Chronos — Release and engineering update

21 October 2026 · Maintained by Zahidul Islam. The frontend and backend are deployed, and the complete source is merged into GitHub main. This dated update is the current release record; earlier technical material follows as preserved history. Live website · Merged release 30c16e5 · Public engineering documentation.

Before and after — verified release

Area	Before	After — 21 October 2026
AI guidance	Earlier help/local inference cohort; local task evaluation passed 3 of 8.	Approved remote route; 9/9 actual held-out tasks and 3 boundaries passed; local engine paused.
Runtime images	Earlier Node/web dependency cohorts.	Patched Node 22.23.3 and Nginx 1.30.4; zero reported vulnerabilities/secrets; one upstream build-style advisory retained.
Payment backend	Stripe SDK 16.6.0 installed.	SDK 20.3.1 immutable cohort; native rollback and test reconciliation passed without shared PHP restart.
Privacy behavior	Native identifier and database-read failure gaps.	Twelve native automated checks passed; accurate erasure counts and explicit read failures.
Public acceptance	Earlier dated storefront evidence.	222 commerce checks and 33 browser checks across three engines passed, including hosted TEST payment.
Recovery and parity	New patches initially unexecuted.	Guide journal-preserving restart and frontend rollback/resume passed; 140 frontend/611 guide files match artifacts.
Source and growth	Earlier publication draft and operating snapshots.	Complete source and nine READMEs merged; future 1M+ session and 99%/99.9% objectives have owned exit gates.

Architecture and AI guidance

React delivers pre-rendered pages; WordPress/WooCommerce owns content, accounts, prices and orders. The project-specific AI guide selects a validated topic and returns maintained answers and internal sources. It has no customer-record access or action tools. Local inference is paused following dependency findings; the approved remote route runs directly. Questions and a previous topic are processed remotely, with explicit disclosure, best-effort sensitive-input filtering and no question logging by the service. Page-only context supports stop/reset. Existing lifetime allowance USD 0.10, durable reservations, deadlines, concurrency and quotas remain enforced; exhaustion retains navigation and contact support.

Growth and availability objectives

The roadmap targets 1,000,000+ simultaneous sessions through measured stages of 500/1,000, 10,000 and 100,000 before the million-session gate. At a 300-second session and 30 requests/session, the example implies about 3,333 session starts/s and 100,000 edge requests/s. A hypothetical 99% hit ratio still leaves about 1,000 origin requests/s; it is an assumption. Private account/payment traffic and AI throughput need separate budgets. Scale through cacheable anonymous delivery, stateless replicas, controlled database writes, bounded queues, media offload and justified multi-zone recovery. Price regional compute, storage, traffic, backups and inference before approval.

Initial availability objective: 99% over 30 days, allowing 432 minutes of unsuccessful service; later 99.9%, allowing 43.2 minutes. Measure browsing, authentication, contact, checkout and AI separately. Representative load, owned alerts, continuous measurements, encrypted off-host restore and achieved RPO/RTO are exit gates. The present single-host boundaries and bounded AI service are not evidence of million-session capacity or an uptime history.

Security, privacy and legal operation

Current real account/contact/test-order data already needs an owner decision on controller applicability, lawful processing, vendor terms and justified retention/deletion; this is due now. Expanded markets and real sales additionally require applicable privacy/transfer, consumer, product-safety, tax, accessibility and merchant PCI review. Hosted payment entry remains Stripe test mode and no goods are shipped. AI disclosure and request privacy flags do not replace processor agreements or legal assessment. Follow the EDPB privacy guide, FTC order guidance and current EU AI Act guidance; the public compliance register maps applicability and unresolved work. Qualified review, manual assistive-technology testing, privilege separation and operating recovery remain explicit gates.

Source, verification and maintenance

PR #43 is merged; the superseded draft #41 is closed with its history retained. The release passed all 13 applicable PR checks and the merged-main workflows. Local checks include 40 Node tests, 47 PHP tests/94 assertions, 19 block tests, 66 Python tests plus two subtests, types, lint, builds and security scans. Hash-pinned certifi 2026.10.19 clears the integration runtime/development audits. Nine first-party READMEs now cover setup and handover alongside architecture, APIs, security, legal governance, operations, recovery and scale. Source/artifact transforms and retained formatting debt are recorded honestly. Zahidul Islam owns maintenance and approved knowledge; use Contact for escalation, without an unverified response-time promise.

Earlier editions — preserved technical reference

The dated material below is retained. Where earlier release status, open pull requests, dependency versions or pending work differ, the 21 October update above governs the current state.

Chronos — Comprehensive Technical Overview

A cinematic watch experience connected to WordPress and WooCommerce

Developed by Zahidul Islam

Latest update: 14 October 2026 • Earlier edition: Sep 15, 2026

Live experience: Chronos

Source repository: Chronos on GitHub

Open source since 24 Sep 2026: MIT (frontend) and GPL-2.0-or-later (custom WordPress plugins).

Chronos combines an accepted cinematic frontend with verified CMS content, customer workflows and test payments. This reference explains the current experience, its engineering, the evidence collected and the additional work required for real commerce or larger audiences.

Update — 14 October 2026: Dependency security repair

On 12 October, a separate reviewed dependency repair (GitHub pull request #42) was merged into main after the scheduled security audit had failed. Both full npm dependency audits then reported zero known findings. Fifteen security regression checks, 19 Gutenberg tests, frontend type/lint/build/prerender/release checks, all 11 pull request checks, and the merged-main Security, CI Pipeline and CodeQL workflows passed. The reviewed merge changed dependency tooling, retained the generated Gutenberg assets, and did not deploy the website.

As checked on 14 October, the separate publication pull request #41 remains open as a draft at its 12 October reviewed commit. Reconcile that branch with the merged dependency repair and rerun release gates before promotion. The verified deployed storefront and test-only checkout were not changed by this repository merge.

Update — 12 October 2026: Storefront, automation, public source and growth plan

Publication and release status

Chronos is maintained by Zahidul Islam. The reviewed source, frontend, WordPress plugins, advertising tools, inventory workflow and public documentation are published in https://github.com/Zahidulislam2222/Chronos/pull/41 at commit 3fbfaed8c1586db7726a28e4349d6d430dfe2b12. The pull request remains a draft: the full Gutenberg dependency audit has 48 findings originating from two upstream packages without a published fix, braces and sprintf-js. Main is unchanged; merge and promotion remain blocked. All six other underlying advisories have been resolved in the candidate lockfile. No audit or security hook has been disabled.

Current deployed website

The connected storefront has five photographed products, three complete journal articles, responsive information pages, grouped footer navigation and a header that hides while scrolling down and returns while scrolling up. Four unfinished source records are retained as drafts. WordPress and WooCommerce remain authoritative for content, prices, stock, accounts and orders. Test checkout uses Stripe's hosted payment page, server-resolved prices, ownership validation, idempotency and payment reconciliation. Physical fulfilment and live payments are outside the current service.

The 12 October release verification recorded 222 browser checks, 107 navigation checks, 33 HTTP responses, a completed hosted test payment, native CMS contact persistence and unchanged stock. All 99 packaged frontend files and two scoped backend files matched local SHA-256 hashes after deployment. The previous release was retained for rollback. These are dated functional and deployment results, separate from capacity or monthly uptime measurement.

Architecture and configuration

React 19, TypeScript 5, Vite 8 and Tailwind 4 provide the pre-rendered frontend. WordPress/WooCommerce, WPGraphQL, JWT authentication, the Chronos Bridge plugin and three Gutenberg blocks provide CMS and commerce behavior. A hardened Nginx container serves versioned frontend artifacts behind HTTPS and CDN delivery; the CMS/database is a separate service boundary. Modern browser requirements are Safari 16.4+, Chrome/Edge 111+ and Firefox 128+.

Public environment settings are validated in the typed frontend configuration and safe examples; every VITE setting is visible to visitors. Maintained JSON owns editorial source, navigation thresholds, catalogue eligibility, build settings and operational assumptions. Provider secrets, recovery details, database exports and private verification evidence remain outside the public repository. Source publication and installed runtime upgrades have separate acceptance checks.

Claude and Meta advertising integration

A local Python MCP server provides eight account-scoped tools, business context, reporting, durable before/after proposals and a separate interactive owner approval CLI. It makes no inference API call; the Claude host supplies conversation and reasoning under its own provider/data settings. Permitted writes create paused awareness campaigns or pause campaigns. Budgets, campaign activation, ad sets, creative publishing and autonomous optimization are outside the present scope. Actual Meta account connectivity and Claude attachment still require owner setup and verification.

The local suite passed 37 tests, including actual stdio MCP initialization and CLI approval behavior. Durable claims, account/configuration/before-state binding and an uncertain-write state protect retry handling. An interrupted or uncertain provider write requires reconciliation before a new attempt. Broader actions need new permissions, contracts, approval policy and real-account verification.

n8n and inventory pipeline

The privately installed n8n Community workflow reads existing published WooCommerce products, preserves supplied BDT prices, maps ordered multi-photo galleries, removes duplicate image URLs, skips unavailable/zero-photo products and processes item references sequentially. It queues internal social listing drafts for human review without duplicating WooCommerce products or publishing external listings. The imported workflow remains unpublished; manual source execution was verified. Supplier browser fetching, a custom median formula and branch geocoding are outside the selected WooCommerce-source flow.

The inventory suite passed 28 tests, including real authenticated loopback HTTP, pagination completeness, restart persistence, sequential processing, stale-batch rejection, malformed-URL handling and container configuration. Review fixes add writable-volume/internal-listener overrides, a persistent latest-batch marker, atomic reconciliation and bounded startup/validation errors. Those fixes require a separate drift-checked rollout to the installed service, followed by a fresh complete sync and manual workflow verification. Completed historical results remain readable; superseded pending items cannot overwrite current queue state.

Security and operating responsibilities

Independent source/setup/documentation review passed after the identified inventory and local-development exposure defects were corrected. Both local WordPress Compose variants now bind published ports explicitly to loopback and have a committed CI regression. Python types, lint, SAST, runtime dependency audits and wheel builds passed for both integrations. Frontend types, lint, dependency audit, build, prerender and release validation passed; PHP passed 38 tests with 60 assertions. Final Gutenberg lint/build and all 19 tests passed. For the reviewed source, GitHub recorded 12 successful checks, one npm-audit failure and four deployment jobs skipped. The full dependency audit remains the release blocker described above.

Security boundaries include server-side authorization, owner-scoped order checks, sanitized CMS HTML, explicit media/link policies, signed payment webhooks, secret separation, restricted containers and local/live parity checks. Browser JWT storage remains script-readable and is a future session-hardening item. A hosted multi-tenant advertising or inventory service would require separate tenant authorization, scoped OAuth, shared durable state, queues and concurrency design.

The maintainer owns code, review, dependency maintenance and artifact verification. The operator owns hosting, DNS/TLS, access/MFA, monitoring, alerts, backup/restore and incident response. The business owner owns product accuracy, rights, markets, fulfilment, taxes, platform approvals and legal review. Define primary/alternate contacts before promising response times. There is no staffed 24/7 on-call service today.

Availability objective and recovery

The initial service objective is 99% over a rolling 30-day window, with a later 99.9% stage. At 99%, the 30-day error budget is 432 minutes; at 99.9%, it is 43 minutes 12 seconds. These are objectives, not measured uptime or a contractual SLA. Independent external probes must record expected content, status and timeout; publish coverage and missing observations alongside results. Probe CMS/API, authentication, contact persistence and test checkout separately so a working cached homepage cannot hide a dynamic outage.

A single-observation probe is available; continuous independent monitoring and alert routing still need operation. Current single-origin/stateful failure domains and the lack of automated off-server backups remain reliability gaps. The roadmap calls for encrypted private off-server backups, restore drills and measured RPO/RTO. Backups must include the CMS database/uploads, release manifest, private settings, inventory state, n8n state and encryption key, and advertising proposal state. Restore into an isolated environment that cannot charge, message or publish. Public workflow artifacts must never contain production databases or private recovery material.

Incident handling records UTC timestamps, user impact, affected release, containment, preserved evidence, rollback, recovery proof, notification decisions and follow-up owners. Fix locally, run the gates, deploy scoped artifacts and prove hashes afterward. Freeze ordinary feature releases when the error budget is exhausted; review recovery and incident actions before resuming them. Project-local interruption checkpoints and the private master dossier preserve the timeline for another maintainer after a power loss.

Roadmap toward 1M+ simultaneous sessions

The target explicitly means one million concurrently active sessions, not registrations or daily users. With the maintained example assumptions of 300-second sessions, five HTML and 25 asset requests per session and 12 MB transferred per session, one million sessions imply about 3,333 session starts per second, 100,000 edge requests per second and 320 Gbps of traffic. Even a hypothetical 99% overall edge hit ratio leaves about 1,000 origin requests per second, exceeding the present aggregate origin guard. This model supplies sizing inputs; it is not a load-test result.

Stage 1 establishes monitoring, alert ownership, private backups, restore evidence, stable endpoints and a measured 99% reporting window. The next stages validate 10k, 100k and then 1M sessions with representative sustained loads, cold-cache/purge behavior, media/device/geographic mixes, failures and recovery. Define p95/p99 latency, error rate, freshness and correctness thresholds before testing. Measure API reads, authenticated operations, checkout writes and provider limits separately.

The target design combines explicit anonymous caching and tag invalidation, replicated immutable frontend artifacts, a stateless multi-zone PHP tier, persistent Redis/Valkey cache, managed HA MySQL and measured replicas/connections, verified WooCommerce HPOS, object storage/media delivery, search services and durable workers. Never cache personal accounts, carts, orders or mutation responses. Preserve authoritative stock reservations, order idempotency, webhook deduplication and reconciliation under retries and replica lag. Add observability, funded on-call and warm regional recovery before the later 99.9% stage.

Provision only after provider traffic/usage approval and a budget review covering edge requests and transfer, media, app workers, databases/cache/search, queues, backups, monitoring, load generators and staffing. Report measured cost per 1,000 sessions and per checkout. No new paid resources or distributed load tests were launched for this publication.

Legal, privacy, accessibility and platform governance

Maintain a market/feature register covering the operator, target countries, products, data purposes, lawful bases, processors, international transfers, retention, tax, consumer obligations and reviewers. Real sales require substantiated specifications and media rights, trader/contact details, total pricing and shipping promises, returns/refunds, warranties, complaints and applicable product-safety roles. Assess GDPR/UK GDPR, applicable US privacy/state breach laws, ePrivacy/PECR and marketing rules by actual activity and jurisdiction.

Under GDPR, supervisory-authority notification under Article 33 generally has a 72-hour requirement where applicable; Article 34 concerns communicating high-risk breaches to affected people without undue delay. Keep those decisions distinct. Retention periods need owner/legal approval by data class; identity-verified access/erasure must cover relevant stores, processors, backup expiry and legal holds. Existing contact exporter/eraser hooks support that process but do not replace it.

Hosted Stripe redirect reduces payment-data scope but does not remove merchant PCI duties or the need to protect the redirecting site. Accessibility targets WCAG 2.2 AA and requires manual assistive-technology review in addition to automated scans. Evaluate AI Act transparency and CRA software-security duties by role, product scope and applicable dates. Preserve MIT/GPL and third-party notices; n8n's internal-use license requires a separate assessment before offering hosted customer access.

The storefront currently has no Meta Pixel or marketing analytics script. Adding remarketing, customer uploads, custom audiences, profiling, new analytics or marketplace publishing requires fresh privacy, consent, account-scope, platform-term and owner-approval review. Actual legal/consumer/tax decisions belong to the business owner with qualified advisers.

Public documentation and next action

All maintained READMEs and the architecture, API, deployment, scalability, availability, threat model, security operations, compliance, verification and roadmap documents are included in the published review branch. New guides cover configuration ownership, reproducible testing and operations/recovery. Read the latest branch through https://github.com/Zahidulislam2222/Chronos/pull/41 while main remains unchanged. Resolve the two unpatched build dependencies through reviewed upstream fixes or a scoped toolchain replacement that preserves all 19 block tests, generated assets and actual editor behavior; rerun the full audit and review before merge/promotion.

Public verification: Source publication verification

Earlier project history follows. It is retained as a dated record; this update gives the current source and deployed-service boundaries.

Current evidence and scope

Chronos is a connected WordPress and WooCommerce storefront with test payments. The public catalogue, journal, pages and navigation read existing CMS records. Account login, customer order history, persistent contact intake and hosted Stripe test checkout are connected. Products and campaign visuals were created for this build; no physical goods are sold and no live payment is accepted.

The connected release has 78 verified deployment files and 25 rendered HTML documents, with 20 indexable canonical URLs. The backend runtime matches local across 86 files. A hosted Stripe test payment was completed and reconciled with a paid WooCommerce order without reducing stock. Capacity of 10,000–1,000,000 readers and 99% availability remain objectives, not measured operating achievements.

What changed in this engineering pass

- Connected the accepted cinematic interface to eight published WooCommerce products, four WordPress posts and eight published pages, preserving existing records.
- Added editable information pages and navigation; verified administrator View links and actual Gutenberg editor save/reload.
- Restored account login and order history, persistent contact intake, authoritative test checkout and verified payment state.
- Fixed guest-cart loss, interrupted-session restoration, historical-receipt cart clearing and false contact success responses.
- Added safe WordPress HTML/media/link handling, genuine missing/draft HTTP 404 responses and a current CMS sitemap.
- Verified publication and editor changes in the browser without rebuilding; preserved marked test content as drafts afterward.
- Deployed locally verified code with before/after hashes, rollback protection and independent review; synchronized the project evidence and stable public reference.

1. Reading Guide and Evidence Policy

“Verified” means a named command or real flow passed and its evidence was retained. “Implemented” identifies source behavior; “target” identifies work still requiring operational proof. “Connected storefront” means real CMS persistence and test payment integration, not a commercial launch. Legacy custom Watch, AI and analytics modules are distinguished from the storefront flows exercised here.

This edition supersedes the disconnected static-only status. Visual polish alone did not establish WordPress integration; the current claims come from browser, API, administrator, editor, payment and deployment checks.

2. Executive Overview

Chronos combines a cinematic luxury-watch presentation with a functioning headless CMS. The frontend origin serves built HTML and assets; the browser retrieves current WordPress and WooCommerce data. Dynamic editorial/product URLs also consult the CMS for publication status before the origin serves them.

The design therefore has two operating boundaries: cacheable frontend delivery and authoritative CMS/database/API work. Login, contacts and test orders exercise persistent backend behavior. Neither the earlier static workload model nor the current functional checks establish commercial transaction capacity.

3. Product Purpose and Client Experience

Dark surfaces, warm accents, serif display typography, macro imagery and restrained motion express the watchmaking theme throughout the hero, collection, gallery, navigation, bag and journal. The accepted campaign design remains intact while actual CMS records populate the collection and editorial routes.

Visitors can browse, search, filter, sort, inspect products, manage a selection, sign in, inspect their orders, submit a contact message and enter clearly labelled test checkout. Keyboard controls, reduced motion, labels, static alternatives and responsive layouts support the experience. No shipping or real purchase is promised.

4. Verified Implementation Snapshot

- Content: eight published products, four posts and eight published pages; three verification records restored to drafts.
- Rendering: 25 HTML documents and 20 indexable canonical URLs; initial HTML refreshes at build time.
- Deployment: 78 of 78 frontend release hashes and 86 of 86 backend runtime-file hashes match local; archive restoration also passed.
- Connected browser suite: 23 checks; administrator: four checks; editor: two checks; publish-without-rebuild browser flow: five checks.
- Payments: hosted Stripe test payment confirmed paid in WooCommerce; stock unchanged; repeated receipt checks passed.
- Code: PHP 38 tests / 60 assertions; Gutenberg 19 tests; frontend types/build/release checks and installed source security scan passed.
- Warnings: frontend lint has zero errors and 12 existing warnings; PHPCS has zero errors and seven warnings. Both npm dependency audits report zero advisories.
- Distributed capacity, continuous uptime, complete manual accessibility assessment and commercial-launch compliance remain unverified.

5. Architecture Overview

Visitor → CDN/frontend origin → rendered HTML and versioned assets → React → GraphQL/REST → WordPress/WooCommerce → test payment provider. CMS publication checks protect dynamic routes; an up-to-date backend-generated sitemap exposes published URLs.

WordPress owns editorial records, navigation and contact persistence. WooCommerce owns product/stock/price/order truth. Validated local content owns cinematic assets, campaign copy and interface messages. Typed settings own public environment values; server-side settings own payment credentials and operational limits.

6. Route and Rendering Model

Existing public routes have build-time HTML, canonical/title/social metadata and readable content. Account and transaction routes are excluded from indexing. The origin validates dynamic CMS routes, returning actual 404 for missing or draft content and failing closed when the CMS cannot validate a route.

New publications and edits appear through browser API reads and the live sitemap without a frontend rebuild. Their initial HTML snapshot is not automatically regenerated: a newly published route initially receives the application shell, and existing snapshots may contain older content until the next build. This is an explicit rendering limitation, not server-side rendering or incremental regeneration.

Canonical index-file and trailing-slash aliases redirect with query strings preserved. Sensitive files and absent assets return 404. People and crawlers receive the same route response.

7. Frontend State and Query Ownership

UI state owns filters, gallery selection, dialogs and the local bag. React Query caches CMS responses and scopes private order/receipt queries to the authenticated identity. Catalogue and editorial adapters follow all cursors rather than assuming the first page is complete.

Guest selections survive sign-in; sign-out clears the active local token and selection. Storage failure has an in-memory fallback. A failed viewer request does not itself revoke a token; explicit authentication rejection does. Browser JWT storage remains sensitive to any successful script injection.

8. API Adapter Layer

GraphQL serves paginated products, posts, identity and order reads. REST supplies site configuration, pages, contact persistence and test checkout operations. Typed settings centralize endpoints, mode, timeout, page size and refresh interval; eight frontend environment variables have documented examples.

Responses are validated where they establish UI contracts or persistence acknowledgement. Only editorial page lookup treats 404 as missing content. Contact success requires a positive saved record ID; HTTP errors and malformed success payloads preserve the user's note. Connected content failure does not silently substitute the built-in local catalogue.

9. WordPress Editorial Ownership

WordPress owns published pages, posts, media, saved blocks and navigation. Administrator View links for published products/posts/pages resolve to the frontend. Four information pages and the primary menu were configured while preserving existing CMS content.

DOMPurify sanitizes editorial HTML before React inserts it. Media origins and internal links are normalized, and unsafe schemes/forms/scripts are excluded. Actual editor save/reload and publication checks established the path from maintained CMS content to rendered frontend content.

10. WooCommerce Domain Ownership

WooCommerce is the authority for product identifiers, raw prices, stock and order totals. The storefront displays the configured BDT currency. Checkout resolves products and availability on the server and rejects invalid quantities; browser display values never set the payable amount.

These are catalogue records, not verified manufactured inventory. Tax, shipping, fulfilment, consumer contracts and real merchant configuration need separate business requirements before any live-commerce launch.

11. Custom Bridge Plugin

Chronos Bridge 2.1 connects site/page/navigation APIs, contact intake, checkout/payment coordination and administrative functions. Its plugin header and runtime version agree. PHP tests, real authorization checks and deployed-file hashes support the current integration claims.

The recovery corrected server pricing, privileged order-field writes, payment completion acknowledgement and duplicate-attempt handling. A source module's presence is not proof that every legacy option, cron task or third-party service has been exercised.

12. Bridge Module Ownership

- Content/API modules expose site, published pages, sitemap, route status and GraphQL extensions.
- Commerce modules resolve server totals, coordinate test sessions and verify persisted order/payment state.
- Contact/privacy modules own persisted intake, administrator access and existing export/erasure integration.
- Operations modules retain cache, scheduled-maintenance and database lifecycle responsibilities; their complete schedule behavior was not load-tested.
- Custom Watch, analytics and AI-assisted administration remain retained capabilities outside the verified storefront path. No paid AI call was made.

13. Gutenberg Block System

Watch Showcase, Watch Collection Grid and Contact Form are registered in the actual WordPress editor. A marked draft saved a custom block and retained it after reload. All 19 unit tests, block lint and asset compilation passed; 17 changed source/build files were synchronized with exact hashes.

Block attributes remain persistent contracts. Future incompatible changes need migration planning. The earlier patched SockJS HTTP/WebSocket check remains maintenance evidence; this release adds real editor integration. A complete matrix of WordPress versions and every custom-watch rendering combination is not claimed.

14. Catalog and Editorial Data Flow

WooCommerce products and WordPress posts/pages → paginated GraphQL or validated REST → normalized media/links and sanitized content → collection, journal and detail routes. CMS navigation resolves through the same frontend origin.

A marked page, post and product were published after building. All three became readable, the new product/post joined their lists and the sitemap changed without rebuilding. An editor-updated page also showed its new text. They were restored to drafts, with missing-page behavior verified separately.

15. Cart and Session Flow

The browser bag records a requested selection and validates restored product/quantity data. It does not reserve stock. Login preserves a guest selection, while checkout revalidates every item against WooCommerce. Stale prices and availability cannot establish order authority.

Checkout retry state stores a stable request identifier and item fingerprint. A paid receipt clears only its matching purchase selection. Returning to an old receipt preserves a later bag; the repeated-receipt regression verifies that boundary.

16. Checkout and Payment Boundary

Checkout is connected in Stripe test mode only. Authenticated customers submit product IDs and quantities; the server calculates the order and creates a hosted test session. The browser only follows the canonical Stripe checkout origin. Live-session identifiers and unauthorized requests are rejected.

A hosted Stripe test card payment was completed, the provider reported a paid test session and WooCommerce persisted a paid processing order. Inventory stayed unchanged and test order emails were suppressed. This is meaningful end-to-end test evidence, not settlement of real money or readiness to ship goods.

17. Payment State and Idempotency

A per-customer idempotency key, database lock and persisted order/session metadata coordinate retries. Return verification and signed webhook handling check ownership, session identity, amount, currency and provider payment state. The endpoint confirms persisted payment completion before reporting paid=true.

Repeated return verification converged on the same order. A different authenticated identity could not read the customer's payment result. Process interruption, provider/network failure and commercial reconciliation still require operational monitoring; functional replay tests do not establish every failure mode.

18. Contact and Customer Data Flow

Contact POST validates and rate-limits name, email, subject and message, then stores a record in the WordPress inbox. Actual administrator inspection confirmed persistence. Anonymous reads and unauthorized privileged operations were rejected. Missing/failed/malformed responses show failure and retain the unsent note.

Accounts, contact records and test orders contain personal data. The published privacy page describes these flows and offers a contact route for access/correction/deletion requests. Contact intake does not send email; no outbound verification message or paid AI processing was performed. A real business must establish retention, lawful purposes, processor terms, transfers and operating rights-request procedures.

19. Search Metadata, Analytics, and Structured Data

The frontend retains semantic headings, canonical/title/description/social metadata, escaped structured data and discovery controls. Website/page/article metadata and connected Product metadata describe the CMS records without merchant offers; campaign disclosures remain visible. The CMS sitemap reflects current published URLs.

Initial HTML snapshots refresh only on rebuild; client-rendered updates and new routes need JavaScript until new snapshots are generated. Search indexing and AI citations are not promised. Google's guidance does not require a special AI schema. Robots/noindex are discovery controls, not authorization. See [1]–[3].

No advertising or behavioral analytics integration was added. Public CSP and runtime checks guard unintended scripts; the no-transform policy remains in place.

20. Caching and Scheduled Maintenance

Hashed scripts/styles have immutable caching; media has a bounded lifetime and range support. HTML emphasizes freshness with no-transform. Error responses use no-store/no-transform. CMS/API and authenticated responses must not be treated as publicly cacheable personalized HTML.

The current connected path still reaches WordPress for browser data and dynamic route status. Cloudflare does not cache HTML by default. High-volume delivery needs measured anonymous cache behavior, safe invalidation and API/database capacity. Existing backend cache and scheduled modules do not establish those properties without runtime evidence. See [3].

21. Security and Privacy Boundaries

The frontend container serves only its built artifact, but the connected system now exposes CMS APIs and persistent account/contact/order flows. Authorization, input validation, server-owned prices, sanitized HTML, session ownership and payment verification therefore matter alongside static delivery controls.

CSP permits the configured backend for API/media traffic while restricting scripts to the same origin. Framing, workers, plugins and native form navigation remain restricted; inline styles support the component/motion system. The origin accepts GET/HEAD only and has sensitive-path, request-size and timeout limits. Backend APIs implement their own method/permission rules. See [4]–[5].

The non-root read-only frontend runtime drops capabilities and uses resource/log limits. Upstream HTTPS certificate verification remains enabled, with an explicit chain depth and bounded timeouts. These controls are not a full host penetration test, perfect bot protection or complete CDN-origin isolation.

22. Failure Behavior and Recovery

Media has static/poster fallbacks and reduced-motion handling. Denied storage avoids import-time crashes. Missing/draft routes return 404; CMS lookup failure fails closed. Contact failures retain text, interrupted viewer requests preserve the token until explicit rejection, and old payment receipts preserve later selections.

Immutable frontend releases and backend before-snapshots support rollback. Promotion verifies prior state, candidate behavior and final hashes; archive restoration was checked. One render attempt timed out on a backend page request and the retry passed, so backend latency remains an operational consideration rather than being hidden as a clean first run.

23. Test and Verification Evidence

- PHP: 38 tests / 60 assertions passed; PHPCS zero errors and seven warnings.
- Gutenberg: 19 tests, lint/build and actual editor registration/save/reload passed; both npm audits report zero advisories.
- Frontend: types, client build, 25-page render, release verification and source security scan passed; lint zero errors and 12 existing warnings.
- Connected compiled storefront: 23 browser checks; actual administrator: four; editor: two; newly published content: five.
- Persistence/payment: real hosted test payment, unchanged stock, repeated receipt three checks, account/inbox persistence two checks and authorization/input eight checks passed.
- Contact failures: missing endpoint, server error and malformed success all retained the note and displayed failure.
- Delivery: isolated actual Nginx route/header/alias/404/range/sitemap checks passed; deployed frontend 78/78 and backend runtime 86/86 hashes match.
- Independent source review passed after receipt, authentication and contact error handling corrections. Public-domain verification passed 16 content/login/responsive checks, three focused login/logout checks and 12 HTTPS checks; six accessibility scans recorded zero violations and the browser recorded zero runtime errors.

24. Test Evidence Interpretation

The evidence covers the connected storefront and named flows. Earlier cinematic/static tests remain historical design evidence, not a substitute for current CMS/API checks. Test payment success does not certify real fulfilment, live webhook operations or all provider failure modes. Automated accessibility scans do not replace manual assistive-technology assessment.

No real secrets were found in the scanned changed-source scope. Full-history scanning returned two vendored JWT documentation keys; exact upstream versions contain the same public examples. The raw history scan is not described as zero findings. Dependency audit counts are dated observations, and lint/PHPCS warnings remain visible.

25. Workflow and Deployment Discipline

Local code is the source of truth. Live files are inventoried and compared before changes; local-ahead changes are validated, deployed and rehashed. Runtime files match; local CI tests are not deployed as application code. Immutable frontend releases retain the previous release and automatic failure recovery.

GitHub workflows remain manually guarded and were not triggered as deployment evidence. Current frontend delivery uses the existing shared server; the WordPress backend uses its existing host. Historical workflow targets are not interchangeable with the current release process. Final repository changes are committed locally, with pushing left to the maintainer.

Update (24 Sep 2026): The repository is now public and open source. CI for PHP, Gutenberg blocks and the frontend, dependency audits and CodeQL now run automatically on every pull request and on pushes to the main branch, without secrets. The retired cPanel pipeline, which worked in production until that hosting expired in May 2026, and the superseded Cloudflare Pages deploy remain in the repository but are switched off. The project runs on VPS hosting, and releases stay manual and local-first with hash parity checks. The paragraph above is kept as the record of the 14 September state.

26. Operational Observability

The availability helper checks expected HTTPS response content and records timestamps; runtime logs are bounded. A static health endpoint alone does not establish CMS health, so the connected operating plan also needs catalogue/API, login and test-payment observations with appropriate non-destructive probes.

No independent always-on scheduler or continuous 30-day record was established. Monitor placement, missing-data treatment and escalation ownership remain necessary. Private checkpoints record before/after release identity, pending commands and recovery actions for power-loss handoff.

27. Verified Limitations and Risk Register

- Current frontend and CMS each retain host/database failure boundaries; no independently verified automatic failover is claimed.
- Large-reader capacity and 99% availability are targets; connected API and route-status traffic require new workload measurements.
- Initial HTML snapshots refresh on rebuild; new publications rely on browser rendering until then.
- Browser JWT storage is sensitive to script compromise; a future commercial session design needs separate review.
- Real merchant identity, processor contracts, retention, taxes, shipping, refunds and market/product facts are not supplied by this portfolio.
- Paid AI, outbound email, comprehensive legacy cron/analytics behavior, manual full accessibility review and host penetration testing were not exercised.
- Existing lint/PHPCS warnings remain documented; zero dependency advisories is not universal security certification.

28. Current Status Summary

The cinematic frontend is connected to WordPress content, WooCommerce records, account/order history, persistent contact intake and Stripe test checkout. Actual administrator/editor, browser/API, payment and release-parity evidence supports those statements.

Commercial operation, large-scale capacity and continuous availability remain separate objectives. The project shows the tested full-stack path while identifying the boundaries that still require business decisions and operational proof.

29. Developer Handoff

Start with README, verification, scalability and security documentation. Use lockfiles and the documented Node/PHP versions; run types, lint, tests, audits, build and release verification. Configure connected endpoints centrally; maintain CMS records in WordPress and cinematic/UI data in validated files. Never place privileged values in public build variables.

Trace a product through server price/stock resolution, authenticated order creation, idempotent test checkout and verified payment state. Trace an editor save/publication through safe frontend rendering and actual HTTP status. Trace a contact acknowledgement to its admin record. Keep credentials and operational recovery private; update the dossier before deriving public documentation.

30. Cinematic Frontend and Motion Engineering

The redesign uses macro watch imagery, campaign films, editorial rhythm and intentional product close-ups. Scroll-linked film seeking supports forward and reverse movement. Responsive variants, self-hosted fonts, optimized images, video posters and motion preferences support a usable experience when autoplay, video loading or animation is unavailable.

Product cards, navigation, buttons and the bag drawer follow the same visual language. Gallery failure recovery, focus return, menu containment and live reduced-motion updates were explicitly exercised. AI-generated imagery and films are disclosed as AI-generated; asset provenance and font/software licences remain part of the handoff. Technical polish is not presented as evidence that the watch specifications shown were manufactured or tested.

31. Dependency and Configuration Maintenance

The frontend moved to the updated Vite 7 and React Router 7 toolchain. Gutenberg tooling uses WordPress scripts 35 with refreshed packages. Patched transitive resolutions address advisories without suppressing audit failures or taking npm's suggested obsolete toolchain downgrade. Both npm trees report zero advisories at the recorded check.

Configuration ownership is explicit: typed settings for public environment values, validated content files for maintained product/copy/search data, and deployment configuration for runtime policy. JSON-LD serialization escapes unsafe delimiters. Private keys, real secrets, environment files, recovery records and personal test evidence are excluded from the public release. Eight public frontend variables are represented in typed settings and the safe environment example. Server payment settings and maintained form/UI content have explicit owners; no real secret was found in the changed-source audit.

32. US and EU Privacy: Applied Privacy Measures

The current privacy notice describes sign-in tokens and local bag storage, WordPress account/contact/order records, test payment processing and provider request metadata. Sign-out clears the active local token and selection. Visitors are asked not to reuse important passwords or submit sensitive information. Data-rights requests use the contact route. A retention/deletion operating schedule and processor arrangements must be established before real business use.

CNIL guidance distinguishes strictly necessary user-requested functionality from consent-requiring trackers. This is not a blanket exemption for all local storage. The storefront adds no advertising or behavioral analytics. Any future tracking feature requires reassessment before loading. See [7]–[8].

GDPR applicability depends on establishment, offering goods/services to people in the EU or relevant monitoring, rather than nationality alone. Where applicable, controller identity, lawful purposes/bases, recipients, retention, rights, transfers, processor arrangements and security/incident responsibilities must be established. The storefront does not invent a controller address, EU representative or DPO. See [9]–[10].

US state privacy rules depend on business and processing facts. California's published adjusted revenue threshold is $26,625,000, alongside other applicability tests; visitor count alone does not decide coverage. Other state laws and child-directed collection require separate review. The current storefront has no sale/sharing-for-advertising feature; that is not a future consent-management platform. See [11].

33. Consumer, Product, Accessibility and AI Law

US consumer protection: the custom product line and AI visuals are disclosed; fabricated reviews, testimonials and unsupported performance claims are excluded. Real merchandise sales require substantiated shipping promises and operational delay/cancellation/refund handling under the FTC's rules. Email marketing would require CAN-SPAM compliance and applicable regional consent analysis. No shipping or unsubscribe service is simulated. See [12]–[14], [18].

EU consumer protection: real distance sales require trader identity, product and total-price information, delivery/complaint arrangements and appropriate withdrawal information. The general withdrawal period is 14 days, subject to exceptions; goods have statutory guarantee rights distinct from optional warranties. These are documented launch obligations, not checkout promises. See [15]–[16].

Product safety: the GPSR introduces responsibilities and online-offer information requirements. Real watches need product classification, manufacturer/responsible-person information and applicable warnings; materials, batteries, connected features, waste rules and national requirements need facts and evidence. Tax/VAT, import duties, sales-tax nexus, trademarks and shipping markets also require business review. No compliance badge is used as a substitute. See [17].

Accessibility: US ADA and EU accessibility scope are considered without claiming that a government website rule automatically governs every private website. Keyboard controls, semantic content, labels, alternatives, reduced motion and contrast improvements are implemented. Automated scans found no violations in the tested pages; manual screen-reader assessment and applicable national-law review remain necessary. See [19]–[20].

AI-generated media: visible AI disclosures support transparency. EU AI Act duties vary by provider/deployer role and content; no claim of cryptographic provenance or full provider compliance is made. US copyright protection depends on human authorship; generated assets are not described as universally exclusive or fully cleared. Preserve model terms, source licences and provenance before commercial reuse. See [21]–[22].

34. Architecture for 10k–1M Simultaneous Readers

The frontend assets and build snapshots can be replicated, but the active browser catalogue, identity/contact/payment operations and dynamic route checks depend on WordPress and its database. The previous static model is therefore only an asset-delivery scenario. It is not a measured capacity model for this connected release.

The maintained example workload assumes a 300-second session, five HTML requests, 25 asset requests and 12 MB transfer. Its arithmetic gives about 1,000 edge requests/second at 10,000 sessions and 100,000 requests/second plus 320 Gbps at 1,000,000 sessions. These are assumptions, not benchmarks or purchased capacity. They omit newly active API/route-status traffic.

Even a hypothetical 99% cache-hit ratio leaves substantial origin traffic; the present 100 requests/second frontend guard does not provide million-reader capacity. A real plan needs anonymous API caching/invalidation, bounded database connections, route-status caching with publication correctness, media bandwidth, independent origins and CMS/database resilience. Personalized and transactional data must stay out of shared caches.

Validation stages remain bounded local tests, isolated representative origin/API tests, distributed 10k, 100k soak/cold-cache/failover and a separately funded 1M test. Define geography, think time, API mix, cache state, latency/error thresholds and recovery. No shared-server load test or paid load service was launched.

35. 99% Uptime Objective and Evidence Plan

The objective is 99% end-to-end availability over a rolling 30-day window. That corresponds to a 432-minute error budget, or 7 hours 12 minutes. Success means expected HTTPS content within the configured timeout, not merely an open network port.

The probe supports a configured 60-second interval when run by an independent scheduler. Reporting must include observation coverage and treat missing intervals as unknown. A handful of successful manual checks is not a measured uptime percentage. No continuous 30-day record or contractual SLA is claimed.

A complete resilience verification requires a second failure domain, matching release hashes, controlled traffic failover, measured recovery time and safe primary restoration. Health checks, process restart and rollback are implemented foundations, but cannot remove a single-host outage by themselves. Any additional paid hosting, routing, monitoring or traffic commitment requires separate approval.

Appendix A. Representative End-to-End Traces

Browsing: WooCommerce/WordPress records → paginated API → validated/sanitized view → collection, journal or page → local selection.

Editorial: actual editor save → publication → current route validation and sitemap → browser fetch; rebuild later refreshes initial HTML.

Test commerce: authenticated product IDs/quantities → server totals/stock → idempotent order/session → hosted test payment → owner/amount/currency verification → persisted paid order.

Contact: validated POST → saved inbox ID → success acknowledgement → administrator-only record inspection; errors preserve the draft.

Release recovery: snapshot/compare → local edit/gates/review → isolated candidate → guarded promotion → public checks and hashes → previous release retained.

Appendix B. Data and Authority Matrix

- Campaign visuals, product ideas and UI messages: validated maintained local content.
- Editorial pages/posts, menu and saved block attributes: WordPress.
- Products, prices, stock and test orders: WooCommerce.
- Payment status: test provider verified against the owning WooCommerce order.
- Contact records: backend persistence with administrator access.
- Filters, dialogs and temporary selection: browser state; never financial authority.
- Public endpoints and limits: typed configuration; privileged values remain private/server-side.
- Capacity and uptime claims: measured evidence; design objectives remain labelled.

Appendix C. Developer Review Checklist

- Run frontend and block audits, types/lint/build/release checks, PHP tests/standards and source scanning.
- Exercise real CMS products/posts/pages, editor save/publication, login/logout, order history, contact persistence and test checkout.
- Include negative authorization, missing/malformed persistence responses, repeated receipts and interrupted navigation.
- Verify draft 404, published routes, sitemap, CSP/media permissions and responsive/accessibility behavior.
- Preserve local-first drift checks, candidate validation, rollback and exact final runtime hashes.
- Reassess session design, retention, provider failure recovery and business obligations before real commerce.
- Keep dossier, public reference, canonical PDF and private recovery checkpoints synchronized.

Appendix D. Client Review and Acceptance Checklist

- Inspect the live cinematic presentation, collection, journal, information pages and mobile layout.
- Use the separately supplied private administrator access to inspect products, posts, pages, contact records and test orders.
- Confirm CMS changes appear in the frontend; understand the separate rebuild requirement for initial HTML snapshots.
- Use only test credentials/card details in checkout; no real goods or live charges are offered.
- Review actual verification results and remaining commercial/operational requirements.
- Treat large concurrency and 99% uptime as objectives until representative evidence exists.

Appendix E. Primary Sources and Further Reading

The following primary sources support the legal and technical analysis. Applicability depends on the real business and deployment facts; this report does not certify universal US/EU compliance.

[1] Understand JavaScript SEO basics

[2] AI features and your website

[3] Default cache behaviour

[4] Content Security Policy Cheat Sheet

[5] HTTP core module; request limit module

[6] Unexpected external redirect advisory; SSR constructor injection advisory

[7] Cookies et traceurs: que dit la loi?

[8] Cloudflare cookies

[9] Guidelines 3/2018 on territorial scope

[10] Regulation(EU)2016/679, GDPR
[11] Adjusted monetary thresholds; FAQs
[12] Advertising and marketing
[13] Final rule banning fake reviews and testimonials
[14] Business guide to the Mail, Internet, or Telephone Order Merchandise Rule
[15] E-commerce, distance and off-premises selling
[16] Consumer guarantees
[17] General Product Safety Regulation summary; GPSR questions and answers
[18] CAN-SPAM compliance guide
[19] Guidance on web accessibility and the ADA
[20] Accessibility of products and services
[21] Transparency obligations under Article 50 of the AI Act; Regulation(EU)2024/1689
[22] Copyright and artificial intelligence

Appendix F. Repository References

Verified release results
Scalability and availability architecture
Security operations
Search, security and legal research

The public deployment is verified. Newly prepared repository documentation becomes available at these branch links after the maintainer pushes the local commit.

Update (24 Sep 2026): The documentation was merged into the main branch. Current versions: Documentation index · Architecture · API reference · Deployment and CI/CD · Availability and disaster recovery · Roadmap · Threat model · Legal and compliance readiness. The branch links above are kept as the record of the 14 September state.

Developed by Zahidul Islam • Cinematic frontend craft, explicit system boundaries and verifiable engineering.