

DESIGN AND AI ASSISTED DEVELOPMENT REPORT



BibleBeacon

BibleBeacon - Web App

Product Design and AI Assisted Development

How Keith Trowbridge defined and directed an offline friendly Scripture web app and used AI to accelerate bounded design, implementation, testing, and documentation work

Visit the online version here: <https://biblebeacon.sardev.io>

PRODUCT BibleBeacon	PRODUCT TYPE Web app and installable PWA	REPORTING PERIOD Verified record September to 15 October 2026	CURRENT STATUS Release 0.16.0 deployed and live verified
-------------------------------	--	---	--

At a Glance

PRODUCT LEAD Keith defined the vision, workflows, visual direction, acceptance criteria, and release decisions.	AI ASSISTANCE AI supported bounded research, exploration, coding, review, testing, and report preparation.	VALIDATION 47 automated tests plus typecheck, lint, and production build passed on 15 October 2026.	RELEASE The public site and 0.16.0 release metadata returned HTTP 200 on 15 October 2026.
---	--	---	---

Figure 1. Verified project ownership, assistance, validation, and release boundary.

1 Executive Summary

BibleBeacon is an offline friendly Scripture learning, reading, listening, discovery, and personal study experience delivered as a responsive web app and installable progressive web app. The product joins Bible games and return loops with a full Bible reader, local search, saved content, plans, prayer and reflection, study metadata, curated discovery, audio, Scripture Fusion, Word Puzzle, and a private progress log. The verified release does not require an account, remote database, payment system, analytics SDK, or runtime Bible API.

Keith Trowbridge directed the product vision, requirements, user journeys, information architecture, visual direction, acceptance criteria, privacy boundaries, and release decisions. AI tools accelerated bounded research, option development, implementation support, code review, test creation, troubleshooting, and documentation. Keith retained decision authority and reviewed, corrected, accepted, or rejected the resulting work. Third party sources and services supplied hosting, source control, Scripture text, study metadata, and chapter audio; they did not define the product.

The selected direction evolved from a lightweight challenge experience into a local first product system. Release records document incremental additions from daily challenges and progression through review, quests, PWA behavior, sharing, discovery, audio, the Bible library, Word Puzzle, and the final four choice Scripture Fusion interaction. On 15 October 2026, the repository passed 47 automated tests, typechecking, linting, and a production build. The public site, release metadata, service worker, and direct game routes returned HTTP 200 and identified release 0.16.0. Formal usability research, audited accessibility compliance, commercial impact, and current Meta store publication remain outside the verified evidence boundary.

KEY TAKEAWAY

Keith shaped BibleBeacon as a coherent, privacy conscious Scripture experience and used AI as a supervised production tool; the web release is live and technically verified, while product adoption and formal compliance outcomes remain unmeasured.

2 Project Context

2.1 Purpose and Opportunity

BibleBeacon began as a compact Scripture challenge and became a broader product intended to support different ways of returning to the Bible: quick play, guided review, reading, search, listening, discovery, saving, planning, prayer, and deeper study. The opportunity was to combine those behaviors with one branded experience while preserving a low friction entry. A user can start without an account and keep progress and personal study material on the current device.

The repository does not contain market sizing, user interviews, conversion data, or revenue evidence. This report therefore frames the opportunity as a product design response to verified requirements and observed application behavior, not as a proven market result.

2.2 Users and Needs

The verified interface supports individual readers and learners who want short Scripture activities or a deeper study session on desktop or mobile. It also anticipates users with intermittent connectivity by bundling text and study indexes and by providing an installable offline shell.

- Start quickly through a clear home screen, daily or replayable game loops, and direct routes to major features.
- Move between reading, search, audio, discovery, saving, plans, prayer, and study tools without creating separate accounts.
- Understand what is stored locally, what source each text or index uses, and which features are available offline.
- Return to prior activity through resume positions, saved items, progress records, missions, quests, and review scheduling.

2.3 Problem

A single quiz could support brief engagement but not the broader ways people read, hear, remember, organize, and revisit Scripture. Adding each capability as a separate mini product would have created fragmented navigation, duplicated state, and inconsistent source or privacy behavior. The design problem was to expand the product without losing the directness of the original challenge, the cinematic visual identity, or the account free operating model.

2.4 Goals Constraints and Scope

Goal	Success Criteria	Constraint or Boundary
Unify play and study	Major destinations are reachable through one responsive navigation and one Library information model.	Preserve existing games, progress, audio, PWA, sharing, and responsive behavior.
Keep core use local first	Reader, search, saved items, plans, prayer, study indexes, and game data work from bundled or device local data.	No account, remote database, analytics SDK, paid API, or runtime Bible API in release 0.16.0.
Make content provenance visible	Translations and study sources carry license, attribution, revision, and checksum records.	Do not import or publish content with unverified file level rights.

Goal	Success Criteria	Constraint or Boundary
Release safely	Automated checks, browser checks, build validation, rollback artifacts, and production route verification are recorded.	Deployment status must remain distinct from local or planned work.

Table 1. Project goals, observable success conditions, and governing boundaries.

2.5 Inputs and Requirements

Input or Requirement	Why It Mattered	Evidence	Status
Preserve the existing challenge product	Expansion could not break the working game, progress, sharing, PWA, or audio surfaces.	README, expansion plan, release records, regression tests	Met
Offline friendly Bible library	Core reading and search needed to avoid runtime dependence on a third-party service.	Bundled Scripture JSON and source manifest	Met
Private personal tools	Notes, highlights, plans, prayer, and reflections needed a clear local storage boundary.	library Store and migration tests	Met locally and in deployed app
Governed external data	Study content required exact source, license, attribution, revision, and checksum handling.	Source registry and import documentation	Met for approved sources
Accounts, synchronization, paid study tier	These require authentication, database, privacy, conflict, and payment decisions.	Expansion plan approval gates	Deferred

Table 2. Requirements and evidence boundaries.

Evidence basis: README.md; docs/BIBLEBEACON_EXPANSION_PLAN.md; docs/SOURCE_REGISTRY.md; tests; release records.

3 Role and Accountability

3.1 Keith Trowbridge Responsibilities

Keith owned the human decisions that shaped BibleBeacon: the product purpose, feature priorities, target workflows, navigation model, visual direction, privacy and source boundaries, acceptance criteria, review expectations, and release decisions. He determined when an idea was ready to move from exploration to implementation and whether a build was suitable for local verification, packaging, deployment, or public presentation.

The repository and release records show that these decisions were translated into staged requirements: preserving the challenge experience, add replay and progress loops, add offline and sharing behavior, introduce discovery and audio, expand into a full Library, and then add distinct game experiences. The public case study should describe Keith as the product and design lead using AI assistance, not as a passive recipient of an autonomous system.

3.2 Contribution Boundaries

Contributor	Contribution	Decision Authority	Evidence
Keith Trowbridge	Vision, requirements, UX and information architecture, visual direction, acceptance criteria, testing priorities, publication and release decisions	Final human authority for product fit and release acceptance	Requirements, release records, approved designs, repository history
AI tools	Bounded research, alternatives, drafting, coding support, test creation, code review, troubleshooting, and documentation	No independent product, content, deployment, or publication authority	Prompts, code changes, tests, change records, produced artifacts
Open source and data providers	Scripture text, study indexes, reference data, source metadata, and supporting software	Authority only over their own licenses and upstream materials	Source registry, manifests, attribution records
Hostinger and GitHub	Hosting and source or deployment infrastructure documented by the project	Platform operation; no product design authority	Deployment status and release records
Meta Horizon	Separate mobile store review channel for an earlier APK line	Platform review and publication gate	Historical deployment status; current status not reverified for this report

Table 3. Ownership, support contributions, authority, and evidence.

HUMAN OWNERSHIP STATEMENT

Keith Trowbridge directed BibleBeacon's goals, requirements, experience design, information architecture, visual direction, acceptance criteria, and release decisions.

AI tools accelerated selected research, exploration, implementation, testing, review, and documentation tasks. Keith reviewed, corrected, accepted, or rejected the outputs and retained accountability for the result.

3.3 Collaboration and Approvals

The evidence supports a founder directed workflow assisted by software tools and external platforms. It does not establish a separate design agency, engineering team, client sponsor, or independent compliance reviewer. Source providers are credited for their materials, and platform services are identified for their operational role. Publication of this report and any changes to authentication, database storage, payments, or destructive deployment behavior remain subject to Keith's explicit approval.

4 Product and Experience Design

4.1 Experience Strategy

The selected product direction uses one cinematic visual system and several clear entry points rather than forcing every behavior into a single dashboard. The home screen highlights the newest engagement paths and keeps six stable destinations visible: Library, Listen, Discover, Scripture Fusion, Word Puzzle, and Beacon Log. The Library then organizes six related personal study tasks: Read, Search, Saved, Plans, Prayer, and Study Tools.

This two-level information architecture separates public content and play surfaces from the private study workspace. It also avoids placing every option in the top navigation. A user can choose a quick action, move deeper into reading or study, and return through browser history, explicit home links, saved items, resume state, or progress surfaces.

4.2 Key Workflows

- Read and compare: open Library, choose a translation, book, and chapter, optionally compare another translation, then save, highlight, note, copy, share, or continue across book boundaries.
- Find and follow: search a word, phrase, topic, or reference; narrow by translation or location; open a result in Read; then save or continue exploring related material.
- Listen and return: choose a supported New Testament book and chapter, use native playback, move between chapters, download the recording, and resume from a device local position.
- Discover and save: filter curated passage cards by theme, read the cited context, follow related references, bookmark a card, or listen when chapter audio is available.
- Play and progress: enter a challenge or game, receive cited feedback or a Scripture reward, then return through replay, daily activity, mission, quest, review, level, or Beacon Log paths.
- Plan and reflect: select a reading plan, open each day's passage in Read, mark progress, and optionally maintain device local prayer and reflection entries.

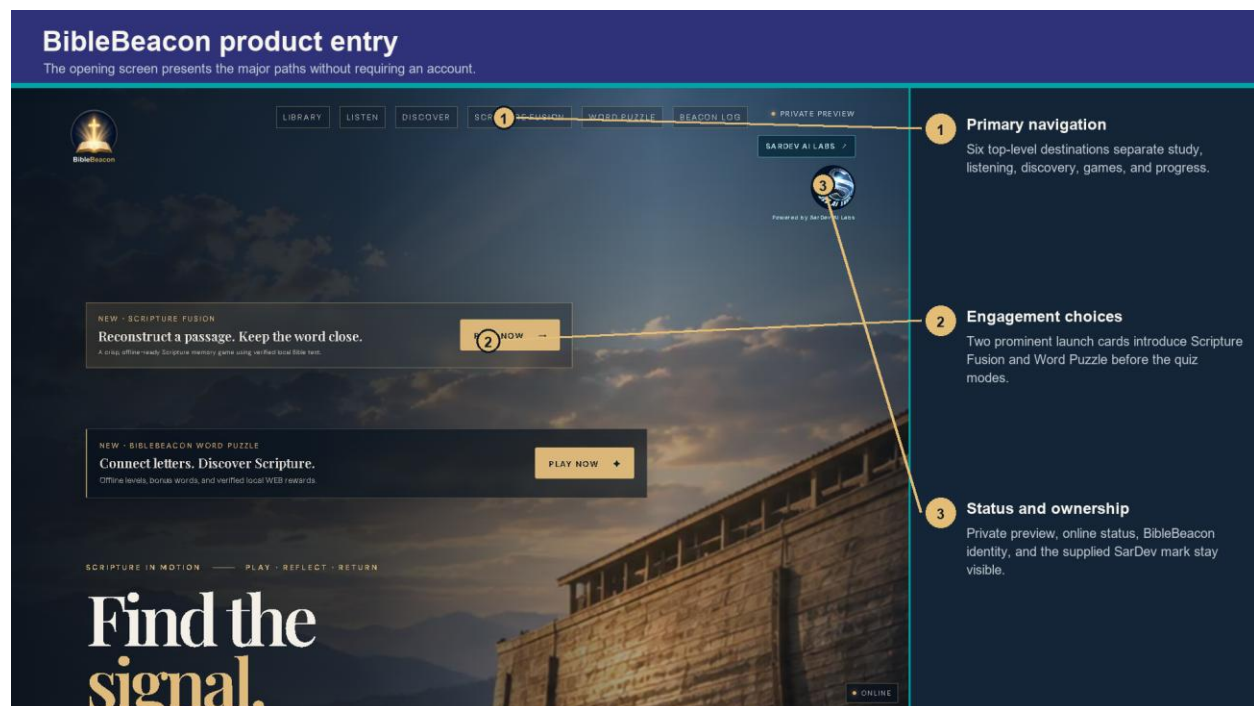


Figure 2. BibleBeacon home and product entry points.

4.3 Important Design Decisions

Decision	Alternatives Considered	Reason Chosen	Validation
Account free local first core	Require sign in and cloud storage from the first release	Reduced setup and kept personal study data on the device while architecture decisions remained unapproved	Storage migration and feature regression tests
Stable top navigation plus Library tabs	One large mega menu or separate standalone tools	Kept major public paths visible and grouped closely related private study work	Desktop and 390 by 844 browser checks

Decision	Alternatives Considered	Reason Chosen	Validation
Bundled Scripture and study indexes	Runtime calls to mutable upstream APIs	Supported offline behavior and pinned source provenance	Manifest hashes, import verification, PWA precache
Native audio controls	Fully custom player controls	Preserved accessibility and browser media behavior while adding product specific chapter navigation	Audio tests, range checks, browser QA
Incremental releases	One large platform rewrite	Protected existing functionality and created rollback points for each major addition	Versioned release and rollback records

Table 4. Major product and interaction decisions.

4.4 Visual Interaction and Accessibility Direction

BibleBeacon uses a dark cinematic surface, gold accents, serif display headlines, restrained borders, and a consistent ark landscape. The visual direction gives the product a distinctive identity while the interaction layer relies on conventional buttons, selects, tabs, native media controls, visible focus, and clear active states. Responsive layouts stack content and wrap navigation instead of shrinking important touch targets.

The source and tests document skip navigation, focus visible states, answer group labels, reduced motion support, responsive layouts, status messaging, and semantic progress behavior. These are verified design and implementation intentions, not a claim of formal WCAG certification or an independent accessibility audit.

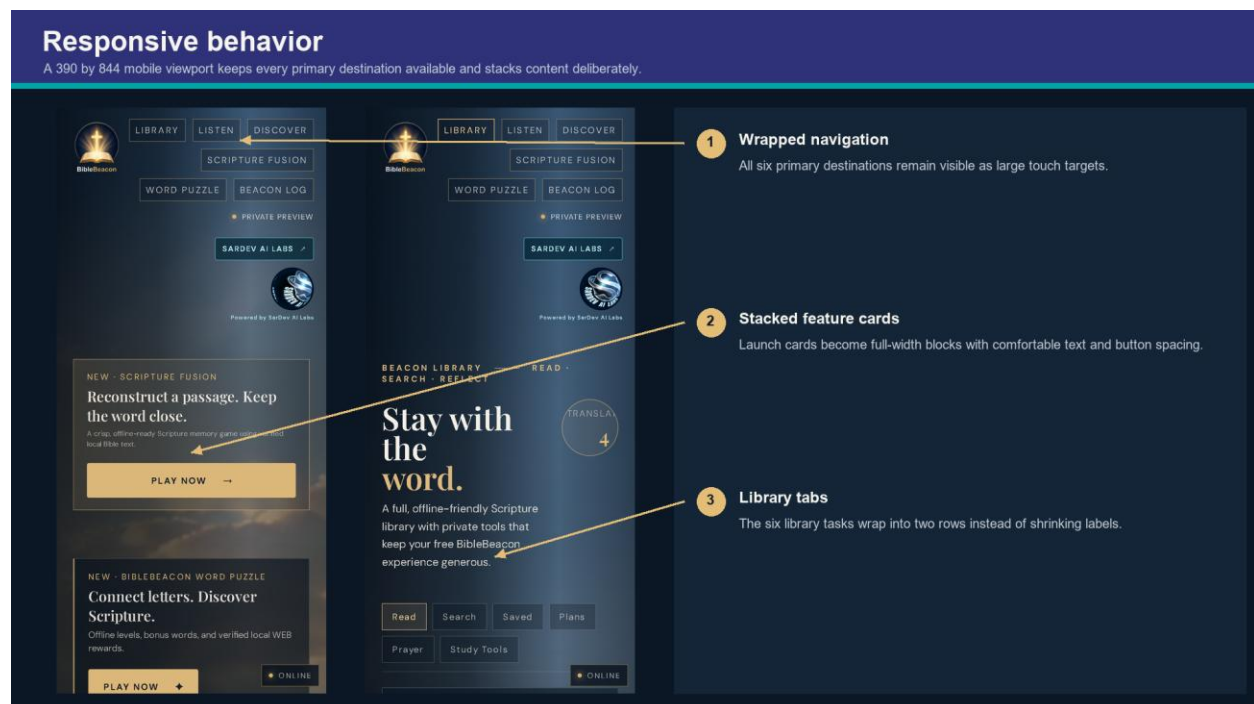


Figure 3. Responsive behavior at a 390 by 844 mobile viewport.

4.5 Feature Walkthrough

4.5.1 Library and Read

- Library is the shared workspace for Read, Search, Saved, Plans, Prayer, and Study Tools.
- Read exposes four bundled translations, book and chapter selection, cross book navigation, resume state, and responsive side by side comparison when text is available.
- Verse actions support bookmarks, four highlight colors, notes, copy, links, device share, and locally generated image cards.

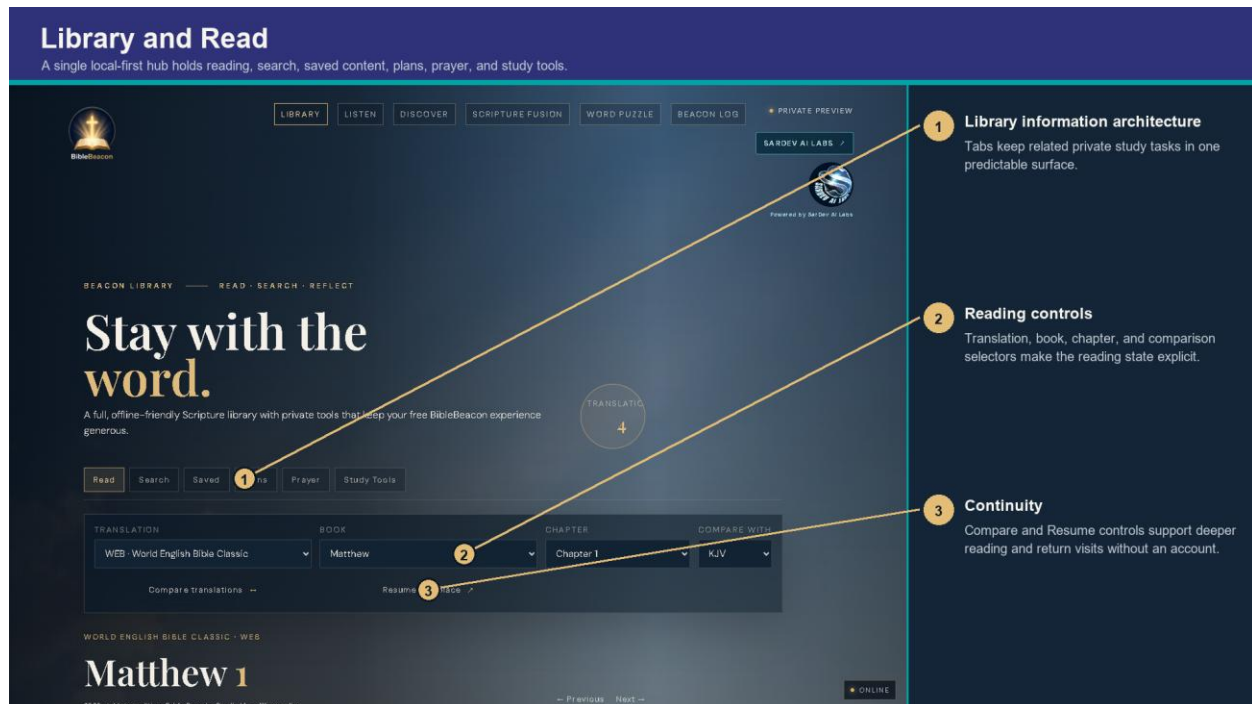


Figure 4. Library information architecture and Read controls.

4.5.2 Search

- Search accepts words, phrases, topics, and direct references against the bundled corpus.
- Translation, book, and chapter filters make the search boundary visible before a result is opened.
- Results retain their translation and reference and link back to the Reader rather than creating a separate reading surface.

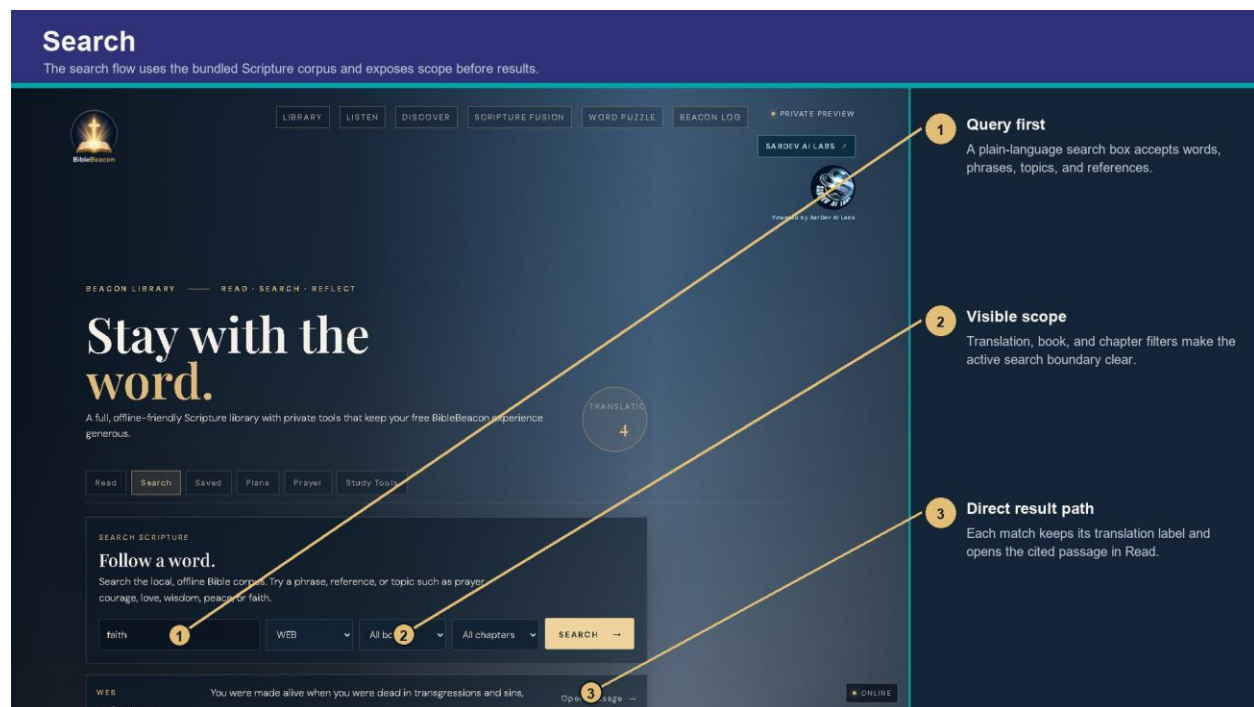


Figure 5. Local Scripture search and scoped result path.

4.5.3 Saved

- Bookmarks, highlights, and notes are consolidated in one searchable collection.
- The interface states that saved material remains on the device until the user exports it.
- Versioned export and import provide a manual portability path while malformed imports are rejected before storage is overwritten.

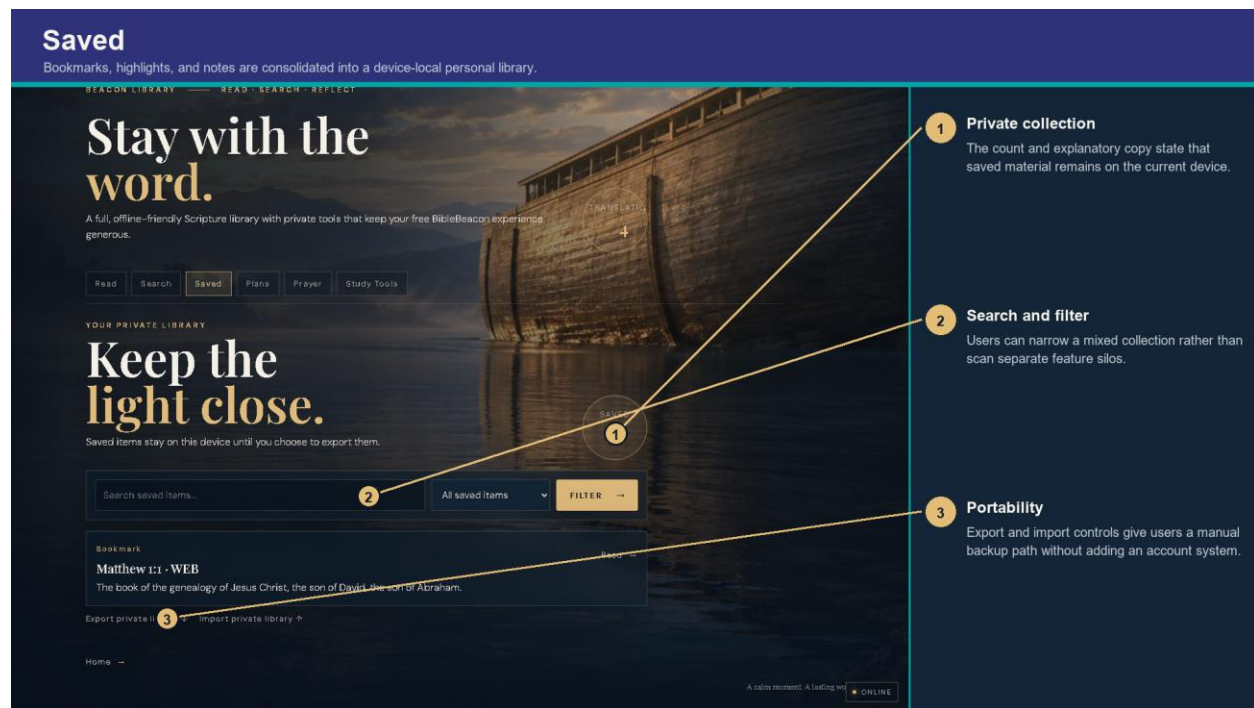


Figure 6. Saved bookmarks highlights and notes.

4.5.4 Plans

- Three short plans offer different reading intents without requiring account setup.
- Each day links into Read and can be marked complete locally, with resume and streak state available on return.
- Browser reminders are optional and best effort; the plan remains usable without notification permission.

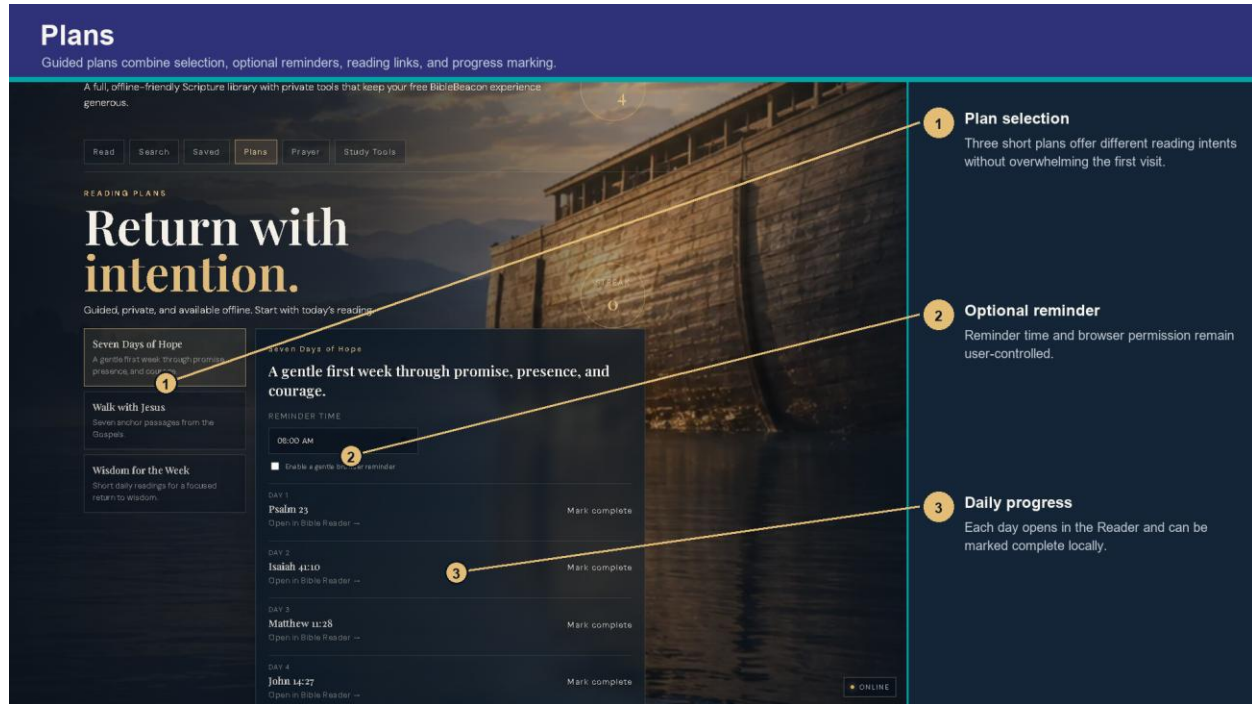


Figure 7. Guided reading plans and local progress.

4.5.5 Prayer

- Prayer and reflection content is private by default and stored locally in the verified release.
- Entries support a title, details, optional Scripture association, search, and active, answered, or archived status.
- The feature does not claim counseling, divine authorship, remote synchronization, or encrypted cloud storage.

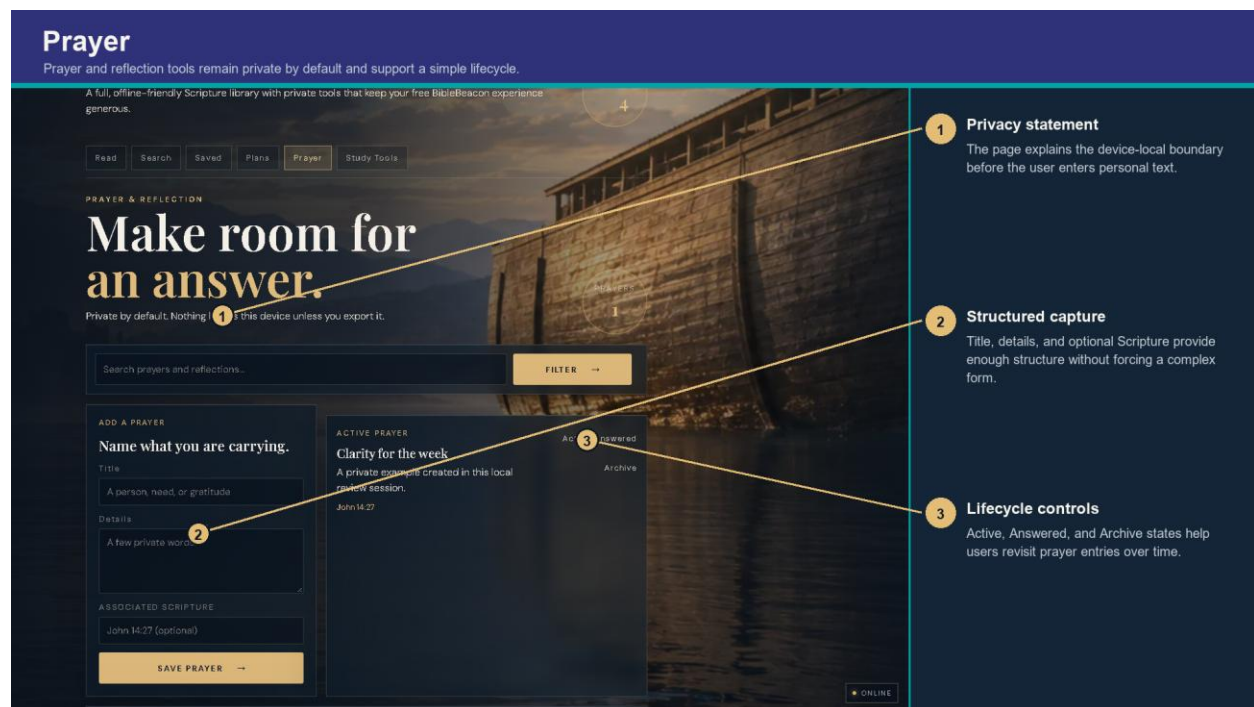


Figure 8. Private prayer and reflection workflow.

4.5.6 Study Tools

- The local index covers people, places, relationships, Hebrew Strong's metadata, and Nave's topical entries from approved sources.
- Source revision, license, attribution, checksum, and status are part of the product boundary, not hidden implementation notes.
- Unapproved commentary, maps, lexicons, dictionaries, and morphology sources remain governed or deferred rather than silently imported.

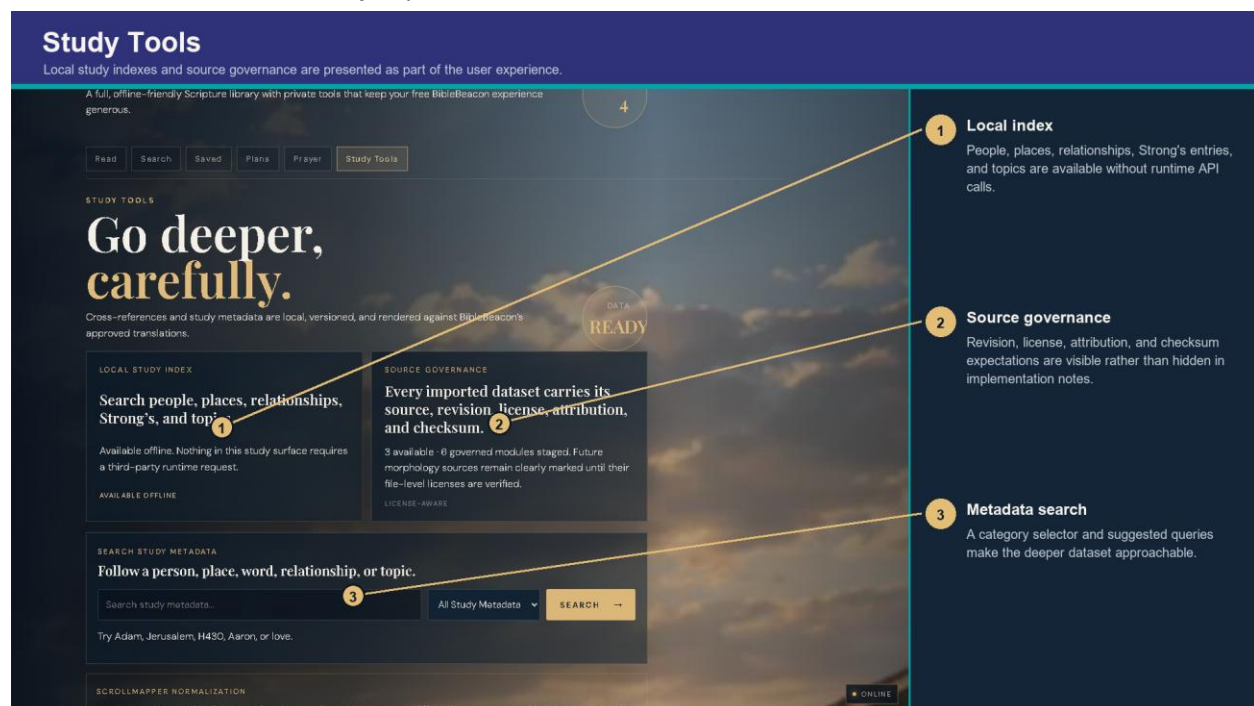


Figure 9. Study metadata and source governance.

4.5.7 Listen

- Listen covers 260 supplied chapter recordings across the 27 New Testament books.
- Book and chapter selectors, previous and next, download, native playback, resume, and low data mode form a single flow.
- Audio is streamed on demand and kept separate from the smaller web deployment package; unavailable chapters remain truthful rather than substituting remote content.

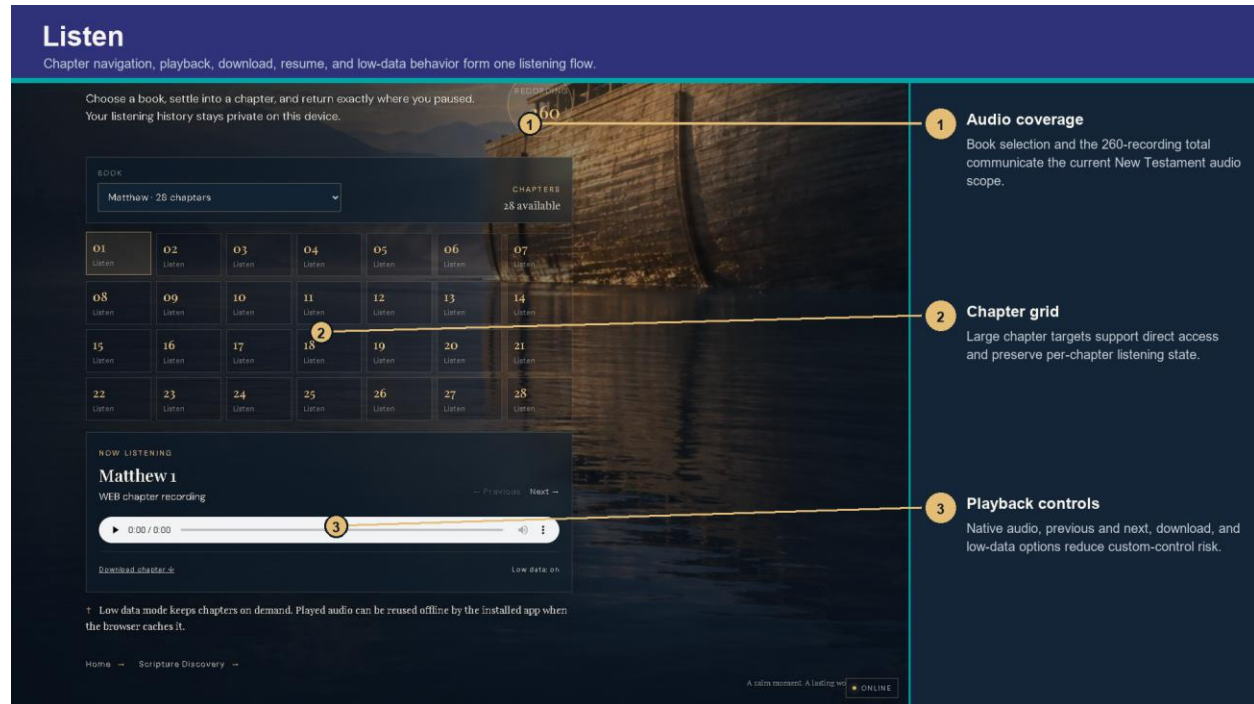


Figure 10. Audio Bible selection playback and return state.

4.5.8 Discover

- Discovery organizes curated cards into Stories, Teachings, People, and Wisdom.
- Each card pairs a cited local WEB excerpt with concise context and related references.
- Bookmarks create a return path and supported New Testament references expose chapter audio without changing the underlying text source.

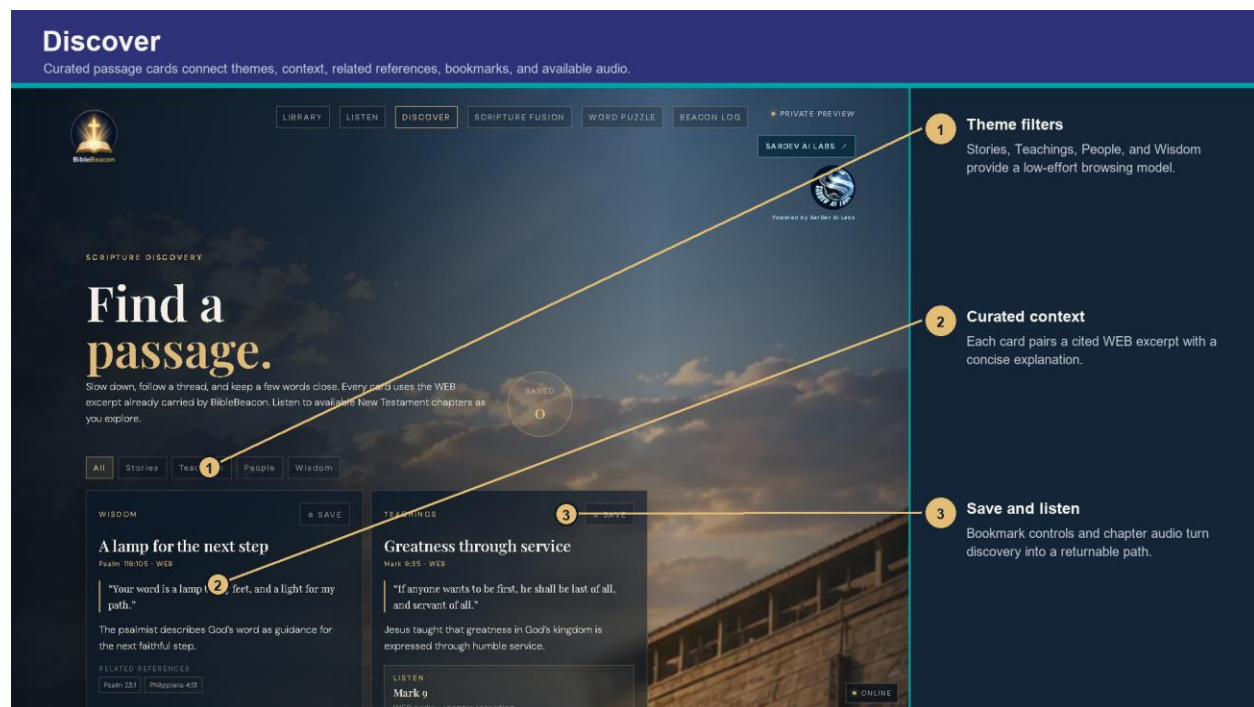


Figure 11. Curated Scripture Discovery cards.

4.5.9 Scripture Fusion

- The final selected interaction replaces an earlier word order concept with one missing word and four choices.
- Incorrect choices reveal the complete passage and allow another attempt; a correct choice completes the verse and triggers celebratory feedback.
- Every passage resolves from pinned local WEB text, while distractors remain clearly game content rather than Scripture.

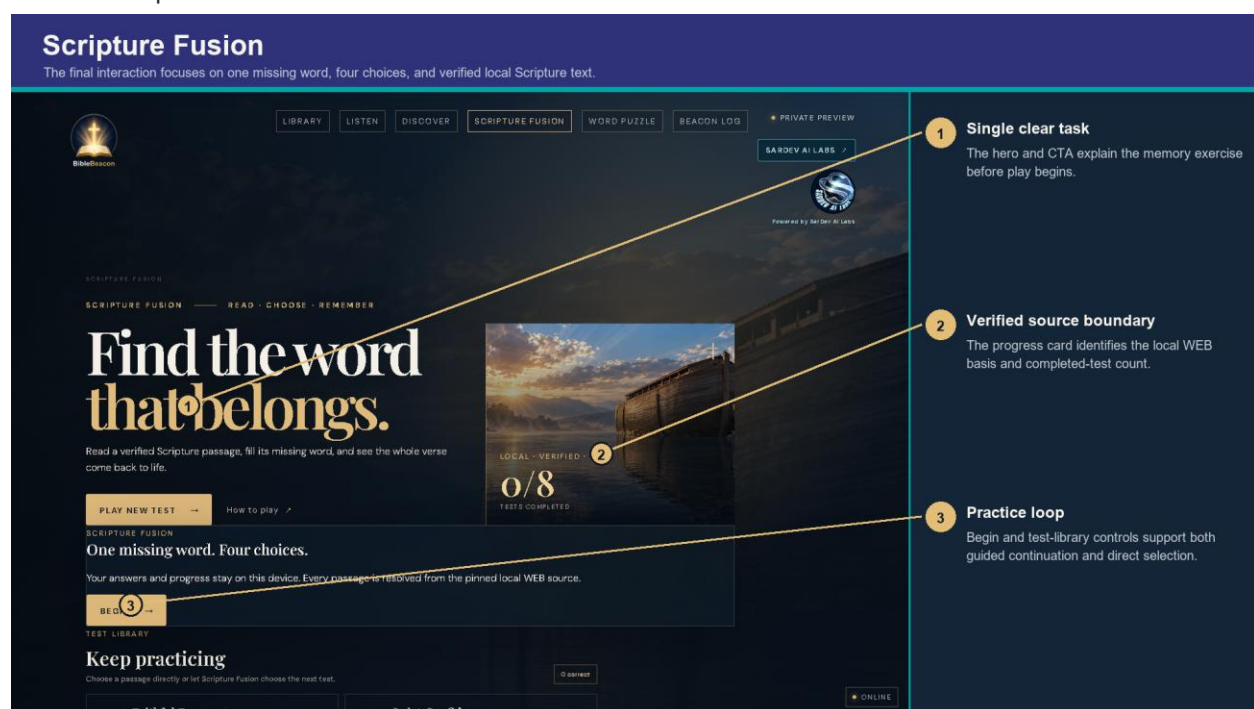


Figure 12. Scripture Fusion missing word experience.

4.5.10 Word Puzzle

- Ten local levels use word connect mechanics, grid placement, bonus words, hints, shuffle, coins, and sequential unlocks.
- Daily challenge, progress, settings, help, and Scripture rewards create several reasons to return.
- Puzzle state uses separate device local namespaces so Reader, audio, Discovery, and other game state remain isolated.

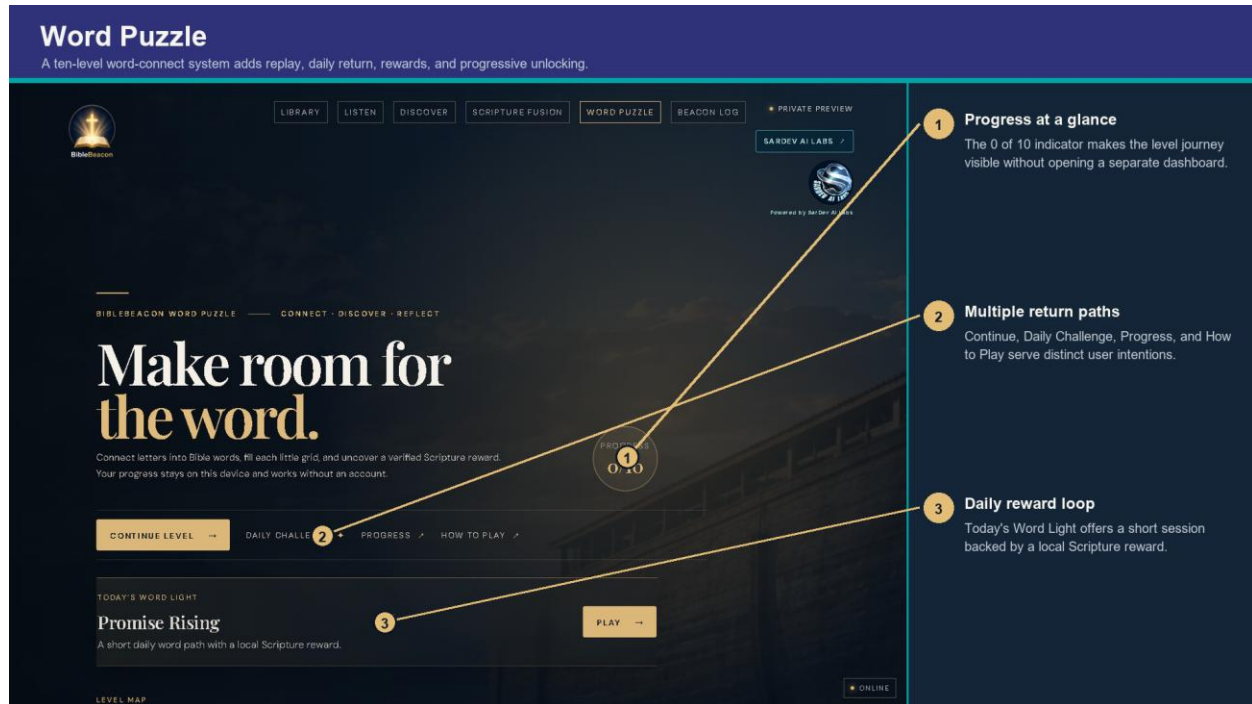


Figure 13. Word Puzzle progression and daily return paths.

4.5.11 Beacon Log

- Beacon Log brings together questions answered, accuracy, best score, sessions, streak, level, path mastery, review state, missions, quests, and recent history.
- Daily missions and weekly quests use deterministic local logic and cap progress at defined targets.
- The summary is a private device local view, not a server leaderboard or account profile.

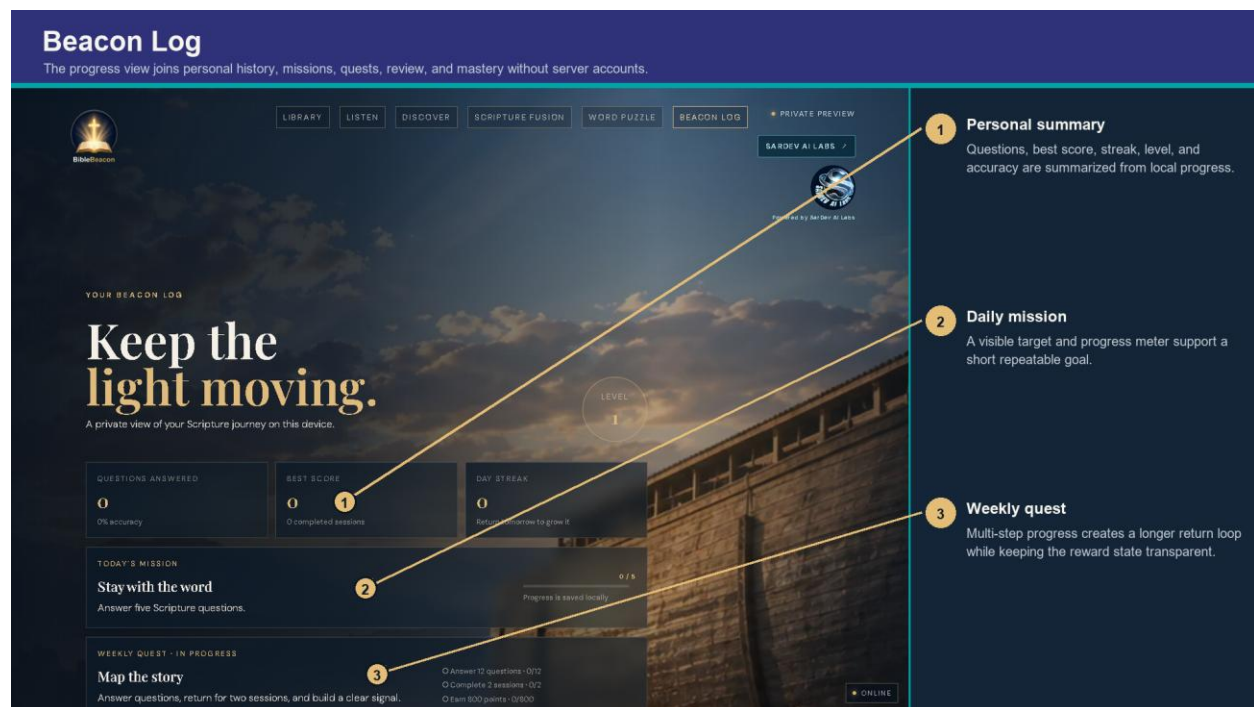


Figure 14. Beacon Log progress missions and quests.

5 Solution and Technical Approach

5.1 System Overview

BibleBeacon is a Vite and TypeScript single page application. A Node 20 compatible static server delivers the built interface and supports single page route fallback and HTTP byte range responses for audio. The browser loads bundled Scripture and same origin study data, registers a versioned service worker, and writes personal state to named local storage records. The separate production audio directory supplies the larger chapter recordings on demand.

The build time data workflow normalizes external sources into local JSON, records hashes and license metadata, and prevents the production browser from depending on mutable upstream repositories. The code includes explicit seams for a future sync provider and study entitlement model, but the verified release keeps synchronization inert and does not activate authentication, a database, Stripe, or paid APIs.

5.2 Architecture

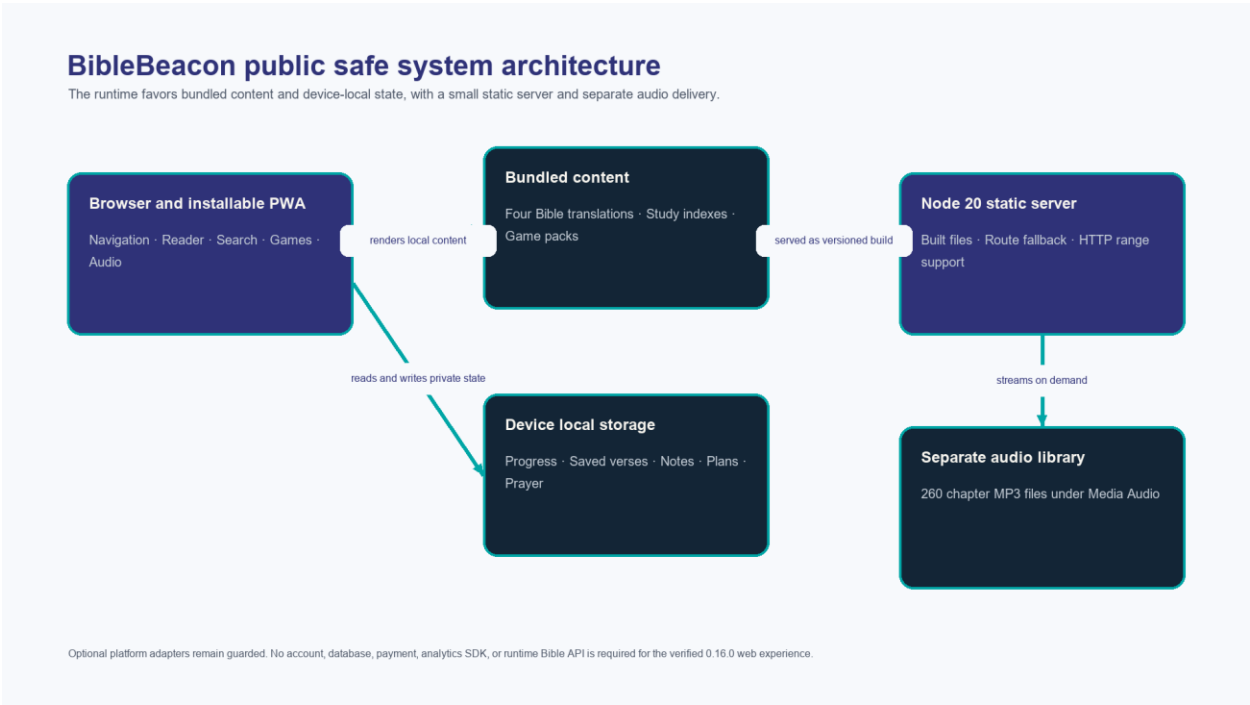


Figure 15. Public safe BibleBeacon system architecture.

5.3 Technology Selection

Layer or Service	Technology	Purpose	Reason Selected
Frontend	TypeScript and Vite	Build the responsive application and hashed production assets	Small dependency surface, typed source, fast static deployment
Runtime	Browser plus Node 20 static server	Render the app, route SPA requests, and support audio range requests	Works with the Hostinger deployment model documented by the project
Content	Bundled JSON and same origin study data	Provide Scripture, search, games, and study metadata	Offline friendly behavior with pinned provenance
Personal storage	Versioned browser local Storage records	Keep progress, saved items, plans, prayer, and settings on the device	No account or remote database required for the verified release
Offline	Web app manifest and service worker	Install the app shell and cache approved assets	Supports repeat use and graceful offline navigation
Hosting	Hostinger with versioned web packages and separate audio sync	Serve the public site and large MP3 library	Matches existing operational constraints and rollback workflow

Table 5. Verified technology choices and reasons.

5.4 Privacy, Security and Resilience

- The verified web release requires no account, remote database, payment, analytics SDK, or runtime Bible API.

- Personal progress, saved items, notes, plans, prayer, reflection, listening positions, and game state use explicit local namespaces and versioned migration logic.
- Import paths validate data and reject malformed or unsupported records before overwriting valid local state.
- Source manifests record URLs, revisions, licenses, attributions, and hashes for approved content; review required sources remain blocked.
- Versioned service worker caches and rollback archives support controlled release recovery; audio remains separated to prevent a web merge from deleting or replacing the large media library.
- These safeguards do not establish encryption at rest, penetration testing, formal compliance, high availability, or secure multi device synchronization.

5.5 Hard Problem and Correction

CHALLENGE AND CORRECTION

Problem: A web deployment that replaced the public root removed the separately managed audio path, causing an expected audio URL to return 404 after the PWA release.

Diagnosis: Release verification separated the small versioned web package from the 549 MB, 260 file audio library and traced the player path across local, legacy, and production locations.

Correction and verification: Later releases moved the production library to /Media/Audio/, preserved fallbacks, used merge only web uploads, synchronized audio separately, and verified all 260 expected URLs plus representative HTTP range delivery. This pattern became an explicit release guardrail.

Evidence basis: package.json; README.md; docs/STUDY_DATA_INTEGRATION.md; DEPLOYMENT_STATUS.md; RELEASE_0.8.0.md through RELEASE_0.11.0.md.

6 AI Assisted Development Method

6.1 Purpose and Boundaries

AI assistance was used as a supervised production accelerator. Appropriate tasks included comparing product options, turning requirements into bounded implementation steps, drafting interface and data models, generating or reviewing code and tests, analyzing failures, preparing release checklists, and assembling documentation. The product does not contain a runtime generative AI service and does not send prayer, notes, search queries, or personal progress to a model.

Keith retained control of the problem definition, acceptance criteria, product fit, source rights, privacy boundaries, and release status. AI output was treated as proposed work until the repository, automated tests, browser behavior, build, or live route supplied evidence. This report does not claim that every historical AI session was independently audited; it documents the human directed method supported by the surviving project artifacts.

6.2 Human Directed Workflow

1. Keith defined the product problem, desired user behavior, nonnegotiable constraints, and release boundary.
2. AI tools helped expand those instructions into options, task plans, working code, tests, review findings, and documentation drafts.
3. Keith evaluated the proposals against the product vision, visual direction, user flow, source rights, privacy boundary, and operational constraints.

4. Unsupported claims, unsafe source assumptions, brittle deployment steps, and poor fit interaction ideas were rejected or corrected.
5. The accepted work was tested and assigned a specific status: planned, verified locally, deployment ready, deployed, or live verified.

6.3 AI Contribution Register

Phase	Keith Trowbridge Direction	AI Assisted Task	Human Review	Evidence
Discovery	Set the Scripture product vision and requested features	Organize requirements, identify dependencies, surface risks and source questions	Selected the staged direction and deferred gated work	Expansion plan and phased prompts
Product design	Defined navigation, workflows, visual direction, privacy language, and acceptance criteria	Generate alternatives, interaction copy, edge cases, and responsive behavior proposals	Accepted, refined, or rejected options against the intended experience	Source UI, screenshots, release notes
Implementation	Approved scope and preserved existing behavior	Draft code, data adapters, migrations, route logic, styles, and test fixtures	Inspected changed files and required complete builds	TypeScript source and tests
Testing	Defined release gates and important journeys	Create automated tests, analyze failures, prepare browser and deployment checklists	Reviewed results and withheld stronger status when evidence was incomplete	Test output and deployment records
Documentation	Set public message, ownership statement, and evidence boundaries	Draft release records, source registries, diagrams, annotations, and this report	Confirmed public safety, factual accuracy, and publication boundary	Docs, annotated assets, report review

Table 6. Human direction, AI assistance, review, and evidence.

6.4 AI Tools Register

Tool or Model Class	Verified Use	Data Boundary	Validation
ChatGPT and Codex assisted workspace sessions	Product exploration, implementation support, review, testing, troubleshooting, release notes, screenshot annotation, and report preparation	Project files and public or licensed source metadata; secrets and credentials were not required for the documented local build and report	Repository inspection, automated checks, rendered UI, live route checks, and Keith's acceptance
No runtime model	BibleBeacon itself uses deterministic local data and application logic	No prayer, note, search, saved item, or progress text is sent to an AI service by release 0.16.0	Source inspection and dependency review

Table 7. Verified AI use and product boundary. Specific model versions were not consistently recorded and are not claimed.

7 Delivery Quality Assurance and Release

7.1 Delivery Phases

Release	Product Change	Verification Boundary	Status
0.3.0	BibleBeacon rebrand, daily challenge, XP, levels, achievements, feedback, replay, responsive and accessibility foundations	Build, typecheck, lint, local and live browser checks documented	Deployed and verified
0.4.0 to 0.7.0	Timeline, missions, Beacon Log, paths, adaptive review, geography, and weekly quests	Automated tests, local and live gameplay, mobile width, rollback records	Deployed and verified

Release	Product Change	Verification Boundary	Status
0.8.0	Installable PWA, versioned cache, offline shell, connection status	Route and browser verification; exposed audio removal risk	Deployed and verified
0.9.0 to 0.11.0	Sharing, Discovery, Listen controls, full Audio Bible library, resume, download, low data mode	Audio path and range checks; 260 recording validation	Deployed and verified
0.12.0 and local completion patch	Full Reader, search, Saved, Plans, Prayer, comparison, sharing, migration, future sync seam	Production 0.12 plus later local completion evidence; not every patch was immediately deployed	Mixed production and local verified history
0.13.0	Offline study data, source governance, four translation corpus	Import and integrity checks documented	Deployed history recorded
0.14.0	BibleBeacon Word Puzzle	Engine tests, desktop and mobile checks, production release record	Deployed and verified
0.15.0 to 0.16.0	Scripture Fusion, branding polish, final four choice missing word flow, browser history behavior	Local browser, live route, release metadata, service worker, audio regression checks	0.16.0 deployed and live verified

Table 8. Major release evolution from the verified repository record.

7.2 Validation Strategy

Quality control combined pure logic tests, data integrity tests, TypeScript checking, linting, production builds, packaged server route checks, desktop and mobile browser journeys, PWA route and cache checks, audio path and range tests, deployment archives, rollback snapshots, and direct production verification. Release notes recorded where a limitation prevented a stronger claim, such as an unavailable headless browser or work that had only been completed locally.

- **Functionality:** focused tests cover audio mapping, challenge links, content validation, library migration, reader links, plans, missions, quests, review scheduling, puzzle rules, Scripture Fusion, PWA behavior, and study data.
- **Content and provenance:** complete canon checks, source manifest hashes, local study index checks, and versioned source records validate approved data.
- **UX and responsiveness:** recorded desktop and 390 by 844 mobile checks cover key flows, layout overflow, focus and control behavior, and game completion states.
- **Deployment:** versioned packages and rollback artifacts separate web files from audio, and live release metadata, service worker, routes, and media behavior are checked after release.
- **Publication:** claims in this report are limited to the evidence above; user adoption, accessibility certification, and business impact are not inferred.

7.3 Verification Record

Check	Environment	Date	Result	Evidence
Automated tests	Current local repository	15 Oct 2026	Pass 47 of 47	npm test output
Typechecking	Current local repository	15 Oct 2026	Pass	npm run typecheck
Lint	Current local repository	15 Oct 2026	Pass with zero warnings	npm run lint
Production build	Current local repository	15 Oct 2026	Pass with bundle size warning	npm run build
Public home	Production HTTP	15 Oct 2026	HTTP 200	biblebeacon.sardev.io

Check	Environment	Date	Result	Evidence
Release metadata	Production HTTP	15 Oct 2026	HTTP 200 and version 0.16.0	/release.json
Service worker	Production HTTP	15 Oct 2026	HTTP 200 and cache biblebeacon-v0.16.0	/sw.js
Direct game routes	Production HTTP	15 Oct 2026	HTTP 200	Scripture Fusion and Word Puzzle URLs
Annotated feature captures	Local production build	14 Oct 2026	13 walkthrough plates plus architecture	Delivered asset folder

Table 9. Current verification performed for this report and supporting project evidence.

7.4 Current Status Boundary

Status	What Is Confirmed
Verified locally	Current 0.16.0 source passed tests, typecheck, lint, and production build on 15 October 2026.
Deployment ready	The current build produced versioned assets, PWA files, study data, and a synchronized 260 item local audio source catalog.
Deployed	Repository release records document Hostinger deployment of 0.16.0 and earlier incremental releases.
Live verified	The public home, release metadata, service worker, and direct Scripture Fusion and Word Puzzle routes returned HTTP 200 on 15 October 2026; release metadata identified 0.16.0.
Not yet verified	No current independent usability study, formal WCAG audit, penetration test, production analytics, business outcome study, or current Meta Horizon store status review is included.

Table 10. Clear separation of local, deployment, live, and unverified status.

7.5 Known Limitations

- The 15 October production check was an HTTP and release metadata verification, not a new exhaustive production browser walkthrough of every feature.
- Responsive QA evidence samples specific desktop and 390 by 844 mobile conditions and does not represent a full device laboratory.
- Accessibility features are implemented and tested in targeted ways, but no independent WCAG conformance audit is documented.
- The current build reports an 18.81 MB minified JavaScript chunk and a Vite chunk size warning; code splitting and data loading should be reviewed before claiming optimized performance on low bandwidth devices.
- Device local storage does not provide multi device synchronization, account recovery, server backups, or verified encryption at rest.
- BSB and any future study resources require current file level license review before broader redistribution or commercialization.
- The repository contains historical Meta Horizon status records, but the store status was not reverified for this report and must not be described as current.

8 Outcomes Evidence and Lessons

8.1 Verified Outcomes

The strongest verified outcome is product capability and release completeness, not a measured business or behavioral result. BibleBeacon now operates as a coherent Scripture product with several connected engagement depths, explicit local privacy boundaries, governed source data, regression coverage, rollback history, and a live public web release.

Outcome or Measure	Baseline	Current Result	Evidence Boundary
Product breadth	Lightweight Scripture challenge	Reader, search, saved items, plans, prayer, study tools, discovery, audio, multiple games, and progress system	Repository and annotated UI; not a usage measure
Automated validation	Early releases recorded smaller test sets	47 current automated tests passed	Current local run on 15 October 2026
Scripture coverage	Question and excerpt content	Four complete 66 book local corpora are present and hash checked	Tests and source manifests; BSB terms remain a review boundary
Audio coverage	No production audio after one replacement deployment	260 New Testament chapter sources validated and separately deployed	Current tests plus release records; live full set not rechecked in this report run
Release state	Incremental local and deployed builds	Public release metadata reports 0.16.0 and direct routes respond	HTTP verification on 15 October 2026
Business and user outcomes	Not measured	Not yet verified	No analytics, research sample, conversion, retention, or revenue evidence in the repository

Table 11. Verified outcomes and measurement limits.

8.2 Evidence Register

Claim	Evidence Source	Verified	Public Safe	Boundary
Keith directed product and release decisions	Requirements, phased prompts, release records, approval boundaries	Repository review	Yes	Describes decision role without inventing team or client authority
AI assisted bounded development tasks	Referenced ChatGPT context, Codex work, code, tests, documentation	Artifact and repository review	Yes	No model version or autonomous authority claim
Release 0.16.0 is public	Live release.json and direct routes	15 Oct 2026	Yes	HTTP and metadata check only
47 tests pass	Current npm test output	15 Oct 2026	Yes	Automated suite, not total quality proof
No account or remote database in verified web release	Source, dependencies, docs, tests	Current repository review	Yes	Does not prove absence of every infrastructure risk
Feature visuals match the application	Local production build screenshots	14 Oct 2026	Yes	Seeded device local examples are illustrative, not user data

Table 12. Evidence sources and publication boundaries.

8.3 Lessons

What Worked	What We Would Do Differently
Incremental releases with rollback artifacts protected existing behavior while the product expanded.	Establish a formal design decision log and stable product requirements baseline earlier so later reports require less reconstruction.
Local first data and explicit source manifests reduced runtime dependencies and made provenance reviewable.	Define the long-term bundle and lazy loading strategy before the corpus and study data make the primary JavaScript chunk exceptionally large.
AI assistance was effective for bounded implementation, test generation, review, and documentation tasks.	Use shorter task scopes, require evidence before status language, and record the exact AI tool and model when that detail matters for audit.
Separating web files from audio preserved the large media library after the deployment issue was understood.	Make the deployment manifest and protected media paths an automated preflight requirement before any platform upload.
A shared Library information architecture let several private tasks feel like one product.	Run structured usability sessions to test whether users understand the split between top level destinations and Library tabs.

Table 13. Evidence grounded lessons.

8.4 Next Steps

Priority	Next Step	Reason	Dependency	Status
High	Run a public safe publication review and approve the exact case study copy and screenshots	The report is ready for owner review but not self-publishing	Keith approval	Pending
High	Split large Scripture and study payloads into lazy loaded chunks and remeasure	Current build emits an 18.81 MB chunk warning	Architecture and regression plan	Planned
High	Conduct keyboard, screen reader, contrast, and multi device accessibility review	Current evidence is targeted rather than formal	Test matrix and remediation budget	Planned
Medium	Run moderated usability sessions across quick play, Library, and return workflows	No user research evidence currently supports usability or adoption claims	Recruiting and research protocol	Planned
Medium	Approve authentication, privacy, conflict, backup, deletion, and recovery design before cloud sync	Personal content is sensitive and local only today	Explicit architecture approval	Blocked pending approval
Low	Recheck the separate Meta Horizon status only if mobile store publication remains a goal	Historical status may be stale	Platform access and release decision	Not yet verified

Table 14. Recommended next steps and approval dependencies.

9 Publication Review

9.1 Public Safe Summary

BibleBeacon is a responsive Scripture web app and installable PWA that brings together reading, search, listening, discovery, personal study, and replayable learning activities. The product began as a

lightweight Scripture challenge and expanded through versioned releases into a broader local first experience. Its current information architecture keeps six primary destinations visible, while the Library organizes Read, Search, Saved, Plans, Prayer, and Study Tools into one predictable workspace.

I directed the product vision, requirements, user journeys, navigation, visual direction, acceptance criteria, privacy boundaries, and release decisions. I used AI tools to accelerate bounded research, design exploration, implementation support, code review, test creation, troubleshooting, and documentation. AI did not hold product or release authority. I reviewed the outputs against the intended experience, corrected defects and unsupported assumptions, and required evidence before work was described as verified or released.

BibleBeacon uses TypeScript and Vite, a Node 20 compatible static server, bundled Scripture and study data, an installable service worker, device local storage, and a separately managed chapter audio library. Core reading, search, saved content, plans, prayer, progress, and games do not require an account, remote database, analytics SDK, payment system, or runtime Bible API in the verified 0.16.0 web release.

The repository passed 47 automated tests, typechecking, linting, and a production build on 15 October 2026. On the same date, the public site, release metadata, service worker, and direct game routes returned HTTP 200, and the release metadata identified version 0.16.0. The evidence demonstrates delivered capability and release discipline. It does not establish adoption, conversion, revenue, audited security, formal WCAG conformance, or current Meta store publication.

9.2 Publication and Redaction Checklist

- ☒ Every factual claim has a traceable project source or a clearly stated verification date.
- ☒ Keith's work, AI assistance, platform services, and source providers are separated.
- ☒ Planned, local, deployment ready, deployed, and live verified statuses are not blended.
- ☒ Release number, public URLs, build scripts, and test count were rechecked on 15 October 2026.
- ☒ No passwords, keys, tokens, private account identifiers, or internal deployment paths are included.
- ☒ No customer data or private prayer, note, or saved item content is shown.
- ☒ Screenshots and diagrams are public safe and include captions and alt text.
- ☒ Accessibility, privacy, security, performance, and reliability claims remain within the evidence.
- ☐ Final publication approval by Keith Trowbridge has not yet been recorded.
- ☐ The final keith-trowbridge.com case study URL must be added after publication.

APPROVAL RECORD

Publication status: Public safe draft prepared for owner review.

Approved by: Not yet approved.

Approval date: Not yet recorded.

Appendix

A Artifact Index

Artifact	Purpose	Location or Identifier	Public Status
BibleBeacon source	Implementation and current verification basis	Project repository	Private source; summarized safely
Public web app	Live product evidence	https://biblebeacon.sardev.io/	Public
Release metadata	Current public version evidence	https://biblebeacon.sardev.io/release.json	Public
Release and deployment records	Version history, tests, deployment and rollback evidence	README, DEPLOYMENT STATUS, RELEASE files	Sanitized summaries only
Source registry	Licensing, attribution, revision, and runtime policy	docs/SOURCE_REGISTRY.md and source manifest	Public safe summary
Feature documentation	Scripture Fusion, Word Puzzle, study data, expansion and local patch evidence	docs folder	Public safe summary
Automated test suite	Functional and data integrity regression evidence	47 current tests	Results public safe
Annotated screenshots	Feature walkthrough for the portfolio case study	Delivered assets/annotated folder	Public safe draft
Raw screenshots	Unannotated source captures from the local production build	Delivered assets/raw folder	Public safe draft
Architecture diagram	Public safe system explanation	Delivered assets/annotated/architecture.png	Public safe draft

Table A1. Report evidence and publication status.

A.1 Annotated Screenshot Index

File	Feature	Alt Text Summary
01-home-annotated.png	Product entry	Home navigation, launch cards, and product ownership marks
02-library-read-annotated.png	Library and Read	Library tabs, reading selectors, comparison, and resume
03-search-annotated.png	Search	Query field, visible search scope, and Reader result path
04-saved-annotated.png	Saved	Private collection, filters, and export or import
05-plans-annotated.png	Plans	Plan choices, optional reminder, and daily progress
06-prayer-annotated.png	Prayer	Privacy statement, structured entry, and lifecycle controls
07-study-tools-annotated.png	Study Tools	Local index, source governance, and metadata search
08-listen-annotated.png	Listen	Audio coverage, chapter grid, and playback controls
09-discover-annotated.png	Discover	Theme filters, curated context, save and listen
10-scripture-fusion-annotated.png	Scripture Fusion	Single task, verified source boundary, and practice loop

File	Feature	Alt Text Summary
11-word-puzzle-annotated.png	Word Puzzle	Progress, return paths, and daily reward
12-beacon-log-annotated.png	Beacon Log	Personal summary, daily mission, and weekly quest
13-responsive-annotated.png	Responsive behavior	Wrapped navigation, stacked cards, and Library tabs at mobile width
architecture.png	Architecture	Browser PWA, bundled content, local storage, Node server, and audio library

Table A2. Delivered annotated asset index.

B Change Record

Document Version	Date	Change	Author or Approver
Version 01	15 Oct 2026	Completed the retained template with verified BibleBeacon design, technical, AI assistance, QA, release, evidence, publication, and artifact content; added annotated screenshots and architecture diagram.	Prepared for Keith Trowbridge review

C Definitions

Term	Definition
AI assisted	AI supported bounded work under human direction and did not hold product authority or release accountability.
Local first	Core content and personal state are bundled or stored on the current device rather than requiring a remote account or API for normal use.
Verified locally	Confirmed in a local development or test environment rather than production.
Deployment ready	Packaged and checked for release but not yet proven in the target production environment.
Deployed	Released to the target environment. Deployment alone does not prove every live behavior.
Live verified	Checked directly in production with recorded evidence and date.
Public safe	Prepared to avoid secrets, private records, internal paths, and unsupported claims; still subject to owner approval before publication.

D About the Author

Keith's life has been shaped by curiosity, adventure, service, and a willingness to step into unfamiliar territory. He is married to his best friend and is the proud father of two Auburn University graduates, both thriving in their respective careers. For all the places he has been to, and the experiences he has gathered, his family remains the center of his life.

His interests span, archaeology and biblical research, skydiving, base jumping, and rock climbing, writing and drawing, and design and development. Across those pursuits runs a common thread: a desire to explore, understand, and create. He is as drawn to questions about the ancient world as he is to the possibilities taking shape in the next generation of technology.

Keith has traveled to every continent except Antarctica, lived in the United States, India, and Israel, and spent more than 20 years immersed in international cultures. Those experiences have deepened his appreciation for different perspectives and the importance of understanding people within the context of their lives. He served in the U.S. Army as a team leader with a special operations unit under Joint Special Operations Command (JSOC) and later supported global anti-human-trafficking organizations through field-based roles.

Today, Keith brings that breadth of experience to his work in AI strategy, design, and development. He sees the present as one of the most extraordinary periods of technological change in human history—and approaches it with the same curiosity that has carried him across the world. His focus is on turning emerging possibilities into useful tools, thoughtful experiences, and businesses that solve real problems. He remains an explorer at heart, with more to learn, more to build, and more to contribute.



Keith Trowbridge

AI Product Designer and AI Strategy Consultant

SarDev AI Labs

keith@keith-trowbridge.com

linkedin.com/in/keith-trowbridge

keith-trowbridge.com