FTM Social Media Automation

Generation, Client Review, Internal Approval, and Delivery — Technical Overview

A public technical edition for client and developer review. It documents the verified workflow, architecture, state transitions, reliability lessons, and handoff controls without exposing credentials or private infrastructure details.

Static Web Interfaces | n8n Orchestration | Media Library | Media-Aware Background Copy | Google Sheets and Drive

Reconciled August 2026 · Developed by Zahidul Islam

Last reconciled: 2026-08-11

1. Snapshot

- Project: Fine Touch Marketing Social Media Automation
- Stage: generally enabled production platform with the Media Library, media-aware client campaign workflow, and background copy generation live
- Primary outcome: ingest approved client media, build a dated campaign, let the client select media and dates, generate media-aware copy, collect client feedback, complete internal approval, and package authorized clean assets for manual publishing
- Current interface: custom admin dashboard, protected Media Library workspace, and an account-scoped client campaign dashboard with calendar, media selection, copy preparation, review, and approval stages
- Automation: n8n workflow suite plus server-side Python/PHP services for media processing, categorization, protected previews, background caption generation, review, approval, and packaging
- Operational data: Google Sheets tabs for the established client/post system plus a dedicated Media Library database for assets, campaigns, selections, revisions, approvals, and delivery records
- Latest recorded release evidence: Phases 10 and 11 completed on 2026-08-08; the 2026-08-11 media-aware dashboard release then passed two 50-post paid HTTPS runs, two fresh media-aware client runs, a real Chrome calendar approval, protected-thumbnail checks, deployment parity, and link cleanup
- Publishing boundary: approved assets are downloaded for manual scheduling; direct social-network publishing is not the current verified product path

2. Problem and Product Intent

Fine Touch Marketing needs a repeatable way to create social content for multiple small-business clients without manually designing every image and writing every caption. Team members onboard businesses, the system researches each brand, AI generates scheduled post ideas and assets, clients review them through a simple portal, and Fine Touch Marketing performs final approval before downloading clean deliverables.

The trial architecture deliberately validates the workflow and commercial process before committing to a full multi-tenant SaaS rewrite. The main product risk is not whether content can be generated; it is whether clients reliably review, request changes, and approve content through the portal.

2A. Media Library and Campaign Setup

The Media Library extends the established generation and approval system with a media-first campaign path. It is designed for clients who already have product photography, short videos, brand artwork, or other approved source material and want Fine Touch Marketing to turn that material into a structured social campaign.

Media intake and processing

- Staff can import a controlled bulk pack or add loose media to a client library.
- Originals remain in private storage and are never exposed as public review files.
- Processing validates file type and size, creates browser-safe derivatives, extracts useful metadata, and records a durable asset entry.
- Staff classification and campaign selection use explicit asset state; a filename or successful upload alone is not treated as approval.
- Image and video assets are supported in the released workflow. Unsupported or incomplete files remain visible for staff correction instead of silently entering a campaign.

Campaign planning

- Staff create a campaign for one authorized client and choose the eligible library assets.
- The campaign defines its date range, cadence, content direction, and review context before client scheduling begins.
- Client-facing links are scoped, revocable, and separate from administrator sessions.

- The client selects dates and media in a guided schedule rather than receiving an unstructured folder of files.
- Every saved choice is checked against campaign scope, asset eligibility, and current state on the server.

Copy and prompt workflow

The new campaign path prepares copy and creative direction from the selected media. It does not create replacement imagery as part of normal campaign generation, so the client-approved source asset remains the visual authority. The workflow produces a reviewable caption, hashtags, and prompt or creative notes where appropriate while making zero new image-generation calls for the released media-first path.

Two-stage review and delivery

The client can review the scheduled campaign, edit written content, approve an item, or request a revision. Fine Touch Marketing then completes a separate internal review and can revise or approve the exact current version. Only internally approved campaign items enter the final package. Delivery uses the clean authorized source associated with that version, and the package includes the corresponding text rather than relying on directory enumeration or browser state.

Release state

Phase 10 completed formal acceptance against all defined criteria. Phase 11 then exercised a controlled media pack and an existing-client browser pilot through schedule selection, video selection, content preparation, client review, staff revision, final approval, and link revocation. The release switch was disabled and restored during a real rollback rehearsal, the established system continued to pass regression checks, and the Media Library was enabled again only after final verification.

The post-release media-aware client flow was deployed and reconciled on 2026-08-11. Completing media selection now starts background caption work automatically; the copy model receives the selected image, or a safe poster frame for video, as visual context while the media-first path continues to record zero image-generation calls. The release was resumed from an interrupted partial deployment only after a read-only drift audit identified the exact files that were local-ahead. Final verification recorded all 38 scoped release files in sync, with no local-ahead, missing, or live-ahead file.

Client information transparency

The authenticated client campaign dashboard includes a persistent read-only summary of the information Fine Touch Marketing has on file: business name, email, phone, website and social/store links, campaign name, start and end month, campaign length, posting cadence, daily cap, and total planned posts. A missing value is shown as N/A, allowing a client to identify an absent profile or contact detail instead of seeing an unexplained omission. The summary remains visible as the client advances through scheduling, media selection, content instructions, and review.

This display does not publish internal setup choices or operational records. Dashboard credentials, token or hash material, internal ownership and status, brand-research notes, staff details, and server paths remain outside the client response. Website and profile values become clickable only when they are valid HTTP or HTTPS addresses, and client-supplied values are rendered as text rather than executable markup.

Developer component map

The released setup has five cooperating surfaces:

- The administrator Media Library view manages import batches, asset readiness, filtering, campaigns, review status, and final packages.
- The protected server gateway validates administrator sessions, client ownership, campaign scope, allowed actions, file paths, and release state before calling library operations.
- Background processing performs bounded normalization and metadata work while retaining a durable batch record that staff can inspect.
- The campaign portal resolves a revocable client link and exposes only the dates, eligible media, and current review records for that campaign.
- The established n8n system continues to own the original brand-research, generated-post, approval, notification, and delivery workflow; the Media Library is an additive path, not a replacement.

The main persisted entities are media assets, import batches, asset categories, campaigns, campaign slots, client selections, generated written content, revision history, approval decisions, delivery packages, and revocable review links. Identifiers connect these records without allowing a browser-provided identifier to bypass ownership or state checks.

Operational controls

The feature has a server-side release switch and a matching administrator-interface switch. The server is the authority: hiding the interface alone does not disable access. Deployment changes follow server-first enablement and interface-first disablement so a visible control is never presented before its protected runtime is ready. Local and deployed artifacts are compared after release, and rollback is verified by observable behavior plus regression tests rather than by configuration text alone.

2B. Media-Aware Client Flow and Background Copy

The released client journey is now one continuous account-scoped flow rather than a series of disconnected forms:

1. Fine Touch Marketing creates or opens a campaign and generates a durable client dashboard link.
2. The client chooses campaign dates in the calendar and selects eligible image or video media from the protected library.
3. A campaign-wide instruction is collected on the selection screen; the duplicate instruction step was removed.
4. Completing selection enqueues caption work automatically instead of keeping the browser request open.
5. A background worker generates caption and hashtag records for each selected post, while the dashboard polls bounded status and renders results as they become reviewable.
6. The client reviews and approves in the calendar; Fine Touch Marketing retains the separate internal approval and delivery boundary.

Visual grounding without replacement imagery

The copy provider receives the selected image bytes, or a video's resolved poster frame, as visual evidence alongside the campaign and brand context. Media paths are resolved and confined inside the configured private media root before any file read. The provider remains copy-only for this path: the deployment evidence records image_calls=0, so visual grounding does not silently replace or regenerate the client-selected media.

Durable asynchronous execution

Copy work is represented as persisted jobs with stable campaign/post scope and idempotent completion behavior. A per-minute worker processes the queue, writes captions, hashtags, usage, and terminal status back to the operational store, and leaves unrelated scheduled jobs intact. The live scale proof created two fresh 50-post campaigns and generated 100 real

captions in total; both runs completed with every item read back, every post moved to review, usage recorded, and no image-generation call.

Responsive protected media

Media labels and controls paint immediately while protected thumbnails hydrate through short-lived preview tickets. Tickets are reused where safe, and a generation-aware server cache avoids repeating expensive protected-preview work without making a private file public. Campaign dashboard URLs persist in account-scoped browser storage across navigation, reload, and a closed tab; revocation still removes the link from the interface and the backend remains the authority.

3. User Roles

Administrator

- Sees all clients and live status counts.
- Opens the Media Library, creates dated campaigns, and generates client dashboard links that remain available across normal navigation.
- Reviews upload readiness, categories, selected media, copy-job progress, and final campaign state.
- Creates, edits, archives, and re-scrapes client records.
- Starts content-generation jobs and sends approval links.
- Reviews client-approved posts and change requests.
- Approves individually or in bulk.
- Requests AI edits or uploads replacement images.
- Downloads approved clean assets and accompanying copy.
- Manages team accounts and notifications.

Team member

- Signs in through the same protected admin surface.
- Creates and manages only clients within the authorized ownership scope.
- Cannot inherit unrestricted administrator visibility merely by calling a shared endpoint.

End client

- Opens a revocable, account-scoped campaign dashboard link.
- Selects eligible media, chooses scheduled dates, provides one campaign-wide instruction, and starts background copy generation.
- Reviews generated captions and hashtags beside the selected media in a calendar-oriented interface.
- Sees watermarked preview images during review.
- Approves individually or in bulk.
- Edits captions and hashtags.
- Requests AI changes with written feedback or uploads a replacement image.
- Receives download access only when the Fine Touch Marketing approval state permits it.

4. Current Architecture

Admin / team dashboard | v Public webhook proxy and protected API boundary | v n8n workflow suite <----> Google Sheets state and Media Library database | +----> website and social brand research +----> AI copy and image generation +----> private media intake, processing, and categorization +----> campaign scheduling, protected media selection, copy jobs, and review +----> per-minute background copy worker and usage records +----> image editing and protected asset utilities +----> approval, notification, and ZIP workflows | v Review derivatives + protected clean assets | v Client dashboard -> schedule and media selection -> background media-aware copy -> client approval -> admin approval -> clean download

The browser applications are static HTML, CSS, and JavaScript. They call a public proxy boundary, which forwards approved routes to n8n and the protected Media Library service. n8n owns the established orchestration and state transitions. Server-side Python/PHP utilities handle media ingestion and normalization, protected-preview tickets, asynchronous copy jobs, image generation/editing, source research, clean-asset access, and archive creation. The Media Library runtime is isolated behind explicit authentication, ownership, campaign-scope, state, path-confinement, and release checks so it can coexist with the established workflow without weakening it.

5. Repository Map

- web/ftm-admin/index.html: - current consolidated admin and team dashboard source.
- web/ftm-portal/index.html: - established generated-post approval portal source.

- web/ftm-library/index.html: - protected staff Media Library browser application.
- web/ftm-client-dashboard/index.html: - current media-first client schedule, selection, generation, and review application.
- web/ftm-library/api/: and `web/ftm-library/lib/` - authenticated Media Library API boundary, preview service, session checks, storage, and campaign-link helpers.
- scripts/lib/: and `scripts/media_library/` - media repository, campaign scheduling/copy/packaging services, background workers, deployment, drift, and integrity tooling.
- admin/: and `portal/` - earlier split frontend sources retained for history; do not assume they are the deployed authority.
- n8n/: - workflow source JSON, payload snapshots, and selected backups.
- scripts/: - operational helpers for workflow inspection, image generation/editing, watermarking, deployment, and cleanup.
- deploy/: - deployment and repair scripts from successive implementation sessions.
- BUILD_PLAN.md: - original system design and workflow plan.
- BUILD_PLAN_SESSION6.md: - reliability, notification, and management expansion plan.
- E2E_TEST_PLAN.md: - full role, generation, approval, download, visibility, and cleanup test sequence.
- SESSION*_PROGRESS.md: - chronological implementation and incident evidence.
- PROGRESS.md: - early consolidated build state; later session files supersede parts of it.

6. Workflow Responsibilities

The workflow filenames are more stable and safer to document than environment-specific workflow IDs.

Access and team management

- wf_login.json: - authenticates admin and team users and returns role-scoped session context.
- wf_add_delete_team.json: and `wf_team_api.json` - add, list, and remove team accounts.
- wf_get_clients.json: - returns clients filtered by role and ownership.
- wf_add_client.json: - validates input, creates a client record, and starts brand research.

Research and generation

- wf_scrape_brand.json: - researches website and configured social sources, then stores a structured brand profile.

- wf_generate_content.json: - schedules post slots, creates caption/hashtag content and image prompts, generates images, watermarks previews, stores clean assets, and appends post records.
- wf_ai_regeneration.json: - applies client/admin change instructions to existing content, with image-edit and regeneration fallbacks.

Client review

- wf_send_approval.json: - prepares the approval stage and returns a client portal link.
- wf_portal_posts.json: - returns only the posts visible to the portal client.
- wf_portal_action.json: - handles approve, reject/change-request, and text-edit actions.
- wf_client_upload_image.json: and `wf_accept_client_upload.json` - validate and apply client-supplied replacement images.

Internal review and delivery

- wf_get_awaiting.json: - returns client-approved posts awaiting Fine Touch Marketing review.
- wf_admin_approve.json: - allows final approval, requests changes, or dismisses a proposed change.
- wf_admin_upload_image.json: - applies an administrator-supplied replacement image.
- wf_download_zip.json: and `wf_portal_download.json` - package approved clean images and copy for authorized download.
- wf_notify_download.json: - records or communicates download-ready events.

Shared utility surface

- wf_utility_api.json: - shared operational actions used by the dashboards.
- Additional edit, delete, notification, and image workflows may exist as deployment-era variants; reconcile them with the live server before editing.

7. Data Model

Clients

Core fields include a stable client identifier, business identity, approved website/social inputs, contact metadata, content frequency, keywords/offers, structured brand profile, ownership, approval state, and timestamps. Sensitive access material must not be placed in client-readable fields.

The brand profile is stored as structured JSON and currently carries services, tone, target audience, location, colors, and social-style evidence used by later generation.

Posts

Core fields include a stable post identifier, client identifier, clean and watermarked image references, caption, hashtags, scheduled date, status, change comment, version, timestamps, and approval evidence.

Watermarked and clean image references serve different trust boundaries. The portal uses the watermarked review asset. Final authorized delivery uses the clean asset.

Users

Stores role, identity, ownership scope, active state, and authentication material appropriate to the current trial design. Passwords and session data require migration to a dedicated authentication system before broader SaaS use.

Settings and logs

Settings hold controlled operational values such as watermark behavior. Logs/notifications record event type, safe message, client context, timestamp, and read state. Credentials must never be logged.

Media Library and campaign records

The additive media-first path persists import batches, media assets, categories, campaigns, schedule slots, selected media, client instructions, copy jobs, generated captions/hashtags, usage records, revision and approval evidence, packages, and revocable campaign links. A ready status is not accepted as proof that bytes exist: integrity checks compare persisted rows with the real private files and fetch every client-visible thumbnail through the authenticated route.

8. Client and Post State Machines

Representative client lifecycle:

new -> scraping -> ready_to_generate -> generating -> pending / in_approval -> client_approved -> approved -> downloaded

Representative post lifecycle:

pending |-- client approves -> client_approved |-- client edits text -> pending |-- client requests change -> regenerating -> pending

client_approved |-- admin allows -> approved |-- admin requests change -> regenerating -> pending

approved -> packaged / downloaded

Other explicit states include image_failed, needs_review, deleted, and transition-specific statuses. UI labels must map to actual backend states rather than inventing a parallel state model.

Representative media-first campaign lifecycle:

draft -> dates selected -> media selected -> copy queued -> copy running -> review -> client approved -> internal approval -> packaged / downloaded

Copy failure remains an explicit recoverable job result; it does not erase the selected media or present the campaign as review-ready.

9. Brand Research and Generation

Brand research starts from the approved client record. It extracts website identity, services, tone, audience, location, and colors, and can enrich the profile with configured social-source style evidence. The result becomes the first context layer for content generation.

The generation workflow combines:

- client brand profile and social style;
- approved keywords and offers;
- schedule duration and posts-per-day settings;
- caption and hashtag generation;

- deep niche-specific image-prompt generation;
- image generation with controlled fallback behavior;
- clean asset storage plus watermarked review rendering;
- one independent post record per schedule slot.

The latest scheduling repair changed multi-post-per-day generation so each slot runs independently. This removed a timeout/error pattern when two or more posts were requested for the same day.

The media-first path is deliberately different. It preserves the client's selected media and asks the copy model to use that visual as evidence for the caption and hashtags. For video, a resolved poster frame supplies the visual context. The work runs through persisted background jobs, and safe retry/idempotency behavior prevents a browser refresh from multiplying paid calls or post records.

10. Approval and Change-Control Design

Approval has two layers:

1. The end client confirms the creative direction.
2. Fine Touch Marketing performs final internal approval before clean delivery.

This distinction prevents a client click from automatically making an asset downloadable as a final agency deliverable. The admin can allow, request changes, edit through AI, upload a replacement, or approve all eligible posts in a controlled batch.

Caption and hashtag edits are direct text changes and remain separate from image regeneration. Image change requests carry the latest post version and current image context into the edit pipeline. The current Session 16 behavior intentionally permits repeated change requests, with fallbacks when the preferred targeted-edit path fails.

11. Asset Protection and Delivery

Review images are visibly watermarked. Clean images are stored separately and are used only by authorized internal review and download workflows. ZIP delivery includes approved clean images and associated caption/hashtag data.

The Media Library uses a different protection model for client-supplied assets. Its normalized and preview references can intentionally resolve to the same processed file, so security checks compare the actual resolved path and authorization context rather than assuming a field named

“preview” is a separate watermarked derivative. Originals and deliverables remain protected by server-side scope and state checks.

Browser controls cannot prevent screenshots, phone-camera capture, or screen recording. Watermarking is the practical protection mechanism. Documentation and sales communication must not claim absolute screenshot prevention.

12. Authentication and Authorization Boundaries

- Admin and team sessions must be verified on every privileged API action, not only in frontend routing.
- Team members must receive only owned/assigned clients from backend filtering.
- Client portal access must remain scoped to one client and must expire or be revocable.
- Clean-image and ZIP access must verify authorization independently of a hidden URL.
- Uploaded images require MIME, size, decode, filename, and ownership validation.
- Webhook routes require rate limits and replay/abuse controls appropriate to their action.

The repository currently contains sensitive operational files and historical progress records. They are not safe client deliverables and should not be committed, copied, or shared without a dedicated secret-removal pass. No credential values are included in this dossier.

13. Verified Test Evidence

Full workflow trial

E2E_TEST_PLAN.md records an 18-phase pass covering:

- admin login;
- team creation and login;
- admin and team client creation;
- brand scraping;
- content generation with images, captions, and hashtags;
- approval-link creation;
- portal viewing;
- client approval, rejection/regeneration, and text editing;
- admin awaiting counts and final approval/rejection;
- clean ZIP generation;
- team visibility isolation;

- exact test-data cleanup.

Reliability suite

Session 8 records a final 17/17 pass after repairs for Sheets quota pressure, notification propagation, regeneration version handling, duplicate response data, and image-generation timing.

Current feature suite

Session 16 records 21/21 live checks passed after adding or repairing:

- clean download behavior;
- portal presentation;
- multi-post-per-day scheduling;
- client and admin bulk approval;
- direct caption/hashtag editing;
- repeated image-change requests;
- live admin status filters;
- administrator AI edit and image upload;
- social-profile brand enrichment.

Media Library acceptance — Phase 10

Phase 10 completed formal acceptance of the new media-first workflow against all 18 defined criteria. The evidence covered private asset intake, processing, classification, library search and selection, campaign creation, client scheduling, image and video selection, copy preparation, review edits, two-stage approval, protected delivery, revocation, authorization, regression safety, and exact test cleanup.

The acceptance work included repeated live scenarios and real-browser coverage on desktop, tablet, and phone layouts. It also confirmed that the new campaign-copy path preserved the selected source media and made no image-generation calls. Browser checks completed without console errors, failed requests, or broken media in the accepted paths.

General release — Phase 11

Phase 11 deployed the feature in controlled stages. A known media pack completed all release checks, followed by an existing-client pilot that exercised the full schedule-to-delivery path with both image and video selection. The client edited and approved review content, staff completed a further revision and final approval, and the temporary review link was revoked after the pilot.

Rollback was rehearsed rather than documented only: the Media Library was disabled at the interface and server boundaries, the established product passed its regression suite while the feature was off, and the server and interface were then re-enabled in a controlled order. Final released-state authentication and security checks, smoke checks, deployment-parity checks, private-file canaries, runtime/schema checks, capacity checks, and artifact cleanup all passed. No known open defect or test artifact remained in the live release scope.

These are dated verification records, not permanent guarantees. Run the relevant suite after workflow, frontend, provider, or schema changes.

Post-release dashboard verification

The client-information update was reconciled against the live deployment before editing, deployed with exact file-hash and server-runtime checks, and then tested against the stored client record without copying any real client values into this public document. The authenticated schedule suite passed 79/79 checks, and the released Chrome/Edge browser suite passed 59/59 across desktop, tablet, and phone layouts. The test followed the information summary through every dashboard stage and found no console, page, network, unexpected-response, or broken-media error. Final deployment parity was 33/33 and the established-system smoke check remained 8/8. The update made no paid AI call; the final cleanup audit found all 60 marked campaigns revoked, zero live test links, and no live test token, path, media, or client.

Media-aware client flow — 2026-08-11

The latest release added visual context for caption generation, automatic generation after media selection, immediate media-label/control rendering, protected preview-ticket reuse, a generation-safe catalog cache, and durable account-scoped dashboard links. Verification included:

- Admin campaign-save browser coverage passed 18/18, including exact link equality after navigation, reload, closing the tab, and signing in again.
- The fresh media-aware client flow passed 23/23 twice. Each run used a new one-post campaign, loaded real protected preview pixels, started paid copy generation automatically, and read the final copy and usage state back from the live store.

- Two independent paid HTTPS scale runs passed 23/23 on fresh 50-post campaigns. All 100 captions/hashtag sets were read back, every post reached review, usage was recorded, and both runs retained image_calls=0.
- A real Chrome calendar click-through passed 22/22: the calendar opened by default, a post modal opened, the approval action produced an observed successful write, the modal refreshed, and the live record became client-approved.
- Protected thumbnail coverage passed 32/32. The final release audit recorded 38 scoped files in sync and the proof links were revoked after verification.

The current evidence is intentionally precise rather than universal. It proves the recorded releases and runs; it does not claim infinite scale, permanent provider availability, or the absence of unrelated historical storage debris.

14. Incident and Repair Lessons

Missing or stale backend behavior behind a working UI

Frontend success does not prove a webhook or workflow is active. Verification must trace the complete action to its data-state result.

n8n routing nodes produced empty outputs

Several approval and regeneration failures came from IF-node behavior, item linking, or branches that emitted no item. Explicit Code/Switch routing and response-path tests were more reliable.

Google Sheets quota pressure

Fast polling plus repeated whole-sheet reads caused transient failures. Polling intervals, deduplication, bounded reads, and workflow design must respect provider quotas.

One logical result was multiplied

Cross-product item execution produced duplicate posts or repeated response rows. Results now require stable post identifiers and explicit deduplication.

Multi-slot generation ran in the wrong execution mode

Running all schedule items as one execution caused timeouts and missed posts. Per-item execution now isolates each schedule slot.

Image editing needed fallback behavior

Targeted AI edits can fail or return unsuitable output. The workflow retains the original until a valid replacement exists and falls back to regeneration where approved.

Source and deployed versions drifted

Local JSON, payload variants, and live workflows can diverge. Every deployment must start from a captured live source and end with parity verification.

15. Known Limitations and Risks

- Google Sheets is practical for a trial but weak for concurrent multi-tenant state, referential integrity, and audited transactions.
- Static browser clients plus webhook orchestration require disciplined backend authorization; frontend role checks are not security boundaries.
- Portal links and clean-asset links need explicit expiry, revocation, and scope tests.
- AI-generated images and copy require human review for brand accuracy, spelling, claims, regulated content, and visual artifacts.
- Social-site scraping is fragile and subject to markup, access, policy, and rate-limit changes.
- Image providers, search sources, Sheets, and messaging providers have quota and availability constraints.
- Automatic posting to social platforms is outside the current verified flow; publishing remains manual.
- Absolute screenshot prevention is impossible.
- Local progress files include contradictory historical status blocks; use the newest completed summary and verify live state.
- Sensitive files and credential-bearing historical notes require removal from version control and secure rotation where exposure is suspected.

- The released Media Library accepts the verified browser-safe image and video formats; broader real-device HEIC coverage remains an additional compatibility gate.
- Direct import from design-tool share links is not part of the released workflow.
- Dynamic responses from the current host can use chunked transfer, so interfaces must not depend on a downloadable response-size header for progress estimates.
- Media Library previews are protected by authorization and campaign scope; the system must not assume that every preview is a separately watermarked file.
- The separate broad storage audit still reports archived test/debris rows whose files are absent and orphan files with no live row. No ready row is currently broken, but that historical hygiene debt is not described as clean and requires a separate ownership-gated cleanup.
- The client's design-tool-link PDF remains parked by owner decision. The public preview images available from those links are below the required social-post resolution, so the released intake path still expects actual media files or a ZIP bundle.

16. Deployment Discipline

1. Capture the current live workflow or frontend source before editing.
2. Reconcile live state with the intended local source of truth.
3. Apply the smallest scoped change.
4. Run syntax and deterministic checks.
5. Deploy only the required frontend, workflow, or utility files.
6. Confirm workflow activation and public proxy routing.
7. Verify local/live parity with hashes or canonical JSON comparison.
8. Run the changed feature plus an unchanged-path smoke test.
9. Clean exact test records and generated assets by captured identifiers.
10. Verify every ready media row against real bytes and render protected thumbnails through the same route the browser uses.
11. Record the result without adding secrets to progress files.

17. Recommended Next Engineering Gates

1. Remove credential-bearing files from the repository, rotate exposed credentials, and add automated secret scanning.
2. Move authentication, sessions, approval tokens, and role enforcement to a dedicated backend identity layer.
3. Replace Google Sheets with a transactional database before scaling beyond the trial workload.
4. Extend the released durable copy-job pattern with operator-visible cancellation and queue back-pressure for every long-running generation/regeneration path.

5. Add signed, expiring asset delivery and explicit clean-image authorization tests.
6. Introduce one manifest that maps workflow filenames to active deployment identifiers without embedding credentials.
7. Add automated live parity and route-health checks after every deployment.
8. Add structured observability for generation latency, quota errors, AI fallbacks, approval age, and download outcomes.
9. Add client-safe content moderation and claim-review gates before final approval.
10. Define the SaaS migration threshold using paying clients, monthly volume, concurrency, support load, and approval completion rate.

18. Maintainer Review Checklist

Before changing generation

- Confirm the brand-profile contract and schedule-slot semantics.
- Use fixed sample inputs and keep AI/provider output bounded.
- Prove clean and watermarked asset paths remain separate.

Before changing approval

- Trace every state transition through portal, workflow, Sheets, admin, and download eligibility.
- Test individual and bulk actions.
- Test a repeated change request against the latest version.

Before changing access

- Test administrator, team, end-client, expired-link, wrong-client, and unauthenticated cases.
- Verify the backend performs ownership checks.
- Scan responses and logs for credentials or private identifiers.

Before declaring completion

- Verify deployed source parity.

- Run the proportional E2E suite.
- Clean exact test artifacts.
- Update this dossier only with evidence that remains safe to share internally.

19. Scope Boundaries and Truth Model

This document describes the source tree and the latest dated integrated test evidence available at reconciliation time. It does not claim that every local workflow snapshot is the active production revision. n8n exports, deployment payloads, repair scripts, and live workflow state can diverge, so a maintainer must confirm active workflow identity and source parity before changing behavior.

The current verified product boundary ends at generation, review, internal approval, and clean-asset packaging. It does not include verified direct publishing to a social network, a production-grade multi-tenant identity platform, guaranteed scraping coverage, or a transactional database. Those are future engineering decisions rather than current capabilities.

The evidence hierarchy used for this edition is:

- Current consolidated browser sources for observable interface behavior.
- Current workflow JSON for intended orchestration and data transitions.
- The newest completed session record for dated live-test results.
- Earlier plans and progress notes for history only when they agree with current evidence.
- Live infrastructure only after an explicit parity check; local notes alone are not proof of deployment.

20. End-to-End Request Lifecycle

20.1 Sign-in and role context

The admin surface submits a login request to the protected automation boundary. The login workflow reads the authorized user store, verifies the submitted secret using the trial authentication design, and returns a scoped session context. The browser retains the session only for the active tab session. Every later privileged request includes that session context, but backend verification remains the authority; hiding controls in JavaScript is only a usability measure.

20.2 Client onboarding

An authorized operator submits business identity, approved research sources, contact metadata, content frequency, keywords, and offers. The onboarding workflow validates and normalizes the input, creates stable client and review identifiers, records ownership, initializes status, and starts research. The response should return only the minimum identifiers the interface needs.

20.3 Brand research

The research workflow loads the canonical client record, marks research in progress, gathers allowed public evidence from the configured website and social sources, combines bounded text, asks the language model for a structured profile, validates the result, and persists the profile. Individual source failures are partial-data conditions unless the remaining context is too weak for safe generation.

20.4 Content generation

The established generated-media path supplies a client identifier, schedule window, frequency, and current campaign inputs. The workflow reloads the client instead of trusting duplicated browser fields, creates one schedule item per intended post, and processes every item independently. Each item produces caption text, hashtags, an image prompt, a clean image, a watermarked preview, and one post record with a stable identifier.

The media-first path begins from client-selected library assets. Completing selection records the dates, media, and campaign-wide instruction, then creates persisted copy work. A background worker resolves each selected image or video's poster frame inside the private media root, gives that visual and bounded campaign context to the copy provider, validates the result, records usage, and advances the post to review. It does not generate replacement imagery.

20.5 Client review

The client opens a scoped campaign dashboard, receives only the authorized campaign, and moves through date selection, protected media selection, background copy preparation, and calendar review. An approval changes the post to the client-approved state. A text edit updates caption and hashtags without pretending that an image changed. A visual change request records the instruction, moves the post into regeneration, and returns a new reviewable version only after replacement validation succeeds.

20.6 Internal review and packaging

Client approval does not bypass agency review. The internal queue loads eligible posts for the authorized operator, who can approve, request an AI change, accept a replacement upload, dismiss a proposed change, or approve an eligible set. Packaging selects only authorized final records, uses clean assets rather than previews, writes accompanying copy, and marks the client lifecycle consistently after a successful result.

21. Admin and Team Interface Behavior

The consolidated admin page is a static browser application with role-aware navigation and a shared request layer. It provides client search and filtering, live status counts, client detail, editable campaign settings, generation controls, durable campaign-dashboard link preparation, Media Library access, copy-job progress, post preview, internal approval queues, image and copy downloads, notification views, and team management.

Important interface rules include:

- Status pills reflect backend states and counts returned from current data, not a hard-coded project plan.
- Long-running generation is acknowledged as a job-like action; the interface refreshes state instead of assuming completion from the initial response.
- Buttons are enabled from explicit state rules. A visible button is not evidence that the backend route is active.
- Bulk approval executes a bounded sequence of individual authoritative actions so one failure can be identified and retried.
- Text, filenames, business names, comments, and server messages must be escaped before DOM insertion.
- Network requests use timeouts and no-store behavior where stale responses could misrepresent approvals or generated assets.
- Team users see the same core workspace but only within backend-enforced ownership scope.

The repository also contains earlier split admin and portal assets. Maintainers must confirm the consolidated files used by deployment before applying a patch; editing a historical copy can create a successful local diff with no production effect.

22. Client Portal Behavior

The client surfaces are intentionally narrower than the internal dashboard. The established portal resolves one generated-post review context. The media-first campaign dashboard adds date selection, protected library selection, one campaign instruction, background caption progress, and a calendar-oriented review view. Both keep internal operations and clean-delivery authority outside the client scope.

The portal maintains several trust boundaries:

- The review token identifies scope but must still be checked by the backend on every read and write.
- Post identifiers are never sufficient by themselves; the post must belong to the resolved client.
- Watermarked previews are the default visible asset during review.
- Clean-image delivery is a separate authorized operation after internal approval.
- A repeated change request must target the latest stored post version and current image, not an original snapshot.
- Upload success means the server decoded, stored, and associated a valid image; selecting a local filename is not success.
- Approve-all must report partial failure instead of showing an unconditional success message.

23. Brand Profile Contract

The brand profile is a structured context object rather than a free-form summary. Current generation expects the profile to carry the business's services, tone, target audience, location, colors, and social-style evidence when available. Keywords and offers remain explicit campaign inputs rather than being silently inferred from research.

Contract requirements:

- Parse model output defensively and reject malformed structures.
- Bound source text and field lengths before sending or storing them.
- Preserve provenance at the source-category level so maintainers can distinguish website evidence from social-style evidence.
- Treat missing social sources as absent evidence, not an empty endorsement.
- Never place passwords, private messages, access tokens, or unrelated personal data into the profile.
- Version the contract before adding fields that change prompt behavior.
- Keep a deterministic fallback profile for sparse but valid client records.

24. Scheduling and Generation Invariants

The scheduling repair recorded in the latest completed test session is a core invariant: one intended schedule slot becomes one independently processed workflow item. A request for multiple posts per day must not run the complete schedule as one opaque item or reuse a single output across slots.

For a requested date window and frequency, generation should guarantee:

- A deterministic set of schedule slots before provider calls begin.
- A stable post identifier per slot.
- At most one committed post record for the same generation key.
- No transition to the final pending state until all committed results are accounted for.
- Explicit image-failed or needs-review handling when copy succeeds but imagery does not.
- Safe retries that reconcile existing records before inserting replacements.
- Independent provider timeouts and bounded fallback behavior.
- Preservation of the last valid image until a replacement has been decoded and stored.
- Persisted copy-job state so navigation, reload, or a closed tab does not cancel accepted work.
- A media-first provider call uses only the campaign's selected asset, or a confined poster frame, as visual context and records no image-generation call.
- Usage and terminal state are read back from the operational store rather than inferred from a successful enqueue response.

The browser must not calculate authoritative slot counts independently. The backend response and persisted records are the source of truth.

25. Approval Consistency Rules

Approval correctness depends on invariants shared by portal, workflows, data records, and the internal dashboard:

- A pending post can be approved or changed by the scoped client.
- A client-approved post is eligible for internal review but not automatically a final deliverable.
- An internal approval records actor and time and enables clean delivery only for the approved version.
- Any content-changing regeneration invalidates approval of the older version.
- Caption and hashtag edits preserve clear state semantics and modification time.
- A rejected or change-requested item cannot remain in a final downloadable set.

- Bulk operations apply the same authorization and transition checks as individual operations.
- Duplicate requests are detected through stable identifiers and current state rather than producing duplicate transitions.

These rules should be expressed in backend code and tests. Frontend labels are a projection of the state machine, not the state machine itself.

26. Image and File Handling

An AI-generated post can have two image representations. The clean asset is the internal master and the watermarked asset is the review derivative. Imported Media Library assets follow their recorded processing contract and can use one normalized file for both normalized and preview references. Authorization therefore compares resolved path values, ownership, campaign scope, and approval state rather than trusting field names. Public review must never rely on the secrecy of a clean URL.

For media-aware copy, the server—not the browser—resolves the selected asset used as model context. Image bytes or a video poster frame may be read only after canonical path confinement inside the private media root. Protected browser previews use scoped tickets and cache controls; a cached derivative does not become a public asset and remains subject to the campaign/session boundary.

Server-side file handling should enforce:

- Allowed image MIME types and a strict maximum upload size.
- Allowed video MIME types, bounded size, and safe browser playback metadata where video is accepted.
- Image decoding and dimension checks rather than trusting extensions.
- Generated filenames derived from server identifiers, never raw client filenames.
- Path confinement to the resolved client and post storage area.
- Re-encoding or metadata removal when appropriate to reduce hidden payload and privacy risk.
- Atomic replacement so a failed edit cannot destroy the last valid image.
- Cache-busting for refreshed previews without weakening authorization.
- ZIP manifests built from approved database records, not directory enumeration alone.
- Expiring or authenticated delivery for clean assets in the SaaS phase.

27. Failure Semantics and Recovery

Provider timeout

The operation records a bounded failure or fallback result. The interface stops presenting an indefinite spinner and offers a safe retry after reconciliation.

Partial research

Reachable approved sources are retained, unreachable optional sources are recorded as unavailable, and generation proceeds only when the remaining profile meets a defined minimum.

Copy succeeds and image fails

The post is not represented as fully ready. It enters an explicit review or image-failed state with enough context to retry only the missing stage.

Regeneration fails

The previous valid version remains available. Version and approval fields change only after the new output passes validation and persistence.

Data-provider quota pressure

Polling backs off, reads are bounded, results are deduplicated, and repeated whole-table calls are consolidated. A quota response is not converted into an empty-success response.

Workflow returns no item

Routing branches must emit a deliberate success or error object. The public response path should never depend on an implicit item-linking assumption.

Deployment drift

Capture live source, compare it with the intended local workflow, deploy the smallest change, confirm activation, and verify parity after deployment. Do not repair drift by overwriting an unknown live workflow with an old export.

28. Observability and Operational Support

The trial needs enough telemetry to explain state without logging sensitive payloads. Recommended event fields are a generated correlation identifier, safe client identifier, post identifier when applicable, workflow action, prior and next state, duration, provider category, attempt count, result class, and a sanitized error code.

Useful operational views include:

- Clients stuck in scraping, generating, or regenerating beyond an agreed threshold.
- Requested schedule slots compared with committed post records.
- Provider latency and timeout rate by generation stage.
- Image-edit fallback frequency.
- Approval age from first review delivery to client and internal decisions.
- Data-provider quota and retry counts.
- Clean-package creation and authorized download outcomes.
- Copy queue age, attempts, terminal failures, per-campaign completion, provider usage, and the invariant that the media-first path records zero image-generation calls.
- Ready-row/file integrity, protected-thumbnail rendering, archived-row debris, and orphan-file counts reported separately so one green metric cannot hide another class of debt.
- Local-to-live parity status for every active workflow and browser artifact.

Logs must exclude prompts containing unnecessary personal information, credentials, session tokens, raw authorization headers, private service URLs, and full uploaded files.

29. Developer Handoff Guide

29.1 Establish the source of truth

Read the newest completed session summary, inspect the consolidated admin and portal files, enumerate workflow JSON, and compare any target workflow with its active deployment before editing. Treat backup and payload variants as evidence, not automatic authority.

29.2 Trace one action completely

For the feature being changed, trace browser event, request contract, proxy route, workflow trigger, authorization check, data reads, routing nodes, provider calls, writes, response object, UI update, and observable stored state. This catches the common failure where a button appears to work while the data never changed.

29.3 Use controlled test records

Create uniquely named test clients and capture every generated identifier. Avoid deleting by broad name fragments. Exercise the changed path plus one adjacent unchanged path, then delete only the exact captured records and generated assets.

29.4 Reconcile long-running work

Generation and regeneration may outlive the initiating request. Tests should poll bounded status with backoff, detect terminal failure, compare expected and actual post counts, and refuse to treat timeout as proof that no work completed.

29.5 Close the loop

Run syntax checks, validate workflow JSON, deploy scoped files, confirm workflow activation, check parity, run the proportional live test, inspect the final data state, clean test artifacts, and update non-sensitive evidence notes.

30. Release Acceptance Matrix

Access

- Administrator login succeeds and invalid credentials fail safely.
- Team login receives role-scoped data only.
- Missing, expired, malformed, and wrong-client review contexts are rejected.
- Every write checks authorization at the backend.

Onboarding and research

- Required fields are validated and normalized.
- Optional source failure produces bounded partial behavior.
- The stored brand profile matches the documented contract.
- Research text and errors contain no credentials or unrelated personal data.

Generation

- One post per schedule slot is committed without duplicates.
- Multiple posts per day complete independently.
- Captions, hashtags, clean images, previews, dates, and versions are consistent.
- Timeout and provider failure reach explicit states.
- Media-first captions use the selected image or video poster frame as confined visual context, retain the selected asset, and record zero image-generation calls.
- Background copy remains durable across navigation/reload and reports persisted usage plus terminal status.

Client review

- Campaign dates, media selection, campaign-wide instruction, generation progress, and review remain scoped to the resolved client campaign.
- Individual approval, approve-all, text edit, visual change, and upload work for owned posts.
- Repeated change requests use the latest version.
- The portal never exposes a clean asset before authorization.
- Partial bulk failure is visible and retryable.

Internal review and delivery

- Awaiting counts match eligible post records.
- Individual and bulk internal approval use identical rules.
- Change requests invalidate stale approval.
- Packages include only approved clean assets and corresponding copy.

Reliability and cleanup

- Requests have bounded timeouts and stale-response controls.
- Retries do not create duplicate posts or transitions.
- Deployed source matches the reviewed source.
- Exact test records and generated files are removed after verification.

Appendix A — Workflow Catalog by Responsibility

Identity and ownership

- Login: verifies an authorized user and issues role context.
- Team management: lists, adds, and removes team users under administrator control.
- Client listing: filters client visibility according to role and ownership.
- Client onboarding: creates a canonical client and initiates research.

Research and creation

- Brand scraping: combines approved website and social evidence into a structured profile.
- Content generation: builds schedule slots, creates copy and imagery, stores assets, and appends posts.
- AI regeneration: changes the latest post version from client or administrator instructions with fallback behavior.

Review

- Approval preparation: creates or returns the scoped portal context.
- Portal post query: returns posts for one verified client context.
- Portal action: handles approval, text edit, and change requests.
- Client image upload and acceptance: validate, store, and attach a proposed replacement.
- Awaiting query: returns posts eligible for internal review.
- Internal approval: allows delivery, requests another change, or dismisses a proposal.
- Administrator image upload: validates and applies an internal replacement.

Delivery and shared operations

- ZIP packaging: selects eligible clean assets and accompanying text.
- Portal delivery: returns authorized final delivery information.
- Download notification: records the delivery event.
- Utility surface: supports shared dashboard actions and carefully scoped data access.

Appendix B — State Transition Reference

Client-level transitions

new -> scraping -> ready_to_generate -> generating -> pending or in_approval -> client_approved -> approved -> downloaded

Exceptional client states should identify whether retry is safe, manual review is required, or the client has been archived.

Post-level transitions

pending -> client_approved -> approved

pending -> regenerating -> pending

client_approved -> regenerating -> pending

approved -> packaged or downloaded

At every transition, store the current version, modification time, and actor category where relevant. Never infer a final state only from the presence of a file.

Appendix C — Safe Client Onboarding Checklist

1. Confirm written authorization and the approved public research sources.
2. Confirm business identity, service wording, location, target audience, tone, keywords, offers, schedule, and review contact.
3. Create least-privilege access through the approved operational process.

4. Validate the client record without placing credentials in business fields.
5. Run research and review the structured profile with a human.
6. Generate a small controlled batch first.
7. Inspect copy, claims, spelling, brand fidelity, images, watermarking, and schedule.
8. Test the scoped portal with a non-privileged browser session.
9. Complete client and internal approval before clean delivery.
10. Remove temporary access and retain only the agreed operational evidence.

Appendix D — Glossary

[omitted]

not an administrator session.

- Brand profile: structured research context used by copy and image generation.
- Clean asset: the internal image master used for authorized final delivery.
- Watermarked preview: the derivative shown during review to identify non-final artwork.
- Client-approved: accepted by the end client but still awaiting the agency's internal decision.
- Deterministic repair: code-enforced correction or validation applied around provider output.
- Generation key: a stable idempotency key representing one intended schedule slot and version.
- Parity check: comparison proving the reviewed local artifact matches the active deployed artifact.
- Reconciliation: reading persisted state after timeout or retry to determine what actually completed.
- Terminal state: an explicit completed, failed, cancelled, or needs-review outcome; running is not terminal.

Appendix E — Public Technical Review Checklist

For client and product review

- Confirm the workflow, role boundaries, schedule rules, approval stages, and publishing boundary.
- Confirm that AI output remains subject to human review.
- Confirm that watermarking is a deterrent and does not prevent screenshots.
- Confirm retention, revocation, and content-claim responsibilities.

For engineering review

- Trace a single post from schedule slot through final delivery.
- Verify every state transition and repeated-request path.
- Verify current source, active deployment, and recorded evidence are time-scoped correctly.
- Verify partial failure, retry, deduplication, and cleanup behavior.

For security and operations review

- Confirm no secret appears in browser source, documentation, logs, exports, screenshots, or process arguments.
- Confirm role, ownership, review scope, upload, and clean-asset checks execute server-side.
- Confirm rate limits, expiry, revocation, observability, backups, and restore procedures.
- Rotate any credential that may have appeared in historical project notes before sharing the repository.

Developed by Zahidul Islam

Social media automation — orchestration, approval systems, AI output controls, operational reliability, and client-safe delivery.