App Improvement Deep-Dive

Masonic Fire · Explain My Build · DenialHelp · ApprovalHelp — 2026-07-11

Prepared by a multi-agent source audit (Fable 5): 6–8 grounded investigators per app reading the current code, then a constraint-screened synthesis. Every critical finding below was independently re-verified against source. Live secrets are redacted — the real values remain only in the repo.

> ***Security handling:*** *this document is a consolidated vulnerability roadmap. Keep it behind authentication. The one live secret found (Masonic Fire encryption key) is* ***not reproduced here*** *— it is referenced as* `STATE.md:155`*.*

Executive summary — the red-lines first

Across the four apps the audit surfaced a cluster of **critical, source-verified issues** that should be addressed before anything else. The DenialHelp/ApprovalHelp items are PHI red-lines and must be fixed **before any promote to production**; the Masonic Fire secret leak is a standing exposure in a pushed repo.

#	App	Critical finding	Effort
1	Masonic Fire	Rotate and purge the committed production secrets (MASONICFIRE_ENC_KEY + CRON_SECRET in STATE.md, pushed to GitHub)	M
2	EMB	Gate the end-user guide on the web funnel — the code violates the monetisation constraint	M
3	EMB	Close the Pro purchase loop in the web app — paying $19/mo currently yields nothing usable there	L
4	DenialHelp	Fix consumer intake cross-patient PHI account-takeover (unverified-email cookie binding)	M
5	DenialHelp	Route insurer-required.ts PHI through the de-id/subscription wrapper (stop direct non-BAA Anthropic egress)	S
6	DenialHelp	Make the HIPAA analytics gate host-aware and truly fail-closed	S
7	ApprovalHelp	Analytics gate fails OPEN on every AH clinical surface — non-BAA GA4/ PostHog fire on PHI pages	S
8	ApprovalHelp	insurer-required.ts sends patient denial-letter text DIRECT to non-BAA Anthropic under a false 'zero-phi' assertion	S
9	ApprovalHelp	OAuth signup (the top AH CTA) dead-ends in an unsubscribable 'pending' account — no trial, no Stripe customer, no checkout path	M

Full detail, ranked priorities, quick wins, constraint checks and the complete finding set for each app follow in the per-app sections.

Masonic Fire

masonicfire.com — verified register/community for regular Freemasons (Next.js + SQLite).

Overall assessment

The design constraints are genuinely implemented in code, not just copy — the Fire is names-never-numbers with an explicit "there must never be a count function" guard, the digest is count-free by construction, trust is strictly lodge-anchored, and route-level auth/validation hygiene is consistently good. But the app has one critical secret-handling failure (the production encryption key that protects ritual lines and welfare notes, and HMAC-signs public Pass tokens, is committed to STATE.md and pushed to GitHub), several verified privacy/ritual seam breaches (unauthenticated member photos, the □ marker sent verbatim to the ASR endpoint, a silent dev-key fallback), and —

the product-level truth — every retention and growth loop is open-circuit at its return point: the built digest has never sent an email, comment replies notify nobody, and visit/confirmation requests to the ~25,000 unclaimed lodges silently dead-end while the UI claims otherwise.

Ranked improvements

1. Rotate and purge the committed production secrets (MASONICFIRE_ENC_KEY + CRON_SECRET in STATE.md, pushed to GitHub)

🔴 **CRITICAL** · effort M · *security/privacy*

> ✅ ***Independently verified against source.*** *Verified: STATE.md:155 holds the live key + CRON_SECRET, git-tracked & pushed.*

What to do: Verified: /Users/micryan/masonic-network/STATE.md:155 contains the live MASONICFIRE_ENC_KEY and CRON_SECRET, git-tracked since the initial commit and pushed to github.com/Micipedia/masonic-fire. Remove secrets from STATE.md (reference the systemd unit / untracked .env), purge history (git filter-repo) and rotate: CRON_SECRET immediately (stateless); for the enc key write a one-off migration that decrypts rehearsal_items.content_enc, welfare_requests.message_enc, pastoral notes, funeral_wishes_enc under the old key and re-encrypts under a new one — do NOT just swap the env var (STATE.md itself warns data becomes undecryptable). Pass tokens (lib/pass.ts HMACs with the same key) re-derive automatically. RED-LINE: confirm the rotation plan with Mic before executing — secret handling requires explicit OK.

Why it matters: This one leak collapses both the membership-data-as-PHI posture and the ritual hard-line at once: anyone with repo access can decrypt every welfare note and every brother's private ritual lines from any DB copy/backup, AND forge Pass tokens to use the public /pass/[token] page as a member-enumeration oracle (name, lodge, standing for sequential ids).

2. Fail loud when MASONICFIRE_ENC_KEY is unset in production (crypto.ts + pass.ts)

🟠 **HIGH** · effort S · *security/privacy*

> ✅ ***Independently verified against source.*** *Verified: crypto.ts + pass.ts silently fall back to a public dev key, no NODE_ENV guard.*

What to do: lib/crypto.ts:16-17 and lib/pass.ts:8-10 silently fall back to sha256('masonic-fire-dev-key') / the literal string — a key published in the repo — whenever the env var is unset, with no NODE_ENV guard. Throw at first use (or refuse to boot) when NODE_ENV==='production' and the key is missing; keep the deterministic fallback dev-only. Then verify the deployed NucBox unit actually sets the key.

Why it matters: Welfare/pastoral/ritual 'encryption' and Pass-token authenticity would silently degrade to a publicly-known key on any unit-file edit or host migration — exactly the kind of quiet

constraint violation the FAIL LOUD doctrine exists to prevent. Flagged independently by three investigators.

3. Ritual hard-line: stop sending the □ marker and raw reference to Whisper; enforce own-infra ASR in code

🟠 **HIGH** · effort S · *ritual hard-lines*

> ✅ ***Independently verified against source.*** *Verified: practice/recognize/route.ts:43 sends the raw decrypted line (□ intact) as Whisper prompt/biasing.*

What to do: app/api/practice/recognize/route.ts:61-62 appends the decrypted ritual line VERBATIM (□ intact) as prompt/biasing — send normalize(reference) (already drops □/punctuation) instead. Validate WHISPER_URL at startup against a private-host allowlist (localhost/Tailscale 100.64.0.0/10) so 'own infra only' is code, not comment. Encrypt rehearsal titles like content_enc, cap the base64 audio field (~10MB), and when a mixed line contains □ with mic on, show one plain line in practice-client.tsx: 'This line holds a guarded secret (□) — pass over it in silence when you recite.'

Why it matters: The '□ never checked/sent to ASR' rule is currently violated (marker position + surrounding ritual context transmitted), and nothing prevents WHISPER_URL pointing at a third party. The mic-guidance line also stops a brother speaking the secret aloud into the recording. These are the letter and spirit of the hardest hard-line in the product.

4. Gate /media behind a session: member photos are currently world-readable with public immutable caching

🟠 **HIGH** · effort M · *security/privacy*

> ✅ ***Independently verified against source.*** *Verified: app/media/[...path]/route.ts has no auth + Cache-Control public immutable.*

What to do: app/media/[...path]/route.ts has no getCurrentMember() call and returns 'Cache-Control: public, max-age=31536000, immutable' (verified). Filenames encode member ids (m{memberId}_{ts36}.jpg). Add the session gate (all legitimate consumers are signed-in same-origin requests so the cookie flows), switch to 'private' caching, add 'X-Robots-Tag: noindex' to image responses, generate filenames from crypto.randomBytes, add app/robots.ts disallow-all, and delete public/u/1.jpg after one SQL check that no profiles.photo_url still references /u/.

Why it matters: Membership-as-PHI/default-closed is bypassed for the most identifying data in the system — photographs of identified Freemasons tied to member ids, retainable by intermediary caches for a year. Video already does this right (signed tokens in api/feed/media); images are the one surface that never got the treatment.

5. Restore the POST-confirm interstitial for magic links; shorten the 7-day token TTL

🟠 **HIGH** · effort S · *security/auth*

> ✅ ***Independently verified against source.*** *Verified: auth/verify GET consumes token + sets 30-day session.*

What to do: app/api/auth/verify/route.ts GET now consumes the token and sets a 30-day session cookie directly, contradicting lib/auth.ts's own prefetch-safe design comments. Render a 'Confirm sign-in' page with ONE large primary button that POSTs the token (the POST handler already exists); consume only in POST. Then drop TOKEN_TTL_MIN (lib/auth.ts:8) from 7 days to 30-60 minutes — the code's own comment says tighten before launch, and the scanner-burn rationale for the long TTL disappears once the interstitial is back.

Why it matters: Email scanners (SafeLinks, corporate AV) follow GETs: they burn the single-use link (brother sees 'expired' — the exact failure the design was built to prevent) and receive an authenticated session in third-party infrastructure. A one-tap big-button confirm also satisfies the 80-year-old bar better than a mystery 'expired' page.

6. Fix the fresh-DB bootstrap crash: feed_items is ALTERed before it is created

🟠 **HIGH** · effort S · *correctness/ops*

> ✅ ***Independently verified against source.*** *Verified: init-db.mjs addColumn(feed_items) L729 precedes CREATE TABLE L908.*

What to do: scripts/init-db.mjs runs addColumn("feed_items", ...) at line 729 but CREATE TABLE feed_items sits at line 908 (verified). Move the feed_* CREATE block above the first addColumn, or make addColumn a loud no-op when PRAGMA table_info returns empty. Then smoke-test a scratch-dir bootstrap after every schema change.

Why it matters: The documented reset/disaster-recovery path (STATE.md: rm db + init-db) crashes outright, and no test environment can be created — for a DB the repo itself calls 'irreplaceable'. It only works today because the live DB already has the table. This also blocks priority 14 (tests need a fresh-DB fixture).

7. Gate the test realm out of the feed spine, digest, relief rail and constitution dropdowns

🟠 **HIGH** · effort S · *data integrity*

What to do: lib/feed.ts (readFeed/readMemberFeed/readComments/firesForItems), the digest cron recipients query, and member-home's reliefCauses query have zero is_test filtering — seed-sandbox creates VERIFIED test members who can post craft-scope to every real brother's feed and weekly digest, and 'Sandbox Grand Lodge' appears in the /trades and /brethren dropdowns (only /visit filters correctly). Add the canonical is_test=0 gate (author + lodge) unless canSeeTestRealm(viewer), plus AND m.is_test=0 on digest recipients and the two dropdown queries.

Why it matters: Routine QA drives fake content into every real member's feed and emails @sandbox.masonicfire.test addresses — corrosive to the dignified, real-world credibility a solemn register trades on, and a direct violation of the codified 'public surfaces always exclude test' rule.

8. Close the two return loops: schedule the already-built weekly digest, and make comment replies actually notify

🟠 **HIGH** · effort M · *product/retention*

What to do: (a) The digest (app/api/cron/digest) is fully built, count-free and scope-gated, but SHIPPED DARK — no timer exists. Add masonicfire-digest.timer (weekly, Sunday evening) mirroring the reminders timer; verify one send against Mic's own account first. (b) 'Tell me when a brother replies to me' (members.reply_notices_off) is a dead toggle — addComment sends nothing. On comment, email the item author and parent-comment author a named, uncounted note ('Bro. John replied to your word' + excerpt + link), honouring reply_notices_off and notify_email, batched per item per hour. Reuse lib/greeting.ts's mail pattern.

Why it matters: Lodge nights are monthly; between them, NOTHING currently pulls a brother back — the best-engineered retention asset in the app has never sent an email, and the app's only interaction primitive (comments) is a conversation nobody learns has continued. These are the single biggest retention levers and both are wiring jobs on constraint-proofed code.

9. Make the officer-less-lodge flows honest and generative (visit requests, confirmation requests, Pass verifier)

🟠 **HIGH** · effort M · *product/growth + honesty*

What to do: (a) request-visit returns {ok:true} and the UI says 'the lodge has been told to expect you' even when the lodge has zero rooted officers (true for ~25,000 lodges) — return an honest state ('not yet on Masonic Fire — confirm with its Secretary before travelling'), fall back to notifying ENQUIRY_NOTIFY, and link /lodge/claim. (b) request-confirmation promises 'we'll notify… the founders' but sends nothing and no admin queue exists — email ENQUIRY_NOTIFY and add an /admin 'Brethren awaiting an unclaimed lodge' section. (c) Add a quiet footer to /pass/[token]: one sentence about Masonic Fire + 'An officer of a lodge may claim its page' → /lodge/claim.

Why it matters: A brother travelling to a lodge on a false 'has been told' promise is the single fastest way to destroy trust in the register; activation currently stalls silently at its most motivated moment; and the Pass — shown to exactly the officer the growth model needs — currently advertises nothing. All three fixes recruit the missing officer instead of dead-ending.

10. Mobile/usability breakage cluster: rail buried below the feed, text-size control a no-op, dead primary CTAs

🟠 **HIGH** · effort M · *UX/usability law*

What to do: (a) member-home.tsx: on <1024px the entire actionable rail (Settle in, 'Words for you', pending introductions, Relief) stacks BELOW 40 feed items and the 'all caught up' terminus — render actionable/personal cards above the composer on small screens, ambient cards stay below. (b) globals.css:160-169: 'html.fs-lg body{font-size:19px}' bumps body px while all Tailwind text-* is rem against the fixed 17px html root — move the bump to html.fs-lg{font-size:19px} so 'Large/Largest' actually enlarges the UI. (c) The empty-feed primary CTA and onboarding 'Say hello' step

anchor to #post-body/#composer which don't exist for unverified members (silent no-op) — gate on verified and lead with 'Find your lodge' instead; and change the three 'Say hello' rail links from /brethren?q=name (trade-filtered, often empty) to /brethren/{id}.

Why it matters: This is the concrete root of 'not well mobile-optimised': the app's warmest, most actionable touchpoints are unreachable on a phone, the one feature built for older eyes does nothing, and the new brother's single primary action silently fails — all direct hits on the 80-year-old Past Master law.

11. Fix GL broadcast audience: recipients ignore the constitution hierarchy

🟠 **HIGH** · effort M · *data correctness*

What to do: app/api/jurisdiction/[id]/broadcast/route.ts pins the feed item to the ROOT constitution but selects email recipients only from lodges attached directly to the raw node — so a sovereign-GL broadcast under UGLE/Scotland-style hierarchies emails almost nobody, while a DISTRICT officer's broadcast is feed-visible to the entire constitution. Expand recipients with the recursive-subtree CTE from lib/jurisdiction-tree.ts, and either scope district broadcasts to the district node (teaching readFeed to match ancestor nodes) or restrict emit-to-root to root officers.

Why it matters: Grand Lodge communication is the institutional-credibility feature — a Grand Secretary whose notice reaches 19 of 370 lodges by email, or a district officer who accidentally addresses the whole constitution, is a trust failure with the exact audience whose endorsement the product needs.

12. Fix lodge dedup identity: name-only keying drops and corrupts legitimately distinct lodges

🟠 **HIGH** · effort M · *data integrity*

What to do: All four import builders dedup on number-stripped lowercase NAME (silently dropping same-named lodges with different warrant numbers), and lib/gl-claims.ts addLodgesToJurisdiction matches on name only, then COALESCE-overwrites the existing lodge's number — live data corruption. Change the key everywhere to (jurisdiction_id, lower(name), coalesce(number,'')); insert rather than update when numbers differ; add a UNIQUE index on that triple after a violation pre-check; and one-off diff the raw scrape files vs imported rows to recover dropped lodges. Also reconstruct and commit the lost /tmp import+geocode scripts (scripts/import-lodges.mjs, geocode-lodges.mjs) so the 25k-lodge flagship dataset is reproducible.

Why it matters: An unknown number of real lodges were silently dropped from the 25k import, the live paste path can renumber officer-corrected rows going forward, and the entire dataset's write/geocode pipeline currently exists nowhere — the register's completeness IS the product.

13. Give the Secretary a roster-invite tool (the whole-lodge adoption engine)

🟠 **HIGH** · effort M · *product/growth*

What to do: Add 'Invite your brethren' to /lodge/manage: paste emails (optionally names), send the existing dignified invite mail from the Secretary's name, pre-create roster membership so each brother lands attached to the lodge on first sign-in and appears in the Secretary's confirm queue. Confirmation stays a separate deliberate per-brother act. Currently onboarding a 40-member lodge = 40 individual invites from /profile (25/day cap) + 40 add-member attempts after each signs in.

Why it matters: STATE.md calls Secretary use the whole-lodge adoption engine, and the Minute Book/summons/dues spine only delivers value once the roster is in — yet the highest-leverage user has the most manual path in the app. Invite ≠ verify, so lodge-anchored trust is fully preserved.

14. Fix the meeting-schedule parser and wire it into every schedule write

🟠 **HIGH** · effort M · *data correctness*

What to do: lib/meeting.ts:14-24 reads only the FIRST weekday but ALL ordinals — '1st Tuesday | 3rd Thursday' becomes '1st AND 3rd Tuesday', showing factually wrong meeting dates on lodge pages, /visit and member-home. Split on '|'/';' into per-segment (ordinal:weekday) pairs, require word-ordinals near a weekday, and decline to compute a date when 'except/dark' months appear (show raw text instead). Also: parseSchedule has ZERO callers in any write path — call it wherever meeting_schedule is written (gl-claims.ts insert+update branches) so meeting_rule can't rot when an officer corrects his schedule.

Why it matters: Visitation is the declared killer utility; a visiting brother acting on a wrong 'Next meeting' date (real multi-part strings exist in the data, plus 1,300+ 'dark July/August' schedules) is the worst possible outcome for the 80-year-old at a strange lodge's door.

15. Add the first test suites over the privacy scope-gate and verification state machine; chain guardrail into the build

🟠 **HIGH** · effort M · *maintainability/trust*

What to do: Zero test files exist. Add vitest with an in-memory better-sqlite3 fixture (unblocked by priority 6) and start with three suites: (1) canSeeFeedItem/readFeed matrix across all six scopes × viewer types, (2) verification tier transitions never lower standing, (3) auth token/code single-use + burn-after-attempts. Wire 'npm run guardrail && npm test && typecheck' into the deploy gate (guardrail-check.mjs — the codified anti-vanity linter — currently runs only by hand), and add import "server-only" to lib/db.ts.

Why it matters: canSeeFeedItem is the single gate keeping membership data default-closed and markMemberVerified is the trust root — a refactor that breaks either leaks member data or corrupts standing with no failing signal. Synchronous SQLite makes this the cheapest codebase imaginable to test, and the guardrail gate makes a constraint regression mechanically unable to ship.

Quick wins (small effort, high value)

- Schedule the digest timer (part of rank 8a) — one systemd timer unit mirroring masonicfire-reminders.timer; the biggest retention feature in the app ships with one file
- Fail-loud crypto key guard (rank 2) — a two-line production check in lib/crypto.ts + lib/pass.ts
- Fix the three dead/wrong CTAs (rank 10c): verified-gate the empty-feed anchor and onboarding 'hello' step; change 'Say hello' links to /brethren/{id}
- Replace all 8 text-xs (12.75px) sites with text-meta and set --text-xs: initial in @theme so the site's own 14px floor becomes uncompilable to break
- Remove the three displayed counts that drift against the WHO-never-how-many law: 'known to N brethren' → voucher names (lodge-manage.tsx:691), 'N gifts recorded' → named givers trailing off (causes-client.tsx:192), 'N brethren' per trade → 'Brethren available' (trades/browse:39)
- Wrap findOrCreateMember, markMemberVerified and pinPost in db().transaction() — pattern already exists in lodge-claims.ts:137
- Tighten amity self-expansion (lib/feed.ts:114-133): require visit_requests.status='acknowledged' and drop self-authored visitation_log from the amity set — visibility must be host-lodge-conferred, never self-conferred
- Per-IP + global rate limits on /api/enquire and /api/auth/request (currently an unauthenticated Paubox mail-spray vector) plus a honeypot field — no CAPTCHA, invisible to elderly users
- Notify the assigned lodge's Secretary on public enquiries (enquire/route.ts currently emails only Mic) — the speed-to-lead pitch depends on it
- Cron sweep to purge expired auth_tokens and sessions (data-minimisation for PHI-grade DB); move CRON_SECRET from query string to Authorization header with timingSafeEqual
- Verified-gate /businesses, /causes and the gift API (currently any email-only unverified account sees named Freemasons); add is_test=0 to the /trades and /brethren constitution dropdowns
- Feed honesty pair: give feed images real alt text (currently alt="" on the main post photo), and give the disabled Fire button visible copy for unverified brethren ('The Fire opens once your lodge confirms you') instead of a touch-invisible title attribute

Constraint checks & code-level violations

Places where the code itself breaks a load-bearing rule, or suggestions that were rejected for violating one.

- CODE VIOLATION (PHI + ritual hard-lines): production MASONICFIRE_ENC_KEY committed to git-tracked STATE.md and pushed to GitHub — welfare/pastoral/ritual encryption and Pass-token authenticity are effectively broken for anyone with repo access (verified at STATE.md:155)

- CODE VIOLATION (ritual hard-line '□ never sent to ASR' + 'Whisper only on own infra'): recognize/route.ts sends the decrypted reference line verbatim, □ intact, as the Whisper prompt/biasing, and nothing enforces WHISPER_URL is own infrastructure; mixed □-lines also keep the mic path open with no keep-the-secret-silent guidance
- CODE VIOLATION (welfare-notes-ENCRYPTED hard-line): lib/crypto.ts silently falls back to a repo-public dev key in production when the env var is unset — encryption 'holds' only nominally
- CODE VIOLATION (membership-data-as-PHI / default-closed): /media serves member photos and scope-gated gallery/post images unauthenticated with public immutable caching and id-derived guessable filenames
- CODE DRIFT (no vanity numbers, WHO never how-many): three displayed counts — 'known to N brethren' per brother in the Secretary console, 'N gifts recorded' per cause card, 'N brethren' per trade category — the only numeric social displays in an otherwise numberless app
- CODE DRIFT (never host/broadcast ritual text): several lib/craft-light.ts entries track working-tools and cardinal-virtue LECTURES near-verbatim in first-person-plural lecture cadence (in many constitutions those lectures ARE printed ritual); reword to descriptive third-person prose about the symbols, keep history/biography entries
- CODE DRIFT (store only hashes of evidence): profiles.certificate_no stored plaintext despite the schema's own 'PHI, owner-only' comment — encrypt like funeral_wishes_enc
- CODE DRIFT (default-closed): /businesses and /causes disclose named Freemasons to any unverified email-only account; the gift API accepts writes from unverified accounts
- CODE DRIFT (lodge-anchored trust): amityMemberIds treats PENDING/declined visit_requests and self-written visitation_log entries as trust edges, letting a brother unilaterally widen his own read scope
- CODE DRIFT (dignity/test-realm rule): verified sandbox brethren can post into every real member's feed and digest, and 'Sandbox Grand Lodge' appears in real members' constitution dropdowns
- INVESTIGATOR SUGGESTIONS REJECTED: none — all 60+ recommendations screened clean against the constraints; notably the pagination fix is a deliberate user-initiated 'Read earlier posts' link (not infinite scroll), reply notices follow the Fire's named-uncounted pattern, and the roster-invite tool keeps invite strictly separate from lodge-conferred verification

Also worth doing (lower priority, nothing dropped)

- Hash session ids at rest (sessions.id stores the raw cookie value; auth_tokens already hash) — accept both forms for 30 days during migration

- Pass token hardening beyond the key rotation: timingSafeEqual over full-length digests, and mix a per-member rotating pass_salt into the HMAC so a leaked pass URL can be revoked (verifier page stays public by design)
- Stripe webhook out-of-order guard: record processed event ids or compare event.created before applySubscriptionStatus, so a stale retried 'active' can't re-publish a cancelled listing
- Entity sequencing before ever flipping listings live: stand up the Masonic Fire NFP/mission entity and its own Stripe account first — 'DenialHelp LLC' on a brother's statement would torpedo Grand Lodge trust (no code change today; add a comment guard on the STRIPE_* gate)
- Honest feed pagination: fetch limit+1 and render a single 'Read earlier posts →' link instead of the currently-false 'You are all caught up, Brother.' at item 40; readFeed's before-cursor already exists with zero callers
- Warm invite landing: /login ignores the ?invite= token — show the inviter's name/note, prefill the invited email, and attribute by token so provenance survives a different sign-in address
- Fix landing/login copy: 'a brother must vouch for you' misstates the actual open-email + lodge-confirmation model and deters legitimate word-of-mouth joiners; add an explicit 'Join the register — free for brethren, forever'
- Make the 'Brethren' tab match its promise: the page lists only trade-opted members — lead with people-by-name search and 'Brethren you may know', make Trade & calling an explicit secondary section (keep all opt-in/verified gates)
- Surface Visiting + the Pass: add a home-rail 'Visiting' card and consider a nav tab — the killer utility is currently three taps deep, worst for the brother at a strange lodge's door
- 48px tap targets for bare action links (globals.css rule covers buttons/inputs/tel only; the rail's 'reply →'/'Say hello →' links are ~28px)
- N+1 batching: pass a precomputed viewer-visibility context into canSeeFeedItem/canModerateItem, batch readComments with IN(...) like firesForItems, and batch the manage/page.tsx per-member credentials/avouchment queries
- Split the 898-line lodge-manage.tsx god-file into components/lodge/*.tsx; extract the 17 copy-pasted inputCls/post() helpers into components/ui/form.tsx + lib/client/post.ts (with 48px targets and role=status feedback baked in) — migrate opportunistically
- lib/rows.ts with one named interface per table to replace the 253 inline 'as {...}' row casts
- Approval dedup guards: approveLodgeClaim should find-or-flag against the 25k existing lodges; approveGlClaim should catch the ux_juris_name unique violation cleanly instead of 500ing
- emitFeedItem update branch should honour input.occurredAt (corrected meeting dates keep stale feed chronology); backfill-feed.mjs needs AND sent_at IS NOT NULL so a future draft-notice feature can't leak unsent summonses

- Rewrite seed-sandbox --drop to enumerate all ~49 referencing tables (or derive from PRAGMA foreign_key_list) in one transaction with a zero-residue assertion — currently FK-aborts midway leaving a half-deleted test realm
- Geocode with country bias: prefer Nominatim hits matching the brother's lodge country and offer a one-tap 'not this one?' picker for ambiguous localities (Perth/Richmond/Newcastle)

- with a typeahead against the existing lodge[v] search (11k+ lodges are silently unreachable today) Time-aware returning-visit greeting instead of the permanent 'You're in, Bro. {name}.' band; compact single-row version on small screens Preserve the documented strength: the no-metrics constraint is enforced in code across every social primitive (feed, Fire, digest, presence, testimony, onboarding) — when wiring notifications/digest, reuse those names-never-numbers primitives verbatim and never add PWA badge counts Appendix — full finding set by dimension Architecture, code quality & maintainability The codebase is in unusually good shape for a solo-built app: lib/ modules are small, single-purpose and heavily documented with the design constraints inline (lib/feed.ts is exemplary — scope-gating, idempotent emits, batched fire reads); 69 of 87 API routes validate with zod and all use one uniform 401/400 error voice; strict TypeScript with zero any; the guardrail-check.mjs constraint linter is a genuine architectural asset. The debt is concentrated and specific: a verified-broken fresh-DB bootstrap (init-db.mjs ALTERs feed_items 180 lines before creating it — the documented reset/recovery path crashes), a silent dev-key fallback in lib/crypto.ts that would leave welfare/pastoral/rehearsal "encryption" using a publicly-known key if the env var is ever missing in prod, zero automated tests over the privacy-load-bearing scope gate and verification state machine, several N+1 query patterns (readMemberFeed re-gating, per-item comments/moderation, per-member credentials), an 898-line client god-file with UI helpers copy-pasted across 17 files, missing transactions around trust-critical multi-statement writes, and dead pagination that makes the home feed's "all caught up" terminus factually wrong past 40 items. Fresh-DB bootstrap crashes: feed_items ALTERed before it is created — 🟠 HIGH · effort S · /Users/micryan/masonic-network/scripts/init-db.mjs:729 (vs CREATE at :908) Problem: The documented reset flow in STATE.md ('rm -f data/masonicfire.db* && node scripts/init-db.mjs') and any new environment/disaster-recovery rebuild fails outright. The script only works against the existing live DB where feed_items already

11,000+ imported lodges in production, so any lodge past the 500th alphabetically simply cannot be invited to a special night — a silent correctness hole, not just slowness. A 500-option native dropdown also fails the 80-year-old usability law (no search, endless scrolling, tiny scroll thumb).

  - Fix: Replace the select with a typeahead backed by a small search endpoint (the /lodges directory search logic already exists to reuse) returning e.g. 20 matches on name/number/locality; keep one obvious input with large text per the usability law.

- **Dead pagination: `before` param has no callers, so the home feed lies at item 41** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/feed.ts:154,212 (opts.before) vs call sites app/member-home.tsx:37, app/brethren/[id]/page.tsx:185, app/profile/page.tsx:145; terminus at components/feed/feed-list.tsx:87-91`
  - Problem: readFeed/readMemberFeed implement occurred_at-cursor pagination, but every caller invokes them bare — the option is dead code. The home feed hard-caps at 40 items and then renders 'You are all caught up, Brother.' even when items 41+ exist, which is factually wrong once a lodge is modestly active; older posts/notices become permanently unreachable through the UI. The 'seen earlier' divider partially masks this today but the gap widens with adoption.
  - Fix: Add a quiet 'Earlier →' link (or server-navigated ?before= param page) at the bottom of FeedList that passes the last item's occurred_at into the existing before option; only render the 'all caught up' terminus when the query returned fewer rows than the limit.

- **253 unchecked `as {...}` casts on raw SQL rows — schema drift breaks silently despite strict TS** — 🟡 MEDIUM · effort M · `/Users/micryan/masonic-network/lib + app (repo-wide pattern; e.g. lib/auth.ts:51,142,192; app/lodge/manage/page.tsx:34,43)`
  - Problem: Every better-sqlite3 .get()/.all() result is hand-cast to an inline object type. tsconfig strict:true and zero `any` show the discipline is there, but the casts are trust-me typing: rename a column in init-db.mjs (or typo a SELECT alias) and the compiler stays green while the runtime silently reads undefined. Row shapes for hot tables (members, feed_items, lodges, lodge_memberships) are re-declared inconsistently at dozens of sites — Member in lib/auth.ts:16-33 is the only canonical one.
  - Fix: Create lib/rows.ts with one exported interface per table (mirroring init-db.mjs, which is already the single schema source) and cast to those named types instead of inline literals; hot paths get one place to update per schema change and greps become reliable. Full query-builder adoption is NOT needed.

- **auth_tokens are never purged and expired sessions only die on explicit sign-out** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/lib/auth.ts:200-207 (only session DELETEs in repo); no DELETE FROM auth_tokens anywhere`
  - Problem: Every sign-in attempt inserts an auth_tokens row (token_hash + code_hash) that lives forever — used or expired. Sessions expire logically (checked at :195) but the rows persist unless the brother signs out. Unbounded slow growth in the two hottest auth tables, and stale hashed

material lingers longer than needed in a database whose contents are treated like PHI (data-minimisation expectation under the Privacy Act framing the repo itself cites in db.ts:2-3).

  - Fix: Add a sweep to the existing cron surface (app/api/cron/reminders pattern, CRON_SECRET-gated): DELETE FROM auth_tokens WHERE expires_at < now or used=1 older than ~7 days; DELETE FROM sessions WHERE expires_at < now. Two statements, idempotent.

- **No `server-only` guard on lib/db.ts, and the guardrail linter is not wired into any gate** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/lib/db.ts:1 (comment only) and scripts/guardrail-check.mjs:5-6 + package.json:12`
  - Problem: Two cheap enforcement gaps. (1) lib/db.ts says 'Server-only' in a comment but nothing enforces it; today every client import from lib/* is type-only or a pure module (verified: comment-thread.tsx:9 and fire-button.tsx:9 use `import type`), but one future value-import of lib/feed from a 'use client' file yields an opaque better-sqlite3 bundling error instead of a clear one. (2) guardrail-check.mjs — the codified anti-vanity constraint linter, one of the repo's best assets — is standalone by its own admission ('NOT wired into next build... Exit 1 ... so Mic can wire it into CI later') and eslint.config.mjs ignores scripts/**, so nothing runs it automatically.
  - Fix: Add `import "server-only"` to lib/db.ts (and transitively everything DB-touching gets protected), and chain the deploy/build step to `npm run guardrail && npm run typecheck && next build` so a constraint regression can never ship silently.

SECURITY & PRIVACY

The security posture is well above typical for an app this young: every SQL statement sampled is parameterized, mutating routes are mostly zod-validated, login tokens/codes are hashed at rest with atomic single-use consumption, Stripe webhooks are signature-verified with a dedicated secret, uploads are re-encoded through sharp (EXIF/GPS stripped), the /media path handler blocks traversal, welfare and ritual content are AES-256-GCM encrypted, the whole site is noindex, and the directory is verified-members + opt-in only. However there is one critical failure that undercuts the whole model: the production MASONICFIRE_ENC_KEY and CRON_SECRET are committed in plaintext to git-tracked STATE.md and pushed to GitHub since the initial commit — the same key encrypts ritual lines and welfare notes AND HMAC-signs the public Pass tokens, so a repo leak yields both decryption of the most sensitive data and a public member-enumeration oracle via /pass/[token]. Below that, the magic-link flow has regressed from its own documented design (GET now consumes tokens, comments still claim prefetch-safety), token TTL is 7 days, the crypto key silently falls back to a hardcoded dev key in production, member photos are served unauthenticated with

immutable public caching, and two unauthenticated endpoints can be used to spray Paubox email with no IP throttle.

- **Production encryption key + cron secret committed to git and pushed to GitHub** — 🔴 CRITICAL · effort M · `/Users/micryan/masonic-network/STATE.md:155 (tracked; in history since initial commit d67db46; remote github.com/Micipedia/masonic-fire)`
  - Problem: Anyone with repo access (or a future repo leak) can (a) decrypt every welfare note and every brother's private ritual lines from any DB copy/backup, and (b) forge makePassToken(id) for sequential member ids and use the public /pass/[token] page as an enumeration oracle over the entire register — name, lodge, Grand Lodge, standing. This collapses both the 'membership data treated like PHI' and the ritual-privacy hard-lines in one shot.
  - Fix: Remove the secrets from STATE.md (reference the systemd unit / an untracked .env instead), purge them from git history (filter-repo) or rotate: rotate CRON_SECRET immediately (it is stateless); for MASONICFIRE_ENC_KEY write a one-off migration that decrypts all rehearsal_items.content_enc and welfare_requests.message_enc under the old key and re-encrypts under a new key (do NOT just swap the env var — STATE.md itself warns data becomes undecryptable), then update lib/pass.ts (all Pass QR codes re-derive automatically since tokens are stateless). Confirm with Mic before rotating (red-line: secret handling).
- **Magic-link token is consumed by an unauthenticated GET — contradicts the code's own prefetch-safe design** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/app/api/auth/verify/route.ts:27-29 vs /Users/micryan/masonic-network/lib/auth.ts:157-160`
  - Problem: Email security scanners (Outlook SafeLinks, corporate gateways, some AV) follow GET links: they burn the single-use token (brother sees 'expired' — the exact failure the interstitial was built to prevent) and, worse, the scanner's request receives a fully authenticated 30-day session cookie, creating live sessions in third-party infrastructure. Sign-in tokens also sit in URLs, which land in access/tunnel logs for up to 7 days while valid.
  - Fix: Restore the interstitial: have GET /api/auth/verify (or a /signin/confirm page) render a 'Confirm sign-in' page with one large primary button that POSTs the token; consume only in the POST handler (the POST handler already exists). This is a one-tap, single-primary-action flow that an 80-year-old can use, and it also fixes the burned-link UX. Update the stale comments either way.
- **MASONICFIRE_ENC_KEY silently falls back to a hardcoded dev key in production** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/lib/crypto.ts:16-17 and /Users/micryan/masonic-network/lib/pass.ts:8-10`
  - Problem: If the env var is ever dropped from the systemd unit (redeploy, unit rewrite, host migration), ritual lines and welfare notes get encrypted under a key that is literally published in the repo, and Pass tokens become forgeable by anyone — silently, with zero error. Given the key is currently 'baked into the unit' by hand, this is a realistic failure mode.

- Fix: In both key()/secret(), throw at first use when NODE_ENV === 'production' and MASONICFIRE_ENC_KEY is unset (fail loud, per the FAIL LOUD principle), keeping the dev fallback only outside production.

- **Sign-in link and 6-digit code stay valid for 7 days** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/auth.ts:8-12`
  - Problem: A 7-day window for a bearer credential sitting in an inbox (and in URL logs) is large; email is the sole authentication factor. The 6-digit code gets 5 guess attempts (good), but the guessing window is a week per issued code. The code's own comment says to tighten this before launch — it hasn't been.
  - Fix: Drop TOKEN_TTL_MIN to 30-60 minutes for public launch (still forgiving for a slow inbox), and once the POST-confirm interstitial is restored the scanner-burn rationale for the long TTL disappears entirely. Keep the resend path prominent so an expired link is a one-tap recovery.

- **Member photos and gallery images served unauthenticated with public immutable caching** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/media/[...path]/route.ts:12-24`
  - Problem: The register is default-closed ('treated like PHI') and the directory itself is gated to verified brethren, yet the photographs of identified members are world-readable to anyone who obtains or guesses a URL, and 'public, immutable' invites intermediary caches (Cloudflare tunnel/CDN) to retain them. A single leaked page-source or shared link permanently exposes a brother's face tied to his member id. The pages that embed these images all require sign-in, so nothing legitimate needs anonymous access.
  - Fix: Add a getCurrentMember() gate to /media (404 or redirect when signed out) and change caching to `private, max-age=...`; also add an `X-Robots-Tag: noindex` header since the HTML metadata noindex does not cover raw image responses.

- **Unauthenticated endpoints can spray outbound Paubox email with no IP or global throttle** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/api/enquire/route.ts:18-95` and `/Users/micryan/masonic-network/app/api/auth/request/route.ts:19-23`
  - Problem: Both endpoints are a spam relay / mail-bomb vector: arbitrary recipients receive Masonic Fire-branded mail containing attacker text, which burns Paubox quota and money, poisons the sender domain's reputation, and floods the enquiries table and ENQUIRY_NOTIFY inbox. Because the tunnel fronts the app, requests are cheap to script.
  - Fix: Add a small per-IP + global sliding-window limiter (a tiny SQLite table or in-memory map keyed on x-forwarded-for/CF-Connecting-IP) to both routes — e.g. 3 enquiries/hour/IP and 10 auth requests/hour/IP — plus an invisible honeypot field on the public enquiry form. Keep responses identical when throttled (no enumeration).

- **Raw rehearsal line — including the '□' secret-gap marker — is sent to the Whisper service as a bias prompt** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/api/practice/recognize/route.ts:63-66 (fd.append("prompt", reference)) with /Users/micryan/masonic-network/lib/rehearsal.ts:26-40`
  - Problem: The ritual hard-line states the '□' marker must never be 'checked/sent to ASR'. The marker (with its framing words) currently is sent — to own infra, and the marker itself carries no secret content, but it violates the letter of the rule and needlessly widens where gap-adjacent ritual context travels. Plaintext titles ('EA obligation, second section') are similarly the only rehearsal field outside the encryption envelope.
  - Fix: Send `normalize(reference)` (which already drops '□' and punctuation) as the prompt/biasing text instead of the raw line, and encrypt the title column the same way as content_enc (or document why titles are exempt). Also cap the `audio` field (currently `z.string().min(1)` with no max at recognize/route.ts:13) at ~10 MB of base64 to prevent memory-exhaustion posts.
- **Session identifiers stored in plaintext in the sessions table** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/auth.ts:183-198`
  - Problem: Any read of the SQLite file or a backup (the same threat model the team already defends against by hashing login tokens and encrypting welfare notes) yields every live session cookie value, allowing silent impersonation of any member for up to 30 days without touching the login flow.
  - Fix: Store sha256(sessionId) in sessions.id and hash the cookie value on lookup — a two-line change to createSession/getMemberBySession/destroySession, with a one-off migration hashing existing rows (or simply let old sessions expire by accepting both forms for 30 days).
- **CRON_SECRET passed as a URL query parameter on all three cron routes** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/app/api/cron/digest/route.ts:22-26, app/api/cron/reminders/route.ts:12-15, app/api/cron/dues-reminders/route.ts:23-27`
  - Problem: Query strings are recorded in access logs, Cloudflare tunnel analytics, and any intermediate proxy, so the cron secret leaks into logging infrastructure on every scheduled run (and it is already burned via the STATE.md commit). Comparison is also non-constant-time, though that is minor.
  - Fix: Move the secret to an `Authorization: Bearer` header (systemd timer curl change is one line), compare with crypto.timingSafeEqual, and rotate the value as part of the STATE.md secret rotation.
- **Pass token HMAC compared with non-constant-time equality and no revocation path** — ⚪ LOW · effort M · `/Users/micryan/masonic-network/lib/pass.ts:12-28`
  - Problem: The `!==` comparison is theoretically timing-observable (low practical risk over a tunnel), and because tokens are stateless and permanent there is no way to invalidate a brother's pass URL if it is shared or leaked — his name/lodge/standing page remains

resolvable forever. Combined with the committed HMAC key (critical finding) this is currently forgeable outright.

  - Fix: Use crypto.timingSafeEqual over full-length digests, and mix a per-member rotating nonce (e.g. a pass_salt column reset on demand or on 'sign out everywhere') into the HMAC payload so a brother can revoke a leaked pass. Keep the verifier page exactly as is — public tyler-check is the intended design.

- **Stripe integration is sound; webhook lacks only out-of-order event protection** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/app/api/stripe/webhook/route.ts and /Users/micryan/masonic-network/lib/stripe.ts:38-49`
  - Problem: Stripe explicitly does not guarantee event ordering and retries for days; a late-arriving stale event can flip a lapsed listing back to published (a brother getting a free listing — the inverse of the 'monetise supply-side' rule, and a books/records inconsistency).
  - Fix: Record processed event ids (or compare event.created against a last_sub_event_at column on members) and skip stale/duplicate events before calling applySubscriptionStatus.
- **Strength: authorization, validation and injection hygiene are consistently good** — ⚪ LOW · effort S · `app/api generally — e.g. /Users/micryan/masonic-network/app/api/lodge/[id]/welfare/route.ts:18-26, app/api/welfare/[id]/status/route.ts:14-21, app/api/credentials/[id]/route.ts:12-16, app/brethren/page.tsx:37-65`
  - Problem: No action needed — recorded so the other findings are read in proportion: the weaknesses are concentrated in secret handling and the auth-link flow, not in the day-to-day authorization fabric.
  - Fix: Keep the established route template (auth → integer-guard → role scope → zod) as the mandatory pattern for new routes; it is currently applied with impressive consistency.

UX, Usability (80-year-old Past Master bar) & Mobile

The foundations are genuinely strong: high-contrast tokens (--muted #b3bed2 is ~9:1 on surface, deliberately not faint grey), a 17px base with a documented 14px floor, global 48px min-heights on buttons/inputs/tel links, a skip link, a 3px brass :focus-visible ring, reduced-motion support, pinch-zoom not blocked, and a plainly-labelled 5-tab bottom nav with aria-current. The dopamine constraints are honoured everywhere I looked (Fire shows names never counts, seen-divider instead of badges, no pagination-bait). Where it falls down — and where Mic's "not mobile-optimised" complaint is real — is structural, not cosmetic: on any phone the entire right rail (Settle in onboarding, private "Words for you", actionable introductions, Relief) stacks BELOW the whole feed and the "all caught up" terminus, so on mobile the app's warmest, most actionable touchpoints are effectively invisible; the accessibility text-size control ("Large/Largest") is broken for most of the UI because it bumps body px while Tailwind sizes resolve in rem against html; the empty-feed primary CTA is a silent dead anchor for exactly the new unverified brother it targets; and several "Say hello" flows route through a trade-filtered search that frequently cannot find the named brother. Each of these fails the 80-year-old-Past-Master bar (clear feedback, one working primary action) in a specific,

fixable way. I found no concrete horizontal-overflow culprits — break-words/min-w-0 are used diligently — so the mobile problem is ordering and dead-ends, not overflow.

- **Text-size control ('Large/Largest') does not enlarge most of the UI — rem vs body-px bug** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/app/globals.css:160-169, /Users/micryan/masonic-network/components/text-size.tsx:24-28`
  - Problem: The comment claims 'Everything sized in em/rem grows with it', but rem resolves against the html root (fixed at 17px by globals.css:78-90), not body. Every Tailwind text utility (text-sm, text-lg, text-xl — 30 occurrences in member-home.tsx alone, plus buttons, nav labels, card metadata) is rem-based and therefore ignores the preference. An older brother who picks 'Largest' sees only unclassed paragraphs grow; headings, feed metadata, bottom-nav labels, and button text stay fixed. The one feature built specifically for older eyes is mostly a no-op.
  - Fix: Move the bump to the root: `html.fs-lg { font-size: 19px } html.fs-xl { font-size: 21px }` (dropping the `body` descendant). All rem-based type AND rem-based spacing then scale together, like browser zoom. Verify the fixed bottom nav and header still fit at 21px.
- **On mobile, the entire actionable rail (onboarding, private words, introductions, Relief) is buried below the whole feed** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/app/member-home.tsx:185-222,224-283`
  - Problem: The grid is `lg:grid-cols-[minmax(0,1fr)_18rem]` with the after the feed column in DOM order, and no order-* utilities exist anywhere (grep confirmed). Below 1024px everything stacks: welcome band → composer → up to 40 feed items → 'You are all caught up, Brother.' → THEN 'Settle in' (first-run onboarding), 'Words for you' (private greetings awaiting reply), 'An introduction asked of you' (a pending decision), Relief causes, next meeting, mentors. A phone-first 80-year-old will never scroll past the terminus to discover a greeting sent to him or an introduction awaiting his answer. This is the most likely root of Mic's 'not well mobile-optimised' complaint — the walker sees all content present, but the human never reaches it.
  - Fix: Split the rail into two slots: actionable/personal cards (Settle in, Words for you, IntroInbox, founder welcome) render ABOVE the composer on small screens (e.g., a `lg:hidden` block before the feed column, `hidden lg:block` copies in the aside — or CSS order utilities on a single flex parent), while ambient cards (Today in the Craft, next meeting, remembrance) stay below the feed on mobile.
- **Empty-feed primary CTA is a silent dead link for the unverified new brother it targets** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/components/feed/feed-list.tsx:44-50, /Users/micryan/masonic-network/app/member-home.tsx:206, /Users/micryan/masonic-network/lib/onboarding.ts:45`
  - Problem: The empty state's big brass primary button 'Introduce yourself to the brethren' is `<Link href="#post-body">`, but the composer (which owns id=post-body) renders only

when `verified && options.length > 0` (member-home.tsx:206). An unverified brother sees no craft-scope items (lib/feed.ts:177 gates 'craft' on isVerified), so he is precisely the person most likely to hit this empty state — and tapping the primary button does nothing at all. Same bug in the 'Say hello to the brethren' onboarding step (onboarding.ts:45, href '#composer'). A silent no-op primary action is the hardest possible failure of the usability law's 'clear feedback' and 'one obvious action' rules.

  - Fix: Pass the viewer's verified state into FeedList (it already receives `viewer`): when unverified, make 'Find your lodge' the single primary action with copy explaining posting opens once his lodge confirms him; keep the anchor CTA only when the composer actually exists. Filter the 'hello' onboarding step the same way.

- **'Say hello' and 'Brethren you may know' links route through the trade-filtered search and often cannot find the named brother** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/app/member-home.tsx:315,333,354, /Users/micryan/masonic-network/app/brethren/page.tsx:64-65`
  - Problem: The home rail links mentors, mentees, and 'brethren you may know' to `/brethren?q=<display name>`, but /brethren only returns members with `p.directory_opt_in = 1 AND p.trade IS NOT NULL AND TRIM(p.trade) <> ''`. A mentor who never listed a trade produces 'No brother answers to that just yet' — a dead end at the exact moment the app invited a warm gesture. The direct profile route `/brethren/[id]` exists and is already used for newBrethren (member-home.tsx:385) and words (line 254); all three broken call-sites already have the member id in hand (mo.id, me2.id, b.id).
  - Fix: Change the three hrefs to `/brethren/${id}` so 'Say hello →' always lands on the brother's page. Keep `/brethren?q=` only for genuine trade/calling searches.

- **'You are all caught up, Brother.' is untrue past 40 items — and older posts are unreachable** — 🟡 MEDIUM · effort M · `/Users/micryan/masonic-network/components/feed/feed-list.tsx:86-92, /Users/micryan/masonic-network/lib/feed.ts:154-155, /Users/micryan/masonic-network/app/member-home.tsx:37`
  - Problem: readFeed caps at 40 items (limit 40, max 100); readFeed's `opts.before` cursor is supported but no caller uses it (grep: only member-home.tsx:37 `readFeed(member)` and digest.ts) and there is no load-more UI anywhere. Once a lodge is active, the terminus fleuron plus 'You are all caught up, Brother.' renders directly under item 40 even when items 41+ exist — the copy lies, and a brother who missed a week can never read back past the cap. For a slow, occasional older user (the core audience), missing posts silently is worse than infinite scroll.
  - Fix: Fetch limit+1; when more exist, replace the terminus with a single plain link/button 'Read earlier posts →' (server-rendered `?before=` page or fetch-append), and only show the fleuron + 'all caught up' when the register is genuinely exhausted. A deliberate, user-initiated 'older posts' step preserves the finite, non-doomscroll design.

- **text-xs (~12.75px) used in 8 places, breaking the site's own 'nothing below 14px' floor** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/member-home.tsx:190, components/feed/feed-list.tsx:19, components/feed/feed-item-card.tsx:40, components/feed/post-composer.tsx:148, app/profile/page.tsx:361, app/pass/page.tsx:77,81, app/pass/[token]/page.tsx:95`
  - Problem: globals.css:19 declares the type scale's floor: 'nothing below 14px so nothing turns fiddly' (--fs-meta = 0.875rem = 14.875px at the 17px root). But Tailwind's stock text-xs (0.75rem = 12.75px) leaks through in 8 places, including load-bearing content: the video-conduct guidance in the composer (post-composer.tsx:148), the 'Around the Craft' card descriptions (profile/page.tsx:361), and the Pass page's instructions to show at a lodge door (pass/page.tsx:77,81 — read under pressure, possibly in poor light). Tiny uppercase-tracked labels ('Seen earlier', '★ Pinned') compound it.
  - Fix: Replace all text-xs with text-meta (the 14px token) — or, to make regressions impossible, disable the stock scale in @theme (`--text-xs: initial`) so text-xs stops compiling and the custom vocabulary (text-meta…text-hero) is the only one.
- **Bottom-nav 'Brethren' tab lands on a trade-only directory — expectation mismatch and frequent empty result** — 🟡 MEDIUM · effort M · `/Users/micryan/masonic-network/components/bottom-nav.tsx:15, /Users/micryan/masonic-network/app/brethren/page.tsx:64-65,100-101`
  - Problem: One of the five primary tabs is labelled 'Brethren', but /brethren lists ONLY verified members with directory_opt_in=1 AND a non-empty trade — a callings directory, not the register. Early on, a verified brother tapping 'Brethren' will often see nobody at all (empty state talks about 'callings gathering'), and even at scale he won't find brothers who never listed a trade. The page title 'The Brethren' with intro about 'their callings' does explain it, but the tab label sets the wrong promise — an 80-year-old will conclude the register is empty or broken.
  - Fix: Make /brethren a warm hub matching its label: put the existing 'Brethren you may know' card (already first on the page, line 118-130) and a people-by-name search front and centre, with trades/callings as an explicit labelled filter section — or relabel the surface honestly (e.g., tab stays 'Brethren', page leads with people, 'Trade & calling' becomes a secondary section). Keep results pull-based and opt-in-gated exactly as now.
- **Key rail action links have ~28px tap targets — the 48px rule covers buttons/inputs but not text links** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/globals.css:99-107,151-158, /Users/micryan/masonic-network/app/member-home.tsx:233,254,279,299,315,373`
  - Problem: globals.css gives 48px min-height to buttons, inputs, and tel:/sms: anchors — but plain / elements get nothing. The rail's primary actions are all bare text links: onboarding steps 'Find or add your lodge →' (member-home.tsx:233, text-lg ≈ 28px line box), 'from … — reply →' (254), 'Lend relief →' (279), 'Your lodge →' (299), 'Say hello →' (315), 'More brethren →'

(373). These are the exact links an older thumb on a phone must hit, several stacked 8px apart in space-y-2 lists.

  - Fix: Add a shared utility (or extend the globals rule) so action links get `min-height:48px; display:inline-flex; align-items:center` — the codebase already does this ad-hoc in ~10 places (`inline-flex min-h-[48px]` in feed-list.tsx:46, member-home.tsx:196); make it the default for the arrow-suffixed action-link pattern.

- **Disabled Fire button gives an unverified brother zero feedback on touch** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/components/feed/fire-button.tsx:83-103, /Users/micryan/masonic-network/components/feed/feed-item-card.tsx:97-105`
  - Problem: For an unverified brother, FireButton renders enabled-looking but `disabled` (fire-button.tsx:87 `disabled={!canGive || busy}`), with the only explanation in a `title` attribute ('The Fire') — which never appears on touch devices. He taps 'Give the Fire', nothing happens, no message. Silent-failure controls are a direct usability-law violation (clear feedback), and confusing precisely for the not-yet-confirmed member who least understands the system.
  - Fix: When !canGive, either hide the button (leaving the givers line if any) or render the pill as plain text with a one-line explanation: 'The Fire opens once your lodge confirms you.' Both keep the constraint that standing comes only from his lodge.
- **Feed photographs are invisible to screen readers (alt="")** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/components/feed/feed-item-card.tsx:77-84`
  - Problem: A post's image renders with `alt=""` , marking meaningful content as decorative — a brother using VoiceOver/TalkBack (realistic in this demographic) is never told a photograph exists, let alone whose it is. The brethren-list and may-know thumbnails correctly use alt="" (genuinely decorative next to the name), but the feed photo IS the content.
  - Fix: Use a modest generated alt, e.g. `alt={item.title ?` Photograph — ${item.title}`: Photograph shared by ${name}}` , and the same for gallery-kind items.
- **'You're in, Bro. {name}.' is permanent first-visit copy on every return visit** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/app/member-home.tsx:161-173`
  - Problem: The welcome band's h1 reads 'You're in, Bro. Mic.' under the eyebrow 'Welcome, Brother' on EVERY visit, forever — arrival copy that stops making sense from day two, occupies the prime top-of-screen slot on mobile (pushing the feed further down, compounding the rail-burial issue), and its casual register ('You're in') sits oddly against the site's otherwise dignified voice. The double-greeting (eyebrow + h1) also spends two lines saying one thing.
  - Fix: Make the h1 time-aware after first visit — 'Good evening, Bro. Mic.' (feedSeenAt already distinguishes first visit from returns) — and consider collapsing the band to a single compact row (avatar + greeting + standing chip) on small screens.

DATA INTEGRITY & CORRECTNESS

The core data machinery is unusually disciplined for a solo-built app: the feed's idempotent emit dedups on a partial-unique (source_kind, source_id) index, the constitution hierarchy roll-up (recursive CTE, one parent per lodge) is provably double-count-free, the migration scripts are genuinely idempotent, sample-lodge reseeding is guarded, and the W2 import even ships a zip-state integrity cross-check. But five real correctness faults stand out: (1) the whole lodge dedup layer — both the four import builders AND the live GL paste endpoint — keys on number-stripped lodge NAME only, which silently drops or merges legitimately distinct same-named lodges; (2) GL broadcasts don't traverse the constitution hierarchy, so a sovereign-GL notice reaches almost no one by email under constitution-first GLs while a district notice reaches the whole constitution in the feed; (3) the meeting-schedule parser mis-derives rules for multi-part schedules (real strings in the data) and is not wired to any write path, so meeting_rule silently rots when schedules are edited; (4) the feed/digest layer has zero is_test gating, so verified sandbox brethren can surface fake content to every real member; and (5) the final DB-import and all geocoding scripts for the 25k-lodge dataset live only in /tmp and are gone, making the flagship dataset unreproducible. All recommendations below preserve the no-counts/no-ranking, lodge-anchored-trust, and members-never-pay constraints.

- **GL broadcast recipients ignore the constitution hierarchy (feed and email audiences disagree)** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/app/api/jurisdiction/[id]/broadcast/route.ts:67-92`
  - Problem: Two-way mismatch. (a) For a constitution-first GL (UGLE: 370 lodges under 19 districts; Scotland: 930 under provinces — per STATE.md), a sovereign-GL broadcast records recipients and emails ONLY members of lodges attached directly to the root node — i.e. almost nobody — silently missing every district/province lodge. (b) Conversely, a DISTRICT officer's broadcast is pinned to the ROOT in the feed, so it is visible to the entire constitution, far wider than his district.
  - Fix: In the recipients query, expand the target node to its full subtree with the same recursive CTE used in lib/jurisdiction-tree.ts (`WHERE l.jurisdiction_id IN (SELECT id FROM sub)`). For the feed side, either scope district broadcasts to the district node and teach readFeed/viewerConstitutionRoots to match ANY ancestor node of the viewer's lodges (not just the root), or restrict can_broadcast emit-to-root to root-node officers.
- **Lodge dedup keys on number-stripped NAME only — silently drops/merges distinct same-named lodges (import scripts AND live paste endpoint)** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/build-us-import.cjs:47-52 (also build-eu-import.cjs:64-68, build-us-w2-import.cjs:51-55, build-ugle-districts.cjs:28-32) and /Users/micryan/masonic-network/lib/gl-claims.ts:186-196`
  - Problem: Within one Grand Lodge, two legitimately distinct lodges sharing a name but holding different warrant numbers (common in large US jurisdictions) collapse to one: the import silently DROPS the second; the live paste endpoint silently RENUMBERS the existing

lodge (data corruption) instead of adding the second. Unknown number of real lodges lost from the 25k import; the live path can corrupt officer-corrected rows going forward.

  - Fix: Change the dedup/find key everywhere to (jurisdiction, lower(name), coalesce(number,'')). In addLodgesToJurisdiction, only treat a row as 'existing' when the number also matches (or is absent on both sides); otherwise insert. Re-run a one-off diff of raw scrape files vs imported rows per GL to recover dropped same-named lodges.

- **Feed, comments, Fire, digest and home-rail causes have NO test-realm gate — verified sandbox brethren can surface to every real member** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/lib/feed.ts (no is_test anywhere), /Users/micryan/masonic-network/scripts/seed-sandbox.mjs:63-77, /Users/micryan/masonic-network/app/member-home.tsx:99-107, /Users/micryan/masonic-network/app/api/cron/digest/route.ts:29-37`
  - Problem: seed-sandbox creates test members with verification_status='verified', and member-home offers every verified member (including test members) the 'All brethren' craft posting scope. readFeed/readComments/firesForItems contain zero is_test filtering, so one craft-scoped sandbox post/comment/Fire (routine when QA 'drives the app end-to-end') appears in every real brother's feed AND weekly digest ('there's been fellowship…'). Separately: (a) the home-rail reliefCauses query joins causes→lodges with no is_test filter, so a sandbox cause shows on every verified member's home; (b) the digest cron selects recipients with no is_test filter, mailing @sandbox.masonicfire.test addresses. The lodges/grand-lodges/brethren/visit pages all filter is_test correctly — the feed spine is the one surface that never got the treatment.
  - Fix: Add the canonical gate to the feed spine: in readFeed/readMemberFeed/readComments/firesForItems join members/lodges and exclude rows where the author or lodge is_test=1 unless canSeeTestRealm(viewer); add `AND m.is_test=0` to digest-cron recipients and `AND l.is_test=0` (plus author check) to the reliefCauses rail.

- **25k-lodge DB-import and ALL geocoding scripts live only in /tmp — flagship dataset is unreproducible** — 🟠 HIGH · effort M · `STATE.md:104-108 (references /tmp/geocode-lodges.cjs, /tmp/geocode-au.cjs, /tmp/geocode-addr.cjs); repo-wide grep`
  - Problem: The repo contains the wf-* scrapers and build-*-import.cjs consolidators, but the step that actually WRITES the 25,000 lodges/473 jurisdictions into the DB, and every geocoding pass (GeoNames stream + AU/GB postcode + town-from-address, 21,958 rows), existed only as /tmp scripts that are now gone. The documented follow-ups — WA/IL-PH/MA-PH partial re-imports, re-geocoding after re-scrapes, re-running the meeting_rule population — cannot be replayed with the same (Gibraltar-hardened, parent-scoped) logic; a future pass will re-invent it and risk re-introducing the cross-parent name-stealing bug.
  - Fix: Reconstruct and commit scripts/import-lodges.mjs (JSON→DB, parent-scoped find-or-create with the codified '(… Constitution)' collision-qualifier) and scripts/geocode-lodges.mjs

(GeoNames stream + postcode passes), both idempotent, so every future re-scrape/refresh replays identical logic.

- **parseSchedule mis-derives rules: only the FIRST weekday is read, so real multi-part schedules produce WRONG 'Next meeting' dates** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/lib/meeting.ts:14-24`
  - Problem: (a) `t.match(/\b(sunday|...)\b/)` is non-global — for '1st Tuesday … | 3rd Thursday …' it captures only 'tuesday', while the ordinal scan collects BOTH 1 and 3 → rule '1,3:2' = 1st AND 3rd Tuesday. The 3rd-Thursday meeting is shown as a Tuesday — a factually wrong date on the lodge page, /visit 'near you this week', and member-home, where a visiting brother may act on it. (b) The unconditional word-ordinal loop (line 24) and digit scan (line 19) pick up strays: 'first floor', '3rd floor', addresses. (c) 1,300+ US schedules carry 'except July, August'/'dark' caveats the rule ignores, so summer 'next meetings' are asserted for dark months. (a) is a hard bug, not the documented 'approximate' limitation.
  - Fix: Split the text on '|'/';' and parse each segment to its own (ords:weekday) pair, storing a multi-part rule (e.g. '1:2|3:4') with nextMeetingDate taking the min across parts; require word-ordinals to sit within a few tokens of the weekday; when /except|dark/i names months, either skip those months in nextMeetingDate or decline to parse and show the raw schedule text instead of a computed date.
- **Amity read-scope is self-expandable via pending visit_requests and self-authored visitation_log entries** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/feed.ts:114-133`
  - Problem: amityMemberIds() treats ANY visit_request (including status='pending' or 'declined') and ANY visitation_log row (an unverified, self-written journal entry) as a 'visited lodge', then adds ALL active members of that lodge to the viewer's amity set. A brother can therefore unilaterally widen which amity-scoped posts he can read — and whose comments he may join — simply by filing visit requests or journal entries against arbitrary lodges. This contradicts the module's own 'every read is scope-gated so membership stays private (default-closed)' contract: the members of the target lodge never did anything to admit him.
  - Fix: Require `status = 'acknowledged'` on the visit_requests branch (host-lodge consent) and drop visitation_log from the amity computation entirely (it is a private journal, not a trust edge). Keep co-lodge and family branches unchanged.
- **Sandbox --drop teardown is incomplete and FK-fragile: leaves test residue and aborts midway once QA has exercised newer features** — 🟡 MEDIUM · effort M · `/Users/micryan/masonic-network/scripts/seed-sandbox.mjs:19-37`
  - Problem: The drop block deletes 12 tables but the schema has ~49. Missing: verifications, avouchments, feed_items/feed_comments/feed_fires, confirmation_requests, lodge_claims, visit_requests, visitation_log, welfare_requests, gallery_images, sessions, auth_tokens, event_reminders, event_invitations, mentorships, member_greetings, member_testimony,

member_businesses, notice_recipients, year_plans, pastoral_followups… Because the connection sets `db.pragma('foreign_keys = ON')` (line 15), the final `DELETE FROM lodges/members WHERE is_test = 1` throws an FK-constraint error as soon as any of those tables reference a test row (e.g. one QA feed post, one verification event from an approved test claim), aborting db.exec midway — a HALF-deleted test realm, which is worse than an intact one.

  - Fix: Rewrite --drop to enumerate every table that can reference a test member/lodge/jurisdiction (child tables first, inside one transaction), or derive the set from PRAGMA foreign_key_list at runtime; end with an assertion that `SELECT COUNT(*) … WHERE is_test=1` is zero across the three flagged tables and fail loudly otherwise.

- **meeting_rule is never recomputed — parseSchedule has zero callers in any write path** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/gl-claims.ts:190-195 (schedule update path); /Users/micryan/masonic-network/lib/meeting.ts:11`
  - Problem: lodges.meeting_rule was populated once by a /tmp script (STATE.md:107 'Populate again after re-scrapes via the parser'). The live paste endpoint updates meeting_schedule via COALESCE but never touches meeting_rule, and repo-wide grep shows parseSchedule is imported nowhere. So the moment an officer corrects his lodge's meeting night, the page shows the NEW schedule text alongside a 'Next meeting' date computed from the OLD rule — a self-contradicting lodge page. The 'computed live so it never goes stale' design only holds for the date, not the rule.
  - Fix: Call parseSchedule(newSchedule) and write meeting_rule in the same UPDATE wherever meeting_schedule is written (addLodgesToJurisdiction insert AND update branches; any future lodge-profile edit route), nulling the rule when the parse fails so no stale date is shown.
- **Proposed-lodge/GL approval inserts blindly: duplicate lodges possible, and a duplicate GL name throws mid-approval** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/lodge-claims.ts:140-155 and /Users/micryan/masonic-network/lib/gl-claims.ts:116-130`
  - Problem: (a) approveLodgeClaim inserts the proposed lodge with no check against the 25k existing rows — a brother proposing a lodge that IS listed (misspelled search, district vs GL confusion) creates a duplicate on approval, and the genesis reviewer gets no 'possible match' signal. (b) approveGlClaim's plain INSERT collides with the global unique index ux_juris_name (init-db.mjs:33): approving a proposed GL whose name already exists throws an unhandled SqliteError inside the transaction → 500 on the admin approve action.
  - Fix: Before insert, find-or-flag: look up (proposed_jurisdiction_id, lower(name), number) among lodges — on a hit, attach the claim to the existing lodge instead of creating; for GLs, surface near-matches (lower(name)) on the admin review card and catch the unique-violation with a clean 'already listed' error.

- **No DB-level uniqueness on lodges — all dedup is script-side convention** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/scripts/init-db.mjs:36-55`
  - Problem: The lodges table has only non-unique indexes (ix_lodge_juris, ix_lodge_region, ix_lodges_latlng). Every 'idempotent' guarantee (import re-runs, GL paste, claim approvals) is enforced in JavaScript with three different keying conventions. One divergent code path re-inserts silently; the 2026-07 audit found 0 dups today, but nothing structural keeps it that way across the four write paths.
  - Fix: Add `CREATE UNIQUE INDEX IF NOT EXISTS ux_lodge_identity ON lodges(jurisdiction_id, lower(name), coalesce(number,''))` after a one-off pre-check for existing violations, and switch script inserts to INSERT-or-rescue so the DB is the final arbiter.
- **emitFeedItem's refresh path never updates occurred_at — re-emitted items keep stale chronology** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/lib/feed.ts:286-301`
  - Problem: The update branch refreshes kind/scope/title/body/link/etc. but not occurred_at, even when the caller passes input.occurredAt. A meeting whose date is corrected (or a notice re-issued) and re-emitted keeps its ORIGINAL feed position and implied date — the feed is ordered by occurred_at DESC, so a moved meeting sits at the wrong point in time and may never resurface for brethren who scrolled past it.
  - Fix: Add `occurred_at = COALESCE(?, occurred_at)` to the update statement, passing input.occurredAt ?? null.
- **Member-locality geocoding takes Nominatim's first hit with no country bias or ambiguity handling** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/lib/geocode.ts:16-35`
  - Problem: `limit=1` with no countrycodes/viewbox bias means globally ambiguous localities resolve arbitrarily ('Perth' → Perth, Scotland for a WA brother; 'Richmond', 'Newcastle' likewise). The wrong lat/lng silently drives /brethren km-radius and /visit 'lodges near you', placing a brother on the wrong continent. The returned label is shown back (some mitigation) but nothing prompts him when the match landed in an unexpected country.
  - Fix: Request limit=3-5 and prefer a hit whose address.country matches the brother's lodge's/constitution's country (lodges.country is already on file); when candidates span countries, show the label prominently with a one-tap 'not this one?' picker — plain-language, single-action, consistent with the usability law.
- **backfill-feed.mjs would publish DRAFT notices (sent_at IS NULL) to the lodge feed** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/scripts/backfill-feed.mjs:81-119 (vs init-db.mjs:489 'sent_at … null = draft')`
  - Problem: The backfill selects `FROM notices WHERE kind != 'minutes'` with no sent_at filter, then uses `sent_at || created_at` as occurred_at. The schema explicitly models drafts (sent_at NULL = draft) and the live emitters only fire at send/issue time. Today every write path stamps sent_at immediately, so this is latent — but the first feature that saves a

draft notice (a natural Secretary ask) makes the next backfill run leak unsent summonses/notices into the brethren's feed.

  - Fix: Add `AND sent_at IS NOT NULL` to the backfill's notices query so the script matches the live emitters' send-time semantics.

PRODUCT, GROWTH & RETENTION

The social overhaul is genuinely well-built and constraint-faithful: the feed is reverse-chronological with comments as the only primitive, the Fire is named-and-uncounted with an explicit "there must never be a count function" guard (lib/feed.ts:478-517), the digest is count-free by construction, onboarding has no progress bar, presence is fuzzy, and business listings buy presence-never-prominence. The product DESIGN creates stickiness without dopamine; the problem is that almost every loop is open-circuit at its return point. The two email hooks that would bring a brother back — the weekly digest and "a brother replied to you" — both exist in code and both send nothing (digest ships dark with no timer; the reply toggle is a dead preference no sender reads). The two viral loops leak at their landing moments: the Pass verifier page has no hook for the scanner, and "Let them know I'm coming" tells the visitor "the lodge has been told to expect you" when, for effectively all 25,000 unclaimed lodges, nothing was sent to anyone. Activation similarly stalls silently: a new brother who asks an unclaimed lodge to confirm him is told "we'll notify the founders" but no notification exists and no admin queue surfaces him. The secretary-pulls-in-the-lodge loop works one brother at a time with no roster-invite tool. The single biggest lever to make brethren return: close the reply loop and schedule the already-built digest — both are S-effort wiring jobs on code that already respects every constraint. The single biggest growth lever: make the no-officer case honest and generative (visit requests, confirmation requests, and the Pass verifier should each recruit the missing lodge officer instead of dead-ending).

- **Visit-request loop silently dead-ends at unclaimed lodges while the UI asserts the lodge was told** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/app/api/lodge/[id]/request-visit/route.ts:43-63` and `/Users/micryan/masonic-network/app/lodges/[id]/visit-button.tsx:19`
  - Problem: STATE.md names visitation the killer utility and top growth loop, but a brother can travel to a lodge on a false promise — nobody at that lodge ever heard. One burned visit destroys trust in the whole register, and the moment that should recruit the missing lodge officer is wasted.
  - Fix: When the lodge has no rooted officer, return an honest state and render honest copy ("This lodge is not yet on Masonic Fire — confirm with its Secretary directly before travelling"), fall back to notifying ENQUIRY_NOTIFY so the founder can broker it, and add a quiet line the brother can share: "An officer of this lodge may claim its page" linking /lodge/claim.
- **"Tell me when a brother replies to me" is a dead toggle — comments never notify anyone** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/app/profile/notifications/`

`notifications-form.tsx:110, /Users/micryan/masonic-network/lib/feed.ts:410-434, /Users/micryan/masonic-network/app/api/feed/comment/route.ts`

  - Problem: Comments are deliberately the ONLY interaction primitive of the feed, yet a brother never learns his post was answered unless he happens to reopen the app. The conversation loop — the core retention mechanic of the social overhaul — is open-circuit, and the UI promises a behaviour that doesn't exist.
  - Fix: On addComment, send a quiet email to the item's author and (for nested replies) the parent-comment author: name + short excerpt + link, honouring reply_notices_off and notify_email, batched per item per hour to avoid floods. Never a count ("Bro. John replied to your word"), never a badge number in-app. This is the single biggest retention lever in the codebase.

- **Weekly digest is fully built, constraint-proofed, and permanently dark — no timer exists** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/app/api/cron/digest/route.ts:9-11, /Users/micryan/masonic-network/lib/digest.ts, STATE.md:163`
  - Problem: For a low-frequency fraternal product, the weekly "a quiet word from your brethren" email IS the retention system — lodge nights are monthly, so nothing else pulls a brother back between meetings. The best-engineered retention asset in the app has never sent a single email.
  - Fix: Add a masonicfire-digest.timer (weekly, e.g. Sunday 17:00 local) hitting /api/cron/digest?key=$CRON_SECRET, mirroring the existing reminders timer unit. Verify one send against Mic's own account first.

- **Confirmation requests to officer-less lodges promise notification that is never sent, and no admin queue surfaces them** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/app/api/lodge/[id]/request-confirmation/route.ts:69-75, /Users/micryan/masonic-network/app/admin/page.tsx:21`
  - Problem: This is the exact moment a brand-new brother does the right thing (found his lodge, asked to be confirmed) at any of the ~25,000 unclaimed lodges — and he falls into a queue nobody reads. Activation stalls silently at its most motivated point; the brother waits for a "we'll let you know" that can never come.
  - Fix: Email ENQUIRY_NOTIFY when a request lands on an officer-less lodge, and add an /admin section "Brethren awaiting an unclaimed lodge" (name, lodge, date) with actions: nudge the brother to ask his Secretary to claim the lodge, or genesis-verify when Mic can vouch directly.

- **No roster-invite tool for the Secretary — the pull-in-the-lodge loop is one brother at a time across two screens** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/app/api/lodge/[id]/add-member/route.ts:31-34, /Users/micryan/masonic-network/app/lodge/`

`manage/lodge-manage.tsx:795-799, /Users/micryan/masonic-network/app/api/invite/route.ts`

  - Problem: STATE.md calls Secretary use "the whole-lodge adoption engine", and the Minute Book/summons/dues spine only delivers value once the roster is in. The highest-leverage user in the growth model is given the most manual, error-prone path in the app.
  - Fix: Add "Invite your brethren" to /lodge/manage: paste a list of emails (optionally names), send the existing dignified invite mail from the Secretary's name, pre-create roster membership so each brother lands attached to the lodge on first sign-in and appears straight in the Secretary's confirm queue. Confirmation stays a separate deliberate act.

- **The Pass verifier page — the loop's landing moment — has zero hook for the scanner** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/pass/[token]/page.tsx:69-99`
  - Problem: STATE.md:101 states "every Pass shown at a host lodge advertises the app to that lodge" — but the advertisement is wired to nothing. The scanner (typically a Tyler/Secretary — exactly the officer the growth model needs) sees a card and has no path in.
  - Fix: Add a quiet footer: one sentence ("Masonic Fire is a verified register of the brethren — free for members, private to Masons") plus "An officer of a lodge may claim its page" linking /lodge/claim, and "Find a lodge" linking /lodges. Dignified, small type, below the disclaimer.

- **Visiting and the Pass are three taps deep — absent from bottom nav and the home rail** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/components/bottom-nav.tsx:11-17, /Users/micryan/masonic-network/app/member-home.tsx:222-409, /Users/micryan/masonic-network/app/profile/page.tsx:209-210`
  - Problem: The self-declared killer utility ("true what's-on near me this week" + the Pass at the door) is the hardest daily surface to reach. An 80-year-old at a strange lodge's door must navigate Home → You → Your Pass under pressure — against the one-obvious-action usability law.
  - Fix: Add a compact "Visiting — your Pass and lodges near you" card to the home rail (verified only), and consider replacing or supplementing a nav tab (e.g. Notices, whose content already appears in the feed) with "Visiting". Keep labels plain and large.

- **Onboarding step "Say hello to the brethren" is a dead anchor for unverified brethren** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/onboarding.ts:41-46, /Users/micryan/masonic-network/app/member-home.tsx:206-214`
  - Problem: The very first-run experience for the exact cohort that needs guidance (brand-new, unverified) contains a silently broken action — corrosive to the "forgiving linear flows, clear feedback" usability law and to activation momentum.
  - Fix: In remainingSteps, include the hello step only when the member is verified; while unverified, the card then naturally leads with the one step that matters ("Find or add your lodge"), which is the correct single obvious action.

- **Landing and login copy misstate the join path — "a brother must vouch for you" deters legitimate self-serve joiners** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/page.tsx:65-67, /Users/micryan/masonic-network/app/login/page.tsx:173-177`
  - Problem: A regular brother arriving by word of mouth (no invite) reads that he cannot join without a voucher and bounces. The primary CTA is also "Sign in", which implies an existing account — there is no "Join" affordance anywhere on the anonymous surface.
  - Fix: Update copy to match the real trust model: landing secondary CTA "Join the register — free for brethren, forever" (same /login), and login footer "New here, Brother? Enter your email to begin — then find your lodge and ask it to confirm your standing." Keep avouchment described as the courtesy it now is.
- **Invite link token is ignored at /login — no warm landing, and attribution breaks on a different email** — 🟡 MEDIUM · effort M · `/Users/micryan/masonic-network/app/api/invite/route.ts:38-39, /Users/micryan/masonic-network/app/login/page.tsx:8-9, /Users/micryan/masonic-network/lib/auth.ts:81-90`
  - Problem: The invited brother gets a generic sign-in page (with the misleading "a brother must vouch for you" copy from the previous finding) instead of "Bro. Michael invited you" — the warmest possible first screen — and the loop's G1/G3 provenance silently fails whenever emails differ (common: invite to work address, joins with personal).
  - Fix: Read the invite param: show the inviter's name and note, prefill the invited email, and stash the token so the invite is marked accepted and proposed_by set by TOKEN when the completing email differs. Still one proposer, no tree, no counts.
- **Public enquiries assigned to a claimed lodge never email that lodge's Secretary — only Mic** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/api/enquire/route.ts:64-77`
  - Problem: "Speed-to-lead that keeps a good man from being lost" is the route's own stated purpose and the day-one value pitch to Secretaries (BLUEPRINT enquiry funnel) — but the lead sits unseen unless the founder manually forwards it. At any scale beyond a handful of lodges this becomes the bottleneck.
  - Fix: When assigned_lodge_id resolves to a lodge with active verifying officers, email them too (same content, honouring notify_email), keeping ENQUIRY_NOTIFY as oversight; keep the /lodge/enquiries triage as the workspace.
- **Monetisation is realistically $0 for a long time, and the DenialHelp-LLC Stripe bridge conflicts with the not-for-profit posture GL relations depend on** — 🟡 MEDIUM · effort L · `/Users/micryan/masonic-network/lib/stripe.ts:1-3,17-19, /Users/micryan/masonic-network/lib/business.ts:1-4, BLUEPRINT.md:20-23`
  - Problem: The one revenue line needs verified-brethren density before any brother pays for a listing, so revenue is structurally $0 until the growth loops above work — fine, but plan for it.

Worse: if the paid gate is ever flipped while charges settle to an unrelated US healthcare company's Stripe, a wary Grand Secretary discovering "DenialHelp LLC" on a brother's statement torpedoes the "we serve the Craft" trust pitch at exactly the moment endorsement is sought.

  - Fix: Keep the gate dark (it is); make the entity a pre-launch blocker: before setting the STRIPE_* envs, stand up the mission entity (BLUEPRINT already lists NFP + trademark as pre-launch) and a Masonic Fire-named Stripe account. No code change needed today beyond a comment guard noting the entity precondition.

- **Sandbox test realm leaks into the /trades and /brethren constitution dropdowns for real members** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/app/trades/page.tsx:36-38, /Users/micryan/masonic-network/app/brethren/page.tsx:52-54, /Users/micryan/masonic-network/scripts/seed-sandbox.mjs:40-48`
  - Problem: Result rows are gated (queryTrustedTrades takes seeTest), so no member data leaks — but a fake Grand Lodge named "Sandbox" appearing in a solemn register's filter undercuts the dignified, real-world credibility the product trades on, and contradicts the documented "public surfaces always exclude test" rule.
  - Fix: Add `AND is_test = 0` to both dropdown queries (mirror /visit/page.tsx:74).
- **STRENGTH — the no-metrics constraint is enforced in code, not just copy, across every social primitive** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/lib/feed.ts:478-543, /Users/micryan/masonic-network/lib/digest.ts:18-25, /Users/micryan/masonic-network/lib/presence.ts, /Users/micryan/masonic-network/lib/testimony.ts:3-6, /Users/micryan/masonic-network/lib/onboarding.ts:4-6`
  - Problem: None — this is the moat. Recorded so the synthesis doesn't mistake missing dopamine mechanics for missing product: the delight-without-metrics bet is faithfully implemented and should be defended against any future 'engagement' suggestion.
  - Fix: Preserve as-is; when wiring the notification/digest levers above, reuse these primitives verbatim (names, never numbers) rather than introducing badge counts on the PWA icon or nav.

RITUAL HARD-LINES + NO-METRICS + NEVER-NETWORKING + PHI-DEFAULT-CLOSED conformance sweep

The design laws are unusually well-internalised in this codebase — lib/feed.ts implements The Fire as names-never-numbers (fire-button.tsx trails off into "and other brethren" with no count), digest.ts emits zero numbers, presence.ts never says "online now", suggestions/introductions are reason-based and directory_opt_in-gated (default 0 in init-db.mjs:668), verification is strictly lodge-anchored (lib/verification.ts:111-127 — peer avouchment confers nothing), Trusted Trades is pull-based patronage sorted alphabetically/by nearness only, the paid /businesses listing is flat-alphabetical "presence never prominence", welfare/pastoral/visit-journal/funeral-wishes/rehearsal content are all

AES-256-GCM encrypted at rest, the □ secret-gap mechanics are structurally sound (normalize drops it, firstLetters passes it through unhinted, isAllSecret short-circuits the mic path), and the whole site is meta-noindexed. The genuine breaches found are concentrated at the seams: the unauthenticated /media/[...path] handler serves member photos and scope-gated gallery/post images to anyone with a URL (the largest default-closed hole); the Whisper recognizer sends the decrypted ritual reference line INCLUDING the □ marker to WHISPER_URL as a biasing prompt with no own-infra enforcement (comment-only guarantee); crypto.ts silently falls back to a repo-public dev key in production if MASONICFIRE_ENC_KEY is unset; and three displayed counts ("known to N brethren", "N gifts recorded", "N brethren" per trade category) drift against the WHO-never-how-many law the rest of the app honours.

- **Unauthenticated /media handler serves member photos and scope-gated images (default-closed breach)** — 🟠 HIGH · effort M · `/Users/micryan/masonic-network/app/media/[...path]/route.ts:12-22`
  - Problem: Membership data is to be treated like PHI/default-closed, yet the raw image bytes behind every avatar, lodge/GL gallery photo, and amity/lodge/family-scoped feed image are fetchable by anyone on the internet who has or guesses a URL, bypassing every scope gate in lib/feed.ts. Filenames leak structure and are semi-enumerable: profile photos are `m{memberId}_{Date.now().toString(36)}.jpg` (photo/route.ts:52) and gallery images `g_{kind[0]}{ownerId}_{memberId}_{ts36}.jpg` (gallery/route.ts:64) — a millisecond timestamp in base36, not a capability token. A leaked/forwarded URL also stays cached publicly for a year.
  - Fix: Require a session in the /media GET (all legitimate consumers are signed-in pages, so the cookie flows with same-origin requests); switch Cache-Control to `private, max-age=...`; and generate filenames from crypto.randomBytes rather than ids+timestamps. Optionally scope-check gallery/post images against gallery_images/feed_items the way api/feed/media does for video.
- **Encryption silently falls back to a repo-public dev key in production** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/lib/crypto.ts:8-18`
  - Problem: Almoner welfare messages, pastoral notes, visitation journals, funeral wishes, and the brother's private ritual lines are all encrypted with key() — but if MASONICFIRE_ENC_KEY is unset in the NucBox systemd unit, key() silently returns sha256('masonic-fire-dev-key'), a constant anyone with repo access can derive. The 'ENCRYPTED at rest' guarantee for Almoner/welfare data (a hard-line) then holds only nominally, with no error, log, or visible symptom.
  - Fix: Fail loud: `if (process.env.NODE_ENV === 'production' && !process.env.MASONICFIRE_ENC_KEY) throw new Error('MASONICFIRE_ENC_KEY required')` (or assert once at startup). Then verify the deployed unit on the NucBox actually sets the key before restarting.

- **Ritual reference line (including the □ marker) is sent to the Whisper endpoint, with own-infra enforced by comment only** — 🟠 HIGH · effort S · `/Users/micryan/masonic-network/app/api/practice/recognize/route.ts:51-63`
  - Problem: Two hard-line gaps in one route. (1) `fd.append("prompt", reference)` / `fd.append("biasing", reference)` ships the brother's decrypted ritual line VERBATIM — including any □ secret-gap markers — to the ASR service, breaching the stated rule that □ is never sent to ASR (the marker's position inside the line is itself information). (2) Nothing enforces that WHISPER_URL is own infra: the comment promises 'ritual audio NEVER goes to a third party', but the code will POST ritual audio + the reference text to ANY URL the env var holds, over plain fetch.
  - Fix: (a) Send `reference.replaceAll(SECRET_GAP, "")` (or normalize(reference)) as the biasing prompt — matching already uses normalize() so nothing is lost. (b) Validate WHISPER_URL at startup against a private-host allowlist (127.0.0.1/localhost/Tailscale 100.64.0.0/10) and refuse to call anything else — turning the own-infra promise from a comment into code.
- **Mic path is open on mixed lines containing □ with no 'leave the secret silent' guidance** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/practice/practice-client.tsx:248-303`
  - Problem: isAllSecret() correctly removes the mic for all-secret lines (line 248 shows 'recite it silently'), but a line mixing ordinary words and □ (e.g. 'I greet you with □') keeps the full mic path. Nothing tells the brother to keep the gap silent while reciting aloud, so his spoken secret can be captured in the audio sent to Whisper — and because normalize() drops □ from the reference, the spoken secret also counts as an insertion error that can fail him on a short line.
  - Fix: When micOn && current.includes(SECRET_GAP), show one plain line above the card: 'This line holds a guarded secret (□) — pass over it in silence when you recite.' Large text, matching the 80-year-old usability bar.
- **'known to N brethren' — a per-brother displayed count in the Secretary console** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/lodge/manage/lodge-manage.tsx:691-693 (fed by app/lodge/manage/page.tsx:75 via lib/verification.ts:131)`
  - Problem: The design law is WHO, never HOW MANY — and this renders a literal number attached to a person: `known to {m.knownBy} {m.knownBy === 1 ? "brother" : "brethren"}`. In a member list it reads as a mini-ranking of brethren by vouches (exactly the pattern The Fire forbids), even though avouchment correctly confers no standing.
  - Fix: Show the voucher NAMES instead (e.g. 'known to Bro. Smith and Bro. Jones', trailing off like fire-button.tsx does after 6 names). More useful to a Secretary vetting a confirmation, and conforms to the no-counts law.

- **'N gifts recorded by the brethren' — a participation counter on every cause card** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/causes/causes-client.tsx:192-194 (fed by app/causes/page.tsx:37)`
  - Problem: `{c.gifts} gift{...} recorded` is a displayed social count that invites cross-cause comparison — the member-home comment (app/member-home.tsx:98) already recognises this hazard ('no fundraising leaderboard... no totals paraded'). The raised-amount + target progress bar is functional relief coordination and fine to keep; the gift COUNT adds nothing functional over it.
  - Fix: Drop the count or replace it with named non-anonymous givers ('Gifts recorded by Bro. A, Bro. B and other brethren' — trailing off uncounted, honouring the anonymous flag), keeping raised/target untouched.
- **/businesses and /causes visible to unverified accounts — anyone with an email can see brethren's names** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/app/businesses/page.tsx:17-19; /Users/micryan/masonic-network/app/causes/page.tsx:25-26; /Users/micryan/masonic-network/app/api/causes/[id]/gift/route.ts:17-18`
  - Problem: findOrCreateMember (lib/auth.ts:49-60) creates an unverified 'brother' account for ANY email that completes magic-link sign-in. /brethren and /trades correctly gate on verification_status === 'verified', but /businesses only checks login+not-family — and its listing prints owner_name (businesses/page.tsx:67), disclosing that a named man is a Freemason to any stranger who signs up. /causes similarly shows lodge relief activity to login-only, and the gift route lets an unverified account write donation records.
  - Fix: Apply the same verified gate (or the /trades pattern of an explanatory gate card for unverified members) to /businesses, /causes, and the gift-recording API.
- **GL certificate number stored in plaintext despite hashes-of-evidence posture** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/scripts/init-db.mjs:724; /Users/micryan/masonic-network/app/api/profile/details/route.ts:55`
  - Problem: The schema's own comment says `certificate_no ... — PHI, owner-only, NEVER on the Pass`, and the repo's stated rail is 'only HASHES of any evidence stored' (init-db.mjs:6, credentials.evidence_hash at line 200) — yet the raw certificate number sits plaintext in profiles, while comparable fields (funeral_wishes_enc, welfare message_enc, pastoral note_enc, rehearsal content_enc) are AES-GCM encrypted. Its only readers are the owner's settings page and his own data export, so plaintext buys nothing.
  - Fix: Store it encrypted (certificate_no_enc via lib/crypto, decrypted server-side for the owner only), with a one-time migration of existing rows.

- **'Today in the Craft' broadcasts near-verbatim degree-lecture paraphrases** — 🟡 MEDIUM · effort S · `/Users/micryan/masonic-network/lib/craft-light.ts:13-14,20,23 (and neighbours)`
  - Problem: The hard-line is never host/ship/broadcast ritual text — the brother brings his own lines. The file's rule ('public Masonic education only... NEVER ritual secrets') is honoured for signs/words/grips, but several entries track the cadence of the working-tools and cardinal-virtue LECTURES almost verbatim (e.g. the 24-inch Gauge 'a part for the service of God and a distressed brother...', the Common Gavel 'divest our minds of the vices and superfluities of life', Fortitude 'that noble and steady purpose of mind...', Relief 'to soothe the unhappy, to sympathise with their misfortunes...'). In many constitutions those lectures ARE the printed ritual, even where US monitors publish them; and this text is broadcast daily to every member across all jurisdictions.
  - Fix: Vet each entry against the 'monitorial/public education' bar and reword the lecture-cadence ones into descriptive third-person prose ABOUT the symbol ('The gavel is taken as an emblem of...') rather than first-person-plural lecture paraphrase. Keep the history/biography entries as-is.
- **'N brethren' per trade category on the browse page** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/app/trades/browse/page.tsx:39-41 (fed by lib/trades.ts:92-103)`
  - Problem: Each category card renders `${n} brethren` — a displayed number users can compare across a grid. lib/trades.ts:91 defends it as 'availability, never a leaderboard of brethren', and it counts categories not persons, so it is the mildest drift found — but the empty/non-empty signal is all a searching brother needs, and the app is otherwise numberless.
  - Fix: Render 'Brethren available' / 'No brethren yet' (or 'A brother available' for the singular) without the numeral, keeping tradeCategoryCounts server-side purely as the boolean source.
- **noindex is meta-only: no robots.txt, no X-Robots-Tag on images, and an orphaned photo in static public/** — ⚪ LOW · effort S · `/Users/micryan/masonic-network/app/layout.tsx:16; /Users/micryan/masonic-network/next.config.ts; /Users/micryan/masonic-network/public/u/1.jpg`
  - Problem: robots:{index:false,follow:false} covers HTML pages, but the repo has no app/robots.ts or public/robots.txt and no X-Robots-Tag header — so /media/* images (which, per the finding above, are currently fetchable unauthenticated) and static files carry no noindex signal to image crawlers, which never see a meta tag. public/u/1.jpg (147 KB, plausibly a member photo from the pre-/media era — the stale comment at api/profile/photo/route.ts:22 still references /public/u) is served statically to the logged-out world.
  - Fix: Add app/robots.ts with a disallow-all rule, set `X-Robots-Tag: noindex, nofollow` on /media responses (one header in the route), and delete public/u/1.jpg after confirming no profiles.photo_url row still references /u/ (one read-only SQL check on the NucBox).

Explain My Build (EMB / Docent)

explainmybuild.com — code→two-guides engine (end-user guide + founder runbook) for non-technical AI-founders.

Overall assessment

The engine is genuinely strong — layered grounding (exact-match citation validation, self-critic, route validator, enforced refusal), a well-guarded zero-key generate path, and a correctly wedge-scoped MCP — but the business around it is broken in three ways: the web funnel (the surface the ICP actually uses) gives the monetised end-user guide away in full while a paying customer gets nothing redeemable there; the marketing copy leads with the free commodity side and makes local-first/citation claims that are factually wrong for the default path — lethal for a product whose differentiator is honesty; and production is a hand-rsynced, unmonitored, unbacked-up single box whose free tier can silently hammer the Claude subscription DH/AH production depends on. Verified against source: every critical claim above was spot-checked in the repo (generate.mjs:131, license.ts:140, README.md:186, index.ts:604, llm.ts, site/index.html:123).

Ranked improvements

1. Gate the end-user guide on the web funnel — the code violates the monetisation constraint

🔴 **CRITICAL** · effort M · *monetisation / wedge*

What to do: In web/generate.mjs, truncate the users' guide server-side before it enters job.result (generous preview: first task section) with an 'Unlock the full guide' CTA; keep runbook + fix-list complete, free, and downloadable. Mirror the desktop's preview-and-unlock pattern in web/app.ts (currently renders + offers full .md download at app.ts:274/286/319-324). Do NOT gate the runbook.

Why it matters: The END-USER guide is the one monetised artifact (runbook is deliberately free), the pricing page sells it at $19/mo, the MCP and desktop correctly gate it — but the primary CTA surface hands the complete artifact to anonymous users at generate.mjs:131 with zero license code in web/. This is the single largest gap between strategy and shipped code; three investigators independently flagged it.

2. Close the Pro purchase loop in the web app — paying $19/mo currently yields nothing usable there

🔴 **CRITICAL** · effort L · *monetisation / ICP*

What to do: Add license redemption to the SPA (accept the token from accounts /return via fragment redirect or paste box, store in localStorage, verify server-side on /api/generate to unlock the full guide), and offer hosted sharing (link + access code) as part of Pro by running the existing publish/server.ts logic on the box instead of requiring self-hosting. Files: web/app.ts, web/generate.mjs, docent-accounts/server.mjs:379-407, publish/server.ts.

Why it matters: Today redemption is 'docent login ' in a terminal and every advertised Pro feature (share, widget, auto-current) lives in a self-hosted Node server — unusable by a non-technical Lovable founder. Rank 1 without rank 2 creates a paywall nobody can pay through; ship them together.

3. Kill the documented and unguarded paywall bypasses

🟠 **HIGH** · effort S · *monetisation / security*

What to do: (a) README.md:186 literally instructs MCP users to set DOCENT_PRO=1 to 'unlock the users' guide' — replace with DOCENT_LICENSE and a pointer to explainmybuild.com. (b) Make license.ts:140/149 honour DOCENT_PRO only with a dev build flag (and never when NODE_ENV=production); electron/main.cjs spreads full process.env so a stray var unlocks shipped builds. (c) SSH the box and verify DOCENT_LICENSE_PUBKEY is NOT the repo's baked-in default (license.ts:37-39) and docent-accounts' LICENSE_SIGNING_KEY matches; consider failing closed when the pubkey equals the known default.

Why it matters: A publicly documented 100% discount on the only monetised artifact, plus an env-var Pro unlock with no prod guard, plus an unverifiable default signing key = three ways revenue leaks even after ranks 1-2 ship.

4. Protect the shared Claude subscription: circuit breaker, health endpoint, instant kill switch

🟠 **HIGH** · effort M · *cost-safety / reliability*

What to do: In web/generate.mjs + site/server.mjs: (a) breaker — after 3 consecutive job errors, pause intake/draining for a cool-off and alert; (b) /healthz returning {ok, queued, running, dayCount, lastJob, engineVersion} wired to the existing com.dh.watchdog/Telegram infra; (c) make ENABLED() also check a sentinel file (touch GENERATE_OFF = instant off, no restart — today the kill switch needs SSH + env edit + a restart that destroys in-flight jobs); (d) skip queued jobs not polled for >60s (client gives up at 6 min, worst-case queue wait ~32 min — abandoned jobs still burn full engine+claude runs); (e) consider a separate budget/lower GLOBAL_DAY until launch.

Why it matters: The free tier runs on the SAME subscription DH/AH production depends on. Today a rate-limit event — a full-stop red-line condition — instead triggers up to dozens more claude invocations, and every failure path is fail-soft with nothing reading generate.jsonl, so the money funnel can be 100% dead invisibly. Guardrails you can't observe aren't guardrails.

5. Script the deploy, reconcile repo⍰box drift, and version-stamp the engine

🟠 **HIGH** · effort M · *reliability / ops*

What to do: Commit scripts/deploy-web.sh (web:build → rsync server.mjs+gitproxy.mjs+generate.mjs+site assets+web/public → restart → automated smoke: /, /app 301, /app/ 200, /gitproxy 403, bad POST 400, /healthz). One-time reconcile: pull ~/explainmybuild down, diff, commit deltas — repo site/index.html still has five pre-fix href="/app" CTAs (lines 114/126/205/225/259) that a naive rsync would revert onto the box. Write .engine-version into ~/explainmybuild-engine on deploy and log it per job. Copy the live explainmybuild.service unit into the repo and add uncaughtException/unhandledRejection handlers to server.mjs.

Why it matters: The entire deploy is a memorised 4-step rsync of three import-coupled files (partial rsync = crash-loop = whole site down, since one process serves marketing + /app + gitproxy + generate). Drift between repo and box is already proven in both directions; engine output can silently come from weeks-old code with no way to correlate incidents to a version.

6. Run the audit-finding verification pass in ALL surfaces, not just the CLI

🟠 **HIGH** · effort M · *trust / correctness*

What to do: Move verifyAuditFindings (currently only in src/index.ts:602-620) into runPipeline in src/pipeline.ts after runEvidence, gated by a verifyFindings option defaulting true; delete the CLI-local copy; emit a pipeline event so the desktop can narrate it.

Why it matters: The 'verified-only' pass is what the code's own comments call the difference between trustworthy findings and 'a raw pattern dump' — yet desktop and MCP (the flagship ICP surfaces) render runbooks and danger zones from UNVERIFIED scanner noise, and the CLI/GUI produce materially different guides from the same repo. This is the trust differentiator failing exactly where the buyer is.

7. Harden the LLM boundary: timeout, retry escalation, shape normalisation, provider-cache fix

🟠 **HIGH** · effort S · *correctness / robustness*

What to do: (a) src/llm.ts runClaudeCli: add DOCENT_LLM_TIMEOUT_MS (~180s) with process kill — the default provider currently has NO timeout and a hung `claude` wedges the desktop UI, blocks ALL subsequent MCP calls, and holds the server queue slot. (b) Don't memoise detectProvider's 'none' result (llm.ts:32-46) — installing claude mid-session currently never takes effect. (c) llmJson: escalate maxTokens on truncation errors (currently retries the identical request 3x, so 'model too big for 8000 tokens' is unrecoverable by design) and short-circuit on 4xx auth. (d) synthesize.ts assertModelShape: normalise extend/operate/personas/glossary to [] — a partial-but-valid JSON reply currently crashes with 'cannot read map of undefined' AFTER the paid synthesis call.

Why it matters: Four small fixes at the same boundary that together eliminate the worst first-run failure modes for the least technical users, on the default zero-key path the whole strategy depends on.

8. Make the site tell the truth and lead with the wedge

🟠 **HIGH** · effort S · *positioning / trust*

What to do: site/index.html + web/index.html: (a) flip the H1 from 'Finally understand the app you built' (the FREE commodity axis) to the wedge — 'Turn the app you built into a guide your users can follow' — with runbook/fix-list as free support acts; reorder the What-you-get cards; add the 'no screen-recording' contrast above the fold (currently appears NOWHERE buyer-facing). (b) Fix the local-first claims — meta description, hero badges, /app header all say 'Runs on your machine' while the DEFAULT path clones the repo to EMB's server and uses EMB's subscription; scope it: 'Public repo? We read it and keep nothing. Private or sensitive? Run it fully in your browser.' (c) Rewrite How-it-works to the real /app flow — the current steps describe Approve/Share/Export, none of which exist where the CTA lands; drop 'auto-current' until a hosted re-check exists. (d) Soften 'a specific… line of code' to 'a specific file or route' until provenance ships (EvidenceRef has no file/line today).

Why it matters: Three constraint violations in one file: local-first is leading instead of qualifying, the free artifact is leading instead of the monetised wedge, and the trust-is-our-differentiator product makes checkably false claims. One skeptical Discord reply ('their default literally uploads your repo to their box') undoes the whole trust message. All copy — S effort, immediate.

9. Close the paid artifact's biggest hallucination surface: feed real UI copy into the guide, strip unverified paths, correct jargon

🟠 **HIGH** · effort M · *trust / wedge quality*

What to do: (a) Pass the already-extracted copyStrings (static.ts:522-553 — currently consumed ONLY by the proofread improvement) plus walker titles into render/public.ts buildPrompt with 'name buttons/labels/fields ONLY from REAL COPY, otherwise describe generically'. (b) Add a deterministic final pass that strips/inline-flags remaining unverified paths — validate.ts's own header promises 'the paid output can't reference a page that doesn't exist' but today enforcement stops at a banner after one corrective pass. (c) Mirror the route-correction pattern for jargon (currently detect-only with a narrow ~18-term regex) and expand the list.

Why it matters: Step-by-step instructions naming buttons/fields the model has never seen are exactly what an end-user follows and where wrongness is most visible — in the one artifact people pay for. The evidence already exists in the model; it just isn't wired to the prompt.

10. Surface the grounding score and visible refusal on the web funnel

🟠 **HIGH** · effort M · *trust / funnel*

What to do: Have generate.mjs read the app model before cleanup and attach groundingReport(model) to job.result so app.ts:279's 'confidence N/100' actually renders on the main path (today: never — no grounding field). Fix the one-liner appName bug (generate.mjs:122 reads .appName; the field is .name). Add a small 'ask about your app' box on the results page backed by the pure-local, LLM-free ask() so the refusal behaviour is demonstrable. Also fix ask() returning confident:true with zero citations for glossary/operate hits (ask.ts:62-66,119-128).

Why it matters: The hero mock sells an 82/100 confidence meter and 'it refuses rather than guesses' — but the flagship zero-key flow shows neither, and refusal is only visible in CLI/MCP (developer surfaces). The differentiator is invisible exactly where the non-technical buyer is; ask() costs nothing on the subscription.

11. Build the viral loop: linked attribution + shareable permalinks

🟠 **HIGH** · effort M · *growth / distribution*

What to do: (a) Add a linked 'Made with Explain My Build → explainmybuild.com' footer (with UTM) to the published reader (publish/server.ts:135 — currently no product mention), the /embed reader, the widget panel (widget.js — currently nothing), and the HTML export (desktop/server.ts:679 — currently plain text, not a link); optional data-attribution=off. (b) For public-repo zero-key runs, offer an opt-in share permalink (explainmybuild.com/g//) with OG tags, preview banner, and share-to-X — today results are deleted after 30 min and the only artifact is a .md download.

Why it matters: Every Pro customer pays to put an EMB guide in front of THEIR users, and the ICP shares artifacts on X/Discord — yet neither surface generates a single referral link. This is the classic 'Powered by' loop, entirely absent, and it is the highest-leverage acquisition mechanic available at S/M effort. Public-repo previews are marketing; the paid tier (your own app, private, hosted, widget) is untouched.

12. Unblock the MCP/CLI/Action channel — right now its external reach is zero

🟠 **HIGH** · effort M · *distribution*

What to do: Publish the already-bundled, minified JS as an npm package (or public installer repo / Release tarball) under a proprietary license — the Pro gate is enforced at runtime via isPro(), so the paywall survives; mirror the living-guide Action into a public repo; list the MCP server in the registries agents browse. Preserve the narrow two-guides tool scoping when doing so (do NOT add generic 'index my codebase' tools).

Why it matters: The MCP is beautifully scoped to the wedge, but `npx github:Micipedia/docent` requires access to a PRIVATE repo and a private-repo Action can't be `uses:`-referenced — the channel reaches exactly zero prospects. Note: this deliberately reverses the README's stated 'no npm publish' decision (flagged in constraint_flags); a closed channel is not a channel.

13. Give the real ICP a path in: private repos and platform exports

🟡 **MEDIUM** · effort M · *ICP / activation*

What to do: (a) Add a browser-local zip drop to the SPA ('export your project from Lovable / GitHub → Download ZIP, drop it here') — unzips in-page, works in Safari/mobile, code never leaves the browser. (b) Add a 'Where's my code?' helper under the URL field with 30-second per-platform instructions. (c) When the clone 404s, route users to the zip/folder path instead of the dead-end 'private repos aren't supported yet'. Longer term: fine-grained read-only GitHub OAuth through the existing pinned proxy.

Why it matters: Lovable/Bolt/Replit founders mostly have PRIVATE repos or no repo at all; the one-button path hard-rejects them (generate.mjs:49) and the fallback demands an API key + Chromium — the two highest-friction asks for this ICP. The most-qualified visitors currently bounce at step one.

14. Box durability: back up the license DB and leads, pin the runtime paths

🟠 **HIGH** · effort S · *reliability / data*

What to do: (a) Nightly cron: sqlite3 'VACUUM INTO' on accounts.db + copy waitlist.jsonl/generate.jsonl into the existing DH offsite (S3/GitHub) pipeline; verify one restore — today a single-disk failure destroys the license→customer mapping and the entire lead list. (b) Version-stable symlinks for node + claude (the systemd unit hardcodes /.nvm/versions/node/v24.14.1/bin/node and generate.mjs assumes claude lives next to node — a routine nvm bump is a latent full outage plus a silent generate outage).

Why it matters: Non-PHI data by design so offsite copies are fine; ~20 lines of shell protecting the funnel's two irreplaceable data assets, plus removing the most likely 'why is everything down weeks after I upgraded node' incident.

15. Measure the hallucination rate — the headline claim currently has no number

🟡 **MEDIUM** · effort M · *trust / measurement*

What to do: Extend src/eval.ts beyond recall: (a) share of synthesized features carrying verified evidence, (b) per-fixture forbiddenClaims scored as hallucinations, (c) guide-level metrics on the rendered public guide (unverified-route count, jargon count). Add a periodic golden-run script analysing the 4 fixture repos on the box's subscription (~$0 marginal) recording scores over time. Also fix the de-id/validator asymmetry (synthesize.ts:141-166 checks refs against RAW evidence while the LLM saw the SCRUBBED digest — dates in commit messages systematically drop valid citations and understate grounding scores).

Why it matters: 'It never makes things up' is asserted, not measured; the monetised artifact is the least-measured output, and regressions in the critic/validator/prompt currently ship undetected. The de-id asymmetry means the honesty layer punishes the model for obeying the privacy layer.

Quick wins (small effort, high value)

- Add a timeout to runClaudeCli in src/llm.ts (DOCENT_LLM_TIMEOUT_MS ~180s + kill) — one hung claude process currently wedges desktop, MCP, and the server queue forever
- Replace README.md:186's DOCENT_PRO=1 MCP example with DOCENT_LICENSE — stop publicly documenting the paywall bypass
- One-liner: generate.mjs:122 read .name not .appName so zero-key results show the app's real name instead of the repo slug
- Copy-only fixes in site/index.html + web/index.html: scope the 'runs on your machine' claims to the paths where they're true, flip the H1 to the users'-guide wedge, add the 'no screen-recording' contrast, align How-it-works to the real /app flow
- Make every 'Made with Explain My Build' mention a hyperlink with UTM (publish reader footer, /embed, widget.js, HTML export) — the seed of the viral loop is S-effort
- Nightly backup cron for accounts.db (VACUUM INTO) + waitlist.jsonl/generate.jsonl into the existing DH offsite pipeline
- Add /healthz to site/server.mjs (stats already in memory in generate.mjs) and point the existing watchdog at it
- Sentinel-file kill switch for generate (touch GENERATE_OFF) so an incident response doesn't require SSH + env edit + a restart that takes the whole site down
- Add PEM private-key-block, JWT, and high-entropy patterns to src/deid.ts — audit-verify.ts sends raw source lines to non-BAA BYO-key endpoints
- Fix the de-id/validator asymmetry in synthesize.ts: build the citation-validation sets from the same scrubbed strings the LLM saw
- Compact JSON.stringify(digest) in the synthesis prompt (synthesize.ts:225) — 25-40% token cut on the engine's largest prompt for free
- SEO basics: robots.txt, sitemap.xml, rel=canonical, FAQPage JSON-LD (the FAQ section maps 1:1), OG tags on /demo (currently zero — the best shareable page renders a blank card on X/Discord)
- SPA header: replace the 📖 emoji with Mic's existing emb-mark.png (asset reuse ONLY — no redraw per the logo red-line) and show the $12 founding price the backend already implements instead of flat $19
- Don't cache detectProvider's 'none' result (llm.ts:32-46) so installing claude mid-session actually works
- Per-IP rate limit: trust cf-connecting-ip, else fall back to socket remoteAddress — never client-controlled X-Forwarded-For (generate.mjs:174, site/server.mjs:50)

Constraint checks & code-level violations

Places where the code itself breaks a load-bearing rule, or suggestions that were rejected for violating one.

- CODE VIOLATES 'MONETISE the END-USER guide': the web funnel (web/generate.mjs:131 + web/app.ts full render/.md download) gives the paid artifact away complete to anonymous users while MCP and desktop gate it — and README.md:186 publicly documents the DOCENT_PRO=1 bypass. Surfaces disagree with each other and with the pricing page.
- CODE VIOLATES the positioning constraint: site H1 ('Finally understand the app you built'), title/OG, and card ordering lead with the FREE runbook/'explain my code' axis; the verified-empty 'users' guide from code, no screen-recording' claim appears nowhere buyer-facing (only in README/mcp.ts).
- COPY VIOLATES 'local-first = QUALIFIER, not the lead' AND the trust table-stakes: meta description, hero badges, and the /app header lead with 'Runs on your machine / nothing uploaded' — factually wrong for the default server-side path, which clones the repo to EMB's box and uses EMB's subscription.
- CODE UNDER-IMPLEMENTS the shared-subscription cost-safety constraint AND the pause-on-rate-limit red-line: during a subscription rate-limit event the generate queue keeps launching fresh claude runs (no circuit breaker, no alert, kill switch requires SSH+restart).
- MARKETING OVERSTATES the citation capability: 'a specific file, route, or line of code' — EvidenceRef carries no file paths or line numbers; either implement provenance or soften the copy (trust message must not be checkably false).
- ACCEPTED WITH EXPLICIT FLAG: publishing the bundled JS to npm (rank 12) reverses the README's stated 'no npm publish, repo access is the gate' decision. It does NOT violate any load-bearing constraint (code stays proprietary, the Pro gate is runtime-enforced, MCP scoping stays wedge-narrow), but it is a deliberate strategy reversal Mic should sign off on.
- SCREENED AND KEPT COMPLIANT: the license device-binding suggestion was accepted only in its no-phone-home form (soft device-id claim at most; hard phone-home would break the local-first/offline-verify promise). The SPA logo fix is compliant only as verbatim reuse of emb-mark.png — no redraw or simplification per the logo red-line. No investigator recommendation had to be rejected outright; none proposed generic 'ask my codebase' repositioning or enterprise-dev targeting.

Also worth doing (lower priority, nothing dropped)

- Implement true file+line citation provenance (extend EvidenceRef with optional file/line captured at extraction — static.ts already knows the source of each route/model/env var; L effort, upgrades the trust message from softened copy to genuinely differentiating)

- Consolidate the triple pipeline orchestration: make src/index.ts's analyze a thin wrapper over runPipeline (the PipelineEvent stream exists for exactly this); remove the unreachable 'phi-gate' state from pipeline.ts/mcp.ts; fixes the drift class at the root (atomic writes and cache pruning currently exist only in pipeline.ts)
- Two-tier the PHI heuristic (strong terms gate alone; 'provider'/'claims'/'eligibility' need >=2 weak hits) and recompute per-run instead of ratcheting forever via run.json — mainstream SaaS apps currently get flagged as health-data apps permanently
- Discovery gaps: move the COPY_EXT check above the CODE_EXT gate in static.ts (plain-HTML copy extraction is currently dead code), add Nuxt .vue / Astro .astro page routes, Go/Spring route regexes, os.environ/os.Getenv, and widen JS_ROUTE_RE to named routers — Bolt emits Astro, Replit emits Python/Go
- Critic robustness: per-section evidence budgets instead of the blind 12,000-char slice (audit hotspots serialize last and fall off first), and log when critique drop-names fail to exact-match any claim
- Extend the self-critic to 'extend' howTo entries and give 'operate' steps evidence refs + inclusion in the grounding score — the highest-blast-radius runbook sections currently bypass every safety layer
- claimConfidence: let >=3 distinct refs within one kind lift low→medium (never high — preserve the runtime/audit ceiling); static-only funnel runs currently look artificially weak
- Validate dotted paths (/index.html-style) instead of skipping them, and drop the whole onboarding step when its route fails verification
- Register improvements.md in the run.json artifacts array (built then dropped at pipeline.ts:299-303), fix the dead hasFile ternary in improvements.ts:43, drop the unused Improvement import, enable noUnusedLocals
- Add orchestration-layer tests with a stubbed provider (guide-error fail-soft, cache prune, run.json merge, containsPhi propagation) — the layer with the most drift has the least coverage
- Anchor the gitproxy path regex and bind method to shape (GET info/refs, POST git-upload-pack); host pin remains the primary gate
- serve.mjs containment: compare against DIR + path.sep like site/server.mjs already does
- Pass the generate child a minimal allowlisted env instead of {...process.env} (the box env holds DH/AH secrets) and add a standing rule that the zero-key path never executes repo code
- Referrer-Policy: no-referrer on the accounts /return page so Stripe session ids don't leak via referer; revisit soft device binding only if revenue leakage shows up
- Reinstate email capture deliberately (footer founding-100 lock-in feeding the still-live /api/waitlist) or delete the dead main.js handler + orphan CSS + endpoint — and use capture as the fail-soft when the free-generation cap or queue-full bounces a launch spike (overnight drain within the same daily budget)

- Queue UX: return queue position in /api/generate/status, start the 6-min client clock only when the job flips to running, and don't charge per-IP/day quota until a job actually starts
- Add an X handle + PH badge + (post-permalink) an opt-in 'recently generated guides' wall — no invented counts or fake logos per the no-overstatement rule
- Compose-and-boot integration test for the assembled prod server in CI (site/server.mjs's imports only resolve in the deploy tree — CI never executes the production server today)
- Cheap SPOF mitigation: Cloudflare Always Online + publish the static site + web/public to Cloudflare Pages as a documented failover origin (the own-key browser path needs no server at all); do NOT buy a second box yet
- 4-6 SEO intent pages per builder platform ('Create a users' guide for your Lovable/Bolt/v0/Replit app — from its code') feeding /app
- Decide founding-price surface consistency everywhere once rank 1-2 ship (site says $12 founding, SPA upsell says $19)

Appendix — full finding set by dimension

ENGINE ARCHITECTURE, CODE QUALITY & ROBUSTNESS

The engine's module-level design is genuinely strong: a clean evidence-source contract (src/evidence/index.ts) that fail-softs per source, deterministic renderers for the runbook/improvements/storyboard, layered hallucination defences (exact-match citation validation, self-critic, route validator, refusal-based ask), and an unusually broad extraction layer (Next/SvelteKit/Remix/Express/FastAPI/Flask/Laravel/Rails/Django/tRPC/SQL/OpenAPI/GraphQL) with 45 unit-test files plus an eval harness. The weaknesses concentrate at the ORCHESTRATION layer and the LLM boundary: the end-to-end flow is implemented three separate times (CLI, desktop pipeline, browser engine) and has already drifted — most consequentially, the audit-finding verification pass ("verified-only", the trust differentiator) runs ONLY in the CLI path, so desktop and MCP users get raw scanner noise; the default zero-key `claude -p` path has no timeout and can hang the desktop/MCP forever; model-shape validation covers only 3 of 7 required arrays so a partial-but-valid JSON response crashes after the paid synthesis; and the PHI keyword heuristic over-fires on ordinary SaaS terms ("provider", "claims") and then ratchets permanently via run.json. Discovery generalises far better than a typical JS-only tool but still returns near-nothing for plain-HTML exports, Nuxt/Astro file routes, and Go/Java backends.

- **Audit-finding verification pass ("verified-only") runs ONLY in the CLI — desktop + MCP surface raw scanner findings** — 🟠 HIGH · effort M · `/Users/micryan/docent/src/`

`index.ts:602-620 vs /Users/micryan/docent/src/pipeline.ts (no audit-verify import)`

  - Problem: The desktop app and the MCP tools (generate_founder_runbook / list_improvements — the flagship surfaces for the ICP) render runbooks, danger zones, and AI-prompts from UNVERIFIED scanner output, exactly the false-positive noise the engine's own comments say destroys trust. The CLI and GUI produce materially different guides from the same repo.
  - Fix: Move the verifyAuditFindings step into runPipeline (after runEvidence, before synthesize), gated by a `verifyFindings?: boolean` option defaulting true, and delete the CLI-local copy. Emit a pipeline event so the desktop UI can narrate it.

- **Default zero-key `claude -p` path has no timeout — a hung CLI wedges the desktop UI and MCP forever** — 🟠 HIGH · effort S · `/Users/micryan/docent/src/llm.ts:61-80`
  - Problem: A stalled `claude` process (network hang, auth prompt, subscription rate-limit stall) hangs synthesize/renderPublic indefinitely: the desktop progress bar freezes at 'synthesize-start', the MCP tool call never returns (its serialised chain in mcp.ts:213 then blocks ALL subsequent tool calls), and the server-side generate on the shared box holds the queue slot.
  - Fix: Add a configurable timeout (e.g. DOCENT_LLM_TIMEOUT_MS, default ~180s) to runClaudeCli: setTimeout → child.kill("SIGKILL") → reject with the same actionable message style fetchT uses.

- **Pipeline orchestration is implemented three times (CLI, desktop, browser) and has already drifted** — 🟠 HIGH · effort L · `/Users/micryan/docent/src/index.ts:498-752, /Users/micryan/docent/src/pipeline.ts:147-397, /Users/micryan/docent/web/engine.ts`
  - Problem: Every behaviour fix must be made 2-3 times and demonstrably isn't (verify pass, atomicity, cache pruning). The header comment in pipeline.ts claims the CLI reuses it, which is only true of leaf functions — a maintenance trap for future changes to PHI gating or artifact contracts.
  - Fix: Make index.ts's analyze command a thin wrapper over runPipeline with a console progress-printer (the PipelineEvent stream already exists for exactly this); keep web/engine.ts as the one justified fork (browser shims) but add a comment inventorying its deliberate differences. Remove the unreachable "phi-gate" member from PipelineEvent/PipelineResult and the mcp.ts:92 handler.

- **assertModelShape validates only 3 of 7 required AppModel arrays — a partial-but-valid JSON reply crashes after the paid synthesis** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/src/synthesize.ts:188-192, with crash sites at synthesize.ts:182, render/private.ts:412, pipeline.ts:271`
  - Problem: An LLM response that omits `extend` / `operate` / `personas` / `glossary` (common under token pressure or with weaker OPENAI_MODEL/GEMINI defaults) parses fine, passes assertModelShape, then throws

`Cannot read properties of undefined (reading 'map')` — failing the whole run with an inscrutable error AFTER the synthesis call was spent, instead of the intended "fail loud with an actionable message".

  - Fix: In assertModelShape (or right after parse), normalise the four secondary arrays to [] when missing and keep the hard failure only for name/features/riskHotspots. One test with a minimal model exercising renderPrivate would lock it in.

- **PHI text heuristic over-fires on ordinary SaaS vocabulary and then ratchets permanently via run.json** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/src/discover.ts:37-38, 88-104; /Users/micryan/docent/src/pipeline.ts:195, 263, 376`
  - Problem: A large slice of non-health apps in the ICP (anything using auth providers or the word 'claims') gets flagged as a health-data app FOREVER: storyboards lose all screenshots (storyboard.ts:68), guides carry "this app looks like it handles health data" warnings, and the MCP appends a health-data note (mcp.ts:117) — confusing, trust-eroding messaging for a mainstream Lovable/Bolt founder. It fails in the safe direction, but with no recovery.
  - Fix: Two-tier the regex: strong single-hit terms (hipaa, phi, ephi, ehr, mrn, patient, medical record, telehealth) gate alone; weak terms (provider, claims, eligibility, encounter, denial) require ≥2 distinct weak hits or one strong hit. And recompute the flag each run from current evidence instead of ratcheting from priorRunPhi (or add an explicit user override that clears run.json's containsPhi).

- **Discovery gaps: plain-HTML copy extraction is unreachable dead code; Nuxt/Astro/Go/Java repos yield near-zero surface** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/src/evidence/static.ts:12-15, 90, 127, 41, 183, 548-552`
  - Problem: For these stacks the skeleton shows 0 routes/0 env vars, assertEnoughEvidence passes only via git history, and the end-user guide degenerates to an empty VALID ROUTES list — the paid artifact is thin exactly where a founder outside the Next.js mainstream tries the tool. The dead COPY_EXT entries also silently kill the 'clarity/proofread' improvement for HTML-templated apps.
  - Fix: Short term: move the COPY_EXT read above the CODE_EXT gate (scan copy-only extensions for copyStrings without counting them as code); add `pages/**.vue` and `src/pages/**.astro` to deriveRoutes. Medium: add Go (net/http/gin/echo `\.(GET|Get|Handle(Func)?)\("/…"`) and Spring (`@(Get|Post|Request)Mapping`) route regexes plus os.environ/os.Getenv to ENV_RE, and widen JS_ROUTE_RE to any identifier assigned from `Router()` .

- **llmJson retries the identical request on every failure — truncation can never succeed and auth errors burn 3 attempts** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/src/llm.ts:213–231, /Users/micryan/docent/src/synthesize.ts:229`
  - Problem: The most likely large-repo failure mode (model too big for the budget) is unrecoverable by design despite the error message promising the fix, and deterministic failures triple latency/cost before surfacing.
  - Fix: In llmJson, detect the truncation error class and retry with maxTokens escalated (e.g. ×1.5, capped); short-circuit (no retry) on 4xx auth/permission errors. Both are string-matchable on the existing error messages it throws itself.
- **Self-critic sees a blind 12,000-char slice of the digest — the audit section (last key) is truncated first, risking drops of audit-backed claims** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/src/critic.ts:54, digest key order at /Users/micryan/docent/src/synthesize.ts:42–104`
  - Problem: The fact-checker can be shown evidence that omits the audit backing a risk hotspot and drop a genuinely audit-grounded claim (the conservative system prompt softens but does not eliminate this), quietly undermining the 'include EVERY critical and high audit finding' guarantee.
  - Fix: Build the critic's evidence view with per-section budgets (e.g. compact-stringify each of the four sections and trim the largest sections first, always preserving audit hotspot areas/severities), instead of one character slice of the whole object.
- **Orchestration layer effectively untested: index.ts (793 lines) has zero direct tests; pipeline.test.ts covers only slugId + detectOnly** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/test/pipeline.test.ts, /Users/micryan/docent/src/index.ts`
  - Problem: The layer with the most duplication and the most recent drift (see the verify-pass finding) is exactly the layer with the least coverage — regressions in artifact contracts, PHI flag propagation, or fail-soft behaviour would ship silently.
  - Fix: Add pipeline tests with a stubbed provider (DOCENT_LLM_PROVIDER + a fake openai-compatible endpoint or an injected llmJson) covering: guide-error keeps the run alive, cache prune leaves one file, run.json merge preserves foreign artifacts, and containsPhi propagation. Once the CLI wraps runPipeline (finding above), a single parseArgs unit test covers the rest cheaply.
- **Dead code cluster in the improvements path: run.json artifact never registered, dead ternary, unused type import** — ⚪ LOW · effort S · `/Users/micryan/docent/src/pipeline.ts:21, 299–303, 366–371; /Users/micryan/docent/src/improvements.ts:43`
  - Problem: The unregistered artifact is a real (if small) contract bug for the run-manifest ecosystem; the dead ternary means the 'a file makes it unambiguously code' intent was never implemented; the unused import is lint noise masking the other two.

- Fix: Spread `...improvementArtifacts` into the artifacts array (with rel() applied like the others); either delete the ternary or implement the intended hasFile bias; drop the unused import. Consider enabling noUnusedLocals in tsconfig to catch this class.

- **detectProvider memoises "none" forever — a desktop/MCP session started before installing `claude` can never recover without restart** — ⚪ LOW · effort S · `/Users/micryan/docent/src/llm.ts:32-46`
  - Problem: The exact onboarding path the error message prescribes silently doesn't work in the GUI/MCP until the app is restarted — a confusing first-run failure for the least technical users.
  - Fix: Don't cache the "none" result (`if (cachedProvider && cachedProvider !== "none") return cachedProvider;`), or have the desktop retry path call _resetProviderCache() before re-running.

- **Synthesis prompt embeds the digest pretty-printed — ~25-40% avoidable token inflation on the engine's largest prompt** — ⚪ LOW · effort S · `/Users/micryan/docent/src/synthesize.ts:225 (and 17)`
  - Problem: Meaningful cost/latency overhead on the API fallback providers and unnecessary context pressure on the subscription path for large repos, for zero informational gain to the model.
  - Fix: Use compact `JSON.stringify(digest)` (or one-line-per-top-level-key) in the prompt; keep the pretty version only for on-disk artifacts. evidenceDigestHash is independent (hashes its own stringify) so caching is unaffected.

Serving-layer security

The serving layer is unusually well-hardened for an indie tool, and several of the hardest problems are solved correctly: the git proxy's SSRF surface is tightly contained (https-only + host allowlist {github.com, codeload.github.com} + redirect:"manual" + the URL parser correctly rejecting userinfo/embedded-host tricks, with no DNS-rebinding surface since only GitHub's own DNS is trusted); the zero-key generate endpoint closes the rate-limit TOCTOU with a genuine synchronous critical section (web/generate.mjs:183-192), kills the whole process group on timeout (line 87), fails CLOSED on an unmeasurable repo size (line 108), and cleans temp dirs in finally; the engine only READS untrusted cloned code (execFile git with an args array + GIT_TERMINAL_PROMPT=0, no shell, no dynamic require/import of repo files, no npm install, no app execution); the Stripe webhook verifies signatures and is idempotent; and the Ed25519 license verify is cryptographically sound (verify over exact claim bytes, expiry re-checked every call, never cached). The real weaknesses are (1) the zero-key generate path gives away the MONETIZED end-user guide free with no entitlement gate, (2) the DOCENT_PRO=1 bypass is unconditional with no prod guard, (3) the free generate runs on the Claude subscription SHARED with DH/AH production with no quota isolation, and (4) de-id misses whole classes of secrets (PEM key blocks, JWTs). None of the recommendations contradict

the load-bearing wedge/positioning constraints; two findings note the code itself sitting in tension with the "monetise the end-user guide" constraint.

- **Zero-key server generate returns the MONETIZED end-user guide free, with no entitlement gate** — 🟡 MEDIUM · effort M · `web/generate.mjs:112-135 (worker), 171-193 (handleGenerate)`
  - Problem: The /api/generate path runs `analyze --persona user` and returns job.result.guides = [{persona:'user', markdown:guide}] AND the runbook + improvements to any anonymous caller. There is no import of src/license.ts and no isPro()/isLoggedIn() check anywhere in the file. The end-user guide is the one artifact the strategy says to MONETISE (the founder runbook is the free giveaway), yet the instant/zero-key path hands the full paid artifact to unauthenticated visitors. Whether intentional as a lead-gen demo, it currently removes all pricing power from the wedge product.
  - Fix: Decide explicitly: if the instant path is the acquisition funnel, ship a TRUNCATED/watermarked user guide (first section + 'unlock the full guide' CTA to docent-accounts) and reserve the complete end-user guide for a verified Pro license; keep the founder runbook + fix-list full and free (that matches the giveaway strategy). Do NOT gate the runbook.
- **DOCENT_PRO=1 is an unconditional Pro bypass with no production guard** — 🟡 MEDIUM · effort S · `src/license.ts:140 and :149`
  - Problem: isPro() and isLoggedIn() both short-circuit to true whenever the env var DOCENT_PRO==="1", with no NODE_ENV / build-flag gate. Any shipped desktop build (electron/main.cjs spreads full process.env into the server at electron/main.cjs:24) or server-side engine run launched from a shell that happens to carry DOCENT_PRO=1 unlocks all Pro enforcement for free. This is exactly the 'dev bypass leaking to prod' risk: a single stray env var in the operator's profile or a leaked build defeats the paywall.
  - Fix: Honor DOCENT_PRO ONLY when a build-time flag says so (e.g. `process.env.DOCENT_PRO === '1' && process.env.DOCENT_ALLOW_DEV_BYPASS === '1'`), and have the release/electron build set neither. At minimum, ignore DOCENT_PRO when NODE_ENV==='production'. Confirm the live box's service env does not carry DOCENT_PRO=1 from DH/AH tooling.
- **Free generate consumes the Claude subscription SHARED with DH/AH production, with no quota isolation or alerting** — 🟡 MEDIUM · effort M · `web/generate.mjs:17 (ENABLED default on), :27 (GLOBAL_DAY=60), :113-116 (spawns engine → claude CLI)`
  - Problem: EMB_GENERATE defaults ON (ENABLED returns true unless explicitly 'off'). Each of up to 60 daily runs spawns the engine, which fires multiple Sonnet calls through the box's `claude` subscription — the SAME subscription DH/AH production depends on (per canonical memory). There is no coordination, no separate quota, and no cost/rate alert to Telegram/watchdog; the only backstop is CONCURRENCY=1 + GLOBAL_DAY=60. A burst of public generations can push the shared subscription into rate-limit and degrade DH/AH's

production LLM calls. The endpoint 'fails soft' for EMB's own users but does nothing to protect the shared dependency.

  - Fix: Give EMB its own subscription/API budget or a hard daily token ceiling distinct from DH/AH; lower GLOBAL_DAY until traffic warrants; on a subscription 429 from the child, back off globally (pause intake for a cool-down) and fire a watchdog/Telegram alert so a runaway is visible. Consider defaulting EMB_GENERATE to OFF and flipping it on deliberately at launch.

- **Baked-in DEFAULT license public key — prod keypair rotation is unverifiable and forge-fatal if skipped** — 🟡 MEDIUM · effort S · `src/license.ts:37-39 (DEFAULT_PUBLIC_KEY_PEM), :44-50 (override)`
  - Problem: A hard-coded default Ed25519 public key ships in the source, with a comment that production 'must' regenerate the keypair and override via DOCENT_LICENSE_PUBKEY (or a build-time replace). If prod ever shipped WITHOUT overriding it, then whoever holds the matching private key can mint valid Pro tokens for every EMB install forever. The verifier itself is sound, so the entire security of Pro reduces to 'did prod replace this default key, and is the matching private key only ever in the docent-accounts env'. I could not verify the live box (serving tree is on the server, not this Mac).
  - Fix: Verify on the live box that DOCENT_LICENSE_PUBKEY (or the built bundle) carries a freshly generated key, NOT the repo default, and that docent-accounts' LICENSE_SIGNING_KEY is the matching private half. If the default is still live, rotate now and re-mint. Consider failing closed (no Pro) when the pubkey equals the known default, to make a missed rotation loud.

- **De-id misses PEM private-key blocks, JWTs, and generic high-entropy secrets that CAN egress via source-line evidence** — 🟡 MEDIUM · effort S · `src/deid.ts:12-25 (PATTERNS); egress at src/evidence/audit-verify.ts:48 (deidentify of raw source lines)`
  - Problem: The scrubber covers emails, SSN/phone/date, a few key prefixes (sk-, ghp_, AKIA, AIza), connection strings, and key=value assignments — but has NO pattern for `-----BEGIN … PRIVATE KEY-----` blocks, no JWT (`eyJ…`) pattern, and no generic high-entropy/bearer token. audit-verify.ts sends actual source LINES (not just route/model metadata) through deidentify() to the LLM. A repo that commits a service-account JSON, a .pem, or a hard-coded JWT would leak those secrets verbatim to a BYO-key user's remote OpenAI/Gemini endpoint (which is NON-BAA per llm.ts:55-59). The subscription-CLI path is BAA-covered so lower risk, but BYO-key remote providers are not.
  - Fix: Add patterns for `-----BEGIN [A-Z ]*PRIVATE KEY-----` … `-----END…`, JWTs (`eyJ[A-Za-z0-9_-]+\.[A-Za-z0-9_-]+\.[A-Za-z0-9_-]+`), and a conservative long-base64/hex high-entropy catch. Keep the deliberate no-name-scrubbing choice. Redaction is cheap insurance on the one path that egresses raw code.

- **Per-IP rate limit trusts client-controllable X-Forwarded-For as fallback** — ⚪ LOW · effort S · `web/generate.mjs:174; same pattern in site/server.mjs:50`
  - Problem: The IP key is `cf-connecting-ip || x-forwarded-for || 'unknown'`. cf-connecting-ip (set by Cloudflare) is trustworthy, but the x-forwarded-for fallback is fully attacker-controlled. If the origin is ever reachable without Cloudflare in front (a misconfig, a direct-IP leak, or a future non-CF fronting), an attacker rotates X-Forwarded-For to defeat PER_IP_HOUR/PER_IP_DAY entirely; only GLOBAL_DAY=60 then protects the shared subscription. Also, all requests missing both headers collapse to the single bucket 'unknown', so one shared bucket throttles many legit users behind a proxy that strips headers.
  - Fix: Trust cf-connecting-ip only; if it is absent, treat the request as untrusted (fall back to the raw socket remoteAddress, as docent-accounts/server.mjs:73 already does — `req.socket.remoteAddress`), NOT to X-Forwarded-For. Optionally gate the endpoint on a shared CF header/secret so direct-origin hits are rejected.
- **Git-proxy path allowlist matches the shape anywhere in the path and does not bind method to shape** — ⚪ LOW · effort S · `web/gitproxy.mjs:16 (ALLOWED_PATH), :39, :67`
  - Problem: ALLOWED_PATH = //(?:info/refs|git-upload-pack)(?:$|?)/ matches `info/refs` or `git-upload-pack` appearing ANYWHERE in the path (e.g. /owner/repo/anything/info/refs), and the check doesn't tie GET→info/refs and POST→git-upload-pack. The host allowlist ({github.com, codeload.github.com}) is doing all the real SSRF containment, so exploitability is low, but the path gate is looser than its 'belt-and-suspenders against SSRF' comment implies, and it will forward arbitrary methods to those shapes.
  - Fix: Anchor and bind: allow exactly `GET …/info/refs?service=git-upload-pack` and `POST …/git-upload-pack`; reject other method/shape combinations. Keep the host pin as the primary gate.
- **Signed Pro license carries no device/nonce binding — fully portable, one token unlocks any machine until expiry** — ⚪ LOW · effort M · `docent-accounts/license.mjs:36-39 (mintPro claims), verified by src/license.ts:98-106`
  - Problem: mintPro emits claims {sub,plan,exp,cid,jti} with NO device binding, and the app verifies purely offline. The token is therefore a portable Pro credential: a paying user can paste it into `docent login` on any number of machines, or share it, and every install unlocks until exp. The /return page also serves the raw token back to anyone holding the (bearer) cs_ session id (server.mjs:398-401), and Stripe session ids can land in browser history/referer. This is the documented 'not tamper-proof, casual copying blocked' tradeoff, but 'casual copying' is actually near-frictionless.
  - Fix: Accept as an explicit indie tradeoff (fine for the ICP), OR add a soft device-id claim + a lightweight periodic re-check for the hosted/desktop app if revenue leakage shows up. Do NOT add a hard phone-home (would break the PHI/local-first promise). Set Referrer-Policy: no-referrer on the accounts /return page so session ids don't leak via referer.

- **Dev static server containment check lacks a path-separator boundary** — ⚪ LOW · effort S · `web/serve.mjs:20-24`
  - Problem: The dev SPA server strips leading `../` then checks `if (!file.startsWith(DIR)) 403`. startsWith without a trailing separator would treat a sibling like `<DIR>-evil` as inside DIR. In practice join(DIR, rel) always inserts a separator so the concrete traversal is not reachable here, and the server binds 127.0.0.1 (line 41, dev only) — but it's an inconsistent pattern next to the production site/server.mjs:59 which does it correctly (`target !== PUBLIC && !target.startsWith(PUBLIC + '/')`).
  - Fix: Mirror the production check: compare against `DIR + path.sep` (and allow the exact DIR). Cheap consistency fix that removes a foot-gun if this server is ever bound beyond localhost.
- **Engine child inherits the full box environment when analyzing untrusted repos** — ⚪ LOW · effort S · `web/generate.mjs:20 (CHILD_ENV = {...process.env}) → :106/:114/:116 spawns`
  - Problem: The zero-key generate spawns git + the engine over an UNTRUSTED cloned repo with the entire box environment (which on this shared box includes ANTHROPIC/Stripe/DH-AH secrets) exposed to the child. Today this is safe because the engine only READS the repo (I confirmed: execFile git with an args array, no shell, no dynamic require/import of repo files, no npm install, no app execution), so nothing in the repo can exfiltrate those vars. But it is a latent blast-radius amplifier: any future engine feature that executes repo-provided code (a plugin, a build step, running the app for a walker) would immediately leak every secret in the environment.
  - Fix: Pass the child a MINIMAL allowlisted env (PATH, HOME, and only the LLM creds the engine actually needs) instead of {...process.env}. Add a standing rule/test that the zero-key analyze path must never execute repo code. Low effort, removes a class of future critical bugs.

Output quality: grounding, citations & hallucination-resistance

The grounding architecture is genuinely layered and better than most LLM-doc tools: an exact-match citation validator (synthesize.ts:136-185) that prunes invented refs, an adversarial self-critic that drops unsupported features/risks (critic.ts), a deterministic consistency layer, a per-claim confidence model with an honest structural ceiling (nothing is 'high' without walker/audit), an enforced lexical-refusal ask() with tested refusal cases (ask.test.ts:36-40), and eval fixtures that REQUIRE a refusal QA (eval-fixtures.test.ts:34). But the trust story has real gaps exactly where the money is: the monetised END-USER guide renders with zero citations and its step-level instructions (button/field names) are generated with no UI-copy evidence — the extracted copyStrings never reach the guide prompt; unverified paths still ship in the paid guide body after one corrective pass; jargon is detect-only with a narrow regex; the marketing site promises line-of-code citations the EvidenceRef type cannot express; the de-id scrubber creates a systematic mismatch that silently

drops valid citations; the visible-refusal differentiator has NO surface on the web funnel the ICP actually uses (no ask UI, and the zero-key path omits the grounding score entirely); and the eval harness measures only recall — hallucination rate (precision) is unmeasured anywhere, and eval never runs end-to-end. The pieces exist; they need to be wired to the wedge artifact and the ICP-facing surface.

- **Marketing promises line-level citations the engine cannot produce** — 🟠 HIGH · effort L · `/Users/micryan/docent/site/index.html:124,181,250 vs /Users/micryan/docent/src/types.ts:198–202 and /Users/micryan/docent/src/synthesize.ts:141–153`
  - Problem: The core TRUST message ('it cites, others guess') is overstated relative to the implementation. A skeptical buyer who checks a citation gets 'static: /dashboard', not code they can look at — and competitors CAN show file:line. The gap is discoverable and undermines the one differentiating promise.
  - Fix: Capture provenance at extraction time (static.ts already knows which file and match offset produced each route/model/env var) and extend EvidenceRef with optional file + line range, threading it through validateEvidence and citeRefs. Until then, soften the site copy to 'a specific file or route'. Given citations ARE the trust message, implement rather than soften.
- **End-user guide steps are generated with no UI-copy evidence — the biggest residual hallucination surface in the paid artifact** — 🟠 HIGH · effort M · `/Users/micryan/docent/src/render/public.ts:35–73 (buildPrompt), /Users/micryan/docent/src/evidence/static.ts:522–553, /Users/micryan/docent/src/improvements.ts:181`
  - Problem: The monetised wedge artifact's step-by-step instructions — the part a non-technical founder hands to USERS — reference UI elements the model has never seen. Route validation catches invented pages but not invented buttons/fields, which is what an end-user actually follows and where wrongness is most visible.
  - Fix: Pass copyStrings (and walker page titles/DOM text where available) into buildPrompt with an instruction like 'when naming a button, label, or field, use ONLY wording from REAL COPY; otherwise describe the action generically'. Optionally add a post-pass that flags quoted UI strings not found in copyStrings, mirroring the route validator.
- **The visible-refusal trust differentiator has no surface on the web funnel the ICP uses** — 🟠 HIGH · effort M · `/Users/micryan/docent/web/app.ts (no 'ask' occurrences; grounding only at :279), /Users/micryan/docent/web/generate.mjs:125–134, /Users/micryan/docent/src/mcp.ts:143`
  - Problem: Enforced refusal + grounding score is named the trust differentiator, but a non-technical founder arriving at explainmybuild.com (the ICP channel) never sees either: no confidence score on their generated guide and no ask box that demonstrably refuses. The differentiator is invisible exactly where the buyer is.

- Fix: Have generate.mjs read app-model.json before cleanup and include groundingReport(model) in job.result (the model is already on disk at that point, generate.mjs:118-122); render the score + thin-ice list in the web results. Add a small 'ask about your app' box on the results page backed by the pure-local ask() (no LLM cost) so refusal is demonstrable in the funnel.

- **No measured hallucination rate: eval is recall-only, never runs end-to-end, and never scores the public guide** — 🟠 HIGH · effort M · `/Users/micryan/docent/src/eval.ts:30-76, /Users/micryan/docent/package.json:26, /Users/micryan/docent/test/eval-fixtures.test.ts`
  - Problem: The headline claim 'It never makes things up' (site/index.html:181) has no number behind it and regressions in the critic/validator/prompt cannot be caught. The monetised artifact (public guide) is the least-measured output.
  - Fix: Add precision metrics to EvalSpec: (a) share of synthesized features carrying verified evidence, (b) an optional forbiddenClaims list per fixture app scored as hallucinations if present, (c) guide-level metrics on the rendered public guide — unverified-route count and jargon count. Add a periodic golden-run script that analyzes the 4 fixture repos on the box's claude subscription (~$0 marginal) and records scores over time.

- **De-identification asymmetry silently drops valid citations** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/src/synthesize.ts:63,141-166,198 and /Users/micryan/docent/src/deid.ts:12-25`
  - Problem: Systematic false-drops of legitimate citations: grounding scores are understated, real provenance is lost, and droppedRefs inflates — precisely on date-heavy AI-built repos (auto-generated commit messages routinely contain dates). The honesty layer punishes the model for obeying the de-id layer.
  - Fix: Build the validator's sets from the SAME transformed strings the LLM saw: apply the identical scrub+deidentify to each set entry (commits, churn paths, routes, audit areas) when constructing staticSet/churnSet/walkerSet/commits, keeping exact-match semantics intact.

- **Self-critic never fact-checks 'extend' guidance or 'operate' steps; operate/glossary carry no evidence and are invisible to the grounding score** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/src/critic.ts:42-57, /Users/micryan/docent/src/synthesize.ts:110-119, /Users/micryan/docent/src/grounding.ts:21-27`
  - Problem: 'How to safely change things' and 'How to operate it' are the runbook sections a founder ACTS on — a hallucinated howTo or operate step is the highest-blast-radius error in the free artifact that anchors trust for the paid one, and it currently bypasses every safety layer.
  - Fix: Add extend entries (area: howTo) to the critique prompt and applyCritique; add an evidence field to operate items in the schema (refs to packageScripts/entryPoints/

envVarNames, all already in the validator's staticSet) and include operate in claimsOf so the grounding score reflects them.

- **ask() can return a 'confident' answer with zero citations, contradicting its own contract** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/src/ask.ts:5-9,62-66,119-128 and /Users/micryan/docent/src/mcp.ts:76`
  - Problem: The one place the refusal/receipts discipline is the whole point can emit uncited confident answers — a skeptical user (or a reviewer testing the trust claim) will hit this on any glossary-ish question.
  - Fix: When the top matches carry no citations, either downgrade to a hedged framing ('from the app model, unverified: …') or require at least one cited entry among the top matches for confident: true; alternatively attach operate/glossary provenance once operate gains evidence refs (see prior finding).
- **Unverified paths still ship inside the paid guide body — validate.ts's own stated contract is not enforced** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/src/validate.ts:1-4 and /Users/micryan/docent/src/render/public.ts:141-159`
  - Problem: The enforcement stops at a warning, so the monetised artifact can still direct end-users to nonexistent pages if the founder skims past the banner — and the second LLM pass can itself introduce new unverified paths that get no further correction.
  - Fix: Add a deterministic final pass that strips or inline-flags each remaining unverified path in the body (e.g. replace with '(page link removed — could not be verified)'), keeping the banner as the explanation. This makes the guarantee mechanical, matching the file's stated contract.
- **Jargon guard on the end-user guide is detect-only and its list is narrow** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/src/render/public.ts:20-27,154-157`
  - Problem: The wedge's differentiation is that the guide is genuinely USER-facing; dev vocabulary leaking into the paid artifact directly erodes that, and today the only mitigation is hoping the non-technical founder rewords it themselves.
  - Fix: Mirror the route-correction pattern: when detectJargon fires, run one corrective pass listing the offending words with 'rewrite in plain language a non-technical user understands'; expand JARGON_RE; add jargon-count to the eval metrics for the public guide.
- **claimConfidence ignores corroboration within a kind — static-only apps are systematically labeled 'low'** — ⚪ LOW · effort S · `/Users/micryan/docent/src/confidence.ts:13-20`
  - Problem: The zero-key web funnel is exactly the static+history tier, so its guides skew to 'low' labels and depressed grounding scores even when claims are multiply corroborated within static evidence — making honest output look weaker than it is to the paying ICP.

- Fix: Count distinct refs as a secondary signal: e.g. ≥3 distinct refs of one kind lifts low→medium (never to high — preserve the runtime/audit ceiling documented in grounding.ts:7-8).

- **Critic fact-checks against a truncated evidence fragment and matches drop-names exactly** — ⚪ LOW · effort S · `/Users/micryan/docent/src/critic.ts:14-15,54`
  - Problem: Truncation biases the critic toward flagging real claims whose evidence fell past the cut (partially mitigated by the 'be conservative' system prompt), and exact-match application means some genuinely-flagged claims are never actually dropped — both silent.
  - Fix: Truncate per-section with priority (keep audit hotspots + routes; trim recentMessages first) instead of a blind prefix slice; log when critique names fail to match any feature/risk so misses are visible.

- **Route extractor skips any path containing a dot, so '/page.html'-style hallucinations bypass validation** — ⚪ LOW · effort S · `/Users/micryan/docent/src/validate.ts:14-15,48-52`
  - Problem: A class of plausible hallucinated paths (AI-built apps often have real .html pages) is invisible to the enforcement layer, and de-routed onboarding steps keep stale instructions.
  - Fix: Validate dotted paths against knownRoutes too (many frameworks expose .html routes; static.ts could record them), or at minimum flag dotted paths as unverifiable rather than silently skipping; drop or flag the whole onboarding step when its route fails verification.

- **Zero-key path reads a nonexistent 'appName' field — guide result is always labeled with the repo slug** — ⚪ LOW · effort S · `/Users/micryan/docent/web/generate.mjs:122 vs /Users/micryan/docent/src/types.ts:206`
  - Problem: The synthesized real app name (a grounded output of the analysis) never reaches the zero-key result — a small but visible output-quality defect on the main funnel ('my-repo-name' instead of the app's actual name).
  - Fix: Read `.name` (with `.appName` as legacy fallback) in generate.mjs.

POSITIONING, WEDGE-FIDELITY & PRODUCT UX

The copy has largely made the pivot to the two-guides wedge — the site subhead, SPA intro line ("we'll write a guide for your users, a runbook for you, and a fix-list"), MCP tool set (generate_end_user_guide / generate_founder_runbook lead), and the zero-key paste-a-GitHub-link flow are genuinely on-wedge and non-technical-friendly, and the accounts/Stripe flow is well hardened (price-label-at-consent, promo codes, dispute/refund revocation). But the MONETISATION side of the wedge is broken in the flagship surface: the /app SPA hands the full end-user guide (rendered + .md download) to anonymous users on both the free server path and the BYO-key path with no Pro gate, while the pricing page says that exact artifact is the $19/mo tier — and a founder who DOES pay receives only a `docent login` CLI token with no way to redeem it in the web app

and no web surface for any advertised Pro feature (share link + access code, widget, auto-current). Layered on top: the marketing "How it works" describes the desktop flow (Approve, Share, Export) that /app doesn't have, the hero H1 leads with the free founder-side value instead of the users'-guide wedge, and the ubiquitous "runs on your machine / only a de-identified summary goes to the AI you choose" claims are factually wrong for the default server-side path — a real risk for a product whose whole trust message is honesty.

- **The monetised end-user guide is given away in full by the web app — no Pro gate on either path** — 🔴 CRITICAL · effort M · `/Users/micryan/docent/web/generate.mjs:118-134, /Users/micryan/docent/web/app.ts:273-290 and 318-324`
  - Problem: The END-USER guide is the artifact the whole business model monetises (runbook free, users' guide paid), and the pricing page sells it as the $19/mo tier ('The public guide that teaches your users', site/index.html:227-231). But the primary CTA surface (/app) delivers it complete and downloadable to anonymous users at ~$0 marginal cost to them. The result-screen upsell (app.ts:287) quietly reframes Pro as only 'sharing + widget + auto-current' — contradicting the pricing page one click away. A founder can download the .md and paste it into Notion/their site forever; only hosted sharing is actually paid. The MCP surface correctly gates generate_end_user_guide behind isPro() (src/mcp.ts:105) — the web app, the surface the non-technical ICP actually uses, has zero license code (grep for license/isPro in web/ returns nothing).
  - Fix: Pick one story and enforce it in /app. Given the constraint (monetise the end-user guide), gate the full users' guide behind Pro in the SPA: show a generous preview (first task section) with the rest blurred/truncated server-side in generate.mjs (truncate `guide` before putting it in job.result so the client never holds the full text), plus 'Unlock the full guide — $19/mo' CTA. Keep runbook + fix-list fully free and downloadable. If instead the intended model is 'view free, share/host paid', then fix the pricing page to say so — but that weakens the wedge's pricing power and should be a deliberate decision, not an accident of two surfaces disagreeing.

- **Paying $19/mo yields nothing usable in the web app — no license redemption, no web surface for any Pro feature** — 🔴 CRITICAL · effort L · `/Users/micryan/docent/docent-accounts/server.mjs:379-407, /Users/micryan/docent/web/app.ts (no license code), /Users/micryan/docent/publish/server.ts:165`
  - Problem: The purchase funnel for the ICP dead-ends in developer tooling. site/index.html:236 and app.ts:287 send buyers to accounts.explainmybuild.com; after upgrading, the return page's redemption instructions are 'docent login ' (server.mjs:383) or 'it signs you in automatically' if 'Explain My Build is open' — which only applies to the desktop/CLI app, not the /app SPA the homepage's primary CTA opens. The SPA has no license/sign-in input at all, and the paid features themselves (share by link + access code, in-app Help widget hosting, auto-current) exist only in publish/server.ts — a self-hosted Node server needing DOCENT_LICENSE env config. A non-technical Lovable founder who pays cannot use anything they bought without a terminal.

- Fix: Close the loop for the web ICP: (1) add a 'I have Pro — sign in' affordance to the SPA that accepts the license token (or better, redirect back from /return with the token in a fragment and store it in localStorage), verified server-side on /api/generate to unlock the full guide; (2) offer hosted sharing as part of Pro — the box already runs publish/server.ts logic, so mint a hosted guide URL + access code from the generate result instead of requiring self-hosting. Until then, the $19 tier is effectively unsellable to the stated ICP.

- **Marketing 'How it works' and pricing promise Approve/Share/Export/auto-current — none exist in /app where the CTA lands** — 🟠 HIGH · effort S · `/Users/micryan/docent/site/index.html:159-162, 171, 232-234 vs /Users/micryan/docent/web/app.ts`
  - Problem: Steps 1-4 on the homepage describe the desktop flow: 'pick the folder your app lives in', 'Read and approve … press Approve when they look right', 'Share a link with an access code, or export a portable file'. The users' guide card (:171) promises 'drop in a Help button … and it stays current as your app changes'; the Pro card (:234) promises 'Export a portable file — and it stays auto-current'. The primary CTA 'Open the app' (:114, :126) opens the /app SPA, which has: GitHub-link paste (not folder-pick, which is buried in Advanced and Chromium-only), no Approve step, no share link, no access code, no portable-HTML export (only raw .md download), and no auto-sync (the 'living guide' mechanism is a GitHub Action requiring YAML + CI secrets — README.md:215-223). A founder following the marketing steps hits a wall at step 3.
  - Fix: Short-term (honesty, S): rewrite the How-it-works steps to match the /app flow (paste link → one button → read your guides → download / upgrade to share) and drop 'auto-current' from the Pro card until a non-technical re-sync exists. Medium-term: build the share + export surface into the SPA (finding 2) and re-add the copy. Non-technical founders can't operate a GitHub Action — 'auto-current' needs a 'we re-check your repo weekly' hosted mechanic to be true for this ICP.

- **Local-first claims are factually wrong for the default zero-key path — and lead the search/social copy** — 🟠 HIGH · effort S · `/Users/micryan/docent/site/index.html:9, 11, 130-134, 183, 223, 258; /Users/micryan/docent/web/index.html:54`
  - Problem: The meta description leads with 'Runs on your machine — only a de-identified summary goes to the AI you choose'; hero trust badges say 'Runs on your machine / Your code stays local'; the free pricing card repeats it; the footer repeats it; and the /app header permanently shows 'Nothing uploaded to us — only a de-identified summary goes to your AI'. But the DEFAULT path (serverGenerate, app.ts:103-136) clones the repo onto EMB's server and analyzes it with EMB's Claude subscription — not the user's machine, not their chosen AI. Two problems: (a) it violates the strategy that local-first is a QUALIFIER, not the lead (og/meta descriptions lead with it, ahead of the two-guides wedge); (b) it's inaccurate for the flagship flow, which is lethal for a product whose differentiator is 'it's honest'. One skeptical Discord reply ('their default literally uploads your repo to their box') undercuts the trust message.

- Fix: Rewrite meta/OG to lead with the wedge ('Turn the app you built into a guide your users can follow — written from your real code, no screen-recording'), and scope the local claims precisely: 'Public repo? We read it for you, keep nothing. Private or sensitive? Run it entirely in your browser or on a local model.' Change the /app header line to be path-aware or neutral ('We keep nothing — and you can run it fully locally'). The in-run notes in app.ts:206-210 already get this right; make the static copy match.

- **Hero H1 leads with the free founder-side value ('Finally understand the app you built'), not the monetised users'-guide wedge** — 🟠 HIGH · effort S · `/Users/micryan/docent/site/index.html:123, 168–173; /Users/micryan/docent/web/index.html:15`
  - Problem: The H1 'Finally understand the app you built' is 'explain my code' framing — it sells the FREE artifact (runbook) and overlaps the commoditized DeepWiki/Cursor-Ask axis. The stated positioning is 'turn the app you built into a guide your USERS can follow — from your real code, no screen-recording.' The subhead does mention both guides (good), but the ordering throughout centres the founder: 'What you get' lists the runbook card first and the paid users'-guide card second (:170-171), and the SPA tab title (web/index.html:15) is 'read your app, in your browser'. The unique, defensible, PAID thing is consistently in second position.
  - Fix: Flip the lead: H1 along the lines of 'Turn the app you built into a guide your users can follow.' with the runbook + fix-list as the 'and you get these free' support acts; reorder the What-you-get cards (users' guide first, runbook second, fix-list third); retitle the SPA page. Also add the 'no screen-recording' contrast somewhere above the fold — it's the one-line proof the category is empty and currently appears nowhere on the site (grep: only in README).

- **README publicly documents the DOCENT_PRO=1 paywall bypass as the way to unlock the paid guide over MCP** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/README.md:186 (also :169)`
  - Problem: The MCP quick-start config literally ships `"env": { "DOCENT_PRO": "1" } // optional — unlocks the users' guide`. license.ts:140 calls this an 'explicit dev/test bypass (kept for tests + local dev)', but the README instructs every MCP user to set it — self-serving a 100% discount on the only monetised artifact. The MCP surface is exactly where technical-ish AI-founders will arrive (Claude Code/Cursor config is copy-paste even for semi-technical users), and the correct mechanism (DOCENT_LICENSE, honoured by loadLicenseToken at license.ts:58-61) is never mentioned in that section.
  - Fix: Change the example env to `"DOCENT_LICENSE": "<your license>"` with a pointer to explainmybuild.com, and note the runbook + improvements + ask tools are free without it. Keep DOCENT_PRO=1 undocumented outside test files.

- **The core ICP's repos are private — and the easy path only supports public github.com, with the private fallback Chromium-only and key-required** — 🟡 MEDIUM · effort M ·

`/Users/micryan/docent/web/generate.mjs:49, /Users/micryan/docent/web/app.ts:79-80, 236-243`

- Problem: Lovable/Bolt/Replit founders who sync to GitHub typically get PRIVATE repos (their whole business is in there). The one-button path rejects anything but public github.com (generate.mjs:49) and the error copy admits 'private repos aren't supported yet' (app.ts:80). The only private-repo route is: open Advanced → get an API key (a separate explainer is needed for 'it's not your ChatGPT login', app.ts:251-259) → use the folder picker, which requires Chrome/Edge/Arc/Brave and a local checkout of the code (which a Lovable-only founder may not have). The '~1 min, one button, no key' promise fails for a large share of the ICP at the very first step.
- Fix: Add a browser-local zip drop: 'Download your project as a .zip (GitHub → Code → Download ZIP, or export from Lovable) and drop it here' — unzips in-page (works in every browser incl. Safari/mobile, keeps the code local, no OAuth build). Longer term, a GitHub 'connect' (fine-grained read-only token or OAuth app) through the existing github-pinned proxy for one-click private analysis. Also soften the failure copy to route users: when the clone 404s, explicitly suggest the zip/folder path rather than just 'private repos aren't supported yet'.

- **The zero-key path drops the confidence score and evidence trail the brand sells** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/web/generate.mjs:125-134, /Users/micryan/docent/web/app.ts:279`
  - Problem: The hero mock's centrepiece is the confidence meter ('Very sure (82/100) — We checked every claim against your real code'), and 'It tells you how sure it is' is one of the three trust cards (site/index.html:182). The BYO-key browser path returns `grounding: groundingReport(model)` (web/engine.ts:136) and the SPA shows 'confidence N/100' (app.ts:279). But the default server path's job.result contains no grounding field and `evidence: []` (generate.mjs:125-134), so the flagship flow never shows the differentiating trust signal it advertised on the way in.
  - Fix: Have the worker read the grounding output the engine already produces (the `docent grounding --format json` command exists per README.md:125) or the app-model and attach `grounding: { score }` plus the evidence summaries to job.result, so easy-path results render the same confidence line as the advanced path.

- **Queue-wait vs client-timeout mismatch: busy hours burn users' rate-limit quota and abandon jobs that keep consuming the shared subscription** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/web/app.ts:122, /Users/micryan/docent/web/generate.mjs:26-33, 151-155, 186-192`
  - Problem: CONCURRENCY=1 and MAX_QUEUE=5 with per-job worst case ~390s (CLONE_MS 90s + ANALYZE_MS 240s + IMPROVE_MS 60s) means a queued job can start ~30 min out, but the client gives up at 6 min (app.ts:122: 'taking longer than expected — please try again'). Each retry POST increments dayCount and the per-IP window BEFORE the

job runs (generate.mjs:188-189), so during a busy hour a user can exhaust their 3/hour allowance receiving nothing, while their abandoned jobs still drain the shared claude subscription (cancel only stops polling — nothing tells the server, and pump() keeps draining the queue).

  - Fix: (1) Return queue position in /api/generate/status and show 'You're #3 in line (~6 min)' instead of a generic spinner; (2) only start the 6-min client clock when state flips to 'running'; (3) don't count a job against the per-IP/day quota until it actually starts (or refund the hit on queue-abandon); (4) add a best-effort DELETE/cancel endpoint so Cancel dequeues a still-queued job.

- **Dead waitlist code shipped on the homepage (no #waitForm exists) plus orphaned CSS and endpoint** — ⚪ LOW · effort S · `/Users/micryan/docent/site/main.js:4-18, /Users/micryan/docent/site/index.html:99-101, /Users/micryan/docent/site/server.mjs:46-54`
  - Problem: main.js exclusively wires a #waitForm/#waitMsg early-access capture, but index.html contains no such elements (grep for waitForm/waitlist in index.html: zero hits) — the script no-ops on every load. The form.wait/#waitMsg CSS (index.html:99-101) and the POST /api/waitlist endpoint (server.mjs:46-54) are likewise orphaned leftovers from the pre-launch waitlist era. Harmless, but it's a wasted script request on the landing page and cruft that will confuse the next edit; alternatively, it's a missed email-capture opportunity now that there's no fallback CTA for visitors not ready to run the app.
  - Fix: Either delete main.js + the wait CSS + the endpoint, or repurpose them intentionally (e.g. a footer 'get launch updates / founding-price reminder' capture — cheap ICP list-building for the creator-channel strategy). Don't leave it half-dead.

- **SPA header uses a 📖 emoji instead of Mic's EMB logo asset; app upsell omits the $12 founding price** — ⚪ LOW · effort S · `/Users/micryan/docent/web/index.html:53, /Users/micryan/docent/web/app.ts:287`
  - Problem: The marketing site correctly uses /assets/emb-mark.png (site/index.html:108), but the /app SPA header brands with '📖 Explain My Build' — a generic emoji at the exact moment the product is judged. Separately, the SPA's only upsell line quotes '$19/mo' flat while the site advertises '$12/mo founding price for the first 100' and the backend implements founding pricing (docent-accounts/server.mjs:282-287) — the surface most likely to convert shows the worse price.
  - Fix: Ship the existing emb-mark.png into web/public and use it in the SPA header (asset reuse only — no redraw), and have the upsell line show the founding price while priceForUpgrade() would still grant it (or just mirror the site's '$12/mo founding price for the first 100' text).

GTM, Distribution & Marketing

The funnel copy is genuinely well-aimed at the ICP — plain-English, zero-jargon, names Lovable/Cursor/v0/Bolt/Replit explicitly (site/index.html:247), the zero-key "paste a GitHub URL, get guides in a minute" path (web/generate.mjs) is a strong activation wow with sensible abuse caps, and the MCP is scoped exactly per the wedge (src/mcp.ts leads with generate_end_user_guide + generate_founder_runbook, Pro-gates the paid artifact, and `ask` visibly refuses — a real strength). But distribution is where it falls down: (1) there is no viral loop anywhere — published guides, the widget, and exports carry no linked attribution, and the zero-key generation result is ephemeral with only a .md download, so the shareable moment evaporates; (2) the MCP/CLI/GitHub-Action channel is install-gated behind a private GitHub repo, so its external reach is zero; (3) the homepage H1 leads with the free commodity side ("Finally understand the app you built") and the verified-empty differentiator ("users' guide from code, no screen-recording") appears nowhere in customer-facing copy; (4) email capture was removed entirely (orphaned form JS/CSS/endpoint remain), there are no community/social links, no sitemap/robots/JSON-LD, and the easy path assumes a public GitHub repo — a real activation mismatch for Lovable/Bolt founders. The single biggest growth lever: make each zero-key generation a shareable, OG-tagged permalink with a linked "Made with Explain My Build" footer, and add the same linked attribution to every published guide/widget/export — that turns both the wow moment and every paying customer's user base into acquisition surfaces. Note: I could only audit the repo source (~/docent/site + ~/docent/web); the live serving tree (~/explainmybuild on the server) and accounts.explainmybuild.com were not verifiable from this Mac.

- **No viral loop: published guides, widget, and export carry no linked EMB attribution** — 🟠 HIGH · effort S · `/Users/micryan/docent/publish/server.ts:135,146; /Users/micryan/docent/site/widget.js; /Users/micryan/docent/desktop/server.ts:679`
  - Problem: Every Pro customer pays specifically to put an EMB-generated guide in front of THEIR users (link, widget, export) — that audience is other app builders' end-users and, for the build-in-public ICP, other founders. Right now that entire surface generates zero referral traffic. This is the classic 'Powered by Intercom/Typeform' loop and it is completely absent, despite the product's whole business model being distribution of its own artifact.
  - Fix: Add a small linked footer — 'Made with Explain My Build ↗ explainmybuild.com' — to the published reader page, the /embed reader, the widget panel bar, and the HTML export (make the existing plain-text line a hyperlink with a UTM). Offer a data-attribution="off" opt-out on the widget if needed for premium feel, defaulting on.
- **Zero-key generation result is ephemeral — the wow moment has no shareable permalink (single biggest growth lever)** — 🟠 HIGH · effort M · `/Users/micryan/docent/web/generate.mjs:33,57,125–135; /Users/micryan/docent/web/app.ts:286–289`
  - Problem: The ICP (build-in-public AI founders on X/Discord/TikTok) shares artifacts, not products. 'It read my repo and wrote my users' guide in a minute' is exactly the kind of post that spreads in Lovable/Bolt communities — but there is literally nothing to link to. DeepWiki won its category with shareable per-repo permalink pages; EMB generates the equivalent artifact and throws it away after 30 minutes.

- Fix: For public-repo zero-key generations, add an opt-in 'Get a share link' that persists the guide to a permalink page (e.g. explainmybuild.com/g//) with proper OG tags, a 'preview — made from this repo's code' banner, a linked Made-with footer, and a prefilled share-to-X button. Label it clearly as a demo/preview so hosted guides for YOUR OWN app (private repo, widget, auto-sync) remain the paid tier.

- **MCP / CLI / GitHub Action distribution channel is install-gated behind a private repo — external reach is zero** — 🟠 HIGH · effort M · `/Users/micryan/docent/README.md:91-101,138-139,179-190,215-223`
  - Problem: The MCP server is beautifully scoped as a distribution play (see strength finding) — but no one outside the org can install it. `npx github:Micipedia/docent` fails without repo access, and a private-repo GitHub Action cannot be `uses:`-referenced by external workflows. As shipped, the MCP, CLI, and CI-Action channels reach exactly zero prospects. The strategy comment in src/mcp.ts:6-9 about deliberate positioning is moot if the artifact is unreachable.
  - Fix: Ship a public install path without open-sourcing: publish the already-bundled, minified JS as an npm package (or public 'installer' repo / GitHub Release tarball) under a proprietary license — the Pro gate is already enforced at runtime via isPro()/license.ts, so the paywall survives. Mirror the Action into a public repo that pulls the bundle. Then list the MCP server in the MCP registries/directories agents actually browse.

- **Homepage leads with the free commodity side; the 'no screen-recording' wedge appears nowhere in customer-facing copy** — 🟠 HIGH · effort S · `/Users/micryan/docent/site/index.html:8,122-124,170-172,179`
  - Problem: The verified-empty market axis is 'a guide your USERS can follow, generated from code instead of a screen-recording' — vs Scribe/Tango-style tools on one side and code-readers on the other. The homepage instead (a) leads with the runbook/understanding promise, which is the FREE, commoditised half the strategy says has no pricing power, and (b) positions inside the crowded 'explain my AI code' category by contrasting only against those tools. A visitor never learns the one claim no competitor can make.
  - Fix: Re-lead the hero on the monetised wedge — e.g. H1: 'Turn the app you built into a guide your users can follow.' with 'From your real code — no screen-recording, no writing it yourself.' Keep 'finally understand it yourself' as the free-tier hook lower down. Add one explicit contrast card: 'Guide tools make you record your screen and re-record every change. This reads your code — and stays current automatically.' Mirror in title/OG/twitter descriptions.

- **Easy-path activation assumes a public GitHub repo — a mismatch with how the ICP's apps actually live** — 🟠 HIGH · effort M · `/Users/micryan/docent/web/app.ts:233–236; /Users/micryan/docent/web/generate.mjs:49`
  - Problem: Non-technical Lovable/Bolt/v0/Replit founders mostly have (a) no GitHub repo at all (project lives in the platform), or (b) a private repo via Lovable's GitHub sync. For them the frictionless path dead-ends immediately, and the escape hatch demands the two highest-friction asks for this ICP: create a developer API key and use a specific browser. The most-qualified visitors are the likeliest to bounce at step one.
  - Fix: Short term: add a 'Where's my code?' helper right under the URL field with 30-second per-platform instructions (Lovable → GitHub sync, Bolt/Replit → download/export), since each platform does offer code export. Medium term: support private repos on the easy path via a GitHub OAuth read-only grant, and/or a drag-in zip for the BYO-key browser path (still local — zip is read in-page, matching the existing folder-picker model).
- **Email capture removed entirely — waitlist form gone from the page while its JS, CSS, and endpoint remain orphaned** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/site/index.html:99–101; /Users/micryan/docent/site/main.js:5–17; /Users/micryan/docent/site/server.mjs:47–54`
  - Problem: The site now has zero owned-audience capture: a visitor who isn't ready to run the app leaves no trace, there's no list to launch to (Product Hunt / founding-100 pricing announcements), and the '$12/mo founding price for the first 100' urgency (index.html:229) has no mechanism to notify interested-but-not-yet buyers. Plus the dead form plumbing is confusing residue.
  - Fix: Reinstate a single capture with post-launch intent — e.g. under pricing: 'Not ready? Get the founding-100 price locked in — email me my code' → the existing /api/waitlist endpoint. Also use email capture as the fail-soft for the free-generation daily cap (see cost-cap finding). Or, if deliberately email-free, delete main.js's handler, the orphan CSS, and the endpoint.
- **Paywall on the end-user guide is inconsistent across surfaces — the browser gives the monetised artifact away in full** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/web/generate.mjs:118–131; /Users/micryan/docent/web/app.ts:274,286,319–324; vs /Users/micryan/docent/src/mcp.ts:105–110 and /Users/micryan/docent/desktop/server.ts:284`
  - Problem: The strategy monetises the END-USER guide, yet the most accessible surface (browser, no key, no signup) hands over the full guide as a downloadable file a founder can paste into Notion or their own site — making $19/mo effectively 'hosting + widget + auto-sync' only. It also undercuts the MCP/desktop surfaces that do gate it: the same user gets refused in Claude Code but served in the browser.
  - Fix: Pick one line and align all surfaces. Least-friction option that preserves the wow: keep the full on-screen view free in the browser (activation), but gate the users'-guide .md download

behind Pro the same way share/export are (runbook + fix-list downloads stay free). Alternatively adopt the desktop's preview-and-unlock pattern in the SPA.

- **SEO plumbing missing: no robots.txt, no sitemap, no canonical, no JSON-LD; demo page has zero OG tags** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/site/ (directory); /Users/micryan/docent/site/demo/index.html:1-9; /Users/micryan/docent/site/index.html:8-20`
  - Problem: The site can't tell crawlers what exists (no sitemap), the best shareable page (/demo — 'a real guide it made') renders a generic card when posted to X/Discord where the ICP lives, and there's zero structured data despite a ready-made FAQ section (index.html:242-254) that maps 1:1 to FAQPage schema. Long-tail queries the ICP actually types ('how to make a user guide for my Lovable app', 'document my Bolt app for customers') have no page to land on.
  - Fix: Ship the S-sized basics now: robots.txt + sitemap.xml, rel=canonical, FAQPage + SoftwareApplication JSON-LD, and OG/twitter tags on /demo (reuse og-image or a demo-specific one). Then M-sized: 4-6 intent pages per builder platform ('Create a users' guide for your Lovable/Bolt/v0/Replit app — from its code') feeding the /app funnel.
- **Free-generation daily cap fails to a hard bounce at exactly the moment a launch spike hits** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/web/generate.mjs:24-33,75-79,187`
  - Problem: These caps are correct cost-safety for the shared subscription — but the GTM consequence is that the exact event the strategy depends on (a Product Hunt feature, a viral X post in a Lovable Discord) saturates the funnel within the first hour, and every subsequent visitor is bounced to the highest-friction alternative (create an API key). The spike traffic is unrecoverable because there's no capture at the bounce (ties to the email-capture finding).
  - Fix: Keep the caps; change the failure mode. At cap/queue-full, offer 'Leave your email + repo URL and we'll generate it overnight and send you the link' (writes to the existing waitlist.jsonl-style log; a cron drains it within the same daily budget). For a planned launch day, pre-agree a temporary GLOBAL_DAY raise + a second concurrency slot as a conscious, reversible decision.
- **No community/social presence or social proof anywhere on the site** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/site/index.html:107-116,257-260`
  - Problem: The stated channel strategy is creator/community distribution (build-in-public X, Lovable/Bolt Discords, PH, short-form) — but the site gives a visitor arriving FROM those channels no thread back to a community, no social proof to trust a $19/mo card, and gives Mic no follow surface to accumulate an audience between launches. For this ICP, 'who else uses this' is the main objection after price.
  - Fix: Add an X/Twitter handle to header+footer and OG twitter:site; once the shareable-permalink loop exists, surface a 'recently generated guides' wall (public repos only, opt-in) as

living social proof; add a PH badge at launch. Keep it lightweight — no fake logos or invented counts.

- **STRENGTH: MCP scoping and funnel copy implement the wedge correctly** — ⚪ LOW · effort S · `/Users/micryan/docent/src/mcp.ts:28-33,54-84,105-110,143; /Users/micryan/docent/site/index.html:159-162,247; /Users/micryan/docent/web/generate.mjs:5-9`
  - Problem: Not a problem — recording that the three things the constraints most worry about (MCP scoped narrow to the two guides rather than a generic 'ask my codebase', enforced refusal as the trust message, monetise-the-users'-guide gate in the MCP) are correctly implemented in code, so no re-scoping work is needed there. The gaps are reach (private-repo install) and copy-lead (homepage H1), covered in other findings.
  - Fix: Preserve this scoping when opening the distribution channel (npm bundle finding) — do not add broader 'index/search my codebase' tools to the MCP to chase generic utility; the narrow surface is the differentiation.

SERVING, DEPLOY & RELIABILITY

The serving code itself is unusually careful for its size — the zero-key generate path (web/generate.mjs) has a genuinely well-designed guardrail set (TOCTOU-free synchronous rate-limit critical section at lines 177-192, process-GROUP kill of the claude grandchild on timeout at lines 86-89, fail-closed repo-size check at line 108, job+poll to clear Cloudflare's ~100s limit, GC that never collects queued/running jobs), and the /app trailing-slash 301 fix is properly root-caused and commented (site/server.mjs:64-75). The reliability risk is almost entirely OPERATIONAL, not code: the entire deploy is a 4-step manual rsync that exists only in a memory file, the live tree ~/explainmybuild is not git-tracked and has already demonstrably drifted from the repo (repo site/index.html still has the pre-fix href="/app" CTAs), the engine runs from an unversioned rsync snapshot (~/explainmybuild-engine), there is no health endpoint/monitoring/alerting on the money path (failures are appended to a jsonl nobody reads), no circuit breaker protecting the shared claude subscription during a rate-limit event, and no backup of waitlist.jsonl or the accounts.db license database. One box + one tunnel is a full-stack SPOF. Note: I could not inspect live prod (~/explainmybuild is on the server, not this Mac) — box-side claims are grounded in the repo code, docent-accounts/DEPLOY.md, and the reconciliation record; where I extrapolate I say so.

- **Entire production deploy is an unscripted, memorised 4-step manual rsync** — 🟠 HIGH · effort S · `/Users/micryan/docent/package.json (scripts), /Users/micryan/docent/scripts/ (only build-cli.mjs), /Users/micryan/docent/site/server.mjs:12-14`
  - Problem: A multi-file, ordering-sensitive deploy done from memory will eventually be done wrong (partial rsync → import crash → whole site down, since one process serves marketing + /app + gitproxy + generate). It is also unreproducible by anyone but the person holding the memory.
  - Fix: Commit scripts/deploy-web.sh: (1) npm run web:build, (2) rsync site/server.mjs web/gitproxy.mjs web/generate.mjs + site static assets to box:~/explainmybuild/, (3) rsync -az --

delete web/public/ box:~/explainmybuild/public/app/, (4) systemctl restart explainmybuild, (5) automated smoke: curl / (200), /app (301→/app/), /app/ (200 + current app-.js resolves), /gitproxy/evil.com (403), POST /api/generate with bad body (400), /healthz (200). Fail loudly on any step.

- **~/explainmybuild is not git-tracked and the repo has ALREADY drifted from the box (proven, both directions)** — 🟠 HIGH · effort M · `/Users/micryan/docent/site/index.html:114,126,205,225,259; /Users/micryan/docent/site/server.mjs`
  - Problem: Source-of-truth-on-a-box: box edits don't reliably land in the repo, repo edits can clobber box fixes, and there is no diff/alarm when the two disagree. This is exactly how the pre-07-10 drift (server.mjs missing gitproxy/generate/301) happened.
  - Fix: Do a one-time full reconcile: pull the box's ~/explainmybuild (minus waitlist.jsonl/generate.jsonl/backups) down, diff against repo site/+web/, commit the deltas (start with the five /app → /app/ CTAs). Then declare the repo canonical, deploy ONLY via the committed script, and add a drift check to that script (ssh + diff of server.mjs/gitproxy.mjs/generate.mjs/index.html before overwriting; abort on unexpected box-side changes).
- **Zero-key engine runs from an unversioned rsync snapshot (~/explainmybuild-engine) with no version stamp anywhere** — 🟠 HIGH · effort S · `/Users/micryan/docent/web/generate.mjs:18,113-116,146`
  - Problem: Silent engine drift: the public zero-key product can run weeks-old engine code with nobody noticing, and incidents can't be correlated to an engine version. The rsync also has no integrity check — a partial transfer leaves a broken engine that fails soft on every job.
  - Fix: Make ~/explainmybuild-engine a git checkout of the private repo pinned to a tag (deploy = fetch + checkout), or at minimum have deploy-web.sh write ~/explainmybuild-engine/.engine-version (git rev-parse --short HEAD) and include it in every generate.jsonl line and in /healthz; add a post-deploy probe that runs the engine against a tiny fixture repo and asserts a guide is produced.
- **No health endpoint, no monitoring, no alerting on the box server — generate can fail 100% silently forever** — 🟠 HIGH · effort S · `/Users/micryan/docent/site/server.mjs (no /healthz route), /Users/micryan/docent/web/generate.mjs:146`
  - Problem: The primary conversion funnel (paste repo URL → guides, no key) can be fully dead while the site looks healthy. Fail-soft without observation = fail-silent.
  - Fix: Add /healthz to site/server.mjs returning {ok, queued, running, dayCount, lastJob: {state,ms,ts}, engineVersion} (all already in-memory in generate.mjs — export a stats() fn). Point the existing watchdog/Telegram infrastructure (com.dh.watchdog pattern already on this box) at it: alert on unreachable, on N consecutive error states in generate.jsonl, and on queue stuck >15 min.

- **No circuit breaker: during a subscription rate-limit every queued job still spawns an engine+claude run against the SHARED subscription** — 🟠 HIGH · effort S · `/Users/micryan/docent/web/generate.mjs:99-149,151-155`
  - Problem: A subscription rate-limit event — which per the operating rules is a full-stop condition for this shared subscription — instead gets up to dozens of additional claude invocations, each burning a visitor's rate-limit allowance on a guaranteed failure.
  - Fix: Add a breaker in generate.mjs: after N (e.g. 3) consecutive job errors whose engine stderr matches rate-limit/auth signatures (or simply 3 consecutive errors of any kind), auto-pause intake+draining for a cool-off (e.g. 30 min), return the existing 503 kill-switch message, and fire the alert from the monitoring finding. Log breaker trips to generate.jsonl.
- **waitlist.jsonl, generate.jsonl and the accounts.db license database have no backup/DR path** — 🟠 HIGH · effort S · `/Users/micryan/docent/site/server.mjs:18,51; /Users/micryan/docent/docent-accounts/store.mjs:16; /Users/micryan/docent/docent-accounts/docent-accounts.service (ReadWritePaths)`
  - Problem: A single-disk failure on the box silently destroys the early-access lead list (the top of the funnel) and the billing/entitlement DB. Stripe can partially reconstruct customers, but license→customer mapping and waitlist are gone.
  - Fix: Nightly cron on the box: sqlite3 accounts.db 'VACUUM INTO ' (WAL-safe) + copy waitlist.jsonl/generate.jsonl, then ship into the already-existing offsite backup pipeline (S3/GitHub) used for DH; verify one restore. ~20 lines of shell, commit it to the repo next to the service unit.
- **The 'canonical' prod server cannot run or be tested from the repo — its imports only resolve in the deploy tree** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/site/server.mjs:8-9`
  - Problem: The single most important process in the product (one server = marketing + SPA + gitproxy + generate) has zero automated integration coverage; the only place it ever runs assembled is production.
  - Fix: Add a tiny compose step (scripts/compose-site.mjs → dist-site/: server.mjs + gitproxy.mjs + generate.mjs + site assets + web/public as public/app/) and a node --test that boots the composed server on an ephemeral port and asserts the smoke routes (/, /app 301, /app/ 200, /gitproxy 403 off-host, /api/generate 400 on bad body, waitlist POST). Run it in ci.yml; make deploy-web.sh rsync dist-site/ so 'what CI tested' === 'what ships'.
- **Kill switch requires SSH + env edit + service restart; the restart itself destroys all in-flight jobs** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/web/generate.mjs:17,57,172`
  - Problem: In an active abuse or cost incident, the fastest response path is a manual SSH session, and executing it takes the whole site (marketing + SPA + gitproxy) through a restart. No automation (e.g. the circuit breaker or watchdog) can flip it.

- Fix: Make ENABLED() also check a sentinel file (existsSync(join(homedir(),'explainmybuild','GENERATE_OFF')), result cached ~5s): `touch` = instant off with no restart, no lost marketing traffic, and scriptable by the breaker/watchdog. Keep the env var as the hard override.

- **Queue math guarantees abandoned-but-still-executed jobs: client gives up at 6 min, worst-case queue wait is ~32 min, and there is no cancel** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/web/app.ts:122, /Users/micryan/docent/web/generate.mjs:28-33`
  - Problem: Under even a small burst (a Product Hunt/Discord spike — the exact ICP channels), later visitors are guaranteed a failure message while the box still burns subscription time on guides nobody will ever see.
  - Fix: Track lastPolledAt per job (update in the status handler); in pump(), skip (mark error) queued jobs not polled for >60s. Return queue position in the status response and have the client show it and extend its timeout to position×7 min. Optionally refund the ipHits entry when a job dies unclaimed.

- **Single box + single tunnel is a full-stack SPOF — including purely static pages that don't need the box at all** — 🟡 MEDIUM · effort M · `/Users/micryan/docent/site/server.mjs:1-3, /Users/micryan/docent/web/README.md:32-36`
  - Problem: Box reboot, cloudflared crash, or the known deploy-restart window takes down not just generate but the marketing homepage and the fully-client-side /app (folder-picker + own-key mode needs NO server at all) — maximum blast radius for minimum reason.
  - Fix: Cheap tier first: enable Cloudflare 'Always Online' for the zone and confirm systemd Restart=always on explainmybuild.service. Next tier: publish site static pages + web/public to Cloudflare Pages as a documented failover origin (generate/gitproxy degrade to the Advanced own-key + folder-picker path, which still fully works client-side). Do NOT re-architect around this — a second box isn't justified yet.

- **Hardcoded nvm node path + 'claude lives next to node' assumption: a routine nvm version bump breaks both services and the generate path** — 🟡 MEDIUM · effort S · `/Users/micryan/docent/docent-accounts/docent-accounts.service (ExecStart), /Users/micryan/docent/web/generate.mjs:19-20`
  - Problem: A routine runtime upgrade is a latent full-outage (units) plus a silent generate outage (claude lookup), with the failure surfacing only at the next restart — possibly weeks later, far from the change that caused it.
  - Fix: Create version-stable symlinks on the box (e.g. /home/micipedia/bin/{node,claude} → current nvm targets, or `nvm alias default` + a stable wrapper), point ExecStart and generate.mjs's CHILD_ENV PATH at that dir, and add 'node -v + which claude' assertions to the deploy script's smoke step.

- **explainmybuild.service systemd unit exists only on the box — restart policy, env, and resource caps are unverifiable and unreproducible** — 🟡 MEDIUM · effort S · `/Users/micryan/docent (find: only docent-accounts/docent-accounts.service exists); /Users/micryan/docent/site/server.mjs:95-101`
  - Problem: The crash-recovery story for the entire public surface is unauditable from source, and rebuilding the box from scratch would require reverse-engineering the unit.
  - Fix: Copy the live unit into the repo as site/explainmybuild.service (mirror docent-accounts' hardening: Restart=always, RestartSec, MemoryMax, NoNewPrivileges, ProtectSystem=strict with ReadWritePaths for waitlist/generate jsonl), have deploy-web.sh install it, and add process.on('uncaughtException'/'unhandledRejection') log-and-exit handlers to server.mjs so systemd restarts cleanly instead of the process wedging.

DenialHelp

denialhelp.com — consumer health-insurance appeal help + DH Pro (clinician B2B). Shared 'appeals' codebase.

Overall assessment

This is a genuinely hardened solo-founder codebase — the de-id core, idempotent billing, prepared-statement discipline, and ops tooling are well above baseline — but the audit surfaced three live PHI red-line violations (intake account-takeover, insurer-required.ts leaking denial-letter text to non-BAA Anthropic, and the analytics gate failing OPEN on clinical surfaces) that must be fixed before any promote, plus a cluster of FAIL-LOUD erosions (verifier fails open, alert-batch bug swallows new alerts, dead validation/dead safety code) that make failures silent. Commercially, the two named pain points are confirmed as concrete code bugs: the outreach reply-classifier bins warm leads and the conversion tracker (lib/track.ts) is dead code with zero importers, so DH is genuinely flying blind on both interest and attribution. The plan sequences PHI/security first, then the correctness/FAIL-LOUD bugs, then the highest-leverage DH commercial fixes (funnel, SEO crawlability, tracking).

Ranked improvements

1. Fix consumer intake cross-patient PHI account-takeover (unverified-email cookie binding)

🔴 **CRITICAL** · effort M · *PHI / Auth*

> ✅ ***Independently verified against source.*** *Verified: intake resolves customerId from unverified email; consumer-auth cookie = bare HMAC(customerId), no exp/nonce.*

What to do: In app/api/intake/route.ts:190-192,780 stop binding the dh_consumer ownership cookie to a pre-existing customer_id resolved purely by unverified email. Either always create a fresh customer row and only merge into the canonical customer after the emailed magic-link (makeClaimToken) verifies, or scope the intake cookie to grant ownership of ONLY the newly-created appeal. Keep verifyClaimToken as the sole cross-appeal ownership path; add an audit_log row whenever an intake resolves to an existing customer_id. Bundle the enabling weakness: lib/consumer-auth.ts:40-68 signs only customer_id with no exp/nonce/version — add expiry + a bumpable cookie_version on the customer row so a single account can be revoked without a global secret rotation.

Why it matters: Confirmed in source: an attacker who knows a victim's email can POST one anonymous intake and receive a valid HMAC cookie owning ALL the victim's prior appeals — full PHI (letters, denial docs, drug/condition), refund, and delete. This is unauthorized PHI disclosure with only a known email; it defeats the entire consumer ownership model and the magic-link flow built to prevent it. Code-level violation of the PHI-default-closed red-line.

2. Route insurer-required.ts PHI through the de-id/subscription wrapper (stop direct non-BAA Anthropic egress)

🔴 **CRITICAL** · effort S · *PHI / AI pipeline*

> ✅ ***Independently verified against source.*** *Verified: insurer-required.ts:352 allowDirect zero-phi → messages.create on denial-letter text (AI_SYSTEM: 'denial letters').*

What to do: In lib/ai/insurer-required.ts:349-353,430-441,534-541 the aiExtract() and verifyFlag() calls use getAIClient({allowDirect:'zero-phi'}) and client.messages.create() directly, but the input is a 4,000-char slice of the patient's own denial letter (prefill.ts:429, locateAppealSection carries page headers with patient name / member ID / claim #). Swap both to safePHITextCall(expectJSON:true) exactly as mine-denials was fixed on 2026-06-10; remove/correct the false 'zero-phi' comment; add a regression test asserting no getAIClient() call remains in the module.

Why it matters: Confirmed in source: the same PHI-bearing section egresses to direct non-BAA Anthropic ~20+ times per letter (Promise.all fan-out), bypassing redact/outboundGate, in production, regardless of HIPAA_LIVE/DEID flags. Identical bug class was already found and fixed in mine-denials. Code-level violation of the PHI-to-external-LLM red-line.

3. Make the HIPAA analytics gate host-aware and truly fail-closed

🔴 **CRITICAL** · effort S · *PHI / Analytics*

> ✅ ***Independently verified against source.*** *Verified: callers use usePathname() (browser path); proxy rewrite keeps browser URL bare; AH clinical paths absent from DENY_PREFIXES → fail open.*

What to do: In lib/analytics-gate.ts:14-38 the AH branch (path==='/ah'||startsWith('/ah/')) never fires because proxy.ts:287-300 rewrites approvalhelp.com clinical paths so the browser URL never contains '/ah', and DENY_PREFIXES (lines 21-24) lists only consumer-shaped paths — so isPublicAnalyticsPath returns TRUE and non-BAA PostHog+GA4 fire pageviews (with full URL incl. patient/chart/appeal IDs) on clinical PHI surfaces. Fix: gate on host — on the AH host allow ONLY an explicit marketing allowlist and deny everything else; keep the /ah branch for the DH host. Add a vitest that enumerates every proxy AH_ROOT_PATHS entry and asserts the gate denies its bare form.

Why it matters: Confirmed in source (DENY_PREFIXES has no /scribe,/patients,/chart,/appeals,etc.; AH_PUBLIC only holds /ah-prefixed dead paths). The gate's own comment says every non-marketing /ah path is 'clinical PHI'; the design is inverted to fail-open by a sibling module. Also root-causes part of the traffic-blind pain point — the gate is host-unaware so DH-vs-AH measurement can't be reasoned about. Shared module + PHI red-line violation.

4. Close the remaining de-id perimeter gaps (system-prompt PHI, dead rehydration check, generator fail-open)

🟠 **HIGH** · effort M · *PHI / AI pipeline*

What to do: Three fixes in the PHI egress path: (a) lib/ai/generation-pipeline.ts:79-108 puts patient-evidence free text into the UN-redacted systemPrompt (used by second-level/external-review/nsa-idr) and subscription-client.ts:68-71 passes it as a --system-prompt CLI argv (process-list visible) — move groundTruth/strategy/toolkit into promptWithPhi like generate.ts, run outboundGate over systemPrompt, and pass system prompt via stdin/temp-file not argv; (b) rehydrateOrThrow/verifyRehydration in lib/de-id/redact.ts:211-312 are dead code (zero call sites) so mangled [NAME_1]/[MEMBER_ID_2] tokens ship silently — wire verifyRehydration into callPHILLM with retry-then-throw to pending_review; (c) lib/ai/generate.ts:512-556 streams full PHI to direct Anthropic when DEID is off with no production fail-closed guard and no audit row — mirror the safePHITextCall H3 guard and add the deid_bypass audit write.

Why it matters: All three confirmed as distinct de-id completeness gaps on the highest-stakes output (the appeal letter). (a) is a partial code-level violation of de-id-before-external-LLM for case-specific free text on 3 escalation endpoints; (b) drops real names/member IDs into delivered letters silently; (c) leaves the flagship path one env-var regression away from un-redacted, un-audited PHI egress.

5. Make the pre-delivery letter verifier fail CLOSED

🟠 **HIGH** · effort S · *PHI / AI safety*

What to do: In lib/ai/verifier.ts:134-137 extractClaims swallows any LLM error and returns {claims: []}, so checkContradictions finds zero violations and passed=true — the blocking safety gate waves everything through exactly when the subscription CLI is degraded. Return a sentinel (passed:false, kind:'verifier_unavailable', high severity) so the existing pending_review path in app/api/generate holds the letter for human review, and Sentry-alert the failure.

Why it matters: Confirmed in source. This gate exists to stop Shannon-class contradictions (letter asserts a criterion met when the evaluator computed not_met) reaching patients; failing open in the AI safety control directly violates FAIL LOUD and self-masks (no block is recorded).

6. Fix the alerting pipeline that silently swallows new alerts (and its silent-drop Telegram SPOF)

🟠 **HIGH** · effort S · *Reliability / FAIL LOUD*

What to do: Two fixes: (a) app/api/internal/audit-anomaly/route.ts:30-31 computes newOnes = alerts.filter((_,i)=>i<stored) but persistAndDedupe skips duplicates in place and returns only a count, so an interleaved [duplicate, new] batch re-sends the duplicate and never telegrams the new alert — which is then dedupe-suppressed for 12h. Change persistAndDedupe to return the actual stored Alert[] and pass that to telegramSend; add a vitest for the interleaved case. (b) lib/telegram-alert.ts:68-83 and the detector send-path drop failures silently ('void fetch().catch(()=>{})', creds-missing cached as 'missing') — add console.error+Sentry.captureMessage on failure, retry telegram_sent=0 rows each tick, and log loudly if creds are missing.

Why it matters: Confirmed in source. This is the single pipe through which cron failures, PHI-access anomalies, de-id drift, and cron staleness reach Mic — and it both drops new alerts on interleave and fails totally silently at the last hop. FAIL LOUD inverted inside the component that implements FAIL LOUD.

7. Enforce the 'required' intake facts (wire the dead validation as a soft gate)

🟠 **HIGH** · effort M · *Consumer funnel / correctness*

What to do: In app/intake/IntakeBodyClient.tsx REQUIRED_FACT_KEYS (line 142) and missingRequiredFacts() (line 857) exist and drive red asterisks, but grep confirms missingRequiredFacts is referenced ONLY at its definition — canAdvance never checks it and the server schema (app/api/intake/route.ts:81-120) is all .optional(). Wire missingRequiredFacts into handleAdvanceClick as a SOFT gate: list missing member ID / claim # / denial reason / patient name with the existing red-highlight plus a 'continue anyway — my letter doesn't show these' acknowledgment (some denials genuinely lack a claim #), and record the acknowledgment for triage.

Why it matters: Confirmed dead code. Today a user can pay $39 with a blank member ID, claim number, and denial reason when prefill misses — producing a weak letter, guarantee refunds, and support load. Direct revenue + quality bug on the paid funnel.

8. Fix the outreach interest pipeline (reply-classifier + role-inbox enrichment + queue starvation)

🟠 **HIGH** · effort M · *Outreach / commercial*

What to do: Three linked fixes in the outreach libs (root-causes pain point b, ~0 real interest): (a) lib/outreach/classify-reply.ts:14-21 — checks unsubscribe first on bare substring 'stop', runs the huge auto_reply regex ('thank you for...reaching') BEFORE 'interested', and matches bare 'yes' last: rewrite with \b word boundaries, score-based classification (interested outranks generic thank-you), classify on full body not subject+snippet, and delete the drifted inline duplicate in app/api/webhook/instantly-reply/route.ts:116; (b) lib/outreach/email-enrich.ts:194-198 + scrape-enrich.ts:279 still generate AND +5 score-boost info@/contact@/office@ addresses that the downstream fix then skips forever (cache never re-enriches) — import isRoleInbox to drop/penalize role locals and add a one-off migration to NULL role-email rows so they re-enrich to a personal address; (c) app/api/internal/outreach-worker/route.ts:119-141 re-selects permanently-blocked oldest rows every tick, starving the daily budget — set status='blocked:' for hard invariants so they leave the SELECT, and dedupe the alert. Then replay stored outreach_replies through the new classifier to recover misfiled leads (read-only).

Why it matters: All confirmed in source. This is a concrete, code-level mechanism for 'sends fine but ~0 real interest': warm replies are binned to ooo/unsub, the sendable pool silently shrinks to role inboxes, and blocked rows can consume the whole daily send budget while reporting 'ran fine'.

9. Wire conversion + attribution tracking (kill the traffic-blindness)

🟠 **HIGH** · effort M · *Analytics / commercial*

What to do: Addresses pain point (a): (a) lib/track.ts has ZERO importers (grep-confirmed) — wire it at 4-6 PUBLIC-path conversion micro-events (pricing CTA click, /intake?vertical= entry, signup CTA) so GA4 (live: G-RSZSCR6RK4) + PostHog get funnel events, keeping all firing behind isPublicAnalyticsPath; (b) read the dh_attr cookie in the AH signup API and persist utm_*/referer columns like app/api/intake does, and surface both in /admin/cac; (c) move recordCrawlerHit + AiReferralBeacon from SeoArticleShell into the root layout on public paths so AI-crawler/referral telemetry covers the homepage and 80+ vertical money pages, not just Tier-1 reference pages; (d) fix proxy.ts:425-433 attribution own-domain check hardcoded to denialhelp.com (self-poisons AH + misfiles cross-brand).

Why it matters: Confirmed: tags are live in the prod bundle but the ONLY events they get are public pageviews — the carefully-built track() wrapper was never wired to a single call site, and the AI-traffic dashboard is blind precisely on the pages where a referral converts. You currently cannot answer 'which channel produced this signup/appeal'.

10. Fix DH SEO crawlability and the canonical-to-homepage bug

🟠 **HIGH** · effort S · *SEO / commercial*

What to do: (a) The homepage ships ZERO crawlable links to the 80+ vertical money pages or the /how-it-works subtree — components/VerticalsBrowser.tsx gates card s behind client-side {isOpen && ...}; server-render the vertical anchors (default one category open or CSS-collapse) and add a footer internal-link block in FooterContact so equity flows into the subtree. (b) app/layout.tsx:108-115 sets alternates.canonical:'/' which metadata-less client pages inherit — app/coverage-check/page.tsx live-declares canonical=homepage (curl-confirmed) and will be dropped from the index. Remove the root canonical (per-page canonicals already exist on 189 files) or wrap client public pages in a server metadata shell; add an audit check that every sitemap URL exports its own canonical.

Why it matters: Confirmed via live curl. Orphaned-from-home money pages + a canonical that says 'I'm a duplicate of the homepage' are fully consistent with the observed 'crawl-fix worked but stuck page 6-7'. Internal linking is the #1 lever Google uses to prioritize which of 9.9k URLs to rank.

11. Fix DH top-of-funnel conversion leaks (hero CTA, bare /intake default, upload-first pricing)

🟠 **HIGH** · effort M · *Consumer funnel*

What to do: (a) app/page.tsx hero (901-931) has no CTA and StickyMobileCTA.tsx:50 anchors to the 86-item browser BELOW the drop zone — move the drop zone / a compact upload button into the hero and point the sticky CTA at it (also validates the queued upload-first-home build). (b) app/intake/page.tsx:42 defaults bare /intake to 'glp1'; 6 shipped CTAs (emergency-room-denial, how-it-works, faq, preventive-recoded, checkout, NewMenu) hit it, so an ER-denial reader lands on 'Which treatment: Wegovy/Zepbound…' — render a lightweight 'what were you denied?' picker instead of the silent GLP-1 fallback. (c) app/intake/IntakeBodyClient.tsx:333 starts drop-zone arrivals at step 2, skipping the step-1-only price+guarantee box — render it on step 2 and move the email field to step 2 so the lead-recovery checkpoint fires for the best-converting cohort.

Why it matters: Confirmed in source. The highest-intent path currently gets no hero entry point, a wrong-context bait-and-switch, and the worst price transparency (the exact '17% bail at Stripe' the manual-path fix targeted). These are the leaks at the top and bottom of the revenue funnel.

12. Fix DH copy-honesty violations (uncited overturn stats, 'pay on delivery', llms.txt physician-review)

🟠 **HIGH** · effort S · *Copy honesty / constraint*

What to do: (a) Vertical landing pages render uncited stat cards — glp1.ts:85-91 'Up to 6 in 10 / 40-60%', biologics '60%+', cancer '60-70%' — while the homepage carefully cites KFF; add a visible source per stat or normalize to the cited 1-in-3 figure, and fix glp1's contradictory 'Under 1 min' vs 'Under 10 minutes'. (b) i18n.ts:252 'We charge once, only when we deliver a letter your doctor can sign' renders on every vertical page but the real sequence is Stripe payment -> generation — reword to the true pre-payment-triage + money-back-guarantee framing. (c) app/llms.txt DH block states 'Appeals are physician-reviewed and physician-signed' (subject =

DenialHelp) — reword to match the FAQ: DH drafts, the patient's OWN treating physician reviews/signs, DH does not file or provide medical review.

Why it matters: Confirmed in source. These are code-level violations of the no-invented-promises / no-market-overstatement copy-honesty red-line — and (c) sits in the file explicitly designed for verbatim AI citation, the worst place for drift.

13. Fix SEO freshness and edge-caching signals

🟠 **HIGH** · effort M · *SEO*

What to do: (a) app/sitemap.ts emits lastmod = request time on all ~9.9k URLs (live curl: every URL carries the exact fetch timestamp) and 110/111 articleJsonLd callers emit datePublished/dateModified = now (lib/seo/json-ld.ts:58-70) — persist a real per-URL last-modified (content-hash or a seo_lastmod table) and pass fixed content-launch dates. (b) The whole tree is force-dynamic with cache-control:private (app/layout.tsx:24), so every crawl is a full SSR round-trip to the Sydney NucBox — add edge caching (Cloudflare Cache Rule s-maxage=3600 SWR) scoped ONLY to public marketing paths (reuse the isPublicAnalyticsPath deny list so PHI/auth pages are never cached), bypass on session cookies, and handle the per-request CSP nonce.

Why it matters: Confirmed via live curl. Google ignores always-'current' lastmod and throttles crawl on a slow origin — together these cap crawl budget and freshness prioritization on a 9.9k-URL programmatic-SEO site, compounding the page 6-7 stall. CONSTRAINT: the caching change MUST exclude every PHI/auth path.

14. Close the cron/deploy observability gaps (OnFailure units, cron_runs coverage, BUILD_ID drift, deep health)

🟠 **HIGH** · effort M · *Reliability / FAIL LOUD*

What to do: (a) Zero systemd units have OnFailure= — add one templated appeals-cron-failure@.service that Telegrams unit+log-tail and attach OnFailure to every appeals-*.service; bump the phi-purge curl timeout from 15s to 290s. (b) Only ~16 of 47 internal cron routes write cron_runs — the daily Stripe overage charger (app/api/internal/overage-batch-charge) and the 7-year retention-purge write nothing and have no tracked timer; wrap every cron route in recordCronRun and add an expected-cron manifest so 'never ran' is as loud as 'ran and failed'. (c) Add a BUILD_ID-vs-git-HEAD drift rule to the existing hourly detector (embed commit SHA at build, expose in /api/health) so the documented 'HEAD moved but build failed, stale code serves' mode pages someone. (d) app/api/health/route.ts never touches SQLite — add a SELECT 1 + 503 so a LUKS/decrypt failure stops reading green everywhere.

Why it matters: Confirmed in source. A dead appeals.service at 04:15 means the HIPAA phi-purge silently doesn't run and the self-tuning staleness rule takes ~3 days to notice; a failed build silently serves stale code with no page; and every monitor reports healthy while every PHI API 500s after a locked-volume reboot.

15. Harden proxy.ts: extract routing logic, add tests, add the AH_ROOT_PATHS sync guard

🟠 **HIGH** · effort M · *Architecture / routing*

What to do: proxy.ts is a 688-line untested monolith (brand routing + rewrites + soft-404 + CSRF + admin gating) whose own comments document 4 prior production incidents. Extract the pure logic (brandFor, brandRewrite, isKnownRoute + the five path lists) into lib/routing/ so it's importable without NextRequest; add vitest table tests for the documented invariants (consumer-308-before-AH-rewrite ordering, no redirect loops, every AH_ROOT_PATHS entry resolves). Add scripts/check-ah-root-sync.mjs to prebuild (only 2 of 5 lists are guarded today) so the next app/ah/ can't silently 404. Also remove the dead-and-harmful KNOWN_TOP_LEVEL supersets 'today','activity','cases' (ls-confirmed no such app dirs) that re-open HTTP-200 soft-404s on denialhelp.com, and extend check-top-level-sync.mjs to fail on extras.

Why it matters: Confirmed as the #1 riskiest module (incident-dense, zero test imports, only browser-tested by the live walker). The unguarded AH_ROOT_PATHS is the exact failure class that already happened twice, and the KNOWN_TOP_LEVEL supersets are live crawl-pollution on the DH host relevant to the SEO remediation.

16. Add targeted tests on the destructive/compliance/revenue seams and make them a release gate

🟠 **HIGH** · effort M · *Architecture / quality*

What to do: ~18 lib modules are tested against 270 API routes + 312 pages; the highest-blast-radius code is untested. Add fixture-based vitest (reuse billing.test.ts pattern, temp encrypted DBs, never prod data) in priority order: retention-purge (assert exactly-eligible rows purged, audit rows written, scraped-source medical_policies untouched, orphan-unlink accounting), proxy brand-routing tables, classify-reply corpus, analytics-gate x AH_ROOT_PATHS, email.ts provider selection, and the anomaly detector's persistAndDedupe/telegram batch (would have caught rank-6). Wire `npm test` into prebuild/deploy so the suite is load-bearing (there is no CI today), and port the self-running lib/savings/programs.test.ts (the Anti-Kickback copay-card guard, currently never executed) into vitest.

Why it matters: Confirmed: under ~10% of critical paths have any test and nothing enforces the ones that exist — a type-check-only `next build` is the de-facto release gate. For a two-brand PHI system the purge/billing/auth seams are exactly where a regression is catastrophic and silent.

17. Fix the DR failover secret set and vendor the untracked load-bearing ops scripts into git

🟠 **HIGH** · effort M · *Resilience / release engineering*

What to do: (a) tools/dr-cold-standby/dr-failover.sh:134-164 writes a 6-secret .env missing DH_CONSUMER_COOKIE_SECRET (consumer auth 500s), PRO_BRAND_HOSTS

(approvalhelp.com serves the DH site — a brand-separation break), INTERNAL_CRON_SECRET (all crons 500), ADMIN_AUDIT_TOKEN, and STRIPE_SECRET_KEY — and the drill smoke only checks /, /pricing, /api/health so a 'green' restore hides all of it; add the full required-secret list to the age-encrypted bundle and extend the drill smoke to hit a consumer-auth endpoint and a Host: approvalhelp.com request asserting AH branding. (b) deploy.sh, backup/restore scripts, appeals.service, and appeals-dr.service are NOT in git (verified) while 21 low-stakes cron units are meticulously tracked — vendor them into ops/ (secrets already live in untracked cred files, so tracking the scripts leaks nothing) and have backups emit machine-readable JSON status instead of the prose-regex cockpit currently parses.

Why it matters: Confirmed. The scripts whose failure loses the business (deploy, backup, restore) have no review, no history, and no git-rebuildability, and the DR runbook would come up green while consumer sign-in is down, AH serves the wrong brand, and no crons run.

Quick wins (small effort, high value)

- Error/warning text is near-invisible in the DEFAULT light theme across the paid funnel (HomeDropZone.tsx:286 text-red-200; IntakeBodyClient.tsx many amber-100/200/300; not-appealable/page.tsx) — one grep sweep to the paired 'text-red-700 dark:text-red-200' pattern that already exists in recover/page.tsx and HardshipApplyClient.tsx. High-value, purely mechanical.
- Homepage vertical cards render literal ''' HTML entities as visible garbage text (app/page.tsx:434,506,522,530,536 — incl. the IBD card title 'Crohn's') — replace with real apostrophes and add a lint grep for &[a-z]+; in VERTICALS.
- Intake progress bar shows 4 segments for a 5-step flow with Insurance/Clinical swapped, and mobile literally reads 'Step 5 of 4' (IntakeBodyClient.tsx:944-976, i18n.ts:1723) — render the real step count and correct label order.
- Remove the dead @anthropic-ai/bedrock-sdk production dependency and the isUsingBedrock()/mapModelId() no-op shim at its 10 call sites (keep the eslint ban as a tombstone; do NOT touch the opus-4-8 pin); fix the stale 'Bedrock Haiku' comment in outreach-enqueue/route.ts:22 (it actually uses callSubscription).
- Point the /pricing primary CTA at the drop zone / #choose instead of looping purchase-intent users back to the homepage top (pricing/page.tsx:83); make VerticalsBrowser 'Start ->' link straight to /intake?vertical={slug} instead of a second landing page.
- Delete 9 committed scratch/backup artifacts in scripts/ (*.bak-2026*, diag{,2,3}.mjs, check-appeals2.cjs, _c.cjs, _find.cjs) — enumerate explicitly and confirm none are referenced by systemd/cron before removing (no bulk rm in data dirs).

- Port the four legacy cron wrappers (run-reminders/policy-watch/mine-denials/rollback.sh) to the safe env-sourcing block already proven in run-internal-cron.sh — the old `export $(grep ... | xargs)` corrupts secrets containing spaces/metachars on the next rotation.
- Add the ALLOWED_EXTS allowlist from start-appeal to /api/intake uploads (reject SVG/zip into the PHI store) and derive the stored extension from the sniffed kind, not the attacker-controlled filename (intake/route.ts:220-244).
- Tighten lib/csrf.ts:30 to accept only sec-fetch-site='same-origin' (drop 'same-site', which trusts every current/future *.denialhelp.com subdomain) to match the strict proxy Origin/Referer gate.

Constraint checks & code-level violations

Places where the code itself breaks a load-bearing rule, or suggestions that were rejected for violating one.

- CODE VIOLATION (PHI default-closed): app/api/intake/route.ts:190-192,780 mints the dh_consumer ownership cookie against an unverified email and reuses the existing customer_id — cross-patient PHI account-takeover. Must be fixed before any promote (rank 1).
- CODE VIOLATION (PHI to external LLM without de-id): lib/ai/insurer-required.ts:349-353,430-441,534-541 sends patient denial-letter text direct to non-BAA Anthropic under a false 'zero-phi' assertion, ~20+ times per letter, in production, regardless of flags (rank 2).
- CODE VIOLATION (PHI default-closed): lib/analytics-gate.ts + proxy.ts URL-cloaking cause non-BAA GA4/PostHog to load and send pageviews (with patient/chart/appeal IDs) on clinical PHI surfaces — the gate's fail-closed AH branch is unreachable (rank 3).
- CODE VIOLATION (de-id before external LLM, partial): lib/ai/generation-pipeline.ts:79-108 places patient-evidence free text in the UN-redacted system prompt on 3 escalation endpoints and passes it as a process argv (rank 4a).
- CODE VIOLATION (FAIL LOUD): lib/ai/verifier.ts:134-137 (patient-facing safety gate fails open); app/api/internal/audit-anomaly/route.ts:30-31 (new alerts silently swallowed); lib/telegram-alert.ts (all alert failures silently dropped); app/intake/page.tsx:42 (bare /intake silently defaults to GLP-1); evening-digest returns ok:true on a failed Paubox send. (ranks 5,6,11,14).
- CODE VIOLATION (copy honesty): uncited 40-70% overturn stats on vertical pages (glp1/biologics/cancer.ts); i18n.ts:252 'we charge only when we deliver' contradicts the pay-then-generate flow; llms.txt states DenialHelp 'appeals are physician-reviewed and physician-signed' (rank 12). Also the DH AH-Enterprise-adjacent copy is out of scope here but flagged in the AH plan.
- CODE VIOLATION (AH-not-an-EHR, shared codebase, AH-facing): app/manifest.ts:14 (LIVE on prod) + app/layout.tsx:41,44 (staging) describe the product as a 'Lite-EHR platform',

contradicting llms.txt's own 'NOT an EHR'. This also creates a keep-branches-aligned / pre-promote-diff hazard: the compliant prod layout copy ('side-car to your EHR') would be REGRESSED by the next staging->master promote. Surface to Mic; belongs primarily to the ApprovalHelp plan but is a live violation on the shared platform.

- PHI-to-non-BAA risk (dormant until Tier 5): app/dashboard/[appealId]/AdsConversion.tsx loads the Google Ads gtag on a PHI dashboard whose ccm/collect ping carries the appeal UUID + payment value to non-BAA Google — fire the conversion from a neutral /checkout/thanks?t= page BEFORE OUTREACH_TIER_5_LIVE is ever flipped.
- GATES (not violations, but require Mic's explicit sign-off before shipping): (a) hardship consent-for-$0-tier change touches data-retention policy — flag, don't just ship; (b) all changes land on staging first and any promote to master needs Mic's approval; (c) the SEO edge-caching change MUST be scoped to public marketing paths only so no PHI/auth page is ever CDN-cached; (d) preserve the deliberate claude-opus-4-8 pin — any recommendation implying un-pinning (e.g. threading model roles) is accepted ONLY as an allow-list of already-pinned IDs, never a casual un-pin.
- No investigator recommendation was rejected for violating a constraint — all eight screened their own fixes (subscription-first routing, age-encrypted DR secrets, rate-limit keys are IP/UA hashes not PHI, medical_policies read-path source-priority enforces rather than overwrites the authoritative-source rule). The only guardrails to carry are the four gates above.

Also worth doing (lower priority, nothing dropped)

- Consolidate the fractured schema management (lib/db.ts): half the DDL now bypasses the version gate and runs ~220 statements + 261 PRAGMA probes on every cold start against the encrypted PHI DB, with failed ensureColumn swallowed by console.error (at odds with FAIL LOUD). Adopt one append-only migrations table applied in a transaction that fails loudly; freeze ensureSchemaInvariants; keep schema.sql as generated docs. (high value, L effort, velocity+integrity).
- Introduce thin typed per-entity query modules for the ~6 hottest tables (appeals, customers, pro_patients, pa_requests, pro_users, outreach_queue) with an eslint no-restricted-syntax ban on getDb() outside lib/ — today there are 1,777 raw prepare() sites, 92 pages querying the DB directly, and 111 unchecked `as unknown as` row casts that compound with schema drift. Migrate opportunistically. Natural enforcement point for the medical_policies source-filter rule.
- When the pending UI/UX audit lands, mechanically split the god client components on the conversion paths (IntakeBodyClient 2,845 lines/26 useState; the DH revenue funnel) by wizard step with a small reducer, and split the 2,343-line i18n DICT by namespace so client pages import only their slice — verified by the existing Playwright walker. Do WITH the UI audit, not before.

- Persist rate limiting (lib/rate-limit.ts is a per-process in-memory Map that resets on every deploy/restart — and the deploy webhook auto-restarts on every push) at least for the admin-login brute-force bucket; back it with the existing SQLite DB (keys are IP/UA hashes, no PHI).
- Harden the admin surface: dh_admin_token is the ADMIN_AUDIT_TOKEN verbatim (a captured cookie = the password for 30 days) with no per-operator identity, and rate-limit-bypass.ts reuses that same secret as its HMAC key — the exact entropy-domain entanglement consumer-auth.ts explicitly refuses. Move to per-session tokens + a dedicated RATE_LIMIT_BYPASS_SECRET.
- Lock down /api/whitelist: it self-serves a 1-year rate-limit-bypass cookie to anyone who guesses a whitelisted (likely founder) email with no session check — require an admin session or provision the cookie out-of-band, shorten the TTL, and audit_log issuance.
- Adopt a FAIL-LOUD catch convention + lint across the 411 bare catch blocks (require a [catch:site] tagged warn or a // silent-ok:): make audit() write failures alert via Sentry (nppes-verify.ts:205 swallows a failed audit_log INSERT), make lib/email.ts suppression-lookup fail CLOSED, and have retention-purge report orphan-unlink failures. Also make the advisory AI passes (strength/fact-check/adversarial/p2p/summary) observable — they all catch{}->null silently, so an outage zeroes the quality layer invisibly.
- Add a source-priority ORDER BY to lib/policies/lookup.ts tier queries (and pa-narrative) so re-seeded config_seed placeholder criteria can't outrank scraped/curated policies and get quoted 'verbatim' as the insurer's own words; suppress the verbatim-quote directive for config_seed rows. Directly enforces the medical_policies authoritative-source rule on the READ path.
- Gate auto_scrape policy auto-promotion (scrape-insurer-policy/route.ts, POLICY_AUTO_PROMOTE on by default with only length checks) on a verbatim-grounding substring check or tag auto_scrape rows to cite by title/URL rather than verbatim quotation until human-reviewed; move the inline claude-opus-4-7 model literal into MODELS and log promote failures.
- Wire the dead fact-check regen loop (lib/ai/fact-check.ts shouldRegenerate/buildCorrectionsBlock have zero callers): on high-severity value-transcription discrepancies run one synchronous correction pass or flip the appeal to a 'corrections suggested' state before letter_ready — highest-leverage letter-quality fix; today known HbA1c/prior-therapy errors ship.
- Thread opts.model through callPHILLM->callSubscription (restricted to already-pinned IDs, keeping opus-4-8) so the MODELS Haiku/Sonnet/Opus escalation ladder isn't a silent prod no-op; log the effective model into ai_call receipts; add a 2-3 spawn semaphore in subscription-client for batch intake on the small box. Or delete the dead escalation branch and document the collapse. Fix the misleading 'Sonnet' comments in scribe.ts/models.ts.
- Make the de-id short-identifier exemption class-aware: lib/de-id/redact.ts outboundGate (<=3 chars) and the sweep (<=2) skip genuine 2-3 char surnames (Ng/Li/Ho/Yu) and member-ID fragments — never exempt class NAME/MEMBER_ID/SSN/PHONE regardless of length; add a 2-char-surname de-id fixture.

- Fix the cross-brand PDF leak: lib/pdf/letterhead.ts defaults productName='DenialHelp' and 3 of 4 render paths (download/pdf, fax, /api/v1 pdf) pass no productName, so an AH practice can send insurers a letter footed 'Drafted with DenialHelp' — resolve brand on all renderLetterheadPdf/ buildAppealPacket call sites, or make the footer brand-required when a pro account is attached.
- Fix /fhir/connect (serves unrewritten on both hosts): hardcoded 'DenialHelp' title/body leaks the consumer brand onto the clinician EHR-connect page, and it lists identical Epic production/ sandbox URLs (sandbox silently targets production) — switch to getBrandIdentity() and fix the Epic sandbox iss.
- Add https://approvalhelp.com/ to the Lightsail gateway healthcheck (currently DH-only) with a brand-marker assertion so an AH-only outage or a PRO_BRAND_HOSTS regression serving DH content on the AH domain is caught — AH is the higher-ARPU brand and is edge-blind today.
- Lower DH_FAIL_THRESHOLD back to ~2 (~6 min) now that the .next.shadow swap serves old code during builds — it was widened to ~15 min for the pre-shadow-build era, so real outages now go unpaged for a quarter hour; use a /tmp/dh-deploying marker file for deploy-scoped suppression instead of a blanket threshold.
- Consolidate the two divergent systemd dirs (deploy/systemd 9 units vs ops/systemd-timers 12 units, both claiming to be source-of-truth, with a real diff on the audit-anomaly unit and 4 legacy prod timers untracked entirely) into one tracked dir + a drift-check script.
- Clarify the DH-branded patient-invite email (app/api/ah/patient/invite): it arrives from 'DenialHelp' with no explanation of who that is or why it's emailing on the practice's behalf — add one sentence mirroring the refer-to-consumer email, keeping the DH sender/Paubox path.
- Reduce hardship-application friction (HardshipApplyClient.tsx): drop the required phone, lower the 200-char stress-statement minimum, and decouple the $0-tier corpus-consent quid-pro-quo — but the consent change touches data-retention policy, so surface it to Mic before shipping (stays on staging).
- Improve the drop-zone error state (HomeDropZone.tsx): on OCR/classification failure keep the uploaded file and offer an inline condition-picker -> /intake?vertical=X&denial=Y (reusing the file already persisted by /api/start-appeal) instead of discarding it and sending the user to browse 86 conditions; add a mobile 'Take photo' affordance mirroring intake step 1.
- Rewrite the top-10-traffic homepage vertical card descriptions in patient language (they currently read as clinician shorthand: 'MHPAEA', 'KDIGO 2024 LN + AURORA-1/2', 'ASTRO Group 1 + ELIANA') — keep the guideline arsenals on the per-vertical landing pages; preserve copy honesty (no new guarantees).
- Bind audit-log/rate-limit client IP to the trusted Caddy hop via a shared-secret header (lib/client-ip.ts trusts spoofable X-Real-IP/X-Forwarded-For first-hop) so a spoofer can't poison the HIPAA access record or borrow rate-limit buckets — strengthens audit_log integrity without weakening it.

- AH-scoped items that touch the shared platform and are worth carrying to the AH plan: staging dev-signin/subscription gate fails OPEN on a single DEPLOY_ENV string (high — add independent locks); AH PATCH endpoints UPDATE by bare WHERE id after a tenant-scoped read (append AND pro_account_id=? to 5 statements); 15+ toLocaleDateString() calls without 'en-US' leak locale-dependent dates on AH surfaces and the build guard only catches type='date' (extend check-no-native-date-input.cjs); AH batch intake silently defaults failed classification to 'glp1'.

Appendix — full finding set by dimension

Architecture & Code Quality

This is a far better codebase than the solo-founder-speed-run baseline in the places that were deliberately hardened: the PHI/LLM compliance gate (client-factory + eslint AST bans + phi-subscription) is genuinely well-engineered, billing is crash-safe/idempotent with deterministic Stripe keys, essentially all 1,777 SQL call sites use prepared statements (the 34 template-literal cases are placeholder/allowlist builds, not injection), TS is strict + noUncheckedIndexedAccess, and there are build-time guards (top-level-sync, vertical-slug-sync, no-native-date-input). But the architecture is accreting faster than it is consolidating: schema management has fractured into three mechanisms with half the DDL now bypassing the version gate; proxy.ts is a 688-line untested choke point holding five hand-maintained path allowlists (only two guarded) whose own comments document four prior production incidents; test coverage is ~18 modules against 270 API routes + 312 pages; and two cross-cutting bugs were confirmed in source: (1) the HIPAA analytics gate's ApprovalHelp fail-closed branch is unreachable because the AH-host rewrites strip the /ah prefix from every browser URL the gate keys on — non-BAA GA4/PostHog will treat clinical surfaces as public; (2) the outreach reply classifier's precedence + substring matching systematically eats "interested" replies, a concrete mechanism for the observed ~0-interest outreach — and it exists as two drifted copies with zero tests. The three riskiest modules: proxy.ts (routing/brand/CSRF monolith, no tests), lib/db.ts (2,640-line schema god file with eroded versioning), and lib/analytics-gate.ts + lib/outreach/classify-reply.ts (small files, outsized blast radius, both broken).

- **HIPAA analytics gate defeated by AH URL-cloaking rewrites — non-BAA GA4/PostHog can run on clinical surfaces** — 🔴 CRITICAL · effort S · `/tmp/dhah-audit/apps/appeals/lib/analytics-gate.ts:14-38, proxy.ts:287-300, components/AnalyticsBootstrap.tsx:48-53, lib/outreach/ga4.tsx:59-61, app/layout.tsx:242-245`
  - Problem: The gate's own comment says 'Fail-closed for ApprovalHelp: only the known /ah marketing pages are public; every other /ah path is clinical PHI', and its AH branch is `if (path === "/ah" || path.startsWith("/ah/")) return AH_PUBLIC.has(path)`. But proxy.ts (AH_ROOT_PATHS block, lines 287-300) deliberately REWRITES approvalhelp.com/scribe, /patients, /chart/, /telehealth, /inbox, /pa, /appeals, /queue, /settings, /forms, /today etc. to /ah/* internally precisely so the browser URL never contains '/ah/'. Both analytics components are mounted in the ROOT layout and gate on usePathname(), which under a

middleware rewrite reflects the browser URL (documented Next.js behavior — runtime value unverifiable from source alone; one staging pageload confirms). So on every AH clinical page the '/ah' branch never fires, DENY_PREFIXES (which only lists consumer-shaped paths: /dashboard, /intake, /me, ...) misses /scribe, /patients, /chart, /telehealth, /inbox, /inbasket, /pa, /queue, /appeals, /eligibility, /schedule, /forms, /avs, /today, /cases, /activity — and isPublicAnalyticsPath returns TRUE: PostHog + GA4 load and fire pageviews (with full window.location.href incl. patient/chart resource IDs) on surfaces the code itself classifies as clinical PHI. The fail-closed design is inverted to fail-open by a sibling module's URL cloaking. This is also entangled with the 'traffic-blind' pain point: the gate is host-unaware, so DH vs AH measurement can't be reasoned about either.

  - Fix: Make the gate host-aware and truly fail-closed: in isPublicAnalyticsPath, when window.location.hostname (or a passed host) matches the AH host, allow ONLY an explicit allowlist of browser-visible marketing slugs ('/', '/pricing', '/about', '/contact', '/baa', '/terms', '/privacy', '/how-it-works', '/enterprise', '/help') and deny everything else; keep the existing /ah/* branch for the DH-host case. Add a vitest table test that enumerates every AH_ROOT_PATHS entry and asserts the gate denies it in bare (non-/ah) form. Verify with one devtools network capture on staging approvalhelp.com/scribe before and after.

- **Outreach reply classifier: precedence + substring bugs suppress 'interested' detection; drifted duplicate copy; zero tests** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/lib/outreach/classify-reply.ts:14-21, lib/outreach/inbox-poller.ts:110-140, app/api/webhook/instantly-reply/route.ts:116-122`

  - Problem: classifyReply checks classes in fixed order with bare substring regexes: (1) unsubscribe first, matching bare 'stop' with no word boundary — a prospect writing 'I want to stop wasting time on denials' (literally the product's pitch language) is classified unsubscribe_request and suppressed; (2) auto_reply third, with an enormous regex including `thank you for (contacting|your email|reaching)` and `this (mailbox|account|inbox)` — a genuinely interested human reply opening 'Thank you for reaching out — yes, send details' is classified auto_reply (kind 'ooo') and never surfaces as a lead; (3) 'interested' is checked LAST and includes bare 'yes' (matches 'eyes'/'yesterday'). The classifier also runs on `blob` = lowercased subject + Gmail snippet only (inbox-poller.ts:110), i.e. ~100 chars of body. This is a concrete, code-level mechanism for the '~0 real interest' symptom: interested replies are being routed to ooo/unsub buckets. Additionally the file's own header admits app/api/webhook/instantly-reply/route.ts:116 carries an inline DUPLICATE that has already drifted (its auto_reply regex is the old 5-term version), and no test file imports classifyReply.

  - Fix: Rewrite with word boundaries (\bstop\b, \byes\b), score-based classification instead of first-match precedence (an 'interested' signal should outrank a generic 'thank you for reaching' auto-reply signal), and classify on the full message body where available, not subject+snippet. Delete the drifted inline copy in instantly-reply/route.ts and import the lib version. Before/after: replay the existing outreach_replies rows (kind column already persisted, inbox-poller.ts:144-155) through the new classifier and diff — that both validates

the fix and recovers historical misfiled interested leads. Add tests/classify-reply.test.ts with a corpus from real replies.

- **proxy.ts is a 688-line untested routing/CSRF/brand monolith with 5 hand-maintained path lists, only 2 guarded** — 🟠 HIGH · effort M · `/tmp/dhah-audit/apps/appeals/proxy.ts (CONSUMER_ONLY_PATH_PREFIXES:82-141, CLINICIAN_ONLY_PATH_PREFIXES:146-178, AH_ROOT_PATHS:287-294, KNOWN_TOP_LEVEL:346-356, KNOWN_VERTICALS:358-382)`
  - Problem: One file interleaves brand host routing, per-brand rewrites/redirects, soft-404 interception, CSRF Origin/Referer enforcement, CSP-Report-Only nonces, admin gating, attribution cookies, and admin session refresh. Its own comments document at least four production incidents caused by exactly this structure: '2026-06-04 brand-leak closure — 11 consumer subbuckets that were 404ing on AH', the 'ORDER-OF-OPS FIX 2026-06-04' (consumer 308 had to run before AH_ROOT_PATHS rewrite or /help soft-404'd), '2026-06-10: was trapping DH visitors on an AH skeleton', and the 2026-05-20 Sec-Fetch-Site CSRF bypass. Of the five hand-maintained lists, only KNOWN_TOP_LEVEL and KNOWN_VERTICALS have prebuild sync guards (scripts/check-top-level-sync.mjs, check-vertical-slugs-sync.mjs). AH_ROOT_PATHS has NO guard: the next app/ah/ feature ships and silently 404s at approvalhelp.com/ (first segment won't be in KNOWN_TOP_LEVEL either) — the exact failure class that already happened twice. And despite being the most incident-dense file in the app, no test imports any of it (grep of tests/ for brandRewrite/isKnownRoute: zero hits) — brandRewrite ordering is only exercised by the live Playwright walker.
  - Fix: Extract the pure logic (brandFor, brandRewrite, isKnownRoute + the five lists) into lib/routing/ so it is importable without NextRequest scaffolding; add vitest table tests asserting the documented invariants (consumer-308-before-AH-rewrite ordering, no redirect loops when envs are misconfigured, every AH_ROOT_PATHS entry resolves, /clinicians alias). Add scripts/check-ah-root-sync.mjs to prebuild mirroring check-top-level-sync.mjs (app/ah/* dirs ⊆ AH_ROOT_PATHS ∪ PRO_REWRITES targets ∪ CONSUMER_ONLY list). This is the #1 riskiest module.
- **Schema management has fractured: half of all DDL now bypasses the version gate and runs on every cold start** — 🟠 HIGH · effort L · `/tmp/dhah-audit/apps/appeals/lib/db.ts:26-41 (SCHEMA_VERSION log), :90-1344 (ensureSchemaInvariants), :1345-2634 (runMigrations), lib/schema.sql (1,589 lines)`
  - Problem: Schema truth is split across four mechanisms: schema.sql (76 tables), version-gated runMigrations() (243 DDL/ensureColumn statements), always-run ensureSchemaInvariants() (221 statements, lines 90-1344 — ~1,250 lines), and always-run ensureColumn calls. ensureSchemaInvariants was documented as being for 'OBSERVABILITY tables (cron_runs, etc.)' but the SCHEMA_VERSION=60 comment shows it has become the default landing zone for real product schema (pro_tasks, med_flags, ai_handling_receipt, experiment_exposures, ad_spend, pro_tier_3_signals) explicitly because the team no longer trusts the version gate ('so a stuck user_version can never leave it missing and 500 the tasks API' — repeated

verbatim for three different features). The version-history comment block itself has gaps (entries 54-56 are missing between 57 and 53). Consequences: every cold start executes ~220 DDL statements + 261 PRAGMA table_info probes against an encrypted PHI database; there is no single place to read 'what is the schema'; and a failed ensureColumn is swallowed by console.error (db.ts:21-23), so partial schema application is survivable-but-silent — at odds with FAIL LOUD. lib/db.ts (2,640 lines) is the #2 riskiest module.

  - Fix: Adopt one boring mechanism: a migrations table with sequential, append-only migration files (id, applied_at), applied in a transaction at startup, failing loudly on error — better-sqlite3 makes this ~50 lines. Freeze ensureSchemaInvariants (no new entries; lint-comment the function) and fold its contents into migration 61 as a one-time consolidation. Keep schema.sql as generated documentation (sqlite3 .schema dump in CI) rather than an execution path. Do it incrementally on staging; the always-run block stays until the migration table has run once on prod.

- **Test coverage reality: ~18 lib modules tested against 270 API routes + 312 pages; PHI-deleting and routing code untested** — 🟠 HIGH · effort M · `/tmp/dhah-audit/apps/appeals/tests/ (23 files, 1,662 lines), vitest.config.ts`
  - Problem: The suite is well-aimed where it exists (billing idempotency, de-id, ai-receipt hash chain, compliance-gate, rate-limit, insurer canon) but imports only ~18 modules out of ~150 in lib/ + lib/ai + lib/pro. Completely untested high-blast-radius code: proxy.ts brand routing (see separate finding), app/api/internal/retention-purge/route.ts (727 lines that DELETE appeals, unlink signed PDFs, and NULL PHI columns — a bug here destroys customer data or violates retention), lib/outreach/classify-reply.ts (revenue), lib/analytics-gate.ts (compliance), lib/email.ts provider gating (1,248 lines), lib/smart-fhir/writeback.ts (1,418 lines writing into external EHRs), and lib/db.ts migrations. The vitest config is node-env only — nothing exercises even pure client logic. Meanwhile runtime confidence rests on the Playwright walker/audit harness, which is good at 'pages render' and weak at 'retention purge deleted exactly the right rows'.
  - Fix: Do not chase coverage %; add targeted tests in priority order: (1) retention-purge against a temp encrypted SQLite fixture — assert exactly-eligible rows purged, audit rows written, scraped-source medical_policies untouched; (2) proxy routing tables; (3) classify-reply corpus; (4) analytics-gate x AH_ROOT_PATHS product; (5) email.ts provider selection (paubox/log/refuse matrix, suppression). Each is a half-day with the existing fixture pattern from billing.test.ts.

- **No data-access layer: 1,777 raw prepare() sites, 92 pages querying the DB directly, hand-typed row casts with no validation** — 🟡 MEDIUM · effort L · `/tmp/dhah-audit/apps/appeals/lib + app (grep: 1,777 .prepare/.exec sites; 92 page.tsx files call getDb(); 91` SELECT * FROM `; 111` as unknown as `casts)`
  - Problem: Every query result is shaped by an unchecked TypeScript assertion (`.get(...) as { n: number }`), so a renamed/retyped column produces undefined at runtime, not a

compile error — particularly dangerous combined with the fractured migration system above (the two failure modes compound: schema drifts, casts lie). 91 `SELECT *` sites couple pages to full row shapes. Server components embed multi-hundred-line SQL inline (e.g. app/admin/outreach-health/page.tsx builds WHERE fragments via brandClause string concat — safe but pattern-fragile; app/ah/dashboard/page.tsx, app/ah/today/page.tsx each run 5+ inline queries). This is the single biggest drag on future velocity: any schema change requires grepping 1,777 sites.

  - Fix: Not an ORM. Introduce thin per-entity query modules for the ~6 hottest tables (appeals, customers, pro_patients, pa_requests, pro_users/accounts, outreach_queue): typed row interfaces defined once next to a zod (already a dependency) or manual runtime guard used in dev/test only, and named query functions. Migrate opportunistically — new code must use them (eslint no-restricted-syntax on getDb() outside lib/, mirroring the existing anthropic-sdk ban pattern), old code migrates when touched.

- **God client components on the conversion paths: IntakeBodyClient 2,845 lines/26 useState; ReviewClient 1,770 lines/37 useState** — 🟡 MEDIUM · effort L · `/tmp/dhah-audit/apps/appeals/app/intake/IntakeBodyClient.tsx (2,845 lines), app/ah/intake/[appealId]/ReviewClient.tsx (1,770 lines), app/ah/telehealth/TelehealthCall.tsx (1,067), app/ah/chart/[id]/VisitActions.tsx (1,159), app/ah/pa/new/PaIntakeWizard.tsx (1,194)`
  - Problem: The DH consumer intake — the revenue funnel — is one client component with 26 useState hooks, inline validation, prefill merging, i18n, scroll orchestration, and upload handling in a single closure scope; ReviewClient (AH) is worse per-line with 37 useState. Any change (and the pending UI/UX audit will demand many) risks regressions across unrelated steps because all state is flat and co-mutated; there is no test coverage on either (node-only vitest). These files also pin the whole 2,343-line lib/i18n.ts DICT into the client bundle (single Record object, not tree-shakeable; imported by 73 files).
  - Fix: When the UI/UX audit lands (it is pending — do this WITH that work, not before): split IntakeBodyClient by wizard step into components owning their slice of state, with a small reducer for cross-step state; extract the UniversalFacts prefill merge into a pure, unit-testable lib function. Split i18n DICT by namespace (nav/hero/intake/...) so client pages import only their slice. Do not redesign behavior — mechanical extraction only, verified by the existing Playwright walker.

- **US date-format discipline is leaking on AH surfaces despite two dedicated modules and a build guard** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/app/ah/patients/[id]/page.tsx:322-420 (7 sites), app/ah/pa/page.tsx:122, app/ah/appeals/AppealsWorkspace.tsx:620, app/ah/appeals/[id]/Detail.tsx:228, app/ah/avs/new/`

`UsDateField.tsx vs components/USDateInput.tsx, lib/format.ts vs lib/format-date.ts`

  - Problem: lib/format-date.ts exists specifically because 'Tester feedback flagged inconsistent dates across pages' and standardizes MM/DD/YYYY — yet 15+ bare `new Date(x).toLocaleDateString()` calls (no 'en-US' locale) remain on AH clinical surfaces. In client components (AppealsWorkspace, Detail, pa list) these render in the VIEWER's browser locale — dd/mm/yyyy for any non-US-locale browser — exactly the bug class the house rule bans; in server components they silently depend on the server locale. The prebuild guard (check-no-native-date-input.cjs) catches `type="date"` but not toLocaleDateString, so the regression class is unguarded. There is also duplication: app/ah/avs/new/UsDateField.tsx (47 lines) reimplements components/USDateInput.tsx (68 lines), and lib/format.ts (formatUSDate) overlaps lib/format-date.ts (formatDate) with different fallbacks and timezone handling.
  - Fix: Sweep the 15 sites to formatDate() from lib/format-date.ts (one mechanical PR); delete UsDateField.tsx in favor of USDateInput; pick one format module (format-date.ts is the more robust — its toDate handles bare SQLite UTC strings) and re-export formatUSDate from it. Extend check-no-native-date-input.cjs to also fail on `toLocaleDateString()` called without an explicit 'en-US' argument in app/ and components/.

- **Silent-catch sprawl (411 sites) including audit-log writes and email-suppression checks — FAIL LOUD partially eroded** — 🟡 MEDIUM · effort M · `/tmp/dhah-audit/apps/appeals (411 bare` catch {} `/` catch { `blocks; lib/pro/nppes-verify.ts:205; lib/email.ts:224-232; app/api/internal/retention-purge/route.ts:464`
  - Problem: Error handling is a mixed regime: 83 sites use the good tagged pattern (`console.warn("[catch:file]", ...)`) but 411 catch blocks are empty or comment-only. Three verified consequential cases: (1) lib/pro/nppes-verify.ts:205 `catch { /* audit is best-effort */ }` swallows a failed audit_log INSERT — an audit write silently failing is in tension with the audit-integrity red-line (the log should be reliable, or its unreliability should page someone); (2) lib/email.ts:230 the suppression-table lookup fails OPEN ('Suppression table may not exist on a fresh DB; fall through') — on a DB error the send proceeds to a possibly-suppressed address (deprecated SES path only, but the pattern is the point); (3) retention-purge:464 unlink failures are best-effort with no orphan-file accounting, so signed PDFs can survive their DB rows on the LUKS volume with nothing reporting it.
  - Fix: Adopt a convention + lint: bare catch requires either a `[catch:site]` tagged warn or a `// silent-ok: <reason>` comment (enforceable with no-restricted-syntax like the existing PHI-log rule). Specifically: make audit() failures alert (Sentry captureException — Sentry is BAA'd per existing sentry-scrub setup) rather than warn; make the suppression lookup fail CLOSED (skip send, return suppressed_check_failed); have retention-purge count and report unlink failures in its response payload so the cron digest surfaces orphans.

- **AH PATCH endpoints update by bare `WHERE id = ?` after a tenant-scoped read — defense-in-depth gap in the multi-tenant write path** — 🟡 MEDIUM · effort S · `/tmp/dhah-`

`audit/apps/appeals/app/api/ah/patients/[id]/route.ts:155, lib/pro/patients.ts:289, app/api/ah/forms/[id]/route.ts:104, app/api/ah/avs/[id]/route.ts:75, app/api/ah/referrals/[id]/route.ts:80`

  - Problem: The pattern is consistent across five AH write endpoints: a SELECT correctly scoped by `id = ? AND pro_account_id = ?` (e.g. patients route.ts:109-111), then the dynamic UPDATE runs with only `WHERE id = ?`. Column names come from constant allowlists and values are parameterized (no injection — verified), and the read-then-write gap is a narrow TOCTOU, so this is not currently exploitable in an obvious way. But it means tenant isolation on writes depends entirely on the preceding read staying in sync as the code evolves — one refactor that reorders or removes the existence check silently becomes a cross-tenant write primitive on PHI tables (pro_patients, pro_forms, pro_avs, pro_referrals).
  - Fix: Append `AND pro_account_id = ?` to all five UPDATE statements and assert `info.changes === 1` (return 404/409 otherwise) — five one-line changes plus params. Make 'writes carry the tenant predicate' a stated convention in CLAUDE.md so new endpoints copy the safe shape.

- **KNOWN_TOP_LEVEL superset entries (today/activity/cases) re-open the HTTP-200 soft-404 hole on the DH host** — ⚪ LOW · effort S · `/tmp/dhah-audit/apps/appeals/proxy.ts:346-356, scripts/check-top-level-sync.mjs`
  - Problem: KNOWN_TOP_LEVEL contains 'today', 'activity', 'cases' (and 'pro'), but app/ has no such top-level directories (verified: ls fails for all four) — they exist only as app/ah/* served via AH-host rewrites, which return before the soft-404 check ever runs on the AH host. So the entries are dead on AH and harmful on DH: denialhelp.com/today, /activity, /cases bypass the proxy's 404 short-circuit and render not-found.tsx with HTTP 200 — precisely the crawl-pollution the allowlist was built to eliminate (per the SOFT-404 FIX comment block, and relevant given SEO is a live remediation project). ('pro' is safe: next.config.ts:71-72 308s it before route matching.) check-top-level-sync.mjs validates only one direction (app dirs ⊆ list), so supersets accumulate silently.
  - Fix: Remove 'today', 'activity', 'cases' from KNOWN_TOP_LEVEL (AH-host requests never reach isKnownRoute for these paths — brandRewrite returns first). Extend check-top-level-sync.mjs to fail on list entries with no corresponding app/ dir, next.config redirect source, or documented exception, closing both drift directions.

- **Dead Bedrock dependency + isUsingBedrock()/mapModelId() shim threaded through 10 call sites; committed scratch files in scripts/** — ⚪ LOW · effort S · `/tmp/dhah-audit/apps/appeals/package.json:27 (@anthropic-ai/bedrock-sdk), lib/ai/models.ts:18-20, lib/ai/client-factory.ts:150-152, 10 call sites (start-appeal, clinical-extract, extract, prefill, pa-clinical-extract, vertical-classify, pa-narrative, generate`

`x2, phi-subscription); scripts/*.bak-2026*, diag.mjs/diag2.mjs/diag3.mjs, check-appeals2.cjs, _c.cjs, _find.cjs`

- Problem: AWS Bedrock was 'permanently denied 2026-05-21' (models.ts comment, matching the Bedrock-HARD-NO decision) yet @anthropic-ai/bedrock-sdk@^0.21.2 remains in dependencies — installed on the PHI server, and its only other reference is the eslint ban of itself. mapModelId(model, isUsingBedrock()) is a no-op identity shim invoked at 10 sites, each one a reader-confusion tax and a place where a future 'cleanup' could resurrect a branch that must stay dead. Both files' comments already say 'follow-up commit can rip the calls out' — that commit never happened. Separately, scripts/ carries 9 committed scratch/backup artifacts (*.bak-20260609, *.bak-20260610, diag{,2,3}.mjs, check-appeals2.cjs, _c.cjs, _find.cjs) — 13% of the directory — which obscure which scripts are load-bearing cron entry points (run-internal-cron.sh etc.).
- Fix: Remove the bedrock-sdk dependency (keep the eslint no-restricted-imports entry as a tombstone — it works without the package being installed); delete mapModelId/isUsingBedrock and pass MODELS. directly at the 10 sites (mechanical, type-checked). Delete or move scratch files to an untracked ops/scratch/ (respecting the no-bulk-rm rule: enumerate the 9 files explicitly, confirm none referenced by systemd/cron units before deleting).

SECURITY: auth & access control

The consumer-appeal and Pro/clinician surfaces are, on the whole, defended by a security-conscious author: state-changing routes are gated by verifySameOrigin + a middleware-level Origin allowlist (proxy.ts is correctly wired as the Next 16 `proxy` convention, verified), Stripe webhooks verify signatures and re-derive the price floor from the trusted DB row, SMART-FHIR and outbound webhooks have real SSRF allowlists with DNS-rebinding re-checks, API keys and Pro session/magic-link tokens are stored hashed, file uploads are magic-byte sniffed, and internal cron routes use timing-safe Bearer auth. However, there is ONE critical flaw that defeats the entire consumer ownership model: /api/intake mints the `dh_consumer` ownership cookie against an unverified email and reuses the pre-existing customer_id when the email already exists — so anyone who knows a victim's email can obtain a session that owns all of that victim's prior appeals (full PHI, refund, delete), bypassing the verified magic-link claim path that exists precisely for this. Beyond that, several defenses are fragile: staging auth bypasses fail open on a single env string, the rate-limit system is per-process in-memory and self-servingly bypassable via an unauthenticated endpoint, and the whole admin surface is one shared static secret reused across entropy domains. The critical intake issue should be treated as a HIPAA-reportable exposure risk and fixed first.

- **Intake mints the ownership cookie for an UNVERIFIED email and reuses the existing customer_id — cross-patient PHI account takeover** — 🔴 CRITICAL · effort M · `apps/appeals/app/api/intake/route.ts:188-206 and :780; amplified by apps/appeals/app/`

`me/refund/page.tsx:37-49 and apps/appeals/app/api/me/refund/route.ts; gate defined in apps/appeals/lib/consumer-auth.ts:181-206`

  - Problem: The consumer ownership model (ownsAppeal) treats the dh_consumer cookie as proof that the bearer owns every appeal whose customer_id matches. /api/intake looks up an existing customer purely by email (`SELECT id FROM customers WHERE email = ?`, line 190), reuses that customer_id if found (`const customerId = existing?.id ?? randomUUID()`, line 192), and at the end unconditionally calls `await setConsumerCookie(customerId)` (line 780) — with NO verification that the submitter controls that email. An attacker submits one anonymous intake using a victim's email (plus any throwaway denial file that passes the magic-byte sniff) and receives a valid HMAC-signed dh_consumer cookie bound to the victim's customer_id. That cookie now satisfies ownsAppeal for ALL of the victim's prior appeals. /me/refund then lists every one of the victim's appeals with id + insurer + drug_requested (the condition/drug being appealed — PHI-adjacent) via `WHERE a.customer_id = ? .all(customerId)`, handing the attacker the victim's appeal UUIDs; from there /api/appeal/[id]/download, /download/packet, and /status disclose the full appeal letter, denial docs and clinical narrative, and /api/me/refund + /me/delete let the attacker refund or destroy the victim's paid appeals. This is unauthorized PHI disclosure achievable with only a known email and a single unauthenticated POST — the magic-link claim flow (verifyClaimToken, emailed to the real customer) exists exactly to prevent this, and intake short-circuits it.
  - Fix: Never bind the dh_consumer cookie to a pre-existing customer identity on an unverified intake. Either (a) always create a fresh customer row for a new intake and only merge into the canonical customer after email verification via the existing makeClaimToken/magic-link path, or (b) scope the cookie set at intake to grant ownership of ONLY the newly-created appeal (e.g. a per-appeal signed grant) and require the emailed magic link for any sibling/prior appeal. Keep makeClaimToken as the sole cross-appeal ownership path. Add an audit_log entry on any intake that resolves to an existing customer_id.

- **Staging auth bypasses fail OPEN on a single env string (dev-signin + subscription gate) — violates FAIL LOUD** — 🟠 HIGH · effort S · `apps/appeals/lib/pro/auth.ts:304-308 (isSubscriptionActive) and :318-343 (devSignInSession); apps/appeals/app/api/ah/auth/dev-signin/route.ts:22-24,54`
  - Problem: devSignInSession mints a full Pro session for ANY pro account looked up by email with no magic link, and isSubscriptionActive returns true unconditionally — both gated only by `process.env.DEPLOY_ENV === 'staging'`. The security rests entirely on prod having DEPLOY_ENV unset. This is a permissive fail-open: the guard grants access when the string matches and the safe state depends on the *absence* of a value. If DEPLOY_ENV="staging" ever reaches prod (env copy-paste from the staging unit file, a templated deploy, a shell export), the result is arbitrary Pro-account takeover by email (GET/POST /api/ah/auth/dev-signin?email=) plus a global billing/subscription bypass, with no second lock. It also inverts the codebase's own FAIL-LOUD principle.

- Fix: Add an independent positive assertion that prod is NOT active before honoring the bypass (require DEPLOY_ENV==="staging" AND an explicit AH_ALLOW_DEV_SIGNIN==="1" AND NODE_ENV!=="production"-with-a-known-staging-host), and hard-crash on boot if DEPLOY_ENV==="staging" is seen together with a production hostname. Mirror the walker/bootstrap route's three-independent-locks pattern (apps/appeals/app/api/walker/bootstrap/route.ts:22-26).

- **Unauthenticated /api/whitelist self-serves a permanent rate-limit-bypass cookie for any guessable whitelisted email** — 🟡 MEDIUM · effort S · `apps/appeals/app/api/whitelist/route.ts:18-49; apps/appeals/lib/rate-limit-bypass.ts:31-72`
  - Problem: GET /api/whitelist?email= is unauthenticated: its only check is `isEmailWhitelisted(email)` against the WHITELIST_EMAILS env list (rate-limit-bypass.ts:40-43). On a hit it sets a 1-year dh_unlimited cookie that makes shouldBypassRateLimit() return true, disabling EVERY rate limit for that browser (ask, generate, start-appeal, checkout, intake, delete-request, admin-login anon buckets, etc.). The whitelisted address is almost certainly a guessable founder email (e.g. mic@denialhelp.com). An attacker who guesses it gets an unforgeable-but-freely-minted bypass cookie and can then run up Anthropic/Stripe/OCR cost and brute-force without any ceiling. The endpoint hands out the credential rather than requiring one.
  - Fix: Require an authenticated admin session (adminCookieMatches against ADMIN_AUDIT_TOKEN) before minting the bypass cookie, or drop the self-serve HTTP mint entirely and provision the cookie out-of-band. At minimum shorten the cookie TTL from 365 days and log issuance to audit_log.

- **Rate limiting is per-process in-memory only — resets on every restart/deploy and cannot cover multi-process; weakens admin brute-force + cost ceilings** — 🟡 MEDIUM · effort M · `apps/appeals/lib/rate-limit.ts:1-76; relied on by apps/appeals/app/api/admin/audit/login/route.ts:37 and all expensive LLM routes`
  - Problem: checkRateLimit stores buckets in a module-level Map with no persistence. The admin-login brute-force defense (10 attempts / 10 min, admin/audit/login/route.ts:37) and every cost-control ceiling therefore evaporate whenever appeals.service restarts — and the deploy webhook auto-restarts on every push (per the deploy notes), so an attacker who can trigger or simply wait out a restart resets the counter. It also silently fails to enforce anything if the service is ever scaled beyond one process. For a single-secret admin login (see next finding), an unbounded-across-restarts guess loop is the realistic threat.
  - Fix: Back the limiter with the existing SQLite DB (a rate_limit_hits table keyed by bucket+window) or a persisted store so counters survive restarts and are shared across processes; keep the in-memory Map as an L1 cache. The comment already flags 'swap to Redis' — at minimum persist the admin-login bucket.

- **Consumer ownership cookie is unrevocable and never expires server-side; signs only customer_id with no exp/nonce** — 🟡 MEDIUM · effort M · `apps/appeals/lib/consumer-auth.ts:40-56 (sign/makeConsumerCookieValue) and :59-68 (verify)`
  - Problem: makeConsumerCookieValue(customerId) = `${customerId}.${HMAC(customerId)}` — the signed payload is ONLY the customer_id, with no expiry, no nonce, and no server-side session record. verifyConsumerCookieValue accepts any correctly-signed customer_id forever. The 365-day maxAge is client-side only; the server has no way to revoke a specific cookie. A forwarded recovery/magic link, a shared/kiosk device, or a leaked cookie grants perpetual access to that customer's PHI, and the only revocation lever is rotating DH_CONSUMER_COOKIE_SECRET, which logs out every consumer at once. This is the design weakness that makes finding #1 so damaging.
  - Fix: Embed an expiry and a per-customer token version into the signed payload (e.g. `customerId.version.exp.sig`) and store a bumpable `cookie_version` on the customer row so a single account can be logged out / revoked (e.g. after a delete, refund dispute, or suspected takeover) without a global secret rotation. Verify exp and version on every read.
- **Admin surface is a single shared static token used verbatim as the cookie value, and that token is reused as the rate-limit-bypass HMAC key (entropy-domain reuse the codebase itself warns against)** — 🟡 MEDIUM · effort M · `apps/appeals/lib/admin-auth.ts:11-18; apps/appeals/proxy.ts:518-540 (refreshAdminSession) and :590-601 (enforceAdminGate); apps/appeals/lib/rate-limit-bypass.ts:22-25`
  - Problem: The entire /admin + /cockpit surface authenticates with one shared secret whose cookie value IS the secret verbatim (dh_admin_token == ADMIN_AUDIT_TOKEN). There is no per-user identity, no per-session token, and no revocation short of rotating the env var (which refreshAdminSession re-stamps to a 30-day sliding cookie, so a captured cookie is effectively a captured password valid for 30 days of inactivity). Separately, rate-limit-bypass.ts:22-24 reuses ADMIN_AUDIT_TOKEN (falling back to INTERNAL_CRON_SECRET) as the HMAC key for bypass cookies — the exact entropy-domain entanglement that consumer-auth.ts:29-37 explicitly refuses to do ('The two secrets must be independent'). A leak of the bypass HMAC context or the admin token cross-contaminates both domains, and admin audit trails cannot attribute actions to a person.
  - Fix: Move admin auth to a per-session token (random value stored server-side, mapped to an operator identity) rather than echoing the env secret as the cookie; give rate-limit-bypass its own dedicated secret (RATE_LIMIT_BYPASS_SECRET) instead of reusing ADMIN_AUDIT_TOKEN/INTERNAL_CRON_SECRET, consistent with the isolation consumer-auth already enforces.
- **Intake upload has no extension allowlist (accepts SVG/zip/text) and kindMatches is satisfiable by client-supplied MIME alone; raw filename extension is used in the stored path** — ⚪ LOW · effort S · `apps/appeals/app/api/intake/route.ts:220-244; apps/`

`appeals/lib/magic-bytes.ts:72-98; contrast apps/appeals/app/api/start-appeal/route.ts:28 (ALLOWED_EXTS)`

  - Problem: start-appeal restricts uploads to {pdf,png,jpg,jpeg} via ALLOWED_EXTS, but /api/intake has no such allowlist — it accepts anything sniffKind can classify, including SVG, generic ZIP, and plain text. kindMatches (magic-bytes.ts:72-98) returns true if EITHER the extension OR the client-controlled MIME matches, so an attacker can always satisfy the 'content matches label' check by setting file.type. The extension used to persist the file (`ext = file.name.split('.').pop()`, then `filename = ${fileId}.${ext}`) is taken raw from the attacker-controlled filename. Upward path traversal is effectively neutralized (splitting on '.' breaks up any '..' sequence and mkdir is non-recursive), so this is not RCE — but accepting active content types like SVG (which can carry script) into the PHI store, and persisting an unvalidated extension segment, is a needless hardening gap versus the stricter sibling route.
  - Fix: Apply the same ALLOWED_EXTS allowlist as start-appeal in intake, reject SVG/zip for denial uploads, derive the stored extension from the *sniffed* kind (not the filename), and sanitize the extension to /^[a-z0-9]{1,5}$/.

- **In-route CSRF check accepts Sec-Fetch-Site=same-site, widening trust to every *.denialhelp.com* / *.approvalhelp.com subdomain* — ⚪ LOW · effort S · `apps/appeals/lib/csrf.ts:27-43`
  - Problem: verifySameOrigin() (the belt-and-braces check some consumer mutation routes call in addition to the proxy gate) returns null (allow) when sec-fetch-site is 'same-origin' OR 'same-site'. 'same-site' includes all subdomains, and ALLOWED_HOSTS additionally trusts staging.denialhelp.com, partners.denialhelp.com, staging.approvalhelp.com, etc. So a page hosted on any current or future sibling subdomain (e.g. a compromised marketing/partners host) can drive state-changing requests that this in-route check would wave through. The primary proxy.ts CSRF gate uses a stricter exact-origin allowlist, so this is defense-in-depth rather than the sole control, but the looser leg means the two layers disagree.
  - Fix: In verifySameOrigin, accept only sec-fetch-site='same-origin' (drop 'same-site'), matching the strict exact-host Origin/Referer check the proxy already enforces, so a hostile subdomain can't be treated as first-party.

- **Audit-log + rate-limit IP is derived from client-spoofable X-Real-IP / X-Forwarded-For first-hop** — ⚪ LOW · effort M · `apps/appeals/lib/client-ip.ts:28-40`
  - Problem: clientIp() trusts X-Real-IP first, then the first hop of X-Forwarded-For, then cf-connecting-ip. These are authoritative ONLY if the fronting proxy (Caddy) overrides/strips any client-supplied copy. Any request path that reaches Node without traversing that header-sanitizing hop — direct hit to the Node port on the LAN, or a proxy layer that forwards rather than replaces X-Real-IP — lets a caller forge the value. That value is written into audit_log (a stated red-line for integrity) as the actor IP and is used as the rate-limit bucket key, so a spoofer can poison the HIPAA access record and evade or borrow rate-limit buckets. Whether

Caddy actually overrides X-Real-IP on every path is a config property not present in this repo (unverifiable from source), but the code's trust order is the observable risk.

- Fix: Have Caddy inject a request over a shared-secret header (e.g. X-DH-Edge-Token) and, in client-ip.ts, only trust the forwarded IP headers when that token is present and valid; otherwise record the raw socket peer. This binds IP trust to the known proxy hop and keeps audit_log values trustworthy.

PHI SAFETY + AI PIPELINE QUALITY

The de-id architecture is genuinely strong at its core — redact→outboundGate→subscription→rehydrate in lib/ai/phi-subscription.ts is well designed, ESLint bans direct Anthropic SDK use outside lib/ai/, vision fallbacks fail closed, audit ai_call writes exist at the real egress point, and the criteria-evaluator + verifier + argument-engine grounding stack is far above typical letter-generator quality. But the perimeter has real holes: lib/ai/insurer-required.ts sends sections of the patient's own denial letter DIRECT to non-BAA Anthropic ~20+ times per letter under a false "zero-phi" assertion (same bug class already fixed in mine-denials on 2026-06-10); the shared generation pipeline for L2/external-review/NSA-IDR moved patient-evidence free text into the un-redacted system prompt, which is also passed as a process argv; the blocking pre-delivery verifier silently fails OPEN when its LLM call errors; the rehydration-completeness check (rehydrateOrThrow) is dead code so dropped names/member IDs ship silently; and the main generator's non-DEID fallback lacks the production fail-closed guard that safePHITextCall has. On quality: the model-role ladder (Haiku/Sonnet/Opus escalation) is a silent no-op under subscription routing, the fact-check regen loop is dead code, and letters are forced to quote coverage_criteria "verbatim" even when the matched row is a config_seed placeholder or unverified auto_scrape text because lookupPolicy ignores the source column's trust hierarchy. Fix the insurer-required leak and verifier fail-open first — both are small diffs with outsized compliance/quality payoff.

- **insurer-required.ts sends patient denial-letter text DIRECT to non-BAA Anthropic under false "zero-phi" assertion** — 🔴 CRITICAL · effort S · `/tmp/dhah-audit/apps/appeals/lib/ai/insurer-required.ts:349-353, 430-441 (aiExtract), 534-541 (verifyFlag), 579-615 (extractInsurerRequirements); caller /tmp/dhah-audit/apps/appeals/lib/ai/prefill.ts:429`
  - Problem: getClient() asserts `allowDirect: "zero-phi"` with a comment claiming it works on "PUBLIC INSURER POLICY text ... not patient PHI" — but the actual input is the patient's own denial letter: prefill.ts pass 5 calls extractInsurerRequirements(text) where text is the extracted denial-document text, and locateAppealSection() slices up to 4,000 chars starting 50 chars BEFORE the appeal-rights header (line 159: `Math.max(0, best.idx - 50)`). Real denial letters carry patient name / member ID / claim # in page headers and inline. aiExtract() and verifyFlag() each call client.messages.create() directly — verifyAllFlags fans out Promise.all over every candidate key, so the same PHI-bearing section egresses to direct (non-BAA) Anthropic up to ~20+ times per letter, bypassing redact/outboundGate entirely, in production, regardless of HIPAA_LIVE/DEID flags (allowDirect passes the gate). The identical

bug class was found and fixed in mine-denials on 2026-06-10 (route.ts:131-135 comment: "prompt embeds verbatim patient denial_reason ... previous direct getAIClient sent it un-redacted to non-BAA Anthropic").

- Fix: Route aiExtract() and verifyFlag() through safePHITextCall (expectJSON:true) exactly as mine-denials was fixed — the prompts are already plain text so it is a mechanical swap. Alternatively run redact() on the section and outboundGate before any direct call. Remove or correct the zero-phi comment. Add a regression test asserting no getAIClient() call in this module.

- **Generation-pipeline puts patient-evidence free text in the UN-REDACTED system prompt (and it ships as a CLI argv)** — 🟠 HIGH · effort M · `/tmp/dhah-audit/apps/appeals/lib/ai/generation-pipeline.ts:79-108, 214-223; /tmp/dhah-audit/apps/appeals/lib/ai/phi-subscription.ts:126-167; /tmp/dhah-audit/apps/appeals/lib/ai/subscription-client.ts:68-71; /tmp/dhah-audit/apps/appeals/lib/argument-engine/patient-evidence-miner.ts (minePriorTherapyStack pushes facts.documentation.priorWeightLossAttempts into patientEvidence); /tmp/dhah-audit/apps/appeals/lib/criteria/canonical-facts.ts:249 (priorAttempts = intakeAnswers free text)`
  - Problem: callPHILLM only redacts and outbound-gates `promptWithPhi`; `systemPrompt` is documented "(no PHI). Prepended verbatim." and transmitted raw. generate.ts (first-level) honors this by putting the criteria ground-truth + argument toolkit in the USER prompt (lines 383-391). But buildFullSystemPrompt() in generation-pipeline.ts — used by /api/generate/ second-level, external-review, and nsa-idr — joins groundTruth + strategyDirective + argumentToolkit into the SYSTEM prompt. The toolkit's patientEvidence includes verbatim intake free text (e.g. priorWeightLossAttempts: "Dr. Smith put me on phentermine at St. Mary's in March 2024") and per-criterion evidence strings. This bypasses redact + outboundGate, and subscription-client passes the system prompt as a `--system-prompt` command-line ARGUMENT (subscription-client.ts:69-71) — visible in process listings on the shared NucBox, unlike the stdin-piped user prompt.
  - Fix: Move groundTruth/strategyDirective/argumentToolkit from endpointSystemPrompt assembly into promptWithPhi (mirroring generate.ts), keeping the system prompt static per (vertical, endpoint). Additionally run outboundGate over systemPrompt in callPHILLM as a belt-and-braces check, and pass the system prompt to the CLI via stdin or a temp file instead of argv.

- **Pre-delivery verifier fails OPEN: LLM extraction error silently returns passed=true** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/lib/ai/verifier.ts:134-137, 191-209`
  - Problem: verifyLetter is the BLOCKING gate that prevents Shannon-class contradictions ("letter asserts criterion met when evaluator computed not_met") from reaching patients. But extractClaims wraps its safePHITextCall in `catch (err) { console.error(...); return { claims: [], rawPreview: null }; }` — zero claims → checkContradictions finds zero

violations → `passed: true`. So exactly when the subscription CLI is degraded (timeouts, empty output, rate-limit — the same conditions that degrade the generator), the safety gate waves everything through. This directly violates the FAIL LOUD automation-integrity principle in a patient-facing safety control, and it also masks itself: /admin/verifier-blocks shows nothing because no block is recorded.

  - Fix: Distinguish "verified clean" from "verification unavailable": on extraction failure return a sentinel (e.g. passed=false, kind="verifier_unavailable", severity high) so the existing pending_review path in /api/generate (lines 413-445) and second-level's FAIL-LOUD branch hold the letter for human review, and Sentry-alert the failure. Small, contained change.

- **rehydrateOrThrow / verifyRehydration are dead code — dropped names and member IDs ship silently** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/lib/de-id/redact.ts:211-312 (definitions); /tmp/dhah-audit/apps/appeals/lib/ai/phi-subscription.ts:176-179 (plain rehydrate used)`
  - Problem: The RANK 7 fix (verifyRehydration: detect residual [CLASS_N] tokens and unconsumed NAME_/*MEMBER_ID*_ placeholders, with RehydrationIncompleteError so callers mark pending_review) exists and is documented, but grep shows ZERO call sites outside redact.ts. callPHILLM uses plain `rehydrate()`, whose own docstring notes invented placeholders are "left unchanged". Consequences: (a) if the model paraphrases/mangles [NAME_1], the patient's real name never returns and the appeal letter ships addressed to nobody — silently; (b) literal `[MEMBER_ID_2]`-style tokens can survive into a delivered letter/PDF. This is both a quality bug and a fail-silent violation on the highest-stakes output.
  - Fix: In callPHILLM, when rehydrateResponse !== false, call verifyRehydration() after rehydrate; on failure either retry once (models often fix token fidelity on retry) then throw RehydrationIncompleteError so route-level pending_review handling fires; record the failure in audit detail. One-file change with existing machinery.

- **Main letter generator fails OPEN if DEID is off in production — unlike every other guarded path** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/lib/ai/generate.ts:512-556 (else-branches), 227-229 (getClient); contrast /tmp/dhah-audit/apps/appeals/lib/ai/phi-subscription.ts:256-269 (H3 fail-closed) and lib/ai/extract.ts:158-166 (fails closed in ALL envs)`
  - Problem: generateAppealLetter branches on isDeidRoutingEnabled(); when false it streams the FULL PHI userPrompt (patient name, DOB, member ID, denial excerpts — see buildUniversalFactsBlock lines 107-205) to direct Anthropic via getAIClient({allowDirect:"legacy-text-fallback"}), with no NODE_ENV=production fail-closed check and no audit/ai_call record at egress. safePHITextCall got the H3 fix ("fail-closed in production ... refuse the call"); the single biggest PHI payload in the app did not. Today prod safety rests entirely on HIPAA_LIVE=on making selectProvider() throw — a one-env-var regression (e.g., unit-file refactor drops HIPAA_LIVE but keeps the API key) silently reverts the

flagship path to un-redacted, un-audited PHI egress. Unverifiable from source whether prod env currently sets both flags; the code-level asymmetry is the finding.

  - Fix: Mirror the H3 guard at the top of the non-DEID branch in generate.ts: if NODE_ENV===production and DEID_FAIL_OPEN!==on, throw instead of streaming; add the same deid_bypass audit write as safePHITextCall. Longer term, collapse generate.ts onto safePHITextCall/runGenerationPipeline so there is exactly one egress decision point.

- **Model-role ladder (Haiku/Sonnet/Opus escalation) is a silent no-op under subscription routing; maxTokens ignored too** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/lib/ai/subscription-client.ts:66-71; /tmp/dhah-audit/apps/appeals/lib/ai/phi-subscription.ts:162-167, 246-254; /tmp/dhah-audit/apps/appeals/lib/ai/generation-pipeline.ts:242-254; /tmp/dhah-audit/apps/appeals/lib/ai/models.ts:28-64`
  - Problem: callSubscription never receives the caller's model or maxTokens: args are `["-p","--model", process.env.CLAUDE_CLI_MODEL || "claude-opus-4-8"]`. So under DEID_VIA_SUBSCRIPTION=on (prod), the entire MODELS registry — Haiku TRIAGE (maxTokens 30 vertical classify), Sonnet GENERATOR, and the WIN2-MOAT-S3 "escalate to Opus on the FINAL regen attempt" logic (generation-pipeline.ts:247-248: `const escalatedModel = isLastAttempt ? MODELS.HARDER_GENERATOR : MODELS.GENERATOR`) — silently collapses to one model. The escalation ladder the comments describe as deciding "median revenue per Pro account" is dead code in prod; trivial classification calls consume the same flat-rate Opus capacity/latency budget (90s+ per spawn, 3 retries) as letter generation; and audit/receipt rows record only "claude-cli-subscription", losing role attribution. There is also no concurrency cap on `claude` CLI spawns (batch intake: 20 files × multiple safePHITextCall passes) on a small home server.
  - Fix: Thread opts.model through callPHILLM→callSubscription into `--model`, restricted to an allow-list of the already-approved pinned IDs in MODELS (keep claude-opus-4-8 as the default/escalation tier — do NOT un-pin). Log the effective model into the ai_call audit detail and receipts. Add a small semaphore (e.g. 2-3 concurrent spawns) in subscription-client. If deliberate single-model routing is preferred, delete the dead escalation branch and document the collapse so future tuning isn't done against a no-op.

- **Fact-check regen loop is dead code — known value-transcription errors ship to patients** — 🟡 MEDIUM · effort M · `/tmp/dhah-audit/apps/appeals/lib/ai/fact-check.ts:103-105, 108-120; /tmp/dhah-audit/apps/appeals/app/api/generate/route.ts:448-537`
  - Problem: factCheckLetter detects exactly the drift the no-hallucination rules worry about ("letter says HbA1c 7.8, truth is 7.1", invented prior therapies) with severity levels, and exports shouldRegenerate()/buildCorrectionsBlock() to drive a correction pass — but grep shows ZERO callers of either helper. In /api/generate the fact-check runs as a detached background Promise AFTER the appeal is already status='letter_ready' and the response has returned; high-severity discrepancies are stored in letter_strength_factors JSON and never trigger regen

or a hold. Additionally fact-check.ts:103 `catch { return null; }` is fully silent (not even console.warn), so a broken fact-checker is indistinguishable from a clean letter.

  - Fix: Highest-leverage letter-quality fix: when fc.discrepancies contains high-severity items, either (a) run one synchronous correction pass via the existing input.corrections plumbing in generateAppealLetter before letter_ready, or (b) flip the appeal to a "corrections suggested" state surfaced in the letter UI/admin. At minimum, log + Sentry the catch and count discrepancy rates per vertical to tune extraction.

- **lookupPolicy ignores the source-trust hierarchy — re-seeded config_seed placeholders can outrank scraped policies and get quoted "verbatim" as the insurer's policy** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/lib/policies/lookup.ts:109-147 (tier queries), 203-207 (safeConfigCriteria), 225-238 (updated_at bump on every seed); /tmp/dhah-audit/apps/appeals/lib/ai/generate.ts:480-483 (policyVerbatimDirective)`
  - Problem: Every tier query is `ORDER BY updated_at DESC LIMIT 1` with no reference to the source column. seedAllPolicies() refreshes its own config_seed rows with `updated_at = datetime('now')` on every run, so after any re-seed a config_seed row (whose coverage_criteria is deliberately the generic safeConfigCriteria pointer text, because "~20% of AI-authored criteria were fabricated") is newer than an older scraped/curated row sharing the same tier identity — and wins the lookup. The letter prompt then forces: "You MUST quote at least one specific clause from coverage_criteria VERBATIM ... attributed to the policy by title" — i.e., the letter quotes a generic placeholder sentence as the insurer's own policy language, weakening exactly the letters the verbatim-quote directive was built to strengthen. The write side is safe (seed can never overwrite scraped rows — the WHERE source='config_seed' guard, and auto_scrape preserves foreign sources); the READ side is the gap.
  - Fix: Add a source-priority ORDER BY to every tier query, e.g. `ORDER BY CASE source WHEN 'scraped' THEN 0 WHEN 'curated' THEN 0 WHEN 'auto_scrape' THEN 1 WHEN 'config_seed' THEN 2 ELSE 1 END, updated_at DESC` (exact source values to be confirmed against prod data — unverifiable from source alone). Also suppress policyVerbatimDirective when the matched row's source is config_seed (its criteria are a pointer, not quotable policy). Same ordering fix applies to pa-narrative.ts's policy SELECT (~line 95, ORDER BY effective_date DESC).

- **auto_scrape policies auto-promote to quotable authority with only length checks — LLM-scraped text becomes "the insurer's own words"** — 🟡 MEDIUM · effort M · `/tmp/dhah-audit/apps/appeals/app/api/internal/scrape-insurer-policy/route.ts:132-195 (auto-promote + quality gate), 24 (inline MODEL pin), header comment lines 1-8`
  - Problem: The scrape route's header says results are "queued in insurer_policy_suggestions for human review", but POLICY_AUTO_PROMOTE defaults ON and passesQualityGate is purely structural (title/URL present, coverage_criteria ≥200 chars, denial_reasons ≥100 chars). Nothing verifies the LLM-authored coverage_criteria against the fetched source text before

the row becomes tier-1 lookup material that generate.ts forces letters to quote VERBATIM in quotation marks attributed to the insurer. The codebase already concluded AI-authored criteria run ~20% fabricated (lookup.ts:200-202) — auto_scrape is the same class of AI-authored text with a web-search step, promoted without the human gate config_seed content gets. Also: the route pins `const MODEL = "claude-opus-4-7"` inline (route.ts:24) instead of a MODELS constant — drifts from the one-file-bump convention and from the app's opus-4-8 pin. Silent `catch {}` at the promote step (route.ts ~193) hides promote failures.

  - Fix: Either (a) gate auto-promotion on a cheap verbatim-grounding check (when the Playwright portal-fetch text is available, require quoted criteria clauses to appear as substrings) with failures routed to the existing human-review queue, or (b) keep auto-promote but tag the letter directive: for source='auto_scrape' rows, instruct citation by policy title/URL rather than verbatim quotation until human-reviewed. Move the model ID into MODELS and log promote failures loudly.

- **Silent catch{} fail-open across advisory AI quality passes — outages invisibly disable the quality layer** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/lib/ai/fact-check.ts:103; lib/ai/letter-summary.ts:138; lib/ai/p2p-prep.ts:164; lib/ai/adversarial-review.ts:134; lib/ai/strength.ts:60; lib/ai/pa-narrative.ts:115 (policy-block fetch); lib/ai/chart-to-pa.ts:116; app/api/generate/route.ts:451-470 (.catch(() => null) ×3) and 537 (.catch(() => {}))`

  - Problem: Every post-generation quality signal (strength score, adversarial review, executive summary, fact-check, P2P prep) swallows errors and returns null with no log, no Sentry, no counter — several catches are completely bare. A subscription outage or systematic JSON-shape drift would zero out the entire quality/win-probability layer (predictWinProbability folds in fc.score and adv.verdict, both silently null) while letters keep shipping and dashboards show nothing. This is the exact "silent failure/fallback" pattern the automation-integrity project (FAIL LOUD) targets, in the AI layer.

  - Fix: One mechanical sweep: replace bare catches with logSafeError + a Sentry breadcrumb tagged by helper name, and increment a per-helper failure counter in the ai_call audit detail so /admin/ai-cost (or a new /admin panel) can chart advisory-pass availability. Keep the null-return contract (advisory passes should not block) — just make failure observable.

- **outboundGate and duplicate-sweep exempt short identifiers (≤3 chars) including short NAMES — class-blind exemption** — ⚪ LOW · effort S · `/tmp/dhah-audit/apps/appeals/lib/de-id/redact.ts:93-99 (sweep skips values ≤2 chars), 331-341 (outboundGate skips originals ≤3 chars)`

  - Problem: To suppress Presidio false positives (state abbreviations, credential tokens), outboundGate skips any mapped value of length ≤3 and the duplicate-occurrence sweep skips ≤2. The exemption is class-blind: a genuine 2-3 character surname ("Ng", "Li", "Ho", "Yu" — common) or a 3-char member-ID fragment flagged by NER gets a placeholder for its first span, but unflagged duplicate occurrences elsewhere in the text are only caught by the

sweep (skips ≤2 chars — so 3-char names ARE swept) and, critically, the last-line outboundGate will never refuse a payload where a ≤3-char name leaked through. The design comment says "Over-redaction is safe; under-redaction is a HIPAA leak" — but this carve-out is an under-redaction acceptance keyed on length rather than identifier class.

  - Fix: Make the exemption class-aware: in outboundGate, skip short values only for low-risk classes (address/other/date components); never exempt class name, member_id, ssn, phone regardless of length (derive class from the placeholder key, e.g. /^[NAME_/). Add a de-id fixture with a 2-char surname to the scorer corpus (apps/appeals/de-id/fixtures) to lock the behavior.

- **AH batch intake silently defaults misclassified/failed uploads to the 'glp1' vertical** — ⚪ LOW · effort S · `/tmp/dhah-audit/apps/appeals/app/api/ah/intake/batch/route.ts:47-98 (readPdfFirstPageText catch → "", autoDetectVertical fallbacks)`
  - Problem: In auto mode, three separate failure paths — PDF text extraction throwing (line 50-53 `catch { return "" }`), missing API creds (line 71-73), and classifier error/unrecognized slug (lines 92-97) — all silently resolve to vertical="glp1" with only a console.warn. The vertical drives which system prompt, policy corpus, criteria rules, and argument packs are used downstream, so a clinician's oncology denial misfiled as GLP-1 produces a confidently wrong appeal scaffold with no operator-visible signal; the appeal row records vertical='glp1' as if classified. clinical_status='needs_review' softens this but nothing flags WHY review is needed.
  - Fix: On classification failure, persist an explicit marker (e.g. denial_extract_json._verticalAutoDetect: 'failed_default_glp1' and/or a distinct clinical_status reason) and surface it in the batch response `processed[]` items so the AH UI can prompt the clinician to pick the vertical. Keep the non-blocking behavior; make the fallback loud and reviewable.

DENIALHELP CONSUMER PRODUCT & UX

The DH consumer funnel is substantively strong where it counts least-visibly — pre-payment triage with a real not-appealable off-ramp, AI prefill with honest "from your letter" badges and fail-loud ambiguity handling, USDateInput used correctly everywhere, camera capture on mobile, lead-recovery checkpoints — but it leaks conversion at the exact top and bottom of the funnel. The homepage hero has NO call-to-action at all (the upload drop zone sits 3-4 screens down, and the mobile sticky CTA scrolls users past it to the 86-item condition browser), which directly validates the queued upload-first-home bet. The highest-intent path (drop-zone → intake) skips the step that shows pricing, so those users first see a price in a modal seconds before Stripe — the exact "17% bail at Stripe" problem a prior audit fixed only for the manual path. Six shipped CTAs link bare /intake, which silently defaults to the GLP-1 weight-loss intake (an ER-denial reader lands on "Which treatment were you denied? Wegovy/Zepbound…"). Error and warning states across the paid funnel use dark-theme-only classes (text-red-200/amber-200) that are near-invisible in the default light theme. The four fields marked required with red asterisks (member ID, claim number, denial reason,

patient name) are never actually enforced — the validation function exists but is dead code — so users can pay $39 for a letter drafted from an empty record. Copy is mostly honest and KFF-cited on the homepage, but vertical landing pages carry uncited 40-70% overturn stats and a "we charge only when we deliver" claim that contradicts the actual pay-then-generate sequence.

- **Homepage hero has no CTA; mobile sticky CTA routes past the drop zone to the 86-item browser** — 🟠 HIGH · effort M · `/tmp/dhah-audit/apps/appeals/app/page.tsx:899-1056, /tmp/dhah-audit/apps/appeals/components/StickyMobileCTA.tsx:50`
  - Problem: The hero section (page.tsx:901-931) renders badge, h1, subtitle, and a KFF source link — no button, and SiteNav is invoked without its optional cta prop (line 899). The first conversion element is the HomeDropZone at line 1031, placed after the how-it-works section with video (933-993) and the guarantee card (995-1021) — 3-4 screens down on mobile. Worse, the mobile StickyMobileCTA ('Start your appeal') anchors to #choose (line 50), the VerticalsBrowser at page.tsx:1056, which is BELOW the drop zone — steering mobile users past the labeled 'Fastest path' into the highest-friction 86-condition browse. A stressed consumer holding a denial letter has to hunt for the way in.
  - Fix: Move the drop zone (or a compact upload button + 'browse by condition' secondary link) into the hero above the video, and point StickyMobileCTA at the drop zone. This is also direct code-level validation that the queued 'upload-first home' build is the right bet: the component, classify API, and prefill handoff already exist and work — only placement is wrong.
- **Error/warning text near-invisible in the DEFAULT light theme across the paid funnel** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/components/HomeDropZone.tsx:286; app/intake/IntakeBodyClient.tsx:1334,1551,1803,2254,2268,2335,2623,2738; app/not-appealable/page.tsx:81,112,220`
  - Problem: globals.css establishes :root = LIGHT as default (line 21 comment 'Default (:root) = LIGHT', color-scheme: light at line 61). Yet the funnel's error/warning states use dark-theme-only tokens with no dark: variant: HomeDropZone upload error is `text-red-200` (pale pink on a white-gradient panel — effectively invisible); intake step-2 clinical-extract error is `bg-red-900/20 border-red-800 text-red-300`; step-5 submission error `text-red-200`; the autofill-incomplete banner `text-amber-200`; per-field blur validation messages `text-amber-300`; the DetectedLevelCard 'uncertain' notice `text-amber-200`; and the OverrideModal's level-selection warning body is `text-amber-100` — near-white text on amber-500/10, i.e., white-on-white for default-theme users choosing what to pay. These are the exact moments users must read to recover or decide.
  - Fix: Sweep these to the paired pattern already used correctly elsewhere in the same codebase (e.g., recover/page.tsx:30 `text-red-700 dark:text-red-200`, HardshipApplyClient.tsx:393 `text-red-700 dark:text-red-200`). One-pass grep fix.
- **Bare /intake silently defaults to the GLP-1 weight-loss intake — 6 shipped CTAs hit it** — 🟠 HIGH · effort S ·

`/tmp/dhah-audit/apps/appeals/app/intake/page.tsx:42; app/emergency-room-denial/page.tsx:124; app/how-it-works/page.tsx:88; app/faq/page.tsx:93; app/preventive-recoded/page.tsx:130; app/checkout/page.tsx:160; components/NewMenu.tsx:93`

  - Problem: intake/page.tsx:42: `const verticalSlug = sp.vertical ?? "glp1";`. Six links point at bare /intake with no vertical param. The emergency-room-denial page promises 'We draft the appeal letter citing the prudent-layperson standard' then its 'Start the appeal →' button drops the user into the GLP-1 intake, where step 3 asks 'Which treatment were you denied?' with options Wegovy/Zepbound/Mounjaro/Ozempic/Saxenda. Same for how-it-works' primary 'Start your appeal — $39' and checkout's no-appeal recovery CTA. This is a silent wrong-context fallback (contra the FAIL LOUD principle) that reads as a bait-and-switch to anyone not seeking a weight-loss drug appeal.
  - Fix: Make bare /intake render a lightweight 'what were you denied?' picker (search box reusing lib/search-keywords + category list) instead of defaulting to glp1, and fix the ER page link to /intake?vertical=out-of-network (that vertical exists and covers ER visits per page.tsx:87-93).

- **Upload-first (drop-zone) users never see pricing until the pre-Stripe modal — the '17% bail' fix only covers the manual path** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/app/intake/IntakeBodyClient.tsx:333,978-1004,1808-1813,2544-2655`
  - Problem: The price+guarantee box ('First-level appeal — $39 … Money-back guarantee … all levels bundle') added after the 2026-06-02 audit note '17% bail at Stripe step. Surface the price + guarantee up-front' (lines 985-1004) renders only inside `{step === 1 && …}`. Drop-zone arrivals start at step 2 (`useState<Step>(hasPreloadedDenial ? 2 : 1)`, line 333) and skip it entirely. Their first price signal is the oblique hardship footnote on step 5 ('If $39 is out of reach right now…', 1808-1813) and then the DetectedLevelCard modal showing '$39/$59/$79/$99' seconds before Stripe. The highest-intent cohort gets the worst price transparency — precisely the bail pattern the fix targeted. Relatedly, in the preloaded flow email is first collected at step 3 (line 1419-1432), so the lead-checkpoint recovery hook (lines 388-407) captures nothing from users who bail at step 2 — lost recoverable leads on the best-converting path.
  - Fix: Render the same price/guarantee box on the first step the preloaded flow actually shows (step 2), and move the email field to step 2 in the preloaded flow so the 24h 'pick up where you left off' nudge can fire for drop-zone bail-outs.

- **'Required' intake facts (member ID, claim number, denial reason, patient name) are never enforced — validation function is dead code** — 🟠 HIGH · effort M · `/tmp/dhah-audit/apps/appeals/app/intake/IntakeBodyClient.tsx:141-147,857-864,874-899; app/api/intake/route.ts:81-120`
  - Problem: REQUIRED_FACT_KEYS (line 142) is documented 'user MUST fill before submitting (validated on step 5)' and FactInput renders a red asterisk + aria-label="required" for these fields (line 2197). But missingRequiredFacts() (857-864) is never invoked anywhere — grep

shows only its definition — and canAdvance() checks nothing on the facts step (step 4 validates only vertical extraFields, step 5 only doctorName+terms, 874-899). Server-side, every universal fact is z.string().optional() (api/intake/route.ts:81-120). So when prefill fails or misses, a user can pay $39 with a blank member ID, claim number, and denial reason — producing a weak letter, guarantee refunds, and support load. A contradictory comment at 1871-73 calls the same fields 'softly required', confirming the intended gate never shipped in either form.

  - Fix: Wire missingRequiredFacts() into handleAdvanceClick on step 4/5 as a SOFT gate: list the missing fields with the existing red-highlight mechanism plus an explicit 'continue anyway — my letter doesn't show these' acknowledgment (some denial letters genuinely lack claim numbers, so a hard block would be wrong). Keep server schema permissive but record the acknowledgment for triage.

- **Copy honesty: uncited 40-70% overturn stats and a 'we charge only when we deliver' claim that contradicts the pay-then-generate flow** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/lib/verticals/glp1.ts:85-91; lib/verticals/biologics.ts:112; lib/verticals/cancer.ts:169; lib/i18n.ts:252-255; app/checkout/page.tsx + api/checkout/route.ts:169-201`

  - Problem: The homepage carefully cites KFF for 'roughly 1 in 3 appealed denials get overturned' (i18n.ts:17-27 with source link). But vertical landing pages render bare stat cards with no source: glp1 'Up to 6 in 10 GLP-1 denials get overturned' + '40–60% GLP-1 appeals overturned when properly argued', biologics '60%+ of biologic denials overturned on appeal', cancer '60–70%'. glp1 also claims 'Under 1 min — From upload to drafted appeal letter, in your hands' on the same page that says 'Four steps. Under 10 minutes.' — internally contradictory, and the letter is only generated after intake + payment. Separately, cta.payOnlyOnDelivery (i18n.ts:252) — 'We charge once, only when we deliver a letter your doctor can sign' — is rendered on every vertical landing page ([vertical]/page.tsx:293), but the actual sequence is Stripe payment → dashboard → generation (with a GenerationFailedCard retry path and manual refund). The truthful framing is the pre-payment triage + money-back guarantee, not pay-on-delivery.

  - Fix: Either add a visible source for each vertical overturn stat (as done for KFF on the homepage) or normalize to the cited 1-in-3/external-review figures; replace 'Under 1 min' with the flow-consistent 'in minutes'; reword payOnlyOnDelivery to guarantee framing, e.g. 'Free pre-payment review — if your denial can't be appealed you don't pay, and there's a full money-back guarantee.' This is a code-level copy-honesty exposure per the no-invented-promises constraint.

- **Intake progress bar shows 4 segments for a 5-step flow, in the wrong order; mobile reads 'Step 5 of 4'** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/app/intake/IntakeBodyClient.tsx:944-976; lib/i18n.ts:1719-1723`
  - Problem: The flow is 5 steps (Upload → Clinical records → Confirm insurance → Review facts → Doctor+terms; type Step = 1|2|3|4|5). The indicator maps only [1,2,3,4] and labels them Upload/Insurance/Clinical/Doctor — Insurance and Clinical are swapped relative to the real order (step 2 is 'Add clinical records', i18n:1743; step 3 is 'Confirm your insurance', i18n:1768), and there is no segment for the longest step (facts review). At step 5 the mobile indicator renders t('intake.steps.indicator') = 'Step {step} of 4' → literally 'Step 5 of 4', and no desktop label is highlighted as current. Progress indicators that lie undermine completion motivation on a long form.
  - Fix: Render 5 segments (or 4 in the preloaded flow which starts at step 2), fix the label order to Upload/Clinical/Insurance/Details/Doctor, and derive 'Step {n} of {total}' from the actual step count.
- **Homepage vertical cards render literal ''' entities as visible text** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/app/page.tsx:434,506,522,530,536-537; components/VerticalsBrowser.tsx:170-174`
  - Problem: Several VERTICALS entries embed HTML entities inside plain JS string literals: 'routinely step-therapy’d' (lupus, line 434), ''’experimental’ denials' (c-diff, line 506; men's health line 522), ''’try 4 antidepressants first’' (line 530), and the IBD title itself: `title: "IBD – Crohn&rsquo;s & UC"` (line 536). VerticalCard renders these via JSX interpolation ({vertical.title}, {vertical.description}) which escapes them — users see the raw characters '’' on the homepage browse grid and in search results within the site. Cheap-looking garbage text on the primary trust surface for exactly the chronic-condition audiences (IBD, lupus) most likely to buy.
  - Fix: Replace the entities with real ’ characters (or plain apostrophes) in the string literals; add a lint grep for `&[a-z]+;` inside VERTICALS.
- **Browse-path double hop: card 'Start →' opens another marketing page, and /pricing's CTA loops to the homepage** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/components/VerticalsBrowser.tsx:155-191; app/[vertical]/page.tsx:68,177-193; app/pricing/page.tsx:83-85`
  - Problem: A user who has already searched/expanded a category and clicked a card labeled 'Start →' lands on /{slug} — a full landing page (hero, stats, how-it-works, FAQ) — and must find and click 'Start my appeal' again to reach /intake?vertical=slug. Meanwhile /pricing's primary CTA (`<Link href="/">{t("pricing.cta.start")}`) sends a purchase-intent user back to the top of the homepage to start hunting again. Each extra hop on mobile costs a meaningful fraction of a stressed audience.

- Fix: On VerticalCard, keep the card body linking to the landing page (SEO/education) but make the 'Start →' affordance a direct link to /intake?vertical={slug}; point /pricing's CTA at /#choose or the drop zone anchor instead of bare '/'.

- **Homepage browse copy speaks clinician, not stressed consumer** — 🟡 MEDIUM · effort M · `/tmp/dhah-audit/apps/appeals/app/page.tsx:29-750 (VERTICALS descriptions), rendered at components/VerticalsBrowser.tsx:174`
  - Problem: The 86 card descriptions shown on the consumer homepage are dense with unexplained clinical/regulatory abbreviations: 'MHPAEA', 'step-therapy'd', 'NCD/LCD + ACC/AHA/HRS citations' (line 234), 'KDIGO 2024 LN + AURORA-1/2 + BLISS-LN turn most' (line 434), 'MASAC + WFH guidelines + bleed-rate logs' (line 210), 'ASTRO Group 1 + ELIANA + Yock medulloblastoma data' (line 410). This vocabulary signals expertise but costs comprehension for the actual buyer — a patient — at the moment they're deciding whether this service understands THEM. The same evidence detail already lives appropriately on the per-vertical landing pages and generation prompts.
  - Fix: Rewrite card descriptions in patient language ('Your plan made you try cheaper drugs first? That's called step therapy — it's often beatable.'), keeping one credibility cue max per card; keep the guideline arsenals on the vertical landing pages. Prioritize the top-10 traffic verticals rather than all 86 at once.

- **Hardship application stacks heavy friction on the most vulnerable users: 200-char minimum essay, required phone, and $0-tier consent quid pro quo** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/app/hardship/apply/HardshipApplyClient.tsx:17,67-79`
  - Problem: validate() requires: a denial-letter upload, a 'stress statement' of MINIMUM 200 characters (STRESS_MIN=200 — an essay-length proof-of-hardship hurdle), an income band, dependents count, an affordability pick, email, a COMPLETE US phone number (lines 75-76 — nothing in the flow indicates anyone will call), attestation, and — if the user selects $0 — mandatory consent to retain a de-identified copy (line 78 `if (isFree && !corpusConsent)`), i.e., the poorest users must trade data consent for free service. Every one of these is a hard block with errors shown only on submit. For a program whose purpose is removing barriers (linked from the homepage, intake step 5, and pricing), this form is more demanding than the paid intake's own gates.
  - Fix: Drop the phone requirement (or mark optional with a stated reason), lower the stress-statement minimum to ~50 chars with an encouraging hint instead of a hard count, and decouple corpus consent from the $0 tier (keep it opt-in). If the consent-for-free trade is a deliberate business rule, surface the trade explicitly BEFORE the user fills the form, not as a submit-time validation error.

- **Drop-zone error state offers no path to the manual upload flow it duplicates; OCR-fail hint points below the fold** — 🟡 MEDIUM · effort M ·

`/tmp/dhah-audit/apps/appeals/components/HomeDropZone.tsx:283-309; app/intake/IntakeBodyClient.tsx:978-1190`

- Problem: When home-page classification fails (including the explicit ocr_unreadable case), the recovery options are 'Try again' and 'Or browse by condition below ↓' (#choose). But the intake's own step 1 accepts the SAME file with camera capture and continues even when prefill fails (advanceFromStep1 catches errors and proceeds manually, IntakeBodyClient.tsx:449-455). A user whose blurry photo fails classification is sent to browse 86 conditions instead of being told 'we'll take it anyway — pick your condition and we'll carry your file in' or being offered the mobile camera re-shoot that step 1 has (capture="environment" input exists only in intake, line 1063-1073). The drop zone also lacks a camera affordance despite being the primary mobile entry.
- Fix: On error, add a third action: keep the uploaded file, open a compact condition-picker inline (reuse LowConfidenceConfirm's layout with search), then route to /intake?vertical=X&denial=Y so the upload isn't discarded (check whether /api/start-appeal already persisted the denialId even when classification failed — if so this is nearly free). Add a 'Take photo' button in the drop zone on mobile mirroring intake step 1.

DH Pro (clinician B2B) product — cockpit, doc factory, pricing/billing, onboarding, API/FHIR, enterprise surfaces

The clinician B2B product is substantially deeper than typical seed-stage code: doc factory with per-form statutory attestation gates (lib/pro/form-attestation.ts, sign route requires z.literal(true)), PA/appeal generation behind safePHITextCall, hashed API keys + signed webhooks, atomic seat-sync with Stripe rollback, insert-only audit_log enforced by SQL triggers, and the medical_policies source-priority rule correctly enforced in SQL (lookup.ts:237). Important framing correction: "DH Pro at $249/$799/$1499" no longer exists in code — every B2B surface (business, /enterprise, /for-clinicians, /pro/) hands off to ApprovalHelp at $499/$1,999/$4,999, and no stale $249 pricing survives anywhere; the memory framing is stale, the code is consistent. The real problems are commercial, not architectural: (1) the two most prominent signup CTAs (Google/Microsoft OAuth) create a dead-end "pending" account with no trial, no Stripe customer, and no post-signup checkout path; (2) the most expensive services (scribe transcription, doc-factory LLM generation) are not gated on an active subscription; (3) the "unlimited everything" pricing pivot was only half-landed in billing — persisted per-account caps still override, the account page renders the 999,999 sentinel the pricing file explicitly forbids showing, and the overage machine can now only mint $0 Stripe PaymentIntents that Stripe will reject; (4) enterprise/API marketing promises SSO/SAML, dedicated CSM, uptime SLAs, "$50K–$250K typical engagements", and webhook events + PA API endpoints that have no implementation — direct copy-honesty red-line exposure. Cockpit is Mic's internal ops surface (admin-cookie-gated), correctly not a customer feature. A top B2B-health-SaaS PM would*

fix the OAuth→trial funnel first, then payment-gate scribe/forms, then finish the unlimited-billing migration, then rewrite the enterprise page to roadmap-honest framing.

- **OAuth signup (top CTA) dead-ends in an unsubscribable 'pending' account — no trial, no checkout path, no attribution** — 🔴 CRITICAL · effort M · `/tmp/dhah-audit/apps/appeals/app/api/ah/oauth/google/callback/route.ts:114-123 (microsoft/callback/route.ts:111-112 identical); /tmp/dhah-audit/apps/appeals/app/ah/signup/signup-client.tsx:159-182; /tmp/dhah-audit/apps/appeals/app/ah/account/page.tsx:27`
  - Problem: The Google/Microsoft buttons are the FIRST CTAs on /ah/signup, above the email form. That path inserts pro_accounts with subscription_status='pending', account_type='provider', seats=1 — discarding the practice/provider toggle, seat count, NPI, state, and the sessionStorage attribution code (grep confirms zero attribution_code references under app/api/ah/oauth). No Stripe customer or checkout session is ever created (checkout.sessions.create exists only in /api/ah/signup, /api/ah/chart-checkout, and consumer /api/checkout). The user lands on /ah/dashboard behind a red 'subscription not active' banner (dashboard/page.tsx:542-554) pointing to /account, whose billing-portal error literally says 'Complete checkout first' — but no post-signup checkout route exists, and re-submitting the email form 409s ('An account already exists for this email'). The 7-day trial advertised by FREE_TRIAL_DAYS is only wired through the email path (signup/route.ts:155 trial_period_days). This is the flagship B2B funnel silently discarding its warmest leads, and it directly explains part of the 'traffic-blind' attribution gap for Pro.
  - Fix: Add a POST /api/ah/subscribe route callable by a pending account owner that creates the Stripe customer + subscription checkout (with trial_period_days: FREE_TRIAL_DAYS), and replace the dead-end red banner with a 'Start your 7-day free trial' CTA. Carry account_type/seats/attributionCode through the OAuth state cookie (they already round-trip a returnTo cookie) and stamp attribution_code in the callback exactly as the email path does.
- **Scribe transcription and doc-factory generation are not gated on an active subscription** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/app/api/ah/scribe/sessions/[id]/transcribe/route.ts:76-79; /tmp/dhah-audit/apps/appeals/lib/forms/auth.ts:20-29; contrast /tmp/dhah-audit/apps/appeals/lib/pa/auth.ts:33`
  - Problem: lib/pa/auth.ts requirePaAuth checks isSubscriptionActive(ctx.account); lib/forms/auth.ts, whose header comment says it 'Mirrors lib/pa/auth.ts', omits that check — requireFormsAuth only requires a session. All scribe routes (session create, transcribe up to 200MB audio, 100 req/hr/account rate limit only) call bare getCurrentProUser(). Combined with the OAuth pending-account path (previous finding), a never-paid account gets unlimited ambient-scribe transcription and doc-factory LLM generation — the two highest-marginal-cost services — indefinitely. Page-level gates exist on some UI routes but the APIs are directly callable.
  - Fix: Add isSubscriptionActive to requireFormsAuth and to the scribe session/transcribe routes (matching requirePaAuth), returning 402/subscription_inactive. Keep the staging bypass

(DEPLOY_ENV==='staging' already returns true inside isSubscriptionActive) so tester flows are unaffected.

- **'Unlimited everything' pricing pivot half-landed in billing: persisted caps override, $0 overage events can poison the Stripe batch, and the UI shows the forbidden 999,999 sentinel** — 🟠 HIGH · effort M · `/tmp/dhah-audit/apps/appeals/lib/pro/billing.ts:31-67,126-146,192-206,298-359,512-535; /tmp/dhah-audit/apps/appeals/lib/pro/pricing.ts:55-58,78,180-196; /tmp/dhah-audit/apps/appeals/app/ah/account/page.tsx:143-156`
  - Problem: pricing.ts declares all paid tiers unlimited (bundledAppealsPerMonth = UNLIMITED_QUOTA_SENTINEL = 999,999; APPEAL/PA overage $0) and warns 'never display this number to customers — branch on unlimited: true'. But: (a) getAllocationSnapshot uses row.monthly_allocation ?? getMonthlyAllocation(tier), and recordAppealUsage backfills with COALESCE — so any account whose row was written under the old 20/100/300 model keeps its cap forever and flips isOverage=true, contradicting 'unlimited' marketing (whether such rows exist in prod is unverifiable from source, but the code preserves them by design); (b) getOverageRate now parses Number('Unlimited'.replace(...)) = 0 while its comment still says '// 10' — chargeOverage then queues amount_cents=0 overage_events, and chargeBatchedOverages will create a Stripe PaymentIntent for $0.00, which Stripe rejects (minimum charge $0.50) → rows cycle charging→pending→failed via the MAX_BATCH_ATTEMPTS=4 poison-batch path, generating failure alerts; (c) /ah/account renders '{used} of {allocation} appeals used' with a progress bar — for post-pivot accounts that is '3 of 999999', exactly what pricing.ts forbids.
  - Fix: Finish the migration: one-time script to NULL out legacy monthly_allocation values (staging first); short-circuit queueOverageEvent when amountCents === 0 (return ok without inserting); branch the /ah/account usage card on tier.unlimited and render 'Unlimited'; fix the stale '// 20'/'// 10' comments so the next reader isn't misled.

- **AH Enterprise page promises capabilities and commitments that don't exist in code — SSO/SAML, dedicated CSM, uptime SLAs, '$50K–$250K typical engagements', PA API access** — 🟠 HIGH · effort S · `/tmp/dhah-audit/apps/appeals/app/ah/enterprise/page.tsx:24,74-96 and the inquiry card copy; cross-check: no SAML implementation (grep 'SAML' in lib/ + app/api hits only marketing strings); /tmp/dhah-audit/apps/appeals/app/api/v1/openapi.json/route.ts (appeals+webhooks only)`
  - Problem: Under 'What's included' (checkmarks, present tense): 'SSO/SAML via Okta, Azure AD, or any SAML 2.0 IdP' — the only auth in the codebase is Google/Microsoft OIDC for individual sign-in, zero SAML code; 'Dedicated customer success manager, named single point of contact' and 'Clinical staff training + change-management onboarding' — invented service commitments for a one-person company; 'SLA-backed support with guaranteed response time + uptime commitments' — the platform is one systemd service on a home NucBox behind a Cloudflare tunnel (deploy docs), an uptime guarantee nobody can honor;

'Typical engagements run $50,000–$250,000 annually' — implies existing enterprise deals (market overstatement red line); 'Public REST API v1 — programmatic access to PAs, appeals, statuses, packets' — /api/v1 has NO PA endpoints (appeals + webhooks + pdf only). The inquiry card also promises 'We'll reply within one business day with a proposed call window' — a reply-SLA + call commitment on Mic's behalf. Related smaller inconsistency: /developers claims 'audit-log retention 7 years' while /faq says '6 years' (schema.sql:36-38 says 7).

  - Fix: Rewrite the Enterprise page with two honest sections: 'Available today' (unlimited usage, custom BAA, de-id pipeline, REST API for appeals, 835 ingest, reporting) and 'Built per enterprise contract' (SSO/SAML, SLAs, training, EHR priority). Delete the CSM bullet, the $50K–$250K sentence, and the one-business-day/call-window promise (replace with 'we'll reply by email'). Align the retention number across /developers and /faq to 7 years per schema.

- **Public API documents webhook events that are never emitted, and API-created appeals silently default to vertical 'glp1' with no way to trigger generation** — 🟠 HIGH · effort M · `/tmp/dhah-audit/apps/appeals/app/developers/page.tsx:154-161; /tmp/dhah-audit/apps/appeals/app/api/v1/webhooks/route.ts:26-29; /tmp/dhah-audit/apps/appeals/app/api/v1/appeals/route.ts:162,235-245,251; /tmp/dhah-audit/apps/appeals/lib/webhooks/deliver.ts`
  - Problem: The /developers page and openapi.json advertise five events (appeal.created, appeal.submitted, appeal.letter_generated, appeal.outcome_reported, appeal.refunded). deliverWebhook is called in exactly two places in the entire codebase — appeal.created (v1/appeals POST) and appeal.submitted (v1/appeals/[id]/submit). A practice that subscribes to appeal.letter_generated ('emitted when the AI-drafted letter is ready for review' — the single most useful event for an integration) will wait forever; the letter-generation and outcome pipelines never fire webhooks. Compounding: POST /api/v1/appeals creates a stub with `body.vertical ?? "glp1"` (route.ts:162) — a cardiology denial submitted without a vertical is classified GLP-1, which feeds vertical-specific letter logic — and there is no API endpoint to trigger generation at all (a clinician must open the UI), which the docs never state. The success URL also uses PUBLIC_BASE_URL ?? denialhelp.com + /ah/appeals/… (route.ts:251), sending API consumers to the consumer hostname and exposing the /ah namespace the proxy layer explicitly hides (proxy.ts:241-245).
  - Fix: Either emit the three missing events from the existing generation-complete and outcome-recording code paths (deliverWebhook is already fire-and-forget), or delete them from /developers + the subscribable enum until they exist. Make vertical required or default to a neutral 'general' vertical, document that generation is UI-triggered (or add POST /api/v1/appeals/{id}/generate), and return the AH base URL.

- **getProTier has no Enterprise concept — a provisioned 25+ seat account would be billed as Practice Growth including metered eligibility the Enterprise tier promises to absorb** — 🟡

MEDIUM · effort M · `/tmp/dhah-audit/apps/appeals/lib/pro/billing.ts:23-28; /tmp/dhah-audit/apps/appeals/lib/pro/eligibility-billing.ts:179-196,222-239; /tmp/dhah-audit/apps/appeals/lib/pro/pricing.ts:204-222`

  - Problem: PRO_ENTERPRISE promises 'truly unlimited everything (scribe, eligibility, PAs, appeals)' with meterEligibility: false ('we absorb the clearinghouse cost'). But ProTier = provider|practice-starter|practice-growth|practice-chart — getProTier maps any practice with seats ≥ 6 to 'practice-growth'. billAllAccounts derives the tier via getProTier, so tierMetersEligibility('practice-growth') = true and a hand-provisioned 25-seat enterprise account gets $0.30/check Stripe invoice items, contradicting the signed offer. The signup route also never selects STRIPE_PRICE_PRO_ENTERPRISE (self-serve intentionally excluded), so the first enterprise deal will be manually provisioned straight into this trap. Minor adjacent bug in the same file: the header comment claims mock AND demo sources are excluded from billing but the SQL only excludes source != 'mock' (harmless today since source is typed 'stedi'|'mock' in eligibility.ts:75, but the comment/query drift invites regression).
  - Fix: Add an explicit enterprise marker (e.g. pro_accounts.plan = 'enterprise' or subscription metadata check, as the practice-chart comment at billing.ts:17-22 already anticipates) and make getProTier return it; exhaustive switch will then force correct handling in allocation, eligibility metering, and the account UI before any enterprise deal closes.

- **Seat packaging fights the product's own role model: 1-seat Independent tier forces the standard prescriber+MA workflow into credential-sharing or a 4x price jump** — 🟡 MEDIUM · effort M · `/tmp/dhah-audit/apps/appeals/lib/pro/pricing.ts:80-104; /tmp/dhah-audit/apps/appeals/lib/pro/roles.ts:1-15; /tmp/dhah-audit/apps/appeals/app/api/ah/oauth/google/callback/route.ts:143-156`

  - Problem: The role model was deliberately built for delegation — coordinator ('full appeal workflow except signing PA') and viewer exist precisely because PA/appeal legwork is done by medical assistants and billers, not prescribers. But every active membership consumes a billable seat (the OAuth invite-acceptance path bumps Stripe quantity whenever activeNow > purchased, with no role distinction), and PRO_INDEPENDENT is maxSeats: 1. So the $499 solo tier — the entry product — cannot legally onboard the MA who will actually operate it: the practice either shares the prescriber's login (destroying the per-user audit trail the enterprise page brags about) or jumps to $1,999. In B2B health SaaS this is the classic silent-churn generator for solo/2-provider practices, the segment the cold-email machine is targeting.
  - Fix: Include 1–2 free non-prescriber (coordinator/viewer) seats on Independent, enforced by role at invite time (canSignPA already exists as the natural billable/non-billable boundary), or price staff seats at a nominal $49. Update pricing.ts + the seat-bump logic to count only prescriber-role memberships against seats_purchased.

- **Marketing surfaces contradict each other on EHR-integration maturity (DrChrono 'active' vs 'Athena in Preview today' vs 'SMART on FHIR coming next')** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/app/integrations/page.tsx:31-75; /tmp/dhah-audit/apps/appeals/app/ah/enterprise/page.tsx (SMART-on-FHIR bullet); /tmp/dhah-audit/apps/appeals/app/developers/page.tsx:175-181`
  - Problem: Three public pages tell three different stories: /integrations lists DrChrono as status 'live-staging' ('Active integration. Production registration in progress') and Epic/Cerner/Athena/eCW/MEDITECH as 'ready-to-connect' with 'Client ID configured'; /ah/enterprise says 'Athena in Preview today; Epic, Cerner, eCW, MEDITECH, DrChrono on the roadmap'; /developers says 'SMART on FHIR (coming next)… Beta access: email us'. Which one reflects reality is unverifiable from source (client IDs are env values), but at most one can be right — the others overstate or understate to prospects comparing pages, and 'Client ID configured' presented as an integration status is promise-inflation for vendors where only a sandbox registration exists. lib/smart-fhir is a real implementation (oauth, client-assertion, us-core-zod, writeback), so this is a copy-coherence failure, not vaporware.
  - Fix: Pick one canonical status source (a small VENDOR_STATUS constant à la lib/pro/pricing.ts) consumed by all three pages, with honest states: 'Preview (Athena)', 'Sandbox-tested', 'Roadmap'. Verify against the live env before promoting (unverifiable from source which vendor is actually connected).
- **/fhir/connect leaks the DenialHelp brand onto the clinician host and lists identical Epic production/sandbox URLs** — 🟡 MEDIUM · effort S · `/tmp/dhah-audit/apps/appeals/app/fhir/connect/page.tsx:8-22,29; /tmp/dhah-audit/apps/appeals/proxy.ts:146-178,287-300`
  - Problem: The page hardcodes metadata title 'Connect your EHR | DenialHelp' and body copy 'DenialHelp uses SMART on FHIR…'. /fhir is neither in CLINICIAN_ONLY_PATH_PREFIXES nor AH_ROOT_PATHS, so it serves unrewritten on BOTH hosts — a clinician who reaches approvalhelp.com/fhir/connect (e.g. from EHR docs or the /developers SMART section) sees consumer-brand chrome on the clinician product, undermining the brand-separation program that proxy.ts:241-245 enforces everywhere else. Separate copy bug in the same file: COMMON_ISS lists 'Epic — production' and 'Epic — sandbox' with the identical URL (https://fhir.epic.com/interconnect-fhir-oauth/api/FHIR/R4 for both), so the sandbox option silently targets production.
  - Fix: Switch the page to getBrandIdentity() like /developers does (title + body use identity.productName), and fix the Epic sandbox iss (Epic's sandbox R4 endpoint differs from production). Consider adding 'fhir' to the clinician-only prefix list so DH visitors are routed to the AH host where the flow belongs.
- **Latent one-flag path to the forbidden 'ApprovalHelp Pro' string in the AH homepage title and signup pill** — ⚪ LOW · effort S · `/tmp/dhah-audit/apps/appeals/app/ah/page.tsx:16-18; /tmp/dhah-audit/apps/appeals/app/ah/signup/signup-`

`client.tsx:106-108; /tmp/dhah-audit/apps/appeals/app/ah/pricing/page.tsx:133-135; /tmp/dhah-audit/apps/appeals/lib/brand-chrome.ts:45,58,74`

- Problem: app/ah/page.tsx builds titleStr = `${productName} Pro — …` when identity.showProPill is true; on the pro host productName is 'ApprovalHelp', so that branch renders the red-lined string 'ApprovalHelp Pro' in the page