequency (default: 1 minute).

Automatic cleanup toggle (default: 1 month).

Per-item cleanup exceptions.

Export path (choose default folder or OPFS).

Toggle prompt on close if unsaved changes.

Import/export preferences (default format: .sj.enc, .md, .json).

Option to enable SHA256 integrity files for exports.

## Notifications

Daily reminder time (default: 8 AM).

Reminder types: tasks due, habits check-in, routine start, streak pause.

Snooze duration options (15 min, 1 hour, until tomorrow).

Quiet hours (disable notifications at night).

Push notifications enable/disable (if PWA installed).

## About

Version number of app and data schema.

Link to documentation.

Link to repository (open source under MIT license).

Acknowledgments and contributors.

Contact/support information.

## UX Considerations

All settings accessible via the Settings tab in navbar.

Search bar for quickly locating settings.

Reset-to-default option per category.

Confirmation prompts for destructive changes (wipe data, disable passphrase).

Settings tab fully accessible: keyboard navigation, ARIA roles, proper labels.

# Security & Privacy

## Purpose

Protect user data and keep the app safe, while staying realistic for a solo-maintained project.

Focus on privacy-by-design and simple, maintainable security measures.

## Data Protection

Local data stored in IndexedDB/OPFS.

Optional passphrase-based encryption; if not enabled, data is plain but still local only.

Encrypted exports use .sj.enc format; plaintext .md/.json allowed but require explicit confirmation.

Autosave snapshots respect encryption setting.

## Authentication

No cloud authentication required for offline mode.

If backend is used (Rails API), rely on JWT with short expiry and refresh tokens.

TLS (HTTPS) required for all server communication.

## Import/Export Integrity

Export files must be re-importable without transformation.

Optional SHA-256 checksum file created at export.

On import, unknown fields preserved (no silent data loss).

## Frontend Security

Strict Content Security Policy (CSP):

No inline JS or CSS.

Scripts/styles must be self-hosted or loaded with SRI hashes.

Serve only over HTTPS.

Service Worker limited to caching and offline; no unsafe eval/dynamic injection.

No third-party analytics or trackers.

## User Safety UX

Critical operations (wipe data, disable passphrase) require confirmation.

Warning shown before exporting unencrypted data.

Auto-lock option after inactivity (user-defined delay).

All dialogs accessible: focus trap, aria-modal, screen reader labels.

## Privacy

No personal data collected server-side.

No hidden tracking or telemetry.

All export/import under explicit user control.

App usable fully offline without backend.

## Security Architecture (Local + Online)

Purpose

Protect user content while allowing a local-first workflow and optional online sync. Keep controls simple, auditable, and realistic for a solo project.

Threat Model (summary)

Device loss or shoulder surfing → mitigate with passphrase + auto-lock.

XSS via content/imports → mitigate with strict CSP + sanitisation + HTML-disabled Markdown.

Network attacker → mitigate with HTTPS/HSTS and short-lived auth tokens.

Server breach → minimise blast radius via client-side encryption or field-level encryption.

Supply-chain risks → pin dependencies and audit updates.

Local (Offline) Data

Storage: IndexedDB / OPFS.

Encryption (recommended): AES-GCM per record/file; 96-bit random IV stored alongside ciphertext.

Key derivation: PBKDF2 (WebCrypto) from user passphrase; non-extractable CryptoKey; kept only in memory; wiped on lock/close.

Key recovery: optional export of a wrapped data key (“Recovery Key”) protected by a second passphrase.

Data separation: keep only minimal non-sensitive metadata in clear (e.g., type, created\_at). Encrypt titles, descriptions, tags where feasible.

Sanitisation: ALL imports (Markdown, HTML, JSON) are sanitised; Markdown renderer has raw HTML disabled; block javascript: URLs and on\* handlers.

CSP/Hardening: strict CSP (no inline/eval), scripts/styles self-hosted or SRI; no third-party analytics; split JS/CSS from HTML.

Online (Rails Backend) Data

Modes:

A) Zero-knowledge sync: client encrypts before upload; server stores ciphertext + minimal metadata.

B) Minimized plaintext: minimal public metadata (e.g., type, timestamps) kept for indexing; sensitive fields encrypted.

Rails defaults:

Auth: short-lived JWT access + refresh; HTTPS required; SameSite=Lax if cookies used.

Storage: PostgreSQL; Active Record Encryption for server-processed fields (if not zero-knowledge).

Authorisation: per-row scoping by user\_id; multi-tenant guards at model/service layer.

Validation: strict JSON schema on import/sync endpoints.

Logs: no personal content; avoid sequential IDs in URLs; prefer UUIDs.

Backups: encrypted at rest; keys managed separately; restore drills documented.

Tags & Search

Local-first search over decrypted content.

If server-side tag search is needed without revealing plaintext tags, store a blind index (HMAC(tag)) in a separate table; plaintext tag remains in encrypted payload.

Retention & Deletion

Auto-cleanup: only for logs/snapshots/tmp; NEVER for backlog items. User can disable or override per item.

User controls: Delete all local data; Delete account + remote data (if online enabled).

Export/Import: global export (encrypted by default) plus optional .sha256 integrity file; import must round-trip without loss.

User Experience for Privacy

Settings toggles: Enable encryption (passphrase), Auto-lock delay (1/5/15/30 min), Export encrypted by default, Delete all local data.

Warnings: explicit confirmation when exporting unencrypted; confirmation when removing passphrase.

Transparency pages: “Where your data lives” (device vs. optional cloud) and “How to back up & recover keys”.

Operational Practices (Solo-Friendly)

Dependency updates: review changelogs; pin versions; run npm audit (production) and Ruby bundle audits on update.

Secrets: never embed server secrets in frontend; use environment vars on server; rotate keys on suspicion.

Error handling: do not leak stack traces or payloads to client or logs in production.

## Auditing and Testing

As a solo developer:

Use browser DevTools security panel to check CSP and TLS.

Run Lighthouse audits locally for security best practices.

Periodically review dependencies (npm audit, bundle audit).

No penetration testing team; issues documented openly in repo if found.

## Brain Dump

The goal is to keep security simple, local-first, and transparent.

Stronger measures (e.g., biometric unlock, advanced key management) may be added later, but are not required at first release.

# Performance & PWA

## Purpose

Ensure Stellar-Journey feels fast, reliable, and usable offline, while staying realistic for a solo-maintained project.

## Core Performance Targets

First load on 4G mobile: usable navbar in < 2 s, interactive in < 3 s

Repeat load (warm cache): < 1 s to interactive

Task interactions (tap/click): response within 100–200 ms

Drag & drop and scrolling: aim for 60 fps on modern devices

## Bundle and Asset Budgets

Initial JS (critical path, post-compression): ≤ 150 KB

Total JS on first route: ≤ 300 KB

CSS critical path: ≤ 40 KB

Fonts: Inter + system stack only; 2 weights (400, 600)

Images: use SVG or lightweight raster, ≤ 150 KB on landing

## Build and Delivery

Minify and tree-shake all JS/CSS

Use modern JS bundles (ES2020+) with fallback transpile only if needed

Serve over HTTPS with TLS 1.3

Use hashed filenames for static assets

Gzip or Brotli compression enabled

## Service Worker (SW)

Provide app shell caching (HTML, core JS/CSS)

Cache static assets with versioning

Network-first for API reads, cache update in background

Offline fallback page for when no network

Background Sync: queue writes if offline, replay on reconnect

Updates: SW checks silently, prompt reload when a new version is ready

## Offline-first Data

Store entities in IndexedDB/OPFS

Autosave snapshots locally (see §5)

All import/export flows work offline

Sync to backend (Rails) when online again

## PWA Installability

Manifest with app name, icons, theme\_color, background\_color

Icons: 192, 256, 384, 512 px (maskable)

Display mode: standalone

Meets Lighthouse PWA checks ≥ 90 (run locally in Chrome)

## Resource and Battery Friendliness

Respect Save-Data setting (skip non-essential requests)

Disable non-essential animations in Reduced Motion Mode

Keep long main-thread tasks under 50 ms where possible

## Monitoring and Diagnostics

No third-party analytics

Use Lighthouse locally for performance and PWA checks

Use browser DevTools to profile JS bundles and long tasks

Optional: WebPageTest or PageSpeed Insights for external check

Document known issues and optimizations in README

# Packaging & Deployment

## Purpose

Define how Stellar-Journey is packaged, distributed, and deployed, keeping the process simple and reproducible for a solo-maintained project.

## Packaging Targets

Deliverables

* Web app (hosted version with Rails backend if used).
* PWA installable package (manifest + icons).
* Android .apk (generated via wrapper, e.g., Capacitor or TWA).
* Desktop optional (Electron or Tauri, future consideration).

File formats

* Source code in GitHub repo (MIT license).
* Exports/imports in .sj.enc, .md, .json formats.

## Deployment Environments

Local

* Run via Docker with Nginx reverse proxy for testing.
* Rails backend launched locally with PostgreSQL.

Production

* Option 1: Self-hosted VPS with Docker + Nginx reverse proxy.
* Option 2: Free/low-cost hosting (Render, Fly.io, or GitHub Pages for frontend-only).
* TLS certificates via Let’s Encrypt.

Offline

App usable fully offline as PWA without backend.

## Build Process

Frontend

* No compilation required for HTML/CSS/JS/Bootstrap core.
* Optional ReactJS build step (if JSX used).
* Bundled with minification and tree-shaking.

Backend (if enabled)

* Rails app containerized with Dockerfile.
* Database migrations managed via Rails CLI.

Scripts

* Simple makefile or npm/yarn scripts for build, lint, test, deploy.

## Deployment Workflow

Development

* Commit to GitHub main branch.
* Run local build + tests.

Staging

* Optional: deploy to a test VPS or Netlify preview.

Production

* GitHub Actions workflow can deploy automatically on tagged release.
* Backup old release for rollback.

## Versioning & Releases

Semantic Versioning (MAJOR.MINOR.PATCH).

GitHub Releases page stores release notes and packaged files.

Each release includes:

* Source code (zip/tar.gz).
* Built .apk (Android).
* Manifest + PWA assets.

Changelog maintained in repo.

## Security Considerations

HTTPS enforced on all deployments.

CSP headers included in Nginx config.

Auto-update PWA via Service Worker (silent update).

No secrets stored in frontend code; config via env vars on backend.

Backups: database dump before major updates.

## Documentation

README includes:

* Build instructions (local + Docker).
* Deployment instructions for VPS and GitHub Pages.
* Brain Dump on offline/PWA installation.

SECURITY.md includes:

* CSP policy summary.
* How to report vulnerabilities.

DEPLOYMENT.md includes:

* Step-by-step VPS setup.
* Nginx reverse proxy config sample.

# Quality & Testing

## Purpose

Ensure Stellar-Journey is reliable, accessible, secure by default, and remains stable as features evolve — with a lightweight, solo-maintainable process. Firefox is the primary test target. The site must remain usable with JavaScript disabled and with cookies refused.

## Test Strategy (Overview)

Browsers

Primary: Firefox (latest ESR and latest stable)

Secondary smoke: Chromium-based (Edge/Chrome) and Safari Technology Preview when available

Test Pyramid

Unit tests → Integration tests → UI/E2E tests → Manual smoke checks

Automation

Run locally and in CI (GitHub Actions). All suites green before release.

## Environments & Modes Under Test

Online mode (Rails backend reachable, HTTPS)

Offline mode (PWA installed; Service Worker; background sync)

Degraded mode A: JavaScript disabled

Degraded mode B: Cookies refused/blocked

Accessibility modes: Reduced Motion, High Contrast, large fonts

Localisation: English, French

Devices

* High-end Android smartphone (~4 years old flagship model)
* Lower-end Android smartphone (recent mid/low range)
* Old iPad (pre-2019 model, Safari)
* Browser simulators/emulators for additional form factors

## Test Suites & Tools

Linting & Static Analysis

ESLint for JavaScript/TypeScript (rules: no eval, no inline events, accessibility plugin)

Stylelint for CSS (no !important defaults; focus-visible styles present)

YAML/JSON schema validation for locales and export metadata

Unit Tests (logic and utilities)

Runner: Vitest or Jest

Coverage targets: 80% lines / 80% branches for core logic (XP rules, Eisenhower classification, time math, streak logic)

Include pure functions for import/export transforms and ID generation

Component / Integration Tests (UI logic without network)

Framework: Testing Library (DOM) with Vitest/Jest

Focus on: forms, dialogs focus trap, keyboard navigation, state transitions, drag/drop adapters

Mock IndexedDB/OPFS; verify read/write paths and reversibility of exports

End-to-End (E2E) UI Tests (real browser)

Playwright with Firefox engine

Core flows: Tasks (create/classify/drag to matrix), Routines (step timers, snooze/skip), Habits (tick, vacation), Schedule (add from existing items), Brain Dump (import/export .md), Import/Export global bundle

Validate offline: switch to offline mode mid-flow; ensure queue and replay on reconnect

Validate PWA install prompt and SW update prompt

Accessibility Automation

axe-core automated checks on key routes and dialogs

Keyboard-only navigation path for each tab (Tab/Shift-Tab/Enter/Space/Arrows)

Verify focus outline visibility and skip-to-content if applicable

Visual Regression (lightweight)

Playwright screenshot snapshots (Firefox) for key screens (light/dark/high contrast)

Tolerances kept small; update snapshots only on intentional UI changes

Performance Spot Checks (browser-agnostic)

Use Firefox DevTools Performance/Network panels

Record first load and repeat load timings, JS bundle sizes, SW cache hits

Targets: see §10 (solo version)

## Degraded Mode Tests (Must Pass)

JavaScript Disabled

* App shell renders basic content (branding, navigation links)
* Brain Dump: viewing and downloading previously exported files remains possible from a static “Import/Help” page
* Clear message explains limited functionality without JS and how to enable full features offline/PWA

Cookies Refused/Blocked

* App runs without cookies (IndexedDB/OPFS and localStorage still functional)
* No cookie wall; no broken UI; preferences persist in local storage
* If backend is used, session/auth falls back to token storage (no cookie dependency)

Both Modes

* No infinite spinners; graceful fallbacks and help links are visible

## Test Data & Fixtures

Seed datasets for: tasks (all Eisenhower quadrants), routines with 3–7 steps and timers, habits with existing streaks (near milestones), notes (GFM tables/code, YAML front-matter), library templates

Edge cases: very long titles, RTL snippet (for future i18n), emoji, large notes (near size limits), missing front-matter (auto-generate)

Corrupt import samples for negative tests (malformed YAML, HTML with scripts)

## Security & Privacy Checks (Lightweight)

CSP verification (no inline JS/CSS; no unsafe-eval). Check response headers via curl or devtools Network

Dependency audits: npm audit (—production); bundler-audit or brakeman for Rails (if backend used)

Import sanitization tests: HTML stripped of scripts/handlers/iframes; Markdown renderer with HTML disabled

No third-party analytics loaded; network panel confirms zero calls to trackers

Mixed content: all requests HTTPS; service worker scope restricted; cache poisoning prevented by versioned assets

## Internationalisation Tests

English ↔ French toggle persists across reloads

Layout remains usable (no overflows/truncation) in both languages

Date/time/number formats adapt via Intl API; 24h/12h preference respected

## Accessibility (Descriptive + Checks)

Screen reader labels on icons and controls; role=tablist with aria-selected on tabs

Dialogs: aria-modal, labelledby, focus trap; ESC closes when appropriate

Reduced Motion and High Contrast modes render critical affordances without transparency loss

No color-only signalling; urgent/important uses color + icon + text

Manual passes at least on NVDA (Windows) and VoiceOver (macOS/iOS) before release

## Release Smoke Checklist (Manual, 10–15 minutes)

Start app fresh (cache cleared), JS enabled → navigate all tabs; create one Task, Routine (with 2 steps), Habit, Note; export and re-import bundle (round-trip OK)

Toggle to offline → complete a task and a routine step → reconnect → verify sync

Notifications permission flow (don’t force); schedule a reminder; receive one

Install PWA; close and reopen from icon; verify golden progress bar during a routine

Switch themes (light/dark/high contrast) and language (EN/FR); check focus rings

Disable JS and reload → verify graceful fallback page and no cookie/cookie-banner blockers

Refuse cookies → verify all features still work that do not need backend cookies

## CI (GitHub Actions) — Solo Friendly

Jobs

lint: ESLint + Stylelint + schema validations

unit: Vitest/Jest

integration: Testing Library

e2e-firefox: Playwright (headless)

a11y: axe-core on key routes

Triggers: on pull request and on main branch push

Artifacts: Playwright traces/screenshots on failure; exported test bundles for debugging

Fail Conditions: any test suite failure; bundle size budgets exceeded (see §10.2)

## Definition of Done (DoD)

All linters pass; no severe warnings

Unit/integration coverage ≥ 80% for core logic modules

E2E happy paths green on Firefox (online/offline)

axe-core reports no critical violations; keyboard-only navigation verified

Import/export round-trip proven on current schema

CSP headers present; no inline/eval violations detected

JS-disabled and cookies-refused fallbacks render without blockers

Changelog updated; version bumped; release notes drafted

## GDPR / RGPD Compliance Checks

Principles

* Privacy by design and by default: no unnecessary data collected.
* All data stays client-side unless user explicitly exports or enables backend sync.
* No third-party analytics, no trackers, no ads.
* Explicit consent required if any online feature requires personal data (e.g., backend account).
* User can delete all local data and exports at any time.

Test Procedures

* Install and run the packaged app (PWA, APK from store).
* Verify that on first launch, no network requests are made except to required APIs.
* Inspect network traffic (Firefox DevTools, mitmproxy) → confirm no analytics or trackers.
* Confirm all settings default to privacy-preserving (no telemetry, no auto-upload).
* Run app with cookies blocked, ensure it still functions offline and exports/imports correctly.
* Confirm "Delete all data" wipes local IndexedDB/OPFS and cached Service Worker data.
* Export file → verify it contains only user data, no hidden identifiers.
* Import/export round-trip → no new identifiers injected without user action.
* For backend-connected mode: confirm TLS enforced, tokens short-lived, no personal identifiers logged.

Documentation

Include a PRIVACY.md in repo and packaged app.

Explicitly state: no personal data is collected, GDPR rights (access/erase) handled locally.

Store listing (Play Store / App Store) must mention:

* App works fully offline.
* All data stays on device unless exported by user.
* No tracking, no ads.

## Brain Dump

Tools suggested are browser-agnostic or Firefox-compatible (Playwright, axe-core, Testing Library, Vitest/Jest). Keep suites small and fast so they’re practical to run often. Prefer many small, reliable tests over one heavy, flaky one.

# Import/Export Schemas

## Task

{

"id": "2-uuid",

"title": "string",

"category": "string",

"quadrant": "UI|NI|UN|NN",

"due": "ISO datetime | null",

"subtasks": [{"title": "string", "done": false}],

"xp": 1,

"notesRef": ["4-uuid"],

"createdAt": "ISO",

"updatedAt": "ISO"

}

## Routine

{

"id": "1-uuid",

"name": "string",

"steps": [{"title": "string", "targetSeconds": 600}],

"tags": ["string"],

"version": "semver",

"estimatedMinutes": 20,

"xpOnCompletion": 5,

"createdAt": "ISO",

"updatedAt": "ISO"

}

## RoutineLog

{

"id": "7-uuid",

"routineId": "1-uuid",

"entries": [{

"stepIndex": 2,

"actualSeconds": 650,

"onTime": false,

"action": "completed|skipped|cancelled|reordered",

"timestamp": "ISO"

}],

"aggregates": {"percentOnTime": 62}

}

## AggregatedStats

{

"id": "8-uuid",

"range": "2025-09",

"tasks": {"completed": 42, "byQuadrant": {"UI": 12, "NI": 18, "UN": 6, "NN": 6}},

"habits": {"currentStreak": 23, "longest": 45},

"routines": {"avgOverrunSec": 90}

}

## Habit

{

"id": "3-uuid",

"name": "string",

"categoryColor": "#RRGGBB",

"streak": 23,

"vacation": [{"from": "ISO", "to": "ISO"}],

"xp": {"daily": 1, "newStreak": 3, "milestone": 2}

}

## ScheduleEvent

{

"id": "5-uuid",

"sourceType": "task|routine|habit|note",

"sourceId": "2-uuid|1-uuid|3-uuid|4-uuid",

"title": "string",

"start": "ISO",

"end": "ISO",

"categoryColor": "#RRGGBB",

"urgent": false

}

## NoteLog (Planning Import)

## Import Compatibility — "Daily Note Markdown" (SJ-MD-DN v1)

{

"id": "9-uuid",

"noteId": "4-uuid",

"title": "string",

"date": "ISO",

"start": "ISO",

"end": "ISO",

"done": false,

"raw": "string",

"source": "import:md-planning",

"sourceHash": "sha256"

}

### Purpose

Reliably import daily note Markdown files with year/month/day headings, Planning tables, TODO checklists, quotes, and code fences. Planning rows are captured as NoteLogs (informational tracking), not as scheduled calendar events.

### Accepted file types

.md UTF-8 (no BOM)

Line endings: LF or CRLF

### Document structure (assumptions)

Top-level sections such as:

# DAY BY DAY

## 2024

### <Current Month>

#### <Current Day>

Sub-sections may include:

##### Planning

##### TODO <context x>

##### TODO <context y>

Planning blocks are GitHub-flavored Markdown tables with a header row. Example:

| Hour | Task | Done? |

|------|------|-------|

| 15h30 -> 16h00 | Task | No |

Checklists use:

- [ ] and - [X] or - [x] (case-insensitive)

Quotes use:

> quoted text

Code fences use:

``` (triple backticks)

### Mapping rules

#### A) Brain Dump

Each day section (#### <weekday> <day>) becomes one Daily Note entity.

Note body stores the full Markdown of the day, excluding elements extracted as Tasks or Planning NoteLogs.

Metadata

note.date = YYYY-MM-DD derived from the enclosing year and month headings plus the day

note.title = "<YYYY-MM-DD> Daily Note"

note.tags = year/<YYYY>, month/<MM>, and contextual tags inferred from headings (e.g., context/pro, context/perso)

#### B) Planning (manual tracking logs)

Each Planning table row becomes a NoteLog linked to the corresponding Daily Note.

Fields

log.source = "import:md-planning"

log.title = value from the Task column

log.date = note.date

log.start and log.end parsed from the Hour column

Recognized formats: "15h30 -> 16h00", "15h30-16h00", "l4h30-15h" (strip stray "l"), "18h-?"

If end is missing or "?", default duration = 1h

If only a start time is present, default duration = 1h

log.done derived from the Done? column ("Yes", "No", "En cours", "Non"; case/locale-insensitive)

log.raw stores the original Markdown row, including table pipes (|)

Important

Planning rows do not create ScheduleEvents. They remain manual tracking logs for later review.

#### C) Tasks (from checklists)

Each checklist line creates a Task.

task.title = text after the checkbox

task.completed = true when [X] or [x]

task.category inferred from surrounding headings

headings containing "pro" → Work

headings containing "perso" → Personal

otherwise → Uncategorized

Subtasks

* nested list items or directly numbered sub-items under a checklist become subtasks of the nearest parent Task
* Project assignment
  + if a project entity with the same name already exists, assign it
  + otherwise add a project/<name> tag
* Unrecognized bullets remain in the Note body

#### D) Code fences and CLI snippets

* Preserved verbatim inside the Daily Note as code blocks
* Not converted into Tasks unless a checkbox is present on the same line

#### E) Quotes and callouts

* Lines beginning with ">" are preserved as quoted paragraphs in the Note body

#### F) IDs and integrity

* All imported objects receive typed IDs: <typePrefix>-<uuid>
  + 1-… Routines,
  + 2-… Tasks,
  + 3-… Habits,
  + 4-… Brain Dump,
  + 5-… ScheduleEvents,
  + 6-… Templates,
  + 7-… RoutineLogs,
  + 8-… AggregatedStats,
  + 9-… NoteLogs
* The composite ID is exported unchanged for integrity checks
* Created Tasks and NoteLogs retain a sourceHash (SHA-256 of original Markdown fragment)

#### G) Time and locale handling

* French day/month labels and hour formats are recognized
* If weekday name conflicts with computed date, the numeric date is trusted and a non-blocking warning is recorded
* Time zone uses the user’s current zone; no TZ shift is applied on import

#### H) Error tolerance and recovery

* Malformed Planning tables are kept in the Note body and flagged in the dry-run report
* Ambiguous hours with no start and no end are stored as NoteLogs without times
* Duplicate checklist lines within the same day are de-duplicated by normalized text + hash

#### I) Encryption, hashing, cleanup

* Encrypted .sj.enc imports are supported; plaintext .md is accepted
* Textarea hashing and optional at-rest encryption follow the Data & Storage policy
* Auto-cleanup default is 1 month; users may disable it globally or per item

#### J) Dry-run and UX

* Dry-run summary displays counts for Daily Brain Dump, Planning rows (NoteLogs), Tasks by status, and warnings
* Import modes include
  + Import All
  + Brain Dump Only
  + Tasks Only
  + Planning Logs Only
* For the same day already present
  + Merge appends new tasks/logs and keeps prior note body separated by a divider
  + Replace overwrites the day

Export symmetry

* Exporting a day reproduces the same structure (headings, Planning table, checklists)
* Re-importing the exported Markdown is idempotent and preserves composite IDs for unchanged entities

## Markdown bundle mapping

* Each schema in §13 maps 1:1 to the JSON body contained in fenced blocks in the Markdown bundle.
* Unknown properties must be preserved on export and re-import.

# Internationalisation (i18n)

## Purpose

Ensure Stellar-Journey can be used in multiple languages and cultural contexts.

Focus on English (default) and French (secondary), with the structure ready for additional languages later.

## Supported Languages

English (en-GB) — default

French (fr-FR)

Future languages possible if contributors join

## Language Selection

User-selectable in Settings (see §8).

Follows browser locale automatically on first launch.

Language preference saved locally and in exports/imports.

## Text Management

All UI text stored in translation files (JSON or YAML).

No hard-coded strings in UI code.

Example:

en.json → { "tasks": "Tasks", "routines": "Routines" }

fr.json → { "tasks": "Tâches", "routines": "Séquences" }

Translation keys are stable identifiers, not English text.

## Date, Time, and Number Formatting

Respect locale for:

Date format (DD/MM/YYYY vs MM/DD/YYYY).

Time format (24h vs 12h).

Decimal separators.

Use Intl API (ECMAScript Internationalisation) for formatting.

## Accessibility of Localisation

All translated strings must preserve meaning and clarity.

No abbreviations that confuse screen readers.

Labels and aria attributes also translated.

Default fallback: English.

## Content to Translate

UI labels (tabs, buttons, dialogs).

Notifications and reminders.

Microcopy and guidance messages.

Error messages and warnings.

Markdown templates and example content (notes, routines).

## Non-Translated Content

Technical words and commands (e.g., JSON, Markdown, SHA-256) remain untranslated.

User-generated content (notes, tasks, habits) is never auto-translated.

## Contribution & Extensibility

Translation files modular; each language in a separate folder (/locales/en/, /locales/fr/).

Developers/contributors can add new languages by copying en.json and translating.

Contributions validated by simple automated test (all keys present).

## Testing

Manual test with Settings switch (English ↔ French).

Check text overflow and alignment in both languages.

Verify date/time formats adapt correctly.

## Brain Dump

English and French are fully supported at release.

Architecture allows more languages if community grows..

# Notifications

## Purpose

Provide reminders and alerts that are helpful without being overwhelming.

Notifications should support productivity, habit tracking, and scheduling while respecting user preferences.

## Notification Types

Task reminders

Important + Urgent tasks auto-surfaced to Schedule.

Optional reminders for Important (not Urgent) tasks.

Routine reminders

Alert when a routine is scheduled to start.

Optional step reminders if a timer expires.

Habit reminders

Daily reminder at user-defined time (default: 8 AM).

Break reminders at the end of a habit sequence.

Schedule events

Alerts for upcoming events with 15-minute default lead time.

Streak & milestone notifications

Notify when user reaches 7/14/28/50/100/250/500/1000-day streaks.

Encouraging but non-intrusive.

## Delivery Channels

In-app notifications (always available).

Push notifications (if PWA installed and permission granted).

Desktop notifications (browser permission required).

No email or SMS.

## Timing & Snooze

Default reminders: 15 minutes before tasks/events.

Snooze options: 15 min, 1 hour, until tomorrow.

Quiet hours: user-defined (default 22:00–07:00) → no notifications during this time.

## Content & Tone

Short, clear, friendly messages.

Consistent with brand voice (see §6 and §7).

Examples:

“Your orbit continues 🌟 Routine ‘Morning Flow’ starts now.”

“Don’t forget: Task ‘Send report’ due in 15 min.”

“Habit check-in: Did you complete ‘Daily Walk’ today?”

“Great work! 🎉 You reached a 14-day streak.”

## Accessibility

All notifications have text, no reliance on sound only.

Option to disable vibration, sound, or haptic feedback.

Respect Reduced Motion and Reduced Sensitivity user settings.

## User Controls

Global toggle for notifications (on/off).

Per-category toggles: Tasks, Routines, Habits, Schedule, Streaks.

Quiet hours configurable.

Reminder time configurable for habits and tasks.

Option to pause all notifications temporarily (1 day to 1 week).

## Technical Implementation

Browser API: Notification API for desktop/web.

PWA Push: Web Push protocol with Service Worker.

Local scheduling: Service Worker + IndexedDB timers.

No background polling: use scheduled alarms and background sync.

Permissions requested on first use, never forced.

## Privacy

No notifications include sensitive or personal information.

Notification payloads minimal; details fetched locally when app is opened.

All scheduling done client-side, no server involvement unless explicitly connected to backend.

## Brain Dump

Notifications must encourage but never pressure.

The default setup is minimal; power users can enable more.

All notification features respect the app’s privacy-first principle.

# Statistics

## Purpose

Provide meaningful insights into tasks, routines, habits, and notes without overwhelming the user.

Focus is on duration tracking, streaks, and progress milestones.

## General Principles

Statistics are always optional and secondary to core features.

Displayed in a clear, non-pressuring way.

Encourages learning and adjustment, not guilt.

Simplified Mode (see §8 Accessibility) hides advanced statistics.

## Tasks

Record completion date and time.

Distinguish Important/Urgent tasks in stats.

Track number of tasks completed per day/week/month.

Aggregate completion times to estimate realistic durations.

XP gained recorded and exportable.

## Routines

Each step has expected vs. actual duration logged.

Mean duration per step and per routine calculated automatically.

Completion rate (steps skipped, cancelled, interverted).

Routine completion streaks (daily, weekly).

XP rewards logged.

Logs stored separately from core data (see §5 Data & Storage).

## Habits

Track streaks (7/14/28/50/100/250/500/1000).

Vacation mode pauses streaks without penalty.

Mean duration of repeatable tasks/habits logged.

Track missed vs. completed days.

XP points: 1 point per tick, plus streak bonuses.

Reminders to define habit before starting for better engagement.

## Brain Dump

Metadata includes created/modified timestamps.

Optionally tag notes with categories.

No scoring, only count and time-tracking of note creation.

## XP & Gamification Stats

XP system unified across Tasks, Routines, and Habits.

1 XP = one completed non-urgent/non-important task OR one step of routine.

Streak bonuses and on-time bonuses logged.

Badges for milestones: completed routines, streak lengths, XP totals.

## Visualisation

Heatmaps for activity (tasks, habits, routines).

Bar charts for completion by category.

Streak counters displayed with progress rings.

Charts simple and accessible; color + labels, no color-only coding.

Respect Reduced Motion/High Contrast settings.

## Data Export

All stats exportable as .json or .md.

Aggregate stats stored in separate file (stats.json) to limit exposure and recomputation.

XP milestones included in exports.

User may delete stats independently of other data.

Brain Dump

* Statistics should inform and encourage but never overwhelm.
* Defaults to minimal views; advanced visualisation available on demand.

# Gamification, Notifications & Feedback

## Purpose

Make daily tasks, routines, and habits more engaging and rewarding without adding pressure.

Gamification is designed to encourage consistency and progress through small, enjoyable feedback loops.

## XP System

Base Unit

* 1 XP = one completed non-urgent, non-important task OR one completed routine step.

Scaling

* Important task completion: +3 XP.
* Urgent task completion: +2 XP.
* Important + Urgent task completion: +5 XP.
* Subtasks can also earn XP, proportional to their classification.

Habits

* Each tick (completed daily check-in) = 1 XP.
* New streak started (≥3 consecutive days) = +3 XP bonus.
* Extended streaks (7, 14, 28, etc.) reward extra XP.

Routines

* Each step: 1 XP + small animation, haptic feedback, optional sound.
* Step completed within its timer: +1 bonus XP.
* Routine completed: +10 XP, with confetti, sound, badge unlock.
* Routine completed within total allocated time: +5 bonus XP.

## Feedback & Rewards

* Confetti on:
  + Important tasks
  + Urgent tasks
  + Important + Urgent tasks
  + Habit streaks
* Routine completions
  + Double confetti if task was Important + Urgent.
  + Badges for milestones (streaks, XP totals, completed routines, important tasks).
* Haptics
  + Gentle vibration on standard task completion.
  + Stronger vibration on Important task completion.
  + Strongest vibration on Important + Urgent task completion.
  + Gentle vibration on routine steps.
  + Stronger vibration on milestones (routine completion, new streak).
* Sounds
  + Subtle success sounds (optional, user can disable).
  + Distinctive celebratory sound for major milestones.
* Progress Indicators
  + Golden progress bar under navbar shows global progression.
  + Streak counters shown as progress rings.

## Streaks & Milestones

Habits

* Streak tracked at 7, 14, 28, 50, 100, 250, 500, 1000 days.
* Vacation mode: user can pause streak without penalty.
* After a sequence, user reminded to take a break.

Routines

* Track consecutive daily completions.

Tasks

* Milestones based on number of Important and Important + Urgent tasks completed.

## Badges

Categories

* Explorer: for first completed routine.
* Consistent: for streaks (habits, routines).
* Achiever: for task milestones.
* Focused: for completing Important (not Urgent) tasks.
* Crisis Master: for completing Important + Urgent tasks.
* Builder: for creating routines or tasks.

Display

* Badges not shown in Library.
* Displayed near Statistics section, grouped by category.
* Biggest earned badge also displayed at the left of the navbar as recognition.

Extensibility

* Future badges may be added without breaking existing data.

## User Controls

Gamification is always optional.

User can disable:

* Confetti
* Sounds
* Haptic feedback

XP and streaks remain tracked even if feedback is disabled.

## 15.6 Data & Export

All gamification stats (XP, streaks, badges) stored in separate stats.json file.

Exported alongside other data (see §5 and §14).

Fully reversible import/export.

## Brain Dump

Gamification is designed to support motivation, not pressure.

Important tasks receive special recognition, as they are the most often delayed.

Rewards are light, fun, and never punish failure.

# Annex — User Stories

## Purpose

Collect user stories that guide design and implementation. Each story follows the template:

As a [user type], I want [feature], so that [goal].

## UX/UI & Responsiveness

### US-UX-01 – Adaptive layout for mobile screens

**As a** mobile user

**I want** the layout to adapt to small screen sizes

**So that** I can access all key features without horizontal scrolling or hidden components

### US-UX-02 – Optimized tablet interface

**As a** tablet user

**I want** the interface to leverage the wider screen real estate

**So that** I can view more content at once and interact with side panels

### US-UX-03 – Clear and simple navigation

**As a** user

**I want** consistent navigation across devices

**So that** I never feel lost between tabs or popups

## Visual Identity

### US-VI-01 – Consistent "stellar" theme

**As a** user

**I want** the interface to reflect a soothing, celestial theme

**So that** I feel calm and focused while using the app

### US-VI-02 – Flexible theming

**As a** user

**I want** to switch between light, dark, and high contrast themes

**So that** I can adapt the visual identity to my needs

**And so** that developers can easily add new themes in the future

## Accessibility & Inclusion

### US-AC-01 – High contrast mode

**As a** user with low vision

**I want** to switch to high contrast mode

**So that** I can read without eye strain

### US-AC-02 – Simplified mode

**As a** user with ADHD or cognitive overload

**I want** a simple mode with fewer details and trends

**So that** I don’t feel overwhelmed

### US-AC-03 – Adjustable font size

**As a** user with visual sensitivity

**I want** to increase or decrease text size

**So that** I can read comfortably

### US-AC-04 – Drag-and-drop alternatives

**As a** user with motor difficulties

**I want** to reorder tasks without drag-and-drop

**So that** I can still organise my work

### US-AC-05 – Full keyboard navigation

**As a** keyboard-only user

**I want** to navigate all features with the keyboard

**So that** I can use the app without a mouse

### US-AC-06 – Screen reader labels

**As a** user relying on a screen reader

**I want** every button clearly labelled

**So that** I know what action it will trigger

### US-AC-07 – Disable sensory overload

**As a** user with sensory sensitivity

**I want** to disable sounds, vibrations, or animations

**So that** I don’t get overloaded

### US-AC-08 – Respect system preferences

**As a** user

**I want** the app to respect my system settings (reduced motion, high contrast)

**So that** I don’t have to reconfigure them

## Data & Storage

### US-DS-01 – Prompt before closing

As a user

I want a reminder to export data before closing the app

So that I don’t lose work

### US-DS-02 – Automatic clean-up

As a user

I want old logs and snapshots to be cleaned up automatically after a month

So that I don’t accumulate clutter

And backlog items are never deleted automatically

### US-DS-03 – Override clean-up

As a user

I want to keep specific tasks, routines, or notes from being auto-deleted

So that I don’t lose important information

### US-DS-04 – Manual cleanup

As a user

I want to delete stale or specific items manually

So that I stay in control of my storage

### US-DS-05 – Encrypted storage

As a privacy-conscious user

I want my local data and exports encrypted with a passphrase

So that no one else can read them

### US-DS-06 – Integrity checks

**As a** user

**I want** export files to include optional SHA-256 checksums

**So that** I can verify they weren’t corrupted

### US-DS-07 – UUID-based IDs

**As a** developer or advanced user

**I want** each item to have a unique ID (routines start with 1, tasks with 2, etc.)

**So that** data consistency can be verified

### US-DS-08 – Global import/export file

**As a** user

**I want to** store everything in one file (settings, stats, notes, tasks, routines)

**So that** I can easily backup and restore my app

## US-DS-09 – Timezone handling and UTC persistence

**As a** user

**I want** my content shown in my local timezone while data is saved in UTC when online

**So that** everything displays correctly for me and remains consistent across devices and servers

## Import/Export

### US-IE-01 – Global import/export

**As a** user

**I want** to export and re-import all my data (settings, stats, notes, tasks, routines)

**So that** I never lose anything

### US-IE-02 – Markdown support

**As a** user

**I want** my data exported in Markdown

**So that** I can version it with GitHub and edit externally

### US-IE-03 – Reversible import

**As a** user

**I want** every export to be re-importable exactly

**So that** nothing is lost or changed

### US-IE-04 – Sanitize all imports

**As a** user

**I want** ALL imported files (Markdown, HTML, JSON) to be strictly sanitized

**So that** no hidden scripts or dangerous content can execute

### US-IE-05 – YAML front-matter

**As a** user

**I want** exported Markdown files to include YAML metadata

**So that** my notes and routines keep consistent IDs and timestamps

### US-IE-06 – Export integrity

**As a** user

**I want** exports to generate an optional SHA-256 checksum file

**So that** I can confirm their integrity

### US-IE-07 – Normalisation

**As a** user

**I want** imported files to be normalised (UTF-8 encoding, LF line endings, ISO 8601 dates)

**So that** I avoid compatibility issues

### US-IE-08 – Autosave prompt

**As a** user

**I want** to be prompted to export unsaved content before closing the browser

**So that** I don’t lose important notes or routines

### US-IE-09 – Safe Markdown rendering

**As a** user

**I want** the Markdown viewer to disable raw HTML and block javascript: or on\* handlers in links/images

**So that** viewing notes is safe even if the source contains

### HTML US-IE-10 – Export notes with images

**As a** user

**I want** my Markdown notes exported with their images bundled

**So that** I can restore them exactly later

**Using** a single archive (e.g., .sj.enc or zip) with stable relative paths

### US-IE-11 – Import notes with images

**As a** user

**I want** to import a Markdown note and its referenced images together

**So that** links are fixed automatically and nothing breaks

**With** strict sanitisation of filenames and no remote downloads without consent

## Tasks

### US-TS-01 – Eisenhower matrix

**As a** user

**I want** tasks displayed in a 2x2 importance/urgency matrix

**So that** I can focus on what really matters

### US-TS-02 – Backlog list

**As a** user

**I want** unscheduled tasks kept in a backlog list

**So that** I can decide later when to schedule them

### US-TS-03 – Drag from backlog to matrix

**As a** user

**I want** to drag and drop tasks from the backlog to the Eisenhower matrix (and back)

**So that** I can quickly reclassify them

### US-TS-04 – Subtasks in matrix

**As a** user

**I want** subtasks to appear in the Eisenhower matrix with a link to their parent task

**So that** I can progress on a subtask without managing the entire task

### US-TS-05 – Task details on click

**As a** user

**I want** task details hidden in the matrix by default and shown only when clicked

**So that** the interface stays clean

### US-TS-06 – Scheduling important tasks

**As a** user

**I want** Important + Urgent tasks to be auto-surfaced into Schedule at their due date (or today)

**So that** I never miss them

### US-TS-07 – Reminder for important tasks

**As a** user

**I want** to be reminded to schedule Important (but not Urgent) tasks

**So that** I don’t keep postponing them

### US-TS-08 – Unschedule tasks anytime

**As a** user

**I want** to unschedule a task at any time

**So that** I can adapt to changes

### US-TS-09 – Task rewards

**As a** user

**I want** Important and Important + Urgent tasks to trigger stronger feedback (confetti, stronger haptics, badges)

**So that** I feel motivated to tackle the tasks I tend to avoid

### US-TS-10 – Category and due date display

**As a** user

**I want** task cards to display their category and due date

**So that** I can prioritise quickly at a glance

## Routines

### US-RT-01 – Step-by-step flow

**As a** user

**I want** to see the current step larger, the previous step faded, and the next step previewed

**So that** I stay oriented in my routine

### US-RT-02 – Timed steps

**As a** user

**I want** each step to have a timer

**So that** I can pace myself correctly

### US-RT-03 – Timer alerts

**As a** user

**I want** alerts before the end of a step and at its exact end

**So that** I don’t miss transitions

### US-RT-04 – Step actions

**As a** user

**I want** to skip, cancel, or reorder steps

**So that** I can adapt when my flow changes

### US-RT-05 – Completion rewards

**As a** user

**I want** completing a step to give haptic feedback, a sound, and XP

**So that** I feel rewarded for progress

### US-RT-06 – Bonus rewards for timing

**As a** user

**I want** extra XP for completing a step or routine within the allocated time

**So that** I feel encouraged to stay on track

### US-RT-07 – Routine completion feedback

**As a** user

**I want** finishing a routine to trigger a large reward (confetti, sound, badge, XP boost)

**So that** I feel motivated to complete routines

### US-RT-08 – Routine streaks

**As a** user

**I want** consecutive daily routine completions to be tracked

**So that** I can build consistency

### US-RT-09 – Log routine outcomes

**As a** user

**I want** my routine logs stored separately with optional tags (e.g. “low energy”, “ADHD fog”)

**So that** I can later filter or analyse them

## Habits

### US-HB-01 – Daily tick

**As a** user

**I want** to log my habit completion with a simple daily tick

**So that** I can track consistency easily

### US-HB-02 – Streak tracking

**As a** user

**I want** streaks tracked at 7, 14, 28, 50, 100, 250, 500, and 1000 days

**So that** I see my long-term progress

### US-HB-03 – Vacation mode

**As a** user

**I want** to pause my streaks without penalty

**So that** I can take breaks and resume later

### US-HB-04 – Break reminders

**As a** user

**I want** to be reminded to rest after finishing a sequence

**So that** I avoid fatigue or burnout

### US-HB-05 – Easy habits

**As a** user

**I want** to be encouraged to choose small, quick habits

**So that** I don’t feel overwhelmed or discouraged

### US-HB-06 – Pre-definition

**As a** user

**I want** to define a habit the day before starting it

**So that** I feel more engaged and committed

### US-HB-07 – Habit penalties

**As a** user

**I want** no penalty if I miss up to 2 days in a week (defaults: x=2, y=1 week, user-configurable)

**So that** missing a few days doesn’t erase all my progress

### US-HB-08 – Habit colours

**As a** user

**I want** to choose from up to 10 colours for my habits, independently of the theme

**So that** I can personalise them

### US-HB-09 – Engagement declaration on habit creation

**As a** user

**I want** a clear call-to-action that asks me to confirm my commitment when creating a new habit

**So that** I feel engaged and increase my likelihood of following through

## Brain Dump & Brain Dumps

### US-NT-01 – Rich Markdown notes

**As a** user

**I want** to take notes in extended Markdown (tables, lists, code, task lists)

**So that** I can capture ideas clearly without HTML

### US-NT-02 – Light/dark note viewer

**As a** user

**I want** to switch between light and dark themes only for the notes editor and viewer

**So that** I can read comfortably even if the global theme differs

### US-NT-03 – Metadata

**As a** user

**I want** my notes to include metadata (title, created/modified timestamps, tags, language)

**So that** they stay organised

### US-NT-04 – Brain dumps

**As a** user

**I want** a quick place to dump thoughts into Markdown files with YAML front-matter

**So that** I can declutter my mind instantly

### US-NT-05 – Shared tags

**As a** user

**I want** tags to be shared across notes, brain dumps, and tasks

**So that** I can easily link ideas and actions

### US-NT-06 – Safe Markdown rendering

**As a** user

**I want** all HTML in my Markdown notes neutralised

**So that** no unsafe code can run in my notes

### US-NT-07 – Export/import notes

**As a** user

**I want** my notes and brain dumps to be exported and re-imported without corruption

**So that** I never lose them

### US-NT-08 – Insert images in Markdown

**As a** user

**I want** to add images to my notes by paste, drag-and-drop, or file picker

**So that** I can illustrate ideas

**With** required alt text for accessibility and optional captions

### US-NT-09 – Local, safe image handling

**As a** privacy-conscious user

**I want** images saved locally (OPFS) and referenced by safe, relative links or blob URLs

**So that** no external requests are made

**And** remote image URLs are blocked unless I explicitly import the file

## Schedule

### US-SC-01 – Unified scheduling

**As a** user

**I want** to schedule tasks, routines, and habits created elsewhere in the app

**So that** I manage all events in one place

### US-SC-02 – Drag and drop

**As a** user

**I want** to drag items into the schedule calendar

**So that** I can quickly plan my day

### US-SC-03 – Resize events

**As a** user

**I want** to change the duration of an event by dragging its bottom edge

**So that** I can adjust timings easily

### US-SC-04 – Color-coded borders

**As a** user

**I want** scheduled elements to display their assigned color as a border

**So that** I identify them at a glance

### US-SC-05 – Suggest next available slot

**As a** user

**I want** the app to propose the next free time slot with a default 15-minute break in between

**So that** I don’t have to search manually for gaps in my day

### US-SC-06 – Break duration preference

**As a** user

**I want** to configure the default break duration between scheduled events

**So that** I can adapt it to my needs

### US-SC-07 – Smooth navigation

**As a** user

**I want** to switch easily between day, week, and month views

**So that** I can plan at the right level of detail

## Schedule

### US-SC-01 – Unified scheduling

**As a** user

**I want** to schedule tasks, routines, and habits created elsewhere in the app

**So that** I manage all time-bound events in one place

### US-SC-02 – Drag and drop to schedule

**As a** user

**I want** to drag items into the schedule calendar

**So that** I can quickly plan my day

### US-SC-03 – Move events

**As a** user

**I want** to drag and drop scheduled events between time slots

**So that** I can easily reschedule them

### US-SC-04 – Resize events

**As a** user

**I want** to change the duration of an event by dragging its bottom edge

**So that** I can adjust timings easily

### US-SC-05 – Color-coded borders

**As a** user

**I want** scheduled elements to display their assigned color as a border

**So that** I identify them at a glance

### US-SC-06 – Suggest next available slot

**As a** user

**I want** the app to propose the next free time slot with a default 15-minute break in between

**So that** I don’t have to search manually for gaps in my day

### US-SC-07 – Disable smart scheduling

**As a** user

**I want** to turn off automatic slot suggestions

**So that** I can place events strictly where I want

### US-SC-08 – Conflict visibility

**As a** user

**I want** conflicting events to both appear in the schedule

**So that** I can decide which one to prioritise

### US-SC-09 – Smooth navigation

**As a** user

**I want** to switch easily between day, week, and month views

**So that** I can plan at the right level of detail

## Gamification

### US-GM-01 – XP system

**As a** user

**I want** XP for completing tasks, routines, and habits

**So that** my progress feels tangible

### US-GM-02 – Task rewards

**As a** user

**I want** Important and Important + Urgent tasks to trigger stronger feedback (confetti, stronger haptics, badges, extra XP)

**So that** I feel motivated to tackle the tasks I might otherwise postpone

### US-GM-03 – Routine rewards

**As a** user

**I want** each step to give me XP, haptic feedback, and optional sound

**So that** I feel rewarded at every step

### US-GM-04 – Timing bonuses

**As a** user

**I want** extra XP for completing steps or routines within their timers

**So that** I stay on track

### US-GM-05 – Routine completion celebration

**As a** user

**I want** finishing a routine to trigger confetti, sound, a badge, and a big XP boost

**So that** I feel encouraged to complete routines

### US-GM-06 – Habit streak bonuses

**As a** user

**I want** streaks at 7, 14, 28, 50, 100, 250, 500, 1000 days to grant XP bonuses

**So that** I stay motivated to maintain habits

### US-GM-07 – Vacation mode streaks

**As a** user

**I want** to pause streaks without penalty

**So that** I don’t lose progress when I rest

### US-GM-08 – Badge collection

**As a** user

**I want** badges for milestones (first routine, Important tasks, streaks, XP totals, builder achievements)

**So that** I have visible recognition of my progress

### US-GM-09 – Badge display

**As a** user

**I want** badges grouped near statistics, with my biggest badge shown left of the navbar

**So that** my achievements are visible but not intrusive

### US-GM-10 – User controls

**As a** user

**I want** to disable confetti, haptics, or sounds independently

**So that** I can enjoy gamification without overload

## Notifications

### US-NO-01 – Task reminders

**As a** user

**I want** Important + Urgent tasks to auto-surface into Schedule and send me reminders

**So that** I never miss critical deadlines

### US-NO-02 – Reminder for Important tasks

**As a** user

**I want** optional reminders for Important (but not Urgent) tasks

**So that** I don’t postpone them indefinitely

### US-NO-03 – Routine step alerts

**As a** user

**I want** alerts before a step ends and exactly at its end

**So that** I transition smoothly between routine steps

**Even** if quiet hours are enabled

### US-NO-04 – Habit reminders

**As a** user

**I want** a daily reminder at a user-defined time

**So that** I remember to log my habits consistently

### US-NO-05 – Break reminders

**As a** user

**I want** to be reminded to take a break at the end of a sequence

**So that** I don’t overexert myself

### US-NO-06 – Streak and milestone notifications

**As a** user

**I want** to be notified when I reach milestones (7/14/28/50/100/250/500/1000 days)

**So that** I stay motivated

### US-NO-07 – Schedule event alerts

**As a** user

**I want** alerts for upcoming events with a 15-minute default lead time

**So that** I’m on time for my commitments

### US-NO-08 – Snooze options

**As a** user

**I want** to snooze notifications for 15 minutes, 1 hour, or until tomorrow

**So that** I can control interruptions

### US-NO-09 – Quiet hours

**As a** user

**I want** to set quiet hours during which notifications are paused

**So that** I’m not disturbed at night

**Except** for critical routine step alerts

### US-NO-10 – Notification channels

**As a** user

**I want** notifications delivered in-app, as desktop notifications, or push notifications (if PWA installed)

**So that** I can choose the most convenient channel

### US-NO-11 – Accessibility of notifications

**As a** user

**I want** notifications to always have text, with optional sound/vibration

**So that** they are accessible regardless of sensory limitations

## Statistics

### US-ST-01 – Task stats

**As a** user

**I want** my completed tasks counted and categorised (Important, Urgent, both, or neither)

**So that** I can review how I spend my time

### US-ST-02 – Task duration

**As a** user

**I want** mean completion times logged for tasks

**So that** I can estimate future tasks more realistically

### US-ST-03 – Routine step duration

**As a** user

**I want** each step’s expected vs. actual duration recorded

**So that** I can compare my planning with reality

### US-ST-04 – Routine completion stats

**As a** user

**I want** completion rates, skipped steps, and reordered steps tracked

**So that** I can understand how I follow routines

### US-ST-05 – Habit streak stats

**As a** user

**I want** streaks tracked (7/14/28/50/100/250/500/1000 days) with pause options

**So that** I can build consistency without losing progress during breaks

### US-ST-06 – Habit duration stats

**As a** user

**I want** repeatable tasks and habits to log their mean duration

**So that** I can adjust my expectations over time

### US-ST-07 – XP and gamification stats

**As a** user

**I want** all XP gains, streaks, and bonuses recorded

**So that** I can follow my progress and milestones

### US-ST-08 – Visualisation

**As a** user

**I want** clear, accessible visualisations (heatmaps, bar charts, streak rings)

**So that** I can understand my progress at a glance

**With** labels and icons, never color alone

### US-ST-09 – Export stats

**As a** user

**I want** to export my statistics as JSON or Markdown

**So** that I can analyse them outside the app

## Cross-Cutting Guidance

### US-CG-01 – Science-based best practices

**As a** user

**I want** short, optional messages about best practices in productivity, routines, and task management

**So that** I can learn from state-of-the-art research

**With** a link to a meta-analysis and a trusted science educator

### US-CG-02 – Consistent color coding

**As a** user

**I want** consistent color coding across tasks, routines, habits, notes, and schedule

**So that** I can understand categories and priorities at a glance

## Developer Utilities & Maintainability

### US-DU-01 – External Markdown viewer template

**As a** maintainer

**I want** the Markdown viewer’s template kept in a separate file

**So that** updating or swapping the viewer is easy and low-risk

### US-DU-02 – Pluggable renderer

**As a** maintainer

**I want** the Markdown rendering pipeline to be modular (viewer, sanitizer, parser)

**So that** I can upgrade components independently without touching app logic

## Library (Templates)

### US-LB-01 – Browse templates

**As a** user

**I want** to browse predefined templates and my own saved templates

**So that** I can start quickly

### US-LB-02 – Create from template

**As a** user

**I want** to create a task, routine, or note from a template

**So that** I don’t have to start from scratch

### US-LB-03 – Save as template

**As a** user

**I want** to save my current task, routine, or note as a template

**So that** I can reuse it later

### US-LB-04 – Import/export templates

**As a** user

**I want** to import and export templates in Markdown

**So that** I can version them in GitHub

### US-LB-05 – No community sharing

**As a** user

**I want** templates to stay local (no public catalog)

**So that** my workflow remains private

### US-LB-06 – Search and filter

**As a** user

**I want** to search and filter templates by tags or categories

**So that** I can find the right one quickly

### US-LB-07 – Safe previews

**As a** user

**I want** template previews to be sanitized if they include Markdown

**So that** previewing is safe

## Internationalisation

### US-IN-01 – Language switch

**As a** user

**I want** to switch the interface language between English and French

**So that** I can use the app comfortably

### US-IN-02 – Locale formatting

**As a** user

**I want** dates, times, and numbers formatted for my locale (24h/12h)

**So that** information reads naturally

### US-IN-03 – Preference persistence

**As a** user

**I want** my language choice to persist across sessions and exports

**So that** I don’t have to reset it

### US-IN-04 – Accessible translations

**As a** user relying on assistive tech

**I want** aria labels and messages to be translated clearly

**So that** navigation remains usable

### US-IN-05 – Fallbacks

**As a** user

**I want** English as a safe fallback when a string is missing

**So that** the interface never breaks

## Testing & Reliability

### US-TR-01 – Works without JavaScript

**As a** user

**I want** a graceful fallback when JavaScript is disabled

**So that** I can still access help and exports

### US-TR-02 – Works without cookies

**As a** user

**I want** the app to function when cookies are refused

**So that** my privacy choices are respected

### US-TR-03 – Offline-first PWA

**As a** user

**I want** the app installable and usable offline

**So that** I can rely on it without network

### US-TR-04 – Multi-device reliability

**As a** user

**I want** the app validated on a high-end (4-year-old) Android phone, a lower-end phone, and an old iPad

**So that** it runs well on my device

### US-TR-05 – Cross-browser and device validation

**As a** user

**I want** the app validated on Firefox, Edge, and Chrome, across desktop, mobile, and simulators

**So that** it works reliably on my environment

### US-DU-01 – External Markdown viewer template

**As a** maintainer

**I want** the Markdown viewer’s template kept in a separate file

**So that** updating or swapping the viewer is easy and low-risk

### US-DU-02 – Pluggable renderer

**As a** maintainer

**I want** the Markdown rendering pipeline to be modular (viewer, sanitizer, parser)

**So that** I can upgrade components independently without touching app logic

# Developer Utilities & Maintainability

**DU-01** [Mandatory] The system shall include an in-app JSON schema validator for Tasks, Routines, Habits, Brain Dump, Schedule, and Templates to ensure imports/exports match the defined spec (§13).

**DU-02** [Mandatory] The system shall provide a debug console or developer mode (local only) that surfaces logs, schema validation results, and entity counts for quick troubleshooting.

**DU-03** [Mandatory] The system shall include a feature flag mechanism (simple config file or env var) allowing the maintainer to enable/disable experimental features without altering production code.

**DU-04** [Mandatory] The repository shall include a Postman (or HTTPie/curl) collection with sample API calls (if backend enabled), covering auth, import/export, and CRUD operations.

**DU-05** [Mandatory] The repository shall include cURL command examples in README or DEVNOTES for critical flows (import/export, unlock, delete all data).

**DU-06** [Mandatory] The Markdown rendering pipeline shall be modular: sanitizer, parser, and viewer implemented as replaceable components.

**DU-07** [Optional] The Markdown viewer template shall be kept in a separate file (not inline in React components) to allow independent editing/swapping.

**DU-08** [Optional] Developer utilities shall log structured JSON (when dev mode enabled) for easier debugging, but logs must be disabled in production builds.

**DU-09** [Optional] A minimal set of unit tests shall exist for developer utilities (schema validation, import/export transforms), runnable in < 2 minutes locally or via CI.

**DU-10** [Optional] A feature flag registry file shall document all available flags, their defaults, and safe values.

# Security Test Plan — Detailed

## Security Architecture Verification Checklist

**STP-SAVC-01** [Mandatory] The system shall encrypt all locally stored user data when passphrase is enabled, using AES-GCM per record/file with a 96-bit IV.

**STP-SAVC-02** [Mandatory] Keys shall be derived with PBKDF2; CryptoKey must remain non-extractable and cleared from memory on lock/close.

**STP-SAVC-03** [Mandatory] The system shall support fallback (unencrypted) storage if passphrase is disabled, while still enforcing CSP; user may enable passphrase later.

**STP-SAVC-04** [Mandatory] All imports (Markdown, JSON, HTML) shall be sanitized to remove scripts, handlers, iframes, and unsafe URLs.

**STP-SAVC-05** [Mandatory] Markdown rendering shall disable raw HTML and block javascript: URLs and on\* handlers.

**STP-SAVC-06** [Mandatory] Responses shall include a strict CSP blocking inline/eval; only self-hosted or SRI’d assets allowed.

**STP-SAVC-07** [Mandatory] JS disabled → app shows static fallback (branding, links, help), no infinite spinners.

**STP-SAVC-08** [Mandatory] Cookies disabled → app still works via IndexedDB/OPFS and localStorage.

## Automated & Manual Solo-Friendly Tests

**STP-AST-01** [Mandatory] CI shall run npm audit (production) and bundler-audit (if Rails backend used); release fails if critical vulns remain.

**STP-AST-02** [Mandatory] CI shall run Playwright E2E (Firefox) on core flows (Tasks, Routines, Habits, Brain Dump, Import/Export, Schedule).

**STP-AST-03** [Optional] Run OWASP ZAP Baseline locally against the app; log and fix P1/P2 findings if discovered.

**STP-AST-04** [Optional] Manually test IDOR by forging UUIDs in API calls (only relevant if backend is enabled).

## Security Headers & CSP Verification

**STP-HDR-01** [Mandatory] Verify via curl -I that responses include CSP, HSTS, Referrer-Policy=no-referrer, X-Content-Type-Options=nosniff.

**STP-HDR-02** [Mandatory] Check Firefox DevTools Security panel: no CSP violations, no inline/eval scripts, no unsafe sources.

**STP-HDR-03** [Optional] Ensure frame-ancestors in CSP restricts embedding to 'self' (prevents clickjacking).

## Import/Export Hardening

**STP-IEX-01** [Mandatory] Schema validation: malformed files fail with clear error; unknown fields preserved.

**STP-IEX-02** [Mandatory] Markdown rendering tests confirm unsafe HTML neutralised; keep 2–3 malicious test files in repo.

**STP-IEX-03** [Mandatory] Exports default to .sj.enc; round-trip re-import restores identical IDs and timestamps.

**STP-IEX-04** [Optional] Enable optional .sha256 checksum export; verify manually once per release cycle.

## Service Worker & PWA Security

**STP-SWP-01** [Mandatory] Verify cached assets are versioned; no cache poisoning possible by query string.

**STP-SWP-02** [Mandatory] Offline fallback page shows only safe static info (never decrypted content if app is locked).

**STP-SWP-03** [Optional] Manually test update flow: new SW prompts reload; ensure no mid-session breakage.

## Authentication, Sessions & Locking

**STP-AUTH-01** [Mandatory] Auto-lock wipes in-memory keys on timeout/visibilitychange; reopening requires passphrase/biometric.

**STP-AUTH-02** [Mandatory] Passphrase unlock works; incorrect entry shows error without leaking details.

**STP-AUTH-03** [Optional] JWT tokens tested for short expiry and refresh rotation (only relevant if backend enabled).

## UX & Data Lifecycle

**STP-PRIV-01** [Mandatory] “Delete all local data” wipes IndexedDB, OPFS, SW caches; verify by fresh reload.

**STP-PRIV-02** [Mandatory] Auto-cleanup only deletes logs/snapshots/tmp; backlog items never touched.

**STP-PRIV-03** [Optional] Maintain PRIVACY.md page with plain explanation of storage, exports, and encryption (5–10 lines).

## Supply Chain & Dependencies

**STP-SUP-01** [Mandatory] Dependencies shall be pinned (package-lock.json / Gemfile.lock); update intentionally.

**STP-SUP-02** [Mandatory] Build artifacts shall exclude secrets; CI verifies no .env or private keys are bundled.

**STP-SUP-03** [Optional] Check that all external fonts/assets are either self-hosted or loaded with SRI.

## Backups & Restore (Solo-Friendly)

**STP-IRB-01** [Mandatory] Perform a “restore drill” once per release: export a .sj.enc file, wipe local data, re-import, confirm counts/IDs match.

**STP-IRB-02** [Optional] If backend enabled: test a DB dump and restore on staging once per quarter.

## UI & A11y of Security-Critical Flows

**STP-UIA-01** [Mandatory] All destructive actions (wipe, disable passphrase, export unencrypted) require confirmation; dialogs are aria-modal with focus-trap.

**STP-UIA-02** [Mandatory] axe-core automated checks on security dialogs must show no critical violations.

**STP-UIA-03** [Optional] Run NVDA/VoiceOver manual pass once per major release on passphrase setup/unlock dialogs.

# Developer Utilities

In-app JSON schema validator; Postman collection; cURL examples; feature flags; debug console (dev only).

# Tab Appearance

## Tasks Tab

Purpose: Provide a clear 2×2 Eisenhower matrix for active tasks, plus a backlog list for parked ideas, styled consistently with Stellar-Journey’s cosmic visual identity.

### Overall Structure

**TAB-TSK-01** The Tasks screen shall render a top row of Task Management Buttons, followed by a 2×2 quadrant grid, followed by a Backlog list; there is no backlog cell inside the grid.

**TAB-TSK-02** The 2×2 grid quadrants shall be labelled exactly (top-left → bottom-right): Important & Urgent; Important; Urgent; Neither important nor urgent.

**TAB-TSK-03** The Backlog shall be displayed as a full-width list below the grid, with the header “Backlog”.

**TAB-TSK-04** The visual structure shall follow this order and spacing:

<Buttons row>

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

| Important & Urgent | Important |

| Urgent | Neither important nor urgent |

-------------------------------------------------------------

==========================================

Backlog

-------------------------------------------------------------

task name | tags | due date

### Task Management Buttons (top row)

**TAB-TSK-05** The buttons row shall include: [Add Task] [Bulk Edit] [Filter] [Sort] [Import] [Export] [Toggle Completed].

**TAB-TSK-06** The Filter control shall filter by Category, Tag, Quadrant, Due (Today / This Week / Overdue), and Status.

**TAB-TSK-07** The Sort control shall provide: Due date (asc/desc), Quadrant order, Last updated, Title.

**TAB-TSK-08** Bulk Edit mode shall enable multi-select to Move (to quadrant/backlog), Complete, Delete, Tag, or Set Due Date.

**TAB-TSK-09** A search field may be provided to filter tasks by title/description.

### Display & Colour System

**TAB-TSK-10** The Tasks screen shall use frosted glass cards on a cosmic gradient background, matching the mock-up (glass panels, subtle shadows, blur).

**TAB-TSK-11** Each quadrant shall have a distinct accent colour used for its header underline, task card left border (4px), and small chips:

• Important & Urgent (IU): #7EA6FF

• Important (I): #86F5E0

• Urgent (U): #8EC8FF

• Neither (NI/NU): #A9B1E0

**TAB-TSK-12** The above colours shall be exposed as CSS variables: --quad-iu: #7ea6ff; --quad-i: #86f5e0; --quad-u: #8ec8ff; --quad-nu: #a9b1e0; and applied consistently to headers, borders, and badges.

**TAB-TSK-13** Task cards shall default to a frosted card style (rounded corners, thin 1px border var(--line)); category chips use the task’s category palette; the quadrant accent colour appears as a left border.

**TAB-TSK-14** Overdue tasks shall show a subtle red dot indicator next to the due date; Today shall show a small filled circle in the quadrant accent colour.

### Task Cards

**TAB-TSK-15** Each task card shall show: Title (mandatory), Category colour chip, Tags (chips), Due date (optional) aligned to the right.

**TAB-TSK-16** Expanding a task shall reveal: Description (sanitised Markdown), Subtasks (checkbox list), References (linked Brain Dump), History (created/edited/completed), and an XP panel (see XP logic).

**TAB-TSK-17** Subtasks shall be collapsible; completing a parent with outstanding subtasks shall prompt either “Complete all subtasks” or “Complete parent only” (configurable default).

**TAB-TSK-18** Task badges may include: Overdue, Today, Snoozed, Recurring; badges appear as small rounded chips below the title.

### Quadrants vs Backlog (semantics)

**TAB-TSK-19** “Neither important nor urgent” quadrant shall represent tasks expected to be done in the near future, but not currently urgent or important.

**TAB-TSK-20** Backlog shall be strictly for unscheduled/parked ideas and long-term items; moving a task from Backlog into any quadrant marks intent to execute.

**TAB-TSK-21** Users shall be able to move tasks between any quadrant and the Backlog by drag-and-drop (desktop/tablet) or via an action sheet (mobile).

### Interactions & Feedback

**TAB-TSK-22** Completing a task shall use a checkbox/tap; the card shall animate, fade, and move to a collapsible Done section within its container (quadrant or backlog).

**TAB-TSK-23** Toast feedback shall confirm actions: created, updated, completed, moved, deleted, imported, exported (bottom-center on desktop, bottom-full-width on mobile).

**TAB-TSK-24** Inline validation errors (e.g., invalid date) shall appear under the field with accessible text and shall be announced to assistive tech.

### Gamification — XP Logic & Display

**TAB-TSK-25** Base XP awards on completion shall be:

• Important & Urgent (IU): 3 XP

• Important (I): 2 XP

• Urgent (U): 1 XP

• Neither (NI/NU): 1 XP

**TAB-TSK-26** A punctuality bonus of +1 XP shall be awarded when a task with a due date is completed on or before the due date.

**TAB-TSK-27** The XP panel inside an expanded task shall show: Base XP by quadrant, any punctuality bonus, and the total XP awarded for completion.

**TAB-TSK-28** On completion, a toast shall show “+{xp} XP — Task completed!” and the navbar XP counter shall increment with a short pulse animation.

**TAB-TSK-29** Confetti shall trigger on Important and Important & Urgent completions; Important & Urgent shall trigger a denser/longer burst. Reduced motion shall replace confetti with a success outline + toast.

**TAB-TSK-30** Haptics (mobile): light vibration for 1 XP total; medium for 2–3 XP; strong triple buzz for 4+ XP (i.e., IU with punctuality bonus).

**TAB-TSK-31** A small circular XP badge may be displayed on the right edge of a task card immediately after completion for 2 seconds showing the awarded total (e.g., “+3”).

### Mobile Behaviour (small screens)

**TAB-TSK-32** Layout shall stack in this order: Buttons row → quadrants (IU, I, U, NI/NU) → Backlog list.

**TAB-TSK-33** A floating action button (FAB) shall appear at bottom-right for [Add Task]; on scroll down it hides; on scroll up it reappears.

**TAB-TSK-34** Toolbar actions (Bulk Edit, Filter, Sort, Import, Export, Toggle Completed) shall collapse into an overflow menu (⋯) anchored under a single toolbar button.

**TAB-TSK-35** Drag-and-drop shall be replaced with long-press (≥350 ms) opening an action sheet with options: Move to [IU | I | U | NI/NU | Backlog], Edit, Complete, Delete.

**TAB-TSK-36** Swipe gestures shall be available on task rows: swipe left reveals quick actions (Edit, Delete, Move); swipe right toggles complete/incomplete with haptic.

**TAB-TSK-37** Touch targets shall be at least 48×48 px; primary buttons and checkboxes shall respect this minimum on all devices.

**TAB-TSK-38** The Backlog header shall be sticky while scrolling the Backlog list; the sticky area shall include a mini Filter + Sort for Backlog-only operations.

**TAB-TSK-39** Completed tasks shall be collapsed by default; a “Show Done” toggle within each quadrant and Backlog expands the list with lazy rendering to preserve performance.

**TAB-TSK-40** Reduced-motion mode shall disable FAB hide/reveal animation and replace panel transitions with instant state changes.

### Accessibility

**TAB-TSK-41** The quadrant grid shall use role="grid" with aria-labelledby pointing to each quadrant title; Backlog shall use role="list"; each task card shall use role="listitem".

**TAB-TSK-42** The Done section toggles shall expose aria-expanded (true/false) and aria-controls pointing to the container they reveal/hide.

**TAB-TSK-43** Keyboard navigation shall support: Arrow keys move focus across quadrants; Tab cycles within a quadrant; Enter expands/collapses a task; Space toggles completion on the focused task.

**TAB-TSK-44** Screen reader output shall include title, due date status (overdue/today/none), category, quadrant name, and completion state; on completion it shall announce the XP awarded (e.g., “Task completed. Three XP gained.”).

**TAB-TSK-45** Colour contrast of text, borders, and chips against background shall meet WCAG 2.1 AA (≥4.5:1 for body text, ≥3:1 for large/bold headers); quadrant accent colours shall be paired with sufficient contrast for borders/chips.

**TAB-TSK-46** Reduced motion preference shall disable confetti and non-essential transitions; success feedback shall still include toast and a static success outline for discoverability.

### Import/Export & Persistence

**TAB-TSK-47** Import shall place items into Backlog by default unless a quadrant is specified in the payload; if both quadrant and due date are provided and conflict with filters, the UI shall still respect the stored quadrant and due date.

**TAB-TSK-48** Export shall preserve quadrant placement, position order within each quadrant/backlog, IDs, timestamps, tags, and all metadata; exports shall allow scope selection (All / Visible / Backlog-only / Specific quadrants).

**TAB-TSK-49** Round-trip import/export shall be lossless: IDs, timestamps, tags, quadrants, and subtask relationships shall remain unchanged; a checksum of the exported payload may be generated to verify re-import integrity.

### Performance & States

**TAB-TSK-50** Initial load of the Tasks tab shall render the grid skeletons (4 placeholders) and a Backlog skeleton, then hydrate with data; total time-to-interaction shall remain under target defined in performance section of the spec.

**TAB-TSK-51** Infinite lists (Backlog and Done sections) shall virtualise long lists to maintain smooth scrolling on mobile (60 FPS target where possible).

**TAB-TSK-52** Offline state shall be visually indicated (e.g., offline badge in toolbar); all task operations remain available and persist locally; any sync (if enabled) is queued for later.

### Error Handling & Validation

**TAB-TSK-53** Inline validation shall be placed directly below the offending field, with concise human-readable messages and programmatic error linking via aria-describedby.

**TAB-TSK-54** Conflicting actions (e.g., moving a task that no longer exists) shall show a non-blocking toast error and auto-refresh the view to reconcile state.

### Wireframes & Screenshots

**TAB-TSK-55** Insert desktop screenshot showing: Buttons row; 2×2 quadrants (Important & Urgent, Important, Urgent, Neither important nor urgent) with quadrant accent colours and card borders; Backlog list beneath with columns “task name | tags | due date”.

**TAB-TSK-56** Insert mobile screenshots showing: stacked quadrants with accent colours; FAB (Add Task) visible; overflow toolbar menu (⋯); action sheet after long-press on a task; Backlog with sticky header and mini Filter/Sort.

**TAB-TSK-57** Insert expanded task card screenshot showing: sanitised Markdown description; subtasks with checkboxes; references (linked Brain Dump); history; XP panel displaying base XP by quadrant, punctuality bonus, and total XP awarded.

### Display & Colour System (reference variables)

**TAB-TSK-58** The following CSS variables shall be defined and used consistently across the Tasks tab:

--quad-iu: #7ea6ff; /\* Important & Urgent accent for headers, borders, chips \*/

--quad-i: #86f5e0; /\* Important accent \*/

--quad-u: #8ec8ff; /\* Urgent accent \*/

--quad-nu: #a9b1e0; /\* Neither important nor urgent accent \*/

--line: rgba(255,255,255,0.18); /\* card borders per mock-up \*/

--text: #ffffff; /\* primary foreground on dark glass \*/

--muted: rgba(255,255,255,0.72); /\* secondary text \*/

**TAB-TSK-59** Task cards shall display a 4px left border in the quadrant’s accent colour; quadrant headers shall show a matching underline; small chips (counts/badges) shall also use the accent colour background with readable foreground.

### Gamification — XP Logic & Display (reference summary)

**TAB-TSK-60** Base XP on completion shall be: IU=3; I=2; U=1; NI/NU=1; punctuality bonus = +1 XP when completed on or before due date.

**TAB-TSK-61** On completion, toast shall show “+{xp} XP — Task completed!”; navbar XP counter shall increment with a brief pulse; for reduced motion users, only the toast and pulse-less counter increment shall be used.

**TAB-TSK-62** Confetti shall trigger on completion for Important and Important & Urgent; IU uses a denser/longer burst; reduced motion disables confetti automatically; haptics pattern shall be: 1 XP = light pulse; 2–3 XP = medium; 4+ XP = strong triple buzz.

## 21.2 Routines Tab

Purpose: Guide users through step-by-step sequences (“routines”), with a focused runner view, timers, progress feedback, and clear completion summaries. Visual style matches Stellar-Journey’s glassmorphic cards over the cosmic gradient seen in the mock-ups.

### Overall Structure

**TAB-RTN-01** The Routines screen shall show (top to bottom): a toolbar (Routine Management Buttons), a “Current Routine” runner panel (if a routine is in progress), and a Library grid/list of saved routines.

**TAB-RTN-02** The visual design shall use frosted glass cards with subtle shadows and thin borders (border: 1px var(--line)), white/ink text (var(--ink)), and dim secondary text (var(--dim)), consistent with styles.css.

**TAB-RTN-03** The “Current Routine” runner shall feature a prominent progress indicator (horizontal progress bar at top of the card) and a step triptych: Previous (dim), Current (enlarged with glow/accent), Next (smaller preview).

**TAB-RTN-04** Routine Library entries shall render as cards showing: Title, Tags, Version, Estimated Duration, and Last Used, with a primary button [Start] and a secondary menu [⋯] for Edit, Duplicate, Delete, Export.

### Routine Management Buttons (toolbar)

**TAB-RTN-05** The toolbar shall include: [New Routine], [Import], [Export], [Filter], [Sort], [Toggle Completed], [View: Grid/List].

**TAB-RTN-06** Filter shall include by Tag, Duration range, Last Used, and Energy tags (e.g., “low energy”).

**TAB-RTN-07** Sort shall include by Title, Last Used, Duration, Recently Edited.

**TAB-RTN-08** “New Routine” opens a create dialog: Title (required), Tags (chips), Estimated Duration (auto/override), Steps table (label + timer duration), Optional Energy tag, Optional Brain Dump link.

### Runner — Layout & Display

**TAB-RTN-09** The runner card shall present the global progress bar along the top edge; underneath, a 3-panel step triptych aligned horizontally: Previous (left, 60–70% scale, 50% opacity), Current (center, 100% scale, glow ring), Next (right, 85% scale, 80% opacity).

**TAB-RTN-10** The Current step panel shall display: Step Title (bold), Countdown timer (mm:ss), Optional Description (sanitised Markdown), and Controls row.

**TAB-RTN-11** Controls row (Current step) shall include: [Complete], [Skip], [Pause/Resume], [Reorder], [Brain Dump], [Cancel], with accessible labels and large touch targets (≥48×48 px).

**TAB-RTN-12** A compact log strip may appear under the controls, appending each completed/skipped step with timestamp and actual duration.

### Runner — States & Interactions

**TAB-RTN-13** Pressing [Complete] shall mark the step done, log the timestamp and duration, award XP per the XP logic, and advance to the next step.

**TAB-RTN-14** Pressing [Skip] shall mark the step skipped (no base XP; see XP logic), and advance; the log notes the skip reason if provided.

**TAB-RTN-15** [Pause/Resume] shall freeze/unfreeze the step countdown and visibly change the timer state; the global progress bar should also pause visually (striped or dimmed).

**TAB-RTN-16** [Reorder] shall enable drag handles on the steps list (drawer or overlay) to move steps; the runner rebinds to the new order without losing history.

**TAB-RTN-17** [Brain Dump] shall open a linked Note (or create a new one) in a side panel or overlay; closing returns to the runner without data loss.

**TAB-RTN-18** [Cancel] shall prompt a confirmation dialog (“Keep partial progress?” Yes/No); Yes preserves logs and partial XP; No discards the in-progress run.

### Steps List & Editing

**TAB-RTN-19** Each step shall have attributes: Label (text), Timer (seconds / mm:ss), Optional Description (Markdown), Optional Energy hint (text/chip).

**TAB-RTN-20** The steps list (editor mode) shall support: Add step, Duplicate step, Delete step, and Drag to reorder (desktop/tablet); on mobile, long-press opens a reorder sheet.

**TAB-RTN-21** Step timers shall accept values from 10s to 2h; invalid values show inline error messages and are prevented from saving.

### XP Logic & Display (Routines)

**TAB-RTN-22** Base step XP shall be 1 XP when a step is completed.

**TAB-RTN-23** On-time step bonus shall be +1 XP if the step is completed with time remaining (timer not expired); if paused, only actual active time counts.

**TAB-RTN-24** Skipped steps shall not earn base XP nor on-time bonus; if “Skip with Reason” is used, the log records the reason for later review in Statistics.

**TAB-RTN-25** Routine completion bonus shall be 2 XP plus floor(total\_steps / 5), capped at +4 XP; example: 1–5 steps = +2 XP, 6–10 steps = +3 XP, 11–15 steps = +4 XP.

**TAB-RTN-26** Perfect routine bonus shall be +2 XP if all steps in the routine were completed on-time (each step earned its on-time bonus).

**TAB-RTN-27** The runner shall show a small XP tally area near the timer that updates on each completion (“+1 XP”, “+2 XP”), and a total XP counter for the current run (e.g., “Run XP: 9”).

**TAB-RTN-28** On final completion, a summary modal shall display: Steps completed/skipped, On-time percentage, Total XP (sum of step base + step bonuses + routine bonus [+ perfect bonus if any]).

**TAB-RTN-29** Confetti shall trigger on routine completion; confetti density shall scale slightly with On-time percentage; reduced motion replaces confetti with a success glow and toast.

**TAB-RTN-30** Haptics (mobile): Short pulse per completed step; stronger double pulse for on-time bonus; strong triple buzz for final completion and bonus awards.

### Completion Summary

**TAB-RTN-31** The summary modal shall include: Routine Title, Duration planned vs actual, Steps completed/skipped list, On-time %, XP breakdown (base steps, on-time bonuses, routine bonus, perfect bonus), and actions [Save as Template], [View Log], [Close].

**TAB-RTN-32** “Save as Template” shall capture the current routine configuration (including any re-ordered steps) into the Library; logs are not included in templates.

### Display & Colour System

**TAB-RTN-33** The runner’s global progress bar shall use the app accent color (e.g., var(--mint)) and animate smoothly (var(--ease), var(--dur)); background remains glass with blur (var(--glass-hi/lo)).

**TAB-RTN-34** The Current step card shall have a subtle glow ring or 4px left border using the accent color; Previous is dimmed (opacity ~0.5), Next is slightly smaller (85–90% scale).

**TAB-RTN-35** Buttons shall be pill-shaped with hover/active transitions, using the existing button styles in styles.css; destructive actions (Cancel) shall use a caution style (e.g., border with higher contrast).

**TAB-RTN-36** Secondary text (timers, hints) shall use var(--dim); borders use var(--line); primary text uses var(--ink).

### Mobile Behaviour (small screens)

**TAB-RTN-37** The runner shall occupy most of the viewport in a single column; the step triptych becomes a horizontal swipe view: (Previous ←) [Current] (→ Next).

**TAB-RTN-38** Controls shall be shown as a sticky bottom bar with large buttons: Complete, Pause/Resume, Skip; overflow menu (⋯) contains Reorder, Brain Dump, Cancel.

**TAB-RTN-39** Swiping shall be disabled while the action sheet is open; gestures shall not interfere with Pause/Resume and Complete buttons (priority capture).

**TAB-RTN-40** Long routines (≥20 steps) shall virtualise the steps list to maintain smooth scrolling in the editor; the runner view always focuses the Current step.

**TAB-RTN-41** Haptics shall fire on key interactions: Complete (short), Skip (short double), Pause/Resume (subtle), Final completion (strong).

### Accessibility

**TAB-RTN-42** The global progress bar shall use role="progressbar" with aria-valuenow, aria-valuemin=0, aria-valuemax=100; text label announces “Routine progress X%”.

**TAB-RTN-43** The countdown timer shall update an aria-live="polite" region once per second (or less when reduced motion is enabled); do not spam the live region on every tick when screen readers are active (throttle announcements).

**TAB-RTN-44** All runner controls shall have explicit aria-labels and visible focus outlines; keyboard shortcuts: Space = Complete, P = Pause/Resume, S = Skip, R = Reorder, N = Brain Dump, Esc = Cancel (with confirmation).

**TAB-RTN-45** Reduced motion shall disable animated scale/glow on Current step and replace with a high-contrast border and static state changes; confetti disabled, toast retained.

**TAB-RTN-46** Colour contrast for all text and controls shall meet WCAG 2.1 AA; timers and diminutive labels must remain ≥4.5:1 against the background.

### Import/Export & Persistence

**TAB-RTN-47** Export of a routine shall include: id, title, tags, version, estimated duration, steps (label, seconds, description), and energy tag; export may exclude run logs by default.

**TAB-RTN-48** Import shall validate schema and on success place the routine in the Library; conflicting ids shall be re-id’d while preserving step order.

**TAB-RTN-49** Round-trip import/export shall be lossless for definitions (not logs): IDs may be regenerated on import if collision occurs, but internal step order and fields must be preserved exactly.

**TAB-RTN-50** Starting a run shall persist an in-progress state so that accidental refresh/close will allow “Resume run?” on next open; the in-progress state includes current step index, remaining seconds, and partial logs.

### Performance & States

**TAB-RTN-51** Initial Routines load shall show skeleton cards for Library and a skeleton runner if a routine is in progress; hydration fills content once data is ready.

**TAB-RTN-52** The runner shall maintain smooth updates (target 60 FPS where possible) by limiting DOM reflows (e.g., requestAnimationFrame for timer rendering) and throttling aria-live updates.

### Error Handling

**TAB-RTN-53** If a timer is invalid or negative, the runner shall refuse to start and show an inline error under the offending step field.

**TAB-RTN-54** If a routine was edited during a run (in another tab), attempting to continue shall prompt to “Reload routine definition” (recommended) or “Continue with cached order” (advanced).

### Wireframes & Screenshots

**TAB-RTN-55** Insert desktop runner screenshot showing: global progress bar (accent color), step triptych (Prev dimmed, Current glow, Next preview), and controls row.

**TAB-RTN-56** Insert mobile screenshots showing: full-height runner with horizontal swipe between steps, sticky bottom control bar, and overflow menu (⋯).

**TAB-RTN-57** Insert completion summary modal screenshot showing: steps list with statuses, on-time %, and XP breakdown (step base, step bonuses, routine bonus, perfect bonus).

**TAB-RTN-58** Insert Library grid/list screenshots with routine cards showing Title, Tags, Version, Estimated Duration, Last Used, and [Start] / [⋯] controls.

## Habits Tab

### Purpose

Track recurring actions via a fast daily check-in UI with streak visuals (today ring + rolling heatmap), gentle gamification, and accessible interactions. Visual style follows Stellar-Journey’s glass cards over a cosmic gradient, consistent with styles.css.

### Overall Structure

**TAB-HAB-01** The Habits screen shall show (top to bottom): a toolbar (Habit Management Buttons), a “Today” panel with daily check-ins and a streak ring, and a rolling 28–35 day heatmap per habit; an optional details drawer opens on selection.

**TAB-HAB-02** The visual design shall use frosted glass cards with subtle shadows and thin borders (border: 1px var(--line)); primary text uses var(--ink), secondary text var(--dim), consistent with styles.css.

### Habit Management Buttons (toolbar)

**TAB-HAB-03** The toolbar shall include: [New Habit], [Filter], [Sort], [Toggle Paused], [Import], [Export], [View: Cards/List].

**TAB-HAB-04** Filter shall include: Category, Tag, Status (Active/Paused), Has Streak ≥ N days, Missed today.

**TAB-HAB-05** Sort shall include: Title, Longest streak, Current streak, Last check-in, Created date.

**TAB-HAB-06** “New Habit” opens a create dialog: Title (required), Category (colour), Tags (chips), Schedule (Daily / Custom days), Start date, Optional Brain Dump link.

### Habit Cards (list/grid)

**TAB-HAB-07** Each habit card shall display: Title (bold), Category colour chip, Current streak count (e.g., “12 days”), and a Today tick control.

**TAB-HAB-08** The Today tick control shall be a large circular checkbox (≥28–32 px on desktop, ≥40–48 px on mobile) with a ring animation on completion.

**TAB-HAB-09** The card shall show a compact heatmap strip (28–35 days) with squares/dots indicating completion; today’s cell is outlined.

### Today Panel & Streak Ring

**TAB-HAB-10 The** “Today” panel shall summarize: total active habits, completed today count, and a circular streak ring that fills proportionally to today’s completions.

**TAB-HAB-11** The streak ring shall display a numeric label at its center (e.g., “7/12 today”); ring uses the app accent colour (e.g., var(--mint)).

**TAB-HAB-12** Tapping a habit’s Today tick in the “Today” panel shall also tick it on its card and update the ring and counters in real time.

### Heatmap (28–35 days)

**TAB-HAB-13** The heatmap shall show one mark per day, left→right, oldest→newest; today is the rightmost cell with a distinct outline.

**TAB-HAB-14** Completion intensity shall vary by completion streak segments (e.g., 1, 3, 7, 14+) while maintaining WCAG AA contrast; reduced motion users shall see static colours without animated transitions.

**TAB-HAB-15** Vacation days (if enabled) shall render with a hatched/diagonal pattern and do not break streaks.

### States & Interactions

**TAB-HAB-16** Ticking a habit today shall animate the Today control, update the card state, update the heatmap latest cell, increment streak, and show feedback (see XP logic & toast).

**TAB-HAB-17** Undo shall be available for 5–10 seconds after ticking, via an inline “Undo” chip and a toast action; undo restores prior streak and counters.

**TAB-HAB-18** Pausing a habit shall move it to a Paused section (collapsible) and disable Today tick controls until resumed.

**TAB-HAB-19** Editing a habit opens a side drawer/modal with fields: Title, Category, Tags, Schedule, Start date, Brain Dump; changes apply immediately on save.

### XP Logic & Feedback

**TAB-HAB-20** Base daily completion XP shall be 1 XP per habit checked today.

**TAB-HAB-21** New streak threshold (first time reaching 3 consecutive days) shall award +3 XP once.

**TAB-HAB-22** Milestones shall award +2 XP at streak lengths of 7, 14, 28, 50, 100, 250, 500, 1000 days (each milestone triggers once per threshold).

**TAB-HAB-23** On completion, a toast shall appear: “Habit done. +1 XP”; on milestone days: “Milestone reached! +2 XP”; on new-streak threshold: “Streak started! +3 XP.”

**TAB-HAB-24** Confetti shall trigger on milestone days; reduced motion replaces confetti with a static celebratory outline and toast; no confetti for regular daily ticks.

**TAB-HAB-25** Haptics (mobile): short pulse for daily tick; double pulse for new-streak threshold; strong pulse for milestone.

### Detail Drawer / Habit View

**TAB-HAB-26** Opening a habit (click/tap title) shall reveal a detail drawer with: full heatmap (up to 90 days), Brain Dump link, Motivation text (optional), Vacation toggle, and a Completion history list (date/time).

**TAB-HAB-27** The Completion history list shall support filters: last 7/30/90 days, and export as CSV/Markdown.

**TAB-HAB-28** The Vacation toggle shall mark selected future dates as vacation (or set a date range); vacation days do not decrement or block streaks.

### Mobile Behaviour (small screens)

**TAB-HAB-29** The layout shall default to a vertical list; each card contains the Today tick, streak count, and a compact heatmap; tapping a card expands details inline (accordion).

**TAB-HAB-30** The toolbar actions (Filter/Sort/Import/Export) shall collapse under an overflow menu (⋯); [New Habit] appears as a bottom-right floating action button (FAB).

**TAB-HAB-31** Swipe right on a habit card toggles Today completion; swipe left reveals actions (Edit, Pause/Resume, Delete).

**TAB-HAB-32** Sticky “Today” summary (ring + counts) shall remain visible at the top while scrolling the list; ring updates smoothly after each tick.

**TAB-HAB-33** Touch targets (Today control, Edit, Pause) shall be ≥48×48 px; spacing ensures no accidental taps.

### Display & Colour System

**TAB-HAB-34** Category chips use the shared 10-colour palette; chips appear left of the title on cards; colour variables follow styles.css tokens (e.g., --cat-blue, --cat-violet).

**TAB-HAB-35** The Today tick control, when completed, shall fill with the habit’s category colour; the ring outline uses the app accent; on hover/focus the control shows a clear focus halo.

**TAB-HAB-36** Heatmap cells use a monochrome scale tinted by the habit’s category colour; contrast ratios must meet WCAG; non-colour cues (check marks or dot density) provide redundancy.

### Accessibility

**TAB-HAB-37** The Habits list shall use role="list" and each habit card role="listitem"; the Today control is a native checkbox or role="checkbox" with aria-checked.

**TAB-HAB-38** The streak ring shall be labelled with aria-label describing progress (e.g., “Today progress: 7 of 12 habits complete”); avoid rapid aria-live updates that spam screen readers.

**TAB-HAB-39** Keyboard support: Tab focuses a habit’s Today control; Space toggles completion; Enter opens the detail drawer; Arrow keys move between habits; Esc closes the drawer.

**TAB-HAB-40** Screen reader announcements on tick shall say: “<Habit title> completed today. Current streak: X days. +1 XP.”

**TAB-HAB-41** Reduced-motion users shall not see animated heatmap transitions or ring spin; updates shall be instantaneous with text/outline cues.

### Import/Export & Persistence

**TAB-HAB-42** Export shall include: id, title, category, tags, schedule configuration, start date, and a compressed completion log (dates only) for the last N days (configurable), excluding PII.

**TAB-HAB-43** Import shall validate schema; on ID collision, a new id shall be assigned while preserving title and logs; the user is notified via toast.

**TAB-HAB-44** Round-trip import/export shall preserve completion history, streak counts, and schedule configuration without mutation.

**TAB-HAB-45** Offline usage shall be fully supported; all ticks persist locally and any optional sync is queued for later.

### Performance & States

**TAB-HAB-46** Initial load shall render skeleton cards and a skeleton ring; hydration fills titles, streaks, and heatmaps; target smooth scrolling at 60 FPS where possible via list virtualisation for long habit lists.

**TAB-HAB-47** Checking a habit shall update only the affected UI elements (Today control, ring, heatmap last cell, counters) to minimize reflow.

### Error Handling

**TAB-HAB-48** If the same habit is toggled rapidly, debouncing shall prevent double entries; on conflict, UI shows a non-blocking warning toast and reconciles to the last user intent.

**TAB-HAB-49** Invalid schedule configurations (e.g., empty day set on custom schedule) shall show inline error text in the editor and prevent saving.

### Wireframes & Screenshots

**TAB-HAB-50** Insert desktop screenshots: Habits list with Today controls, streak counts, and compact heatmaps; “Today” summary ring at top.

**TAB-HAB-51** Insert mobile screenshots: vertical list with large Today controls, sticky Today ring summary, swipe actions, and expanded accordion detail.

**TAB-HAB-52** Insert detail drawer screenshot showing extended heatmap, vacation toggle, motivation text, and completion history export action.

## 21.4 Schedule Tab

Purpose: Provide a week/day planner that places Tasks, Routines, Habits and Brain Dump onto a time grid. Visual style matches Stellar-Journey’s glass cards and cosmic gradient.

### Overall Structure

**TAB-SCH-01** The Schedule screen shall show (left→right): a sidebar (agenda/shortcuts) and a main calendar area; on small screens, the layout stacks vertically (sidebar above calendar).

**TAB-SCH-02** The main calendar shall support Day view (default) and Week view; Month view may be provided as a high-level overview (non-editable).

**TAB-SCH-03** The calendar grid shall render a left time column and a right slots column; hours are divided into fixed rows.

**TAB-SCH-04** The visible hour range shall be 06:00–21:00 by default (as in the mock-up), configurable in Settings.

### Calendar Grid & Hours

**TAB-SCH-05** The grid shall use two columns: .hours { grid-template-columns: 80px 1fr } with .hour-col on the right drawing horizontal guides at each time row.

**TAB-SCH-06** Each hour row shall be 46px tall (per styles.css .hour-col rows: repeat(16, 46px)), with a dashed bottom border (var(--line)).

**TAB-SCH-07** Time labels in the left column shall show “06:00, 07:00, … 21:00” in dim text (var(--dim)), aligned to the corresponding row.

**TAB-SCH-08** A “now” marker (thin line) shall be displayed if the current time is within the visible range; scrolled into view on load.

### Blocks (Events)

**TAB-SCH-09** Scheduled items shall render as absolutely positioned .block elements inside .slots, with top = pixels from 06:00 and height proportional to duration.

**TAB-SCH-10** Blocks shall show a title (top-left, bold) and a meta line (bottom-left) of either “HH:MM–HH:MM” or a single time for punctual tasks.

**TAB-SCH-11** Routine blocks shall include an internal progress bar near the bottom of the block: .progress { height: 6px } with a gradient fill (.progress i) using #8ec8ff → #86f5e0 as in styles.css.

**TAB-SCH-12** Task blocks shall use the .block.task style (lighter purple-tinted glass with border-color rgba(200,180,255,.24)); other block types use the default .block style (blue-tinted glass).

**TAB-SCH-13** Overlapping blocks shall shift horizontally (smart offset) to keep titles readable; the rightmost block can overlay with increased z-index when hovered/focused.

### Creating & Editing Events

**TAB-SCH-14** New events shall be created only from existing entities: Task, Routine (sequence), Habit (reminder), Note (reference).

**TAB-SCH-15** Users shall be able to create an event by: (a) dragging on the grid to select a time span (desktop/tablet), (b) tapping a “+” in the hour gutter (mobile), or (c) converting an entity via “Schedule” action.

**TAB-SCH-16** Resizing an event shall be done by dragging the bottom edge; moving an event shall be done by dragging the block vertically (and horizontally into another day in Week view).

**TAB-SCH-17** When an event is moved/resized, a snap-to grid of 15 minutes shall apply (configurable), with a visual ghost preview.

### Event Details & Source Link

**TAB-SCH-18** Clicking a block shall open a compact details popup anchored to the block: title, type (Task/Routine/Habit/Note), category chips, time span, and source link [Open source].

**TAB-SCH-19** The details popup shall provide actions: Edit time, Change day, Duplicate, Delete, and for Routines: [Start in Runner]; for Tasks: [Mark done]; for Habits: [Mark today].

TAB-SCH-20 The [Open source] action shall navigate to the originating entity in its tab, setting an internal navigation flag that suppresses the beforeunload export prompt (as in schedule.html’s internal-nav logic).

### Suggestions & Conflicts

**TAB-SCH-21** The planner shall propose the next free slot for a selected entity, ensuring a default 15-minute break between items; user can override.

**TAB-SCH-22** Conflicting events shall be indicated with a subtle red edge on both blocks and an inline message in the popup, offering “Keep overlap” or “Shift to next free slot”.

### Runner Overlay (for Routines)

**TAB-SCH-23** Starting a Routine from the calendar shall open the Runner overlay (full-screen glass panel) centered on the viewport (per #runner in schedule.html).

**TAB-SCH-24** The Runner overlay shall display the Routine title, elapsed/remaining time, a horizontal progress bar (accent var(--mint)), and controls [Start/Pause/Stop/Skip/Close] with icons (per mock-up).

**TAB-SCH-25** The Runner overlay shall be dismissible via [Close] or Esc; dismissing pauses the routine unless explicitly Stopped.

**TAB-SCH-26** On Runner progress, the corresponding calendar block’s .progress i width shall update (e.g., style="width:30%") live; reduced motion throttles visual updates.

### Display & Colour System

**TAB-SCH-27** The calendar container shall use a glass panel background (rgba(16,20,44,.30)) with a 1px border var(--line) and 14px radius (styles.css .calendar).

**TAB-SCH-28** Blocks (.block) shall use the blue glass background with border rgba(150,200,255,.24); task blocks (.block.task) shall use a purple glass background with border rgba(200,180,255,.24).

**TAB-SCH-29** Titles shall use var(--ink), meta lines use var(--dim); focus outlines shall use var(--mint) on keyboard focus (matching global focus style).

**TAB-SCH-30** The internal progress gradient shall be exactly linear-gradient(90deg, #8ec8ff, #86f5e0) as in styles.css .progress i.

### Toolbar & Sidebar

**TAB-SCH-31** The top toolbar shall include: [Today], [Previous], [Next], [View: Day/Week/Month], [Filter], [Export], [Import].

**TAB-SCH-32** [Filter] shall filter by Type (Task/Routine/Habit/Note), Category, Tag, and Status (Completed/Active); filters persist per session.

**TAB-SCH-33** The sidebar shall list the day’s items in chronological order (compact agenda), with quick actions (open source, start runner, mark done); clicking an agenda item scrolls the grid to its block.

### Mobile Behaviour (small screens)

**TAB-SCH-34** The layout shall stack vertically: toolbar → agenda (sidebar) → calendar grid; the agenda becomes the default view for narrow screens (< 768px).

**TAB-SCH-35** Creating an event on mobile shall use a “+ Add at…” sheet offering common start times (Now, +30m, +1h, Evening) and an option “Pick time…”.

**TAB-SCH-36** Blocks shall be draggable only vertically on mobile; resizing uses a handle with increased hit area; all controls must meet ≥48×48 px touch targets.

**TAB-SCH-37** The details popup shall present as a bottom sheet with large buttons; swipe down to dismiss; [Start in Runner] is sticky and prominent for Routines.

### Accessibility

**TAB-SCH-38** The time grid shall use role="grid"; hour labels act as row headers; blocks are focusable elements with aria-label “<Title>, <Type>, <Start–End>”.

**TAB-SCH-39** Keyboard: Arrow keys move the time focus by 15 minutes; Shift+Arrow extends the selection; Enter opens the details popup; Delete removes the selected event.

**TAB-SCH-40** Focus outlines for blocks and toolbar items shall use the global focus style (outline: 3px var(--mint)); popup sheets shall be aria-modal with focus trap.

**TAB-SCH-41** Reduced motion: disable animated block transitions and progress bar easing; instead, update progress steps discretely; screen readers receive polite announcements for schedule changes.

### Import/Export & Persistence

**TAB-SCH-42** Export shall include a “ScheduleEvent” array with: id, type (task/routine/habit/note), ref (entity id), day (YYYY-MM-DD), start (HH:MM), dur\_min, and optional meta; import shall validate schema and re-link to existing entities by id.

**TAB-SCH-43** On import, if a referenced entity is missing, the event is imported in a “detached” state with a warning badge and a toast linking to remediation (“Map to existing…”).

**TAB-SCH-44** Round-trip import/export shall preserve positions, durations, and links to entities; when a collision occurs (same id and time), user chooses Replace, Duplicate, or Skip.

### Performance & States

**TAB-SCH-45** Initial load shall render skeleton hours and a skeleton block list; hydration applies .block elements; target smooth scroll/drag at 60 FPS where possible.

**TAB-SCH-46** Virtualize long days (many blocks) to maintain performance; only visible blocks are mounted; off-screen blocks mount on scroll.

**TAB-SCH-47** Offline operation: creating/moving/resizing events shall persist locally; if sync is enabled, changes queue for later; an “Offline” badge appears near the toolbar.

### Error Handling

**TAB-SCH-48** Invalid times (end before start) shall be prevented and flagged inline; snapping corrects minor overlaps if user confirms.

**TAB-SCH-49** If an entity is deleted while its event is open, closing the popup shall show a non-blocking toast and remove or detach the block as appropriate.

### Wireframes & Screenshots

**TAB-SCH-50** Insert desktop Day view screenshot showing: hour grid (06:00–21:00), a Routine block with internal progress bar, and a Task block; include overlapping example.

**TAB-SCH-51** Insert Runner overlay screenshot showing title, elapsed/remaining, progress, and Start/Pause/Stop/Skip/Close controls.

**TAB-SCH-52** Insert mobile screenshots showing: agenda list (default), bottom-sheet event details with large actions, and drag-resize handles on a block.

## Brain Dump Tab

### Purpose

Provide a fast, secure Markdown note-taking interface with split Editor/Preview (desktop), Daily Brain Dump, cross-references (^ref), local image attachments (OPFS), and safe rendering (sanitised).

### Overall Structure

**TAB-BDP-01** The Brain Dump screen shall present (top to bottom): a toolbar (Note Management Buttons), a sidebar list of notes (filterable), and a main area with Editor/Preview; on mobile, the list and editor/preview are separate views with a toggle.

**TAB-BDP-02** The visual design shall use frosted glass cards on a cosmic gradient background, thin borders (1px var(--line)), primary text var(--ink), secondary text var(--dim), consistent with styles.css.

### Note Management Buttons (toolbar)

**TAB-BDP-03** The toolbar shall include: [New Note], [New Daily], [Import], [Export], [Search], [Filter], [Sort], [Toggle Preview], [Pin/Unpin], [Delete], and a “⋯” overflow for less-used actions (Duplicate, Convert to Template).

**TAB-BDP-04** “Search” shall search titles, body text (Markdown), tags, and refs; results highlight terms within the list and optionally in preview.

**TAB-BDP-05** “Filter” shall include: Tag(s), Date range, Pinned only, Has attachments, Has backlinks, Edited recently.

**TAB-BDP-06** “Sort” shall include: Last edited (desc), Created (desc), Title (A–Z), Most backlinks, Most words.

### Sidebar List

**TAB-BDP-07** The sidebar shall display note rows with: Title (bold), snippet (first line or H1), tags (chips), last edited timestamp; selected row uses a subtle accent border/indicator.

**TAB-BDP-08** The sidebar shall support infinite/virtualized scrolling for large collections; keyboard up/down navigates notes, Enter opens in editor.

### Main Area Layout

**TAB-BDP-09** Desktop default shall be split view: Editor (left) and Preview (right) with a draggable splitter; toggle allows Editor-only or Preview-only modes.

**TAB-BDP-10** Tablet shall default to Editor with a Preview toggle button; mobile shall switch between List and Editor/Preview via a segmented control at the top of the page.

**TAB-BDP-11** The main area shall use a glass card container with padding 16–24px; focus outlines use the global focus style (3px var(--mint)).

### Editor

**TAB-BDP-12** The editor shall be a plain-text Markdown editor with monospaced font option, line numbers (optional), soft-wrap, and a large caret; supports indentation with Tab/Shift+Tab for lists.

**TAB-BDP-13** The editor toolbar shall provide: Bold (Ctrl/Cmd+B), Italic (Ctrl/Cmd+I), Link (Ctrl/Cmd+K), Unordered List, Ordered List, Checklist, Blockquote, Inline Code, Code Block (```), Table, Insert Date/Time, Insert Ref (^ref), Undo/Redo.

**TAB-BDP-14** Inserting a link shall open a small inline dialog for URL and text; URLs are validated and normalised; javascript: and data: schemes (except data:image/\* local attachments) are blocked.

**TAB-BDP-15** Inserting a table shall add a 3×3 Markdown table scaffold; Tab navigates cells; Shift+Tab moves reverse; Enter inserts a new row.

**TAB-BDP-16** Insert Date/Time shall insert ISO/local formatted text (configurable) at the caret.

**TAB-BDP-17** Insert Ref (^ref) shall prompt for an entity id (Task/Routine/Habit/Note); accepted values insert ^ref(<id>); invalid ids show inline error.

### Preview (Sanitised Renderer)

**TAB-BDP-18** The Preview shall render GitHub-flavoured Markdown (GFM), with raw HTML disabled and sanitised; on\* handlers, scripts, iframes, and javascript: URLs are removed.

**TAB-BDP-19** Headings, lists, code blocks, tables, checklists, links, images, and blockquotes shall style per styles.css (consistent with site typography).

**TAB-BDP-20** Checklists in Preview shall be interactive when “Preview is interactive” setting is on; otherwise, they render read-only.

### Daily Brain Dump

**TAB-BDP-21** “New Daily” shall create a note titled with the current date (YYYY-MM-DD or localized), pre-seeded with a template (configurable) and tags (#daily).

**TAB-BDP-22** Opening today’s date shall reuse the existing Daily note (if present) rather than creating duplicates; a toast indicates reuse.

### Tags & Metadata

**TAB-BDP-23** Tags shall be chips under the title bar; adding a tag via the toolbar or typing #tag in the editor updates metadata.

**TAB-BDP-24** Pinned notes shall appear at the top of the sidebar with a pin icon; Pin/Unpin is available in toolbar and via right-click/context menu.

### Attachments (Local Images & Files via OPFS)

**TAB-BDP-25** Dragging an image/file into the editor shall prompt to “Attach locally”; upon confirm, the file is saved in OPFS and a Markdown link is inserted (e.g., ![alt](/opfs/notes/<id>/<filename>)).

**TAB-BDP-26** The Insert Image dialog shall enforce alt text; missing alt inserts placeholder “[image]” and flags a warning until edited.

**TAB-BDP-27** Attached files shall be listed in a right-side panel (toggleable), with options to rename, remove, or copy link; deleting a note prompts to also delete its attachments.

### Cross-References & Backlinks

**TAB-BDP-28** Typing ^ref(<id>) shall create a cross-reference; valid ids become clickable in Preview (and in Editor via Ctrl/Cmd+Click).

**TAB-BDP-29** Backlinks panel shall list all notes that reference the current note; selecting a backlink navigates to that note and highlights the reference occurrence.

**TAB-BDP-30** Hovering a ^ref link in Preview shall show a small tooltip preview (title + first line) of the referenced entity; keyboard users can open the same preview via a focusable icon.

### Security & Safety

**TAB-BDP-31** The renderer shall block raw HTML; only Markdown is allowed; embedded links must be https or relative; data: URLs allowed only for local images from OPFS.

**TAB-BDP-32** Pasted content shall be sanitised; if HTML is detected, it shall be converted to Markdown where possible and stripped of active content; a discreet toast states “HTML sanitised”.

### Autosave & Version History

**TAB-BDP-33** Brain Dump shall autosave on pause typing (≈500–800 ms) and on blur; a “Saved” indicator appears in the toolbar.

**TAB-BDP-34** Version history shall store lightweight snapshots; “View Changes” shows a side-by-side diff (added/removed lines) with timestamps and an option to restore a version.

**TAB-BDP-35** Restoring a version shall create a new snapshot before replacing the current content, enabling undo.

### Search & Find-in-Note

**TAB-BDP-36** Global Search (toolbar) shall filter the sidebar list live; Enter focuses the first result; Esc clears.

**TAB-BDP-37** Find-in-Note (Ctrl/Cmd+F) shall highlight matches in the editor and optionally in Preview; Find Next/Prev supported; case/regex toggles optional.

### Mobile Behaviour (small screens)

**TAB-BDP-38** Mobile default shall show the sidebar list; tapping a note transitions to the Editor; a top segmented control toggles Editor/Preview; back button returns to list.

**TAB-BDP-39** The editor toolbar shall collapse into a scrollable row of icons; less-used actions appear under an overflow (⋯) menu.

**TAB-BDP-40** Long-press on a word/selection shall open a selection menu: Bold, Italic, Link, Checklist item, Code; copy/paste use native system menus.

**TAB-BDP-41** Touch targets (toolbar icons, chips) shall be ≥48×48 px; sticky Save indicator appears at the top when autosave completes.

### Display & Typography

**TAB-BDP-42** Headings shall scale responsively (H1 > H2 > H3), with comfortable line-height; code blocks use a monospaced font and glass card background with subtle border.

**TAB-BDP-43** Links in Preview shall underline on hover and show a focus outline; external links open in a new tab with rel="noopener".

**TAB-BDP-44** The editor caret and selection colours shall maintain sufficient contrast; focus-visible outlines appear on all inputs and buttons.

### Accessibility

**TAB-BDP-45** The editor shall expose role="textbox" with aria-multiline="true"; the Preview region shall be labelled and not steal focus unexpectedly.

**TAB-BDP-46** All toolbar buttons shall have aria-labels and visible text/tooltips; keyboard shortcuts shall be documented in a help tooltip (Shift+/).

**TAB-BDP-47** The attachments panel and backlinks panel shall be navigable via keyboard with logical tab order; dismissable via Esc.

**TAB-BDP-48** Reduced motion shall disable animated transitions for splitter drag, preview fade-ins, and diff animations; rely on static state changes.

### Import/Export & Persistence

**TAB-BDP-49** Export shall support: single note (Markdown), selected notes (zip of .md files + metadata.json), or all notes; attachments export as files in structured folders.

**TAB-BDP-50** Import shall accept: .md files (one per note), .zip (notes + metadata), or .html (converted to Markdown with sanitisation); on ID collision, new IDs are generated but backlinks are remapped.

**TAB-BDP-51** Round-trip import/export shall preserve titles, tags, ^ref links (rewritten if IDs changed), and attachments; a migration report lists any remapped IDs.

### Performance & States

**TAB-BDP-52** The sidebar list shall virtualise items for large collections; editing large notes shall avoid layout thrash by batching updates and debouncing preview rendering.

**TAB-BDP-53** Preview rendering shall be incremental: re-render only the changed blocks where possible; show a lightweight skeleton or “Rendering…” chip when needed.

### Error Handling

**TAB-BDP-54** Failing to save to OPFS/IndexedDB shall show a persistent warning banner with retry; the system shall buffer edits in memory until storage is available or user exports.

**TAB-BDP-55** If an attachment write fails, the UI shall show a clear error and offer “Retry” and “Save elsewhere” (download file).

### Optional Enhancements (Non-blocking)

**TAB-BDP-56** Quick Brain Dump mode shall allow a small, transient note composer with minimal chrome; promote to full note moves it to the main list with tags.

**TAB-BDP-57** Templates for notes shall be stored in the Library and insertable via toolbar; placeholders like {{date}} and {{title}} auto-resolve on insert.

**TAB-BDP-58** Word count and reading time shall appear in the footer; togglable in Settings.

### Wireframes & Screenshots

**TAB-BDP-59** Insert desktop screenshots: split Editor/Preview with toolbar, backlinks panel open, attachments panel open.

**TAB-BDP-60** Insert mobile screenshots: list view, editor view with collapsed toolbar, preview view; show long-press selection menu.

**TAB-BDP-61** Insert version history/diff screenshot and an attachments management screenshot (rename/remove/copy link).

## Library Tab

### Purpose

Provide a central place to manage reusable templates for Tasks and Routines. Templates speed up repeated creation and encourage consistency.

### Overall Structure

**TAB-LIB-01** The Library screen shall present (top to bottom): a toolbar (Library Management Buttons) and a main panel of Template cards in a grid layout (desktop/tablet) or list layout (mobile).

**TAB-LIB-02** The visual design shall use frosted glass cards (rounded 14–16px corners, 1px var(--line) border, subtle shadow var(--shadow)) over a cosmic gradient background (per styles.css). Primary text uses var(--ink), secondary text var(--dim).

### Library Management Buttons (toolbar)

**TAB-LIB-03** The toolbar shall include: [New Template], [Import], [Export], [Search], [Filter], [Sort], [View: Grid/List].

**TAB-LIB-04** “Search” shall filter templates by Title, Tags, Type (Task/Routine).

**TAB-LIB-05** “Filter” shall allow filtering by: Type (Task vs Routine), Tag(s), Duration range (for Routines), Last Used, Recently Added.

**TAB-LIB-06** “Sort” shall support: Title (A–Z), Last Used (desc), Duration, Date Created.

**TAB-LIB-07** “New Template” shall open a creation dialog: select base type (Task/Routine), Title (required), Tags, Category (for Tasks), Steps (for Routines), Duration (for Routines).

### Template Cards (Grid/List)

**TAB-LIB-08** Each template card shall display: Title (bold), Type icon (Task square / Routine circular arrows), Tags (chips), Version, Duration (for Routines), and Last Used date.

**TAB-LIB-09** Action buttons on each card shall include: [Use], [Edit], [Duplicate], [Delete], and [⋯] overflow for Export.

**TAB-LIB-10** Hovering a card (desktop) or long-pressing (mobile) shall highlight it with a glow accent and reveal secondary buttons.

**TAB-LIB-11** Editing a template shall open a side drawer/modal with structured fields depending on type:

• Task template: Title, Category, Quadrant default, Tags, Due offset (optional), Brain Dump link.

• Routine template: Title, Tags, Steps list (label + duration + optional description), Energy tag, Estimated Duration.

**TAB-LIB-12** Template previews shall sanitise any embedded Markdown before rendering to avoid unsafe HTML or scripts.

### Using Templates

**TAB-LIB-13** Selecting [Use] on a template shall spawn a new Task or Routine pre-filled with the template’s fields; the spawned entity is independent from the template thereafter.

**TAB-LIB-14** For Task templates, the new Task shall appear in the selected quadrant or backlog (configurable); for Routine templates, the new Routine appears in the Library with its own ID.

**TAB-LIB-15** A toast shall confirm instantiation: “Template applied — Task created” or “Routine created”.

### Import/Export

**TAB-LIB-16** Export shall support single template (.json), multiple selected templates (.zip with JSON files), or all templates. Metadata includes id, type, title, tags, version, and fields.

**TAB-LIB-17** Import shall accept .json or .zip; schema validation required. On ID collision, the imported template shall be re-id’d with a new UUID.

**TAB-LIB-18** Round-trip import/export shall preserve all template fields, tags, and metadata without mutation.

**TAB-LIB-19** Import shall provide a dry-run preview modal showing: number of templates detected, types, titles, and warnings.

### Mobile Behaviour (small screens)

**TAB-LIB-20** The layout shall default to a vertical list of templates; toolbar actions collapse into an overflow menu (⋯); [New Template] is exposed as a floating action button (FAB) at bottom-right.

**TAB-LIB-21** Tapping a template opens a detail sheet with [Use], [Edit], [Duplicate], [Delete].

**TAB-LIB-22** Swipe left on a template row reveals quick actions (Use, Edit, Delete); swipe right pins/unpins template.

**TAB-LIB-23** Touch targets for actions shall be ≥48×48 px; cards shall use comfortable spacing to avoid accidental taps.

### Display & Colour System

**TAB-LIB-24** Template type icons shall use consistent accent colours: Tasks use var(--quad-i) mint-tinted; Routines use var(--quad-u) blue-tinted; pinned templates show a gold border.

**TAB-LIB-25** Tags are chips styled with the shared 10-colour palette. Chips shall truncate with ellipsis if overflowing.

**TAB-LIB-26** Version number shall appear as a subtle badge (small font, dim text) at bottom-right of the card.

### Accessibility

**TAB-LIB-27** The template grid shall use role="grid" and each template card role="gridcell"; list mode shall use role="list" and role="listitem".

**TAB-LIB-28** Action buttons shall have aria-labels (“Use template”, “Edit template”).

**TAB-LIB-29** Keyboard navigation: Arrow keys move between cards in grid; Enter selects and opens; Esc closes drawers/modals.

**TAB-LIB-30** Screen readers shall announce: Title, Type, Tags, Last Used date, and Version.

**TAB-LIB-31** Reduced motion setting shall disable hover scale/glow animations; highlight via static border colour instead.

### Performance & States

**TAB-LIB-32** The template list/grid shall virtualise items for large collections.

**TAB-LIB-33** Editing and saving templates shall persist instantly to IndexedDB/OPFS; a “Saved” toast shall confirm.

**TAB-LIB-34** Offline use: creating, editing, and using templates shall work fully offline; optional sync queues changes.

### Error Handling

**TAB-LIB-35** Invalid fields (e.g., negative duration, missing title) shall block save and show inline error messages with aria-describedby.

**TAB-LIB-36** Importing malformed files shall fail gracefully with a toast “Import failed: Invalid schema”; valid templates in the file are still imported.

### Wireframes & Screenshots

**TAB-LIB-37** Insert desktop screenshots: template grid with Task and Routine templates; card with hover highlighting actions.

**TAB-LIB-38** Insert mobile screenshots: vertical list with FAB (New Template), swipe actions visible, detail sheet open.

**TAB-LIB-39** Insert template edit modal screenshot: fields for Title, Tags, Quadrant, Steps (for Routines).

## Settings Tab

### Purpose

Provide a central hub for all configuration options: General, Security, Accessibility, Data & Storage, Notifications, and About. Design must remain simple, grouped, and responsive.

### Overall Structure

**TAB-SET-01** The Settings screen shall display grouped categories of options, each within frosted glass cards, separated by headings: General, Security, Accessibility, Data & Storage, Notifications, About.

**TAB-SET-02** On desktop, categories shall be laid out in a tabbed interface (horizontal tab bar or side nav). On mobile, categories shall appear as collapsible accordion sections.

**TAB-SET-03** Visual design: frosted glass panels with 14–16px rounded corners, 1px var(--line) borders, subtle shadows (var(--shadow)), and consistent cosmic gradient background.

### General Category

**TAB-SET-04** General shall include: Language selector, Theme (Light / Dark / Auto), Date/Time format, First-day-of-week selector, and Time zone (auto/manual).

**TAB-SET-05** Language selector shall use a dropdown with preview text (“Bonjour / Hello”) for clarity.

**TAB-SET-06** Theme selection shall preview the choice in a small swatch or live background.

### Security Category

**TAB-SET-07** Security shall include: Passphrase setup/change, Biometric unlock (if supported), Auto-lock timeout, Export key (QR/Base64), and Wipe all data.

**TAB-SET-08** Passphrase setup shall include fields: New passphrase, Confirm, Strength meter, and Save. Validation prevents weak or mismatched entries.

**TAB-SET-09** Auto-lock timeout shall be a dropdown (e.g., 1 min, 5 min, 15 min, 1h, Never).

**TAB-SET-10** Wipe all data shall require typing CONFIRM and pressing a red destructive button; confirmation dialog is aria-modal with focus trap.

### Accessibility Category

**TAB-SET-11** Accessibility shall include: Reduced motion (toggle), Font scaling (slider 80%–200%), High contrast mode (toggle), Simplified interface (toggle), Sound/haptics (toggles).

**TAB-SET-12** Reduced motion disables animations/confetti and uses static outlines instead.

**TAB-SET-13** Simplified interface hides advanced settings, showing only essentials.

**TAB-SET-14** Font scaling previews the selected scaling immediately in a sample text.

### Data & Storage Category

**TAB-SET-15** Data & Storage shall include: Auto-cleanup (toggle), Per-item overrides (checkbox per entity type), Default export format (dropdown: JSON, .sj.enc, Markdown, HTML), Enable SHA256 checksums (toggle).

**TAB-SET-16** Auto-cleanup shall allow specifying retention (7/30/90 days).

**TAB-SET-17** Export defaults shall persist until changed, applied across all tabs.

**TAB-SET-18** Import/Export paths shall be listed, with a [Clear history] button for local storage references.

### Notifications Category

**TAB-SET-19** Notifications shall include: Quiet hours (time range picker), Snooze (preset intervals: 5m, 10m, 30m, 1h), Per-category toggles (Tasks, Routines, Habits, Brain Dump).

**TAB-SET-20** Notification preview shall allow sending a test notification to confirm system/browser permissions.

### About Category

**TAB-SET-21** About shall show: Version number, Maintainer info, Project description, License type, Credits (fonts, icons, libraries), and a Support link (to GitHub issues or email).

**TAB-SET-22** Tagline (“Explore your orbit. Follow your path.”) shall be displayed with brand font in accent colour.

### Controls & Interactions

**TAB-SET-23** All settings changes shall save immediately on toggle or selection; destructive actions (Wipe all data, Reset to defaults) shall require explicit confirmation.

**TAB-SET-24** A “Reset to default” button shall be available in each category card (optional).

**TAB-SET-25** A search bar shall allow quick filtering of settings by keyword; matches highlight corresponding controls.

### Display & Colour System

**TAB-SET-26** Section headers shall use bold var(--ink) text with accent underline; inactive tabs use var(--dim).

**TAB-SET-27** Toggles shall use accent colours when active (var(--mint)) and muted grey when inactive.

**TAB-SET-28** Sliders (e.g., font scaling) shall show value labels and coloured track (accent for filled, var(--line) for unfilled).

### Mobile Behaviour (small screens)

**TAB-SET-29** Layout shall collapse into accordion sections (General, Security, Accessibility, etc.); only one section may expand at a time.

**TAB-SET-30** Floating action buttons are not required; destructive actions appear inline (with red highlight).

**TAB-SET-31** All touch targets (toggles, sliders, dropdowns) shall meet ≥48×48 px minimum.

### Accessibility

**TAB-SET-32** All controls shall use native elements (checkbox, radio, select, slider) with appropriate aria roles and labels.

**TAB-SET-33** Keyboard navigation: Tab cycles through controls; Space/Enter toggles; Esc closes modals/drawers.

**TAB-SET-34** Screen reader announcements: each setting’s name, current value, and state (on/off); live region updates for Save confirmation (“Setting updated”).

**TAB-SET-35** Reduced motion disables transitions between tabs/accordions; focus states remain visible and consistent.

### Import/Export & Persistence

**TAB-SET-36** Export/Import shall apply to all settings when selected, storing values in a config.json alongside entity data; import shall validate schema.

**TAB-SET-37** Round-trip import/export shall be lossless; invalid or unknown keys in imported config.json are ignored with a toast warning.

### Performance & States

**TAB-SET-38** Settings shall load instantly (<200ms target); skeletons not required due to small dataset.

**TAB-SET-39** State changes (toggles, sliders) shall debounce to avoid rapid re-saves; offline state has no effect (local-only persistence).

### Error Handling

**TAB-SET-40** Invalid input (e.g., invalid time format for Quiet hours) shall show inline error messages; input is blocked until corrected.

**TAB-SET-41** Failure to save to storage (rare) shall show a persistent warning banner and retry automatically when possible.

### Wireframes & Screenshots

**TAB-SET-42** Insert desktop screenshots: tabbed Settings with General and Security cards open; toggles, sliders, and dropdowns visible.

**TAB-SET-43** Insert mobile screenshots: accordion layout with expanded Security section, showing Passphrase and Auto-lock controls; collapsed others.

**TAB-SET-44** Insert Wipe data confirmation modal screenshot with red destructive button and “CONFIRM” typed.

## Navbar (Global)

### Purpose

Provide global navigation between major tabs (Tasks, Routines, Habits, Schedule, Brain Dump, Library, Settings). Must remain visible, consistent, and responsive across all devices.

### Overall Structure

**TAB-NAV-01** The Navbar shall display three aligned zones: Left (Logo/Title), Center (Primary Tabs), Right (Global Actions).

**TAB-NAV-02** The Navbar shall be fixed at the top of the viewport, with frosted glass background (rgba(16,20,44,.30)), 1px var(--line) border at bottom, and subtle shadow (var(--shadow)).

**TAB-NAV-03** On desktop, primary tabs shall appear as horizontal text buttons in the center zone; on mobile, they collapse into a hamburger/overflow menu.

### Left Zone (Logo/Title)

**TAB-NAV-04** The left zone shall display the app logo (icon or wordmark “Stellar-Journey”), aligned left with padding 12–16px.

**TAB-NAV-05** Clicking the logo shall return to the default tab (Tasks) without triggering export prompts.

### Center Zone (Primary Tabs)

**TAB-NAV-06** Primary tabs shall include: Tasks, Routines, Habits, Schedule, Brain Dump, Library, Settings.

**TAB-NAV-07** Tabs shall be styled as pill-shaped buttons with subtle hover and active states:

• Active: bold text, accent background (var(--mint)), contrasting foreground (var(--ink)).

• Hover: underline + semi-transparent background.

• Inactive: dim text (var(--dim)).

**TAB-NAV-08** The active tab shall expose aria-selected="true"; inactive tabs aria-selected="false".

**TAB-NAV-09** Keyboard navigation: Tab cycles through tabs; Arrow keys move left/right; Enter activates.

### Right Zone (Global Actions)

**TAB-NAV-10** The right zone shall include optional global actions: Search icon (opens overlay), Theme toggle (light/dark/auto), Account/Profile shortcut (if enabled).

**TAB-NAV-11** On mobile, the right zone may collapse into a single overflow menu (⋯) containing Search and Theme toggle.

**TAB-NAV-12** Notifications icon may be displayed if system notifications are enabled in Settings; unread state shall show a small accent badge.

### Responsive Behaviour

**TAB-NAV-13** On tablet/mobile, the Navbar shall collapse:

• Left: Logo remains.

• Center: Hamburger icon replaces tabs.

• Right: Search/Theme/Profile condensed into overflow menu.

**TAB-NAV-14** Hamburger menu shall slide from left with a frosted glass panel containing primary tabs in vertical list; list items have ≥48×48 px touch targets.

**TAB-NAV-15** Active tab in hamburger list shall be bold with accent bar on the left.

### Display & Colour System

**TAB-NAV-16** Navbar background shall be semi-transparent dark (rgba(16,20,44,.30)) with blur; border-bottom uses var(--line).

**TAB-NAV-17** Active tab background uses var(--mint) (accent); inactive uses var(--dim).

**TAB-NAV-18** Hover/focus outlines shall use var(--mint) with ≥3px visible outline.

**TAB-NAV-19** Global actions use consistent icon set (Feather/Lucide); icons are white (var(--ink)) at rest, accent (var(--mint)) on hover.

### Accessibility

**TAB-NAV-20** Navbar shall use role="navigation" with aria-label="Main".

**TAB-NAV-21** Tab list shall use role="tablist"; each tab role="tab"; aria-selected reflects active state.

**TAB-NAV-22** Hamburger panel shall be aria-modal when open, with focus trap and Esc to close.

**TAB-NAV-23** Screen readers shall announce “<Tab name>, selected” or “<Tab name>, not selected” when navigating.

**TAB-NAV-24** Reduced motion disables slide-in animation of hamburger menu; replaced with instant panel display.

### Performance & States

**TAB-NAV-25** Navbar load shall be instant (<100ms); skeletons are not required.

**TAB-NAV-26** Tabs shall update instantly on selection without layout shifts.

**TAB-NAV-27** Offline state may show a small “Offline” badge in right zone; sync status optional.

### Wireframes & Screenshots

**TAB-NAV-28** Insert desktop screenshot: Navbar with logo left, tabs centered, search/theme/profile actions right.

**TAB-NAV-29** Insert mobile screenshots: Navbar with logo left, hamburger menu open showing vertical tab list, overflow menu (⋯) open with Search and Theme toggle.

## Popups & Dialogs

### Purpose

Provide consistent modal interactions for confirmations, forms, and quick actions. Design must be accessible, responsive, and aligned with the glassmorphic style of Stellar-Journey.

### Overall Structure

**TAB-POP-01** Popups and dialogs shall be displayed as frosted glass panels with 14–16px rounded corners, 1px var(--line) border, subtle shadow (var(--shadow)), and cosmic gradient dimming overlay behind.

**TAB-POP-02** Dialogs shall be centered on desktop/tablet; on mobile, they may appear as bottom sheets (slide-up panels).

**TAB-POP-03** The background behind a dialog shall dim with a semi-transparent overlay (rgba(0,0,0,.60)); clicking/tapping outside does not close destructive dialogs.

### Types of Dialogs

**TAB-POP-04** Confirmation dialogs shall be used for destructive actions (Delete, Wipe, Reset). They shall include: Title (bold), Message text, Confirm button (destructive style, red highlight), Cancel button.

**TAB-POP-05** Input dialogs shall collect small amounts of data (e.g., new passphrase, rename template). They shall include: Labelled input fields, Confirm/Cancel, inline validation.

**TAB-POP-06** Informational dialogs shall provide warnings, status updates, or help. They may include a single [Close] button.

**TAB-POP-07** Action sheets (mobile only) shall slide up from bottom, listing actions with large touch targets (≥48×48 px). Example: Move Task to Quadrant, Edit, Delete, Complete.

### Controls & Interactions

**TAB-POP-08** All dialogs shall trap focus inside the modal until closed; Tab/Shift+Tab cycles elements; Esc closes (except destructive confirm, which requires button click).

**TAB-POP-09** Confirm buttons shall require explicit click/tap; pressing Enter in a text input shall trigger confirm only when valid.

**TAB-POP-10** Error messages (e.g., “Invalid passphrase”) shall appear inline beneath the input field and be announced to assistive tech.

**TAB-POP-11** Progress dialogs shall show a progress bar with aria-valuenow; for example, during import/export.

### Display & Colour System

**TAB-POP-12** Dialog headers shall use var(--ink), body text var(--dim); background uses frosted panel (rgba(16,20,44,.30)); destructive confirm buttons use accent red (#ff5555 or var(--alert)); safe buttons use var(--mint).

**TAB-POP-13** Bottom sheets shall have a visible drag handle (small bar at top), indicating swipe-to-close gesture.

**TAB-POP-14** Focus outlines shall use var(--mint) 3px border; hover states shall brighten buttons/icons slightly.

**TAB-POP-15** Animations: desktop dialogs fade/scale in; mobile sheets slide up; reduced motion disables animations, using instant state change.

### Accessibility

**TAB-POP-16** All dialogs shall use role="dialog" or role="alertdialog" with aria-modal="true"; titles linked via aria-labelledby; body text via aria-describedby.

**TAB-POP-17** Screen readers shall announce dialog type and focus the first actionable element.

**TAB-POP-18** Esc key closes non-destructive dialogs; destructive dialogs require explicit Cancel/Confirm.

**TAB-POP-19** Action sheet items shall be read as buttons with aria-labels matching the action (“Move task to Important & Urgent”).

**TAB-POP-20** Reduced motion disables scale/slide animations; overlay appears instantly.

### Mobile Behaviour

**TAB-POP-21** On small screens, most dialogs shall default to bottom sheets; confirm dialogs may still center if short.

**TAB-POP-22** Swipe down on a bottom sheet shall close it, unless destructive; destructive sheets require explicit Cancel.

**TAB-POP-23** Buttons shall be large (≥48×48 px) with comfortable spacing to prevent accidental taps.

### Performance & States

**TAB-POP-24** Opening a dialog shall mount it instantly (<100ms); skeleton not required.

**TAB-POP-25** Progress dialogs shall throttle UI updates (max 5/sec) to avoid screen reader spam.

### Error Handling

**TAB-POP-26** Invalid input shall block confirmation until corrected; error text inline; confirm button disabled if mandatory field missing.

**TAB-POP-27** Failure states (e.g., import error) shall show persistent warning dialog with actions [Retry], [Export error log], [Dismiss].

### Wireframes & Screenshots

**TAB-POP-28** Insert desktop screenshot: centered confirmation dialog with Title, Message, Confirm (red), Cancel.

**TAB-POP-29** Insert mobile screenshots: bottom action sheet with task actions (Move, Edit, Delete, Complete); swipe-to-close visible.

**TAB-POP-30** Insert progress dialog screenshot showing progress bar and Cancel button.

## Statistics (UI)

### Purpose

Provide visual feedback on user progress across Tasks, Routines, and Habits. Statistics motivate users through streaks, completion percentages, and XP earned.

### Overall Structure

**TAB-STT-STR-01** The Statistics screen shall display frosted glass panels grouped by entity type: Tasks, Routines, Habits, and Global Summary.

**TAB-STT- STR-02** On desktop, panels shall be arranged in a 2×2 grid layout; on tablet, a single column with two panels per row; on mobile, stacked vertically.

**TAB-STT- STR-03** Each panel shall include a title, chart or visualisation, key metrics, and a footer with filter options (time range, tags).

### Tasks Statistics

**TAB-STT-TST-04** The Tasks panel shall show quadrant throughput: number of tasks completed in each quadrant (IU, I, U, NI/NU).

**TAB-STT-05** A bar chart shall display weekly task completions per quadrant, stacked or grouped by quadrant colour (IU=var(--quad-iu), I=var(--quad-i), U=var(--quad-u), NI/NU=var(--quad-nu)).

**TAB-STT-06** Key metrics shall include: Total tasks completed, % per quadrant, Average time to completion (days).

**TAB-STT-07** Filters: Last 7 days, Last 30 days, Custom date range.

### Routines Statistics

**TAB-STT-08** The Routines panel shall show completion vs skipped rates, and on-time percentage.

**TAB-STT-09** A donut chart shall visualise proportion of completed vs skipped steps, with colours: Completed=var(--mint), Skipped=var(--alert).

**TAB-STT-10** A line chart shall show routine completion times (expected vs actual) across runs.

**TAB-STT-11** Key metrics shall include: Average % on-time, Total routines completed, Average XP per run.

**TAB-STT-12** Filters: Last 10 runs, Last 30 runs, Custom tag filter (energy tags).

### Habits Statistics

**TAB-STT-13** The Habits panel shall show streaks and milestones.

**TAB-STT-14** A heatmap calendar (28–35 days) shall visualise daily completions, tinted by category colours; missed days grey, vacation days hatched.

**TAB-STT-15** Milestone counters shall list milestones achieved (7, 14, 28, 50, 100, etc.) with completion dates.

**TAB-STT-16** Key metrics shall include: Longest streak, Current streak, Total completions, Average daily habits completed.

**TAB-STT-17** Filters: Last 7 days, Last 30 days, Last 90 days.

### Global Summary

**TAB-STT-18** The Global Summary panel shall show total XP earned by entity type (Tasks, Routines, Habits).

**TAB-STT-19** A stacked bar or ring chart shall visualise XP distribution across entities.

**TAB-STT-20** Key metrics shall include: Total XP earned, XP this week, XP this month, Average daily XP.

**TAB-STT-21** Filters: This week, This month, All time.

### Controls & Interactions

**TAB-STT-22** Each chart shall provide tooltips on hover/tap, displaying exact values and dates.

**TAB-STT-23** A time range picker shall appear at top of screen, applying globally to all panels; individual filters override global range.

**TAB-STT-24** Export buttons shall allow exporting current stats as JSON (raw data) or Markdown (summary report).

**TAB-STT-25** Clicking a bar/point/heatmap cell shall open the underlying entity list (e.g., tasks completed that day).

### Display & Colour System

**TAB-STT-26** Quadrant charts shall use accent colours defined in Tasks tab (IU=--quad-iu, I=--quad-i, U=--quad-u, NI/NU=--quad-nu).

**TAB-STT-27** Completed steps (Routines) shall use var(--mint); Skipped steps shall use var(--alert) (red/pink); neutral values use var(--dim).

**TAB-STT-28** Habits heatmap cells shall use tinted scale of category colour; missed days grey (#888); vacation days hatched overlay.

**TAB-STT-29** XP ring/bars shall use app accent colour (var(--mint)) for earned XP; secondary entities use matching quadrant/habit colours.

**TAB-STT-30** Fonts: Titles bold (var(--ink)), metrics medium weight, tooltips small text with dark background and white text.

### Mobile Behaviour (small screens)

**TAB-STT-31** Layout shall stack panels vertically; charts shall switch to compact mode (simplified bars/lines, smaller heatmap cells).

**TAB-STT-32** Filters shall appear as a sticky horizontal scroll row below panel titles.

**TAB-STT-33** Touch tooltips shall activate on tap; tapping outside dismisses.

### Accessibility

**TAB-STT-34** All charts shall provide text summaries (e.g., “You completed 24 tasks this week: 10 Important & Urgent, 8 Important, 4 Urgent, 2 Neither”).

**TAB-STT-35** Tooltips shall be accessible via keyboard focus; chart elements shall be focusable with aria-labels (e.g., “Tasks completed: 5 on 2025-09-20”).

**TAB-STT-36** Colour shall never be the sole means of conveying data; icons, patterns, or labels shall accompany colours.

**TAB-STT-37** Reduced motion shall disable animated chart transitions; updates appear instant.

### Performance & States

**TAB-STT-38** Statistics shall pre-aggregate data (in IndexedDB or local store) to avoid recomputation; visual updates shall complete within 200ms on average hardware.

**TAB-STT-39** Long ranges (90+ days) shall virtualise heatmap rendering to preserve performance.

### Error Handling

**TAB-STT-40** If no data exists, panels shall display an empty state: “No tasks completed in this range” with a suggestion (“Try completing a task to see stats!”).

**TAB-STT-41** Import/export failures shall show non-blocking toast with retry option.

### Wireframes & Screenshots

**TAB-STT-42** Insert desktop screenshots: 2×2 grid with Tasks bar chart, Routines donut chart, Habits heatmap, Global XP ring.

**TAB-STT-43** Insert mobile screenshots: stacked panels with compact charts and sticky filter row.

**TAB-STT-44** Insert tooltip screenshot on bar/heatmap cell, showing exact count/date.

## Import/Export — UX Flows

### Purpose

Provide a safe and user-friendly workflow for importing and exporting app data. Import must validate before committing, export must allow scope and format selection. Designed to support round-trip integrity.

### Overall Structure

**TAB-IEX-01** The Import workflow shall follow this sequence:

1. User triggers [Import].

2. File picker opens (accepts .json, .md, .html, .sj.enc).

3. File is validated in dry-run mode.

4. Validation modal appears (showing summary, warnings, errors).

5. User confirms or cancels.

6. Data is applied; progress bar and toast feedback appear.

**TAB-IEX-02** The Export workflow shall follow this sequence:

1. User triggers [Export].

2. Export modal appears (scope selector, format selector, encryption toggle).

3. User confirms.

4. Data is packaged; download begins.

5. Toast feedback confirms completion.

### Import Workflow Details

**TAB-IEX-03** File picker shall accept: .json (data schema), .md (notes), .html (notes/pages), .sj.enc (encrypted backup).

**TAB-IEX-04** Validation modal shall show: number of entities detected (Tasks, Routines, Habits, Brain Dump, Templates), warnings (schema mismatches), and errors (invalid format).

**TAB-IEX-05** Errors shall block import; Warnings allow user to continue. Example: “Unknown field ignored: mood.”

**TAB-IEX-06** User must confirm import before applying changes; destructive actions (overwriting existing entities) shall require explicit confirm checkbox.

**TAB-IEX-07** Progress bar shall display during import with aria-valuenow updates; after completion, a toast appears (“Import complete: 52 Tasks, 4 Routines, 12 Brain Dump”).

### Export Workflow Details

**TAB-IEX-08** Export modal shall provide:

• Scope selector: All data / Selected entities / By type (Tasks, Routines, Habits, Brain Dump, Templates) / By date range.

• Format selector: JSON, Markdown (notes only), HTML (notes/pages), Encrypted (.sj.enc).

• Encryption toggle: Yes/No (only for .sj.enc).

**TAB-IEX-09** User shall confirm export scope and format before start.

**TAB-IEX-10** On success, file download begins; toast shows “Export complete: tasks\_2025-09-25.sj.enc”.

### Conflict Resolution

**TAB-IEX-11** If imported entities have the same IDs as existing ones, the conflict resolution dialog shall appear:

• Options per entity: Keep existing, Replace with import, Merge fields.

• Batch actions: Apply to all of same type.

**TAB-IEX-12** Conflict resolution dialog shall display entity diffs in a side-by-side or stacked card view, highlighting differences in title, tags, fields.

### Display & Colour System

**TAB-IEX-13** Import/export modals shall use frosted glass card design with accent headers. Confirm buttons use var(--mint); destructive replace buttons use var(--alert).

**TAB-IEX-14** Progress bars shall use accent gradients (linear-gradient 90deg, var(--quad-u), var(--quad-i)).

**TAB-IEX-15** Validation modal shall use coloured icons: Green check for valid, Orange triangle for warnings, Red circle for errors.

### Mobile Behaviour (small screens)

**TAB-IEX-16** On mobile, Import modal shall appear as a bottom sheet with file picker button and validation summary; Confirm and Cancel are sticky at bottom.

**TAB-IEX-17** Export modal shall appear as bottom sheet with stacked controls: Scope selector (dropdown), Format selector (radio list), Encryption toggle (switch).

**TAB-IEX-18** Conflict resolution shall use accordion lists: each conflict expandable to show diffs; batch actions appear as sticky buttons.

### Accessibility

**TAB-IEX-19** File picker shall use native input[type=file]; aria-label “Choose file to import”.

**TAB-IEX-20** Validation modal shall use role="dialog", aria-modal="true", and aria-describedby for warnings/errors summary.

**TAB-IEX-21** Progress bar shall expose aria-valuenow, aria-valuemin, aria-valuemax; screen reader announces “Import 42% complete”.

**TAB-IEX-22** Conflict resolution cards shall be focusable with aria-labels summarising the entity and its differences.

**TAB-IEX-23** Reduced motion disables progress bar animation; updates appear as discrete steps.

### Performance & States

**TAB-IEX-24** Validation shall parse up to 10,000 entities without blocking UI; large imports stream in chunks with progress updates.

**TAB-IEX-25** Exports shall be generated in worker threads to keep UI responsive; large files stream progressively.

**TAB-IEX-26** Offline state: imports/exports always work locally; if sync enabled, imported data is queued for upload.

### Error Handling

**TAB-IEX-27** Invalid files (.exe, .zip without manifest) shall be rejected with toast “Unsupported file type”.

**TAB-IEX-28** Corrupted .sj.enc files shall fail decryption gracefully with clear error “Invalid encryption key or corrupted file”.

**TAB-IEX-29** Partial imports (some entities invalid) shall import valid entities and show warning: “12 imported, 3 skipped”.

### Wireframes & Screenshots

**TAB-IEX-30** Insert desktop screenshots: Import validation modal with summary and coloured icons; Export modal with scope/format controls.

**TAB-IEX-31** Insert mobile screenshots: Import bottom sheet with file picker; Export bottom sheet with toggles.

**TAB-IEX-32** Insert conflict resolution screenshot: side-by-side diff cards, batch actions visible.

## Security Settings — UX Flows

### Purpose

Provide clear, consistent security-related workflows, ensuring data protection while maintaining user-friendliness. Visuals must match Stellar-Journey’s glassmorphic design.

### Overall Structure

**TAB-SEC-01** Security workflows shall be accessible from the Settings tab (Security section). Each workflow opens in a frosted glass modal or drawer, with accent colours for clarity.

### Passphrase Setup & Change

**TAB-SEC-02** Passphrase setup shall display fields: New passphrase, Confirm passphrase, and an inline Strength meter.

**TAB-SEC-03** The Strength meter shall classify passphrases as Weak, Medium, Strong based on entropy, updated live as user types.

**TAB-SEC-04** A Save button shall remain disabled until both fields match and meet minimum strength criteria.

**TAB-SEC-05** Error states (mismatch, weak) shall appear inline with aria-describedby; screen readers announce the error immediately.

**TAB-SEC-06** A toast shall confirm successful save: “Passphrase updated”.

### Unlock Dialog

**TAB-SEC-07** The Unlock dialog shall appear on app launch or after auto-lock. It shall include: Passphrase field, Show/Hide toggle, and [Unlock] button.

**TAB-SEC-08** Incorrect passphrase shall trigger an inline error “Incorrect passphrase”; the input field shall shake slightly (reduced motion disables the shake).

**TAB-SEC-09** Optional Biometric unlock button shall appear if device supports it; fallback always available via passphrase.

**TAB-SEC-10** Unlock dialog shall trap focus until dismissed; Esc closes only if Unlock is not required (e.g., optional relock).

### Auto-Lock Modal

**TAB-SEC-11** Auto-lock shall trigger after configured inactivity; a modal shall overlay the app with “Session locked” message.

**TAB-SEC-12** Modal shall include [Unlock] and [Cancel session] buttons; [Cancel session] wipes in-memory keys and requires full login.

**TAB-SEC-13** Countdown (optional) may be displayed before auto-lock engages, with toast “Locking in 10s…”.

### Data Wipe Confirmation

**TAB-SEC-14** Wipe All Data shall open a destructive confirmation dialog: Title “Delete all data?”, Message “This cannot be undone. Type CONFIRM to proceed.”

**TAB-SEC-15** Confirm button shall remain disabled until the user types CONFIRM in uppercase letters.

**TAB-SEC-16** On confirm, local data (IndexedDB, OPFS, localStorage) shall be deleted; app reloads to onboarding state.

**TAB-SEC-17** A success toast shall appear after reload: “All data wiped”.

### Export Key Dialog

**TAB-SEC-18** Export Key shall open a modal showing the encryption key in two forms: Base64 string and QR code.

**TAB-SEC-19** A warning message shall be displayed: “Keep this key private. Anyone with it can read your data.”

**TAB-SEC-20** Buttons: [Copy to clipboard], [Download .key file], [Close]. Copy shows toast “Key copied”.

**TAB-SEC-21** Export key modal shall be aria-modal, with Esc closing and focus returning to Settings.

### Display & Colour System

**TAB-SEC-22** Passphrase fields shall use frosted inputs with accent border (var(--mint)) when focused.

**TAB-SEC-23** Strength meter shall use colours: Weak=var(--alert), Medium=#f2c94c (yellow), Strong=var(--mint).

**TAB-SEC-24** Error messages shall be red (#ff5555), success toasts green (var(--mint)).

**TAB-SEC-25** Wipe confirm dialog shall style Confirm button as red destructive (var(--alert)); Cancel button dim (var(--dim)).

**TAB-SEC-26** Export Key modal QR code shall be displayed in high contrast (black/white only).

### Mobile Behaviour (small screens)

**TAB-SEC-27** All security modals shall appear as full-screen sheets on mobile, with sticky action buttons at bottom (Confirm/Cancel).

**TAB-SEC-28** Passphrase fields shall support password managers (autocomplete="current-password/new-password").

**TAB-SEC-29** Typing CONFIRM in wipe dialog shall auto-capitalise input on mobile keyboards for convenience.

### Accessibility

**TAB-SEC-30** All dialogs shall use role="dialog" or role="alertdialog" with aria-modal="true"; titles linked via aria-labelledby; error/help text via aria-describedby.

**TAB-SEC-31** Passphrase strength meter shall include a text label for screen readers (e.g., “Strength: Weak”).

**TAB-SEC-32** Unlock errors shall be announced politely but immediately (“Incorrect passphrase”).

**TAB-SEC-33** QR code in Export Key shall be accompanied by a selectable text version for screen readers.

**TAB-SEC-34** Reduced motion disables shaking effect on incorrect passphrase; error feedback remains via text and outline.

### Performance & States

**TAB-SEC-35** Unlock modal shall appear instantly (<100ms); biometric prompt defers to device OS timing.

**TAB-SEC-36** Wipe operation shall complete within 2 seconds for typical datasets; a progress spinner appears if longer.

### Error Handling

**TAB-SEC-37** Invalid passphrase entries shall not allow unlock until corrected.

**TAB-SEC-38** Export key failures (clipboard denied, file write error) shall show clear inline error with retry.

**TAB-SEC-39** If wipe fails (e.g., storage locked), user shall see persistent banner “Data wipe failed. Please retry.”

### Wireframes & Screenshots

**TAB-SEC-40** Insert desktop screenshots: Passphrase setup with strength meter, Unlock dialog with error, Wipe confirmation modal.

**TAB-SEC-41** Insert mobile screenshots: full-screen Unlock sheet with sticky Unlock button, Export Key QR code modal with Copy/Download buttons.

**TAB-SEC-42** Insert data wipe success state screenshot (post-reload onboarding).

## Static Pages Appearance

### Purpose

Provide static informational pages with consistent design, accessible layout, and clear separation of content. These pages inform the user about privacy, legal aspects, and project background.

### Overall Structure

**TAB-SPG-01** Static pages shall be accessible from the Settings tab (About section) and from the footer of the web app (if enabled).

**TAB-SPG-02** All static pages shall use frosted glass containers with 16px padding, 14–16px rounded corners, subtle shadow (var(--shadow)), and cosmic gradient background.

**TAB-SPG-03** Static pages shall use a simple article layout: Title (H1), Section headers (H2), Body paragraphs, and optional lists.

**TAB-SPG-04** A sticky back button [← Back] shall return to the previous screen or Settings tab.

### Privacy Policy Page

**TAB-SPG-05** The Privacy Policy page shall include the following sections:

* Data Storage — “All data stays on your device by default.”
* Telemetry — “No telemetry, analytics, or ads are collected.”
* Third Parties — “No third-party trackers are embedded.”
* Export/Import — “Exported files remain under your control.”
* Encryption — “Encrypted .sj.enc is recommended.”
* Sync — “Optional online sync uses HTTPS, no personal content logged.”
* Data Deletion — “You may delete all data anytime from Settings.”

**TAB-SPG-06** Each section shall be collapsible (accordion style) on mobile, expanded by default on desktop.

**TAB-SPG-07** Privacy text shall be written in plain, concise language; font size minimum 16px for readability.

### Legal Notice Page

**TAB-SPG-08** The Legal Notice page shall include the following sections:

* Project — “Stellar-Journey, productivity app (PWA).”
* Maintainer — “Independent developer (indie project).”
* Contact — Placeholder email or GitHub repo link.
* Hosting — e.g., “GitHub Pages” or VPS provider.
* Intellectual Property — “Source code licensed under MIT unless otherwise specified.”
* Liability — “Provided as-is, without warranties.”
* Jurisdiction — “Personal, non-commercial, open-source project.”

**TAB-SPG-09** Section headers shall use bold text with small icons (e.g., book for Legal, shield for Privacy).

**TAB-SPG-10** Content shall follow Markdown-like styling: paragraphs, bullet lists, horizontal rules between sections.

### About Page

**TAB-SPG-11** The About page shall include:

* Project purpose — “A calm, structured productivity app designed for neurodivergent users.”
* Maintainer — “Independent solo developer.”
* License — MIT open-source.
* Credits — Fonts (Inter, SIL OFL), Icons (Lucide, MIT), Libraries (React, Bootstrap).
* Support — “File issues on GitHub repo.”
* Tagline — “Explore your orbit. Follow your path.” displayed in brand font/accent colour.
* Version number — displayed at bottom.

**TAB-SPG-12** About page may include a small section with “Changelog” linking to GitHub releases.

### Display & Colour System

**TAB-SPG-13** Titles (H1) shall use bold var(--ink) at 24–28px; section headers (H2) use accent underline (var(--mint)); body text uses var(--dim).

**TAB-SPG-14** Links shall be accent-coloured (var(--mint)), underlined on hover, with focus outline visible.

**TAB-SPG-15** Icons (Feather/Lucide) shall be used sparingly next to section headers; size 18–20px.

**TAB-SPG-16** Reduced motion disables accordion animation on mobile; expands/collapses instantly.

### Mobile Behaviour

**TAB-SPG-17** Layout shall collapse into a single vertical scroll with collapsible sections (accordion).

**TAB-SPG-18** The Back button shall appear sticky at top; bottom navigation hidden while static page is active.

**TAB-SPG-19** Links open in new tab (target=\_blank) with rel="noopener"; external links clearly marked.

### Accessibility

**TAB-SPG-20** Pages shall use role="document"; each section shall use role="region" with aria-labelledby referencing its header.

**TAB-SPG-21** Accordion sections shall expose aria-expanded states and aria-controls linking to content.

**TAB-SPG-22** Text shall meet WCAG AA minimum font size (≥16px body) and contrast (≥4.5:1).

**TAB-SPG-23** Screen readers shall announce “Section expanded/collapsed” when toggled.

**TAB-SPG-24** Reduced motion disables collapse/expand animations; state change is announced via aria-live.

### Performance & States

**TAB-SPG-25** Static pages shall load instantly (<100ms); content is lightweight and text-based.

**TAB-SPG-26** Offline state: all pages shall remain accessible even when offline.

### Wireframes & Screenshots

**TAB-SPG-27** Insert Privacy Policy screenshot: collapsible sections visible on mobile, expanded on desktop.

**TAB-SPG-28** Insert Legal Notice screenshot: article-style layout with section headers and small icons.

**TAB-SPG-29** Insert About page screenshot: tagline highlighted, credits list, version number footer.