ALAI Frontend Engineering Spec v1 — Build, Auth, Cache, i18n, CSP, CI Gate (MC #106089)

ALAI Frontend Engineering Spec v1

Document ID: ALAI-FE-ENG-001
Version: 1.0
Date: 2026-07-20
Status: Active draft for cross-product PR/CI adoption
Owner: Vizu — Brad Frost + Lea Verou route, with Securion for security review
MC: #106089
Scope: Every ALAI product frontend using Next.js/App Router or React-based web UI. Product-level docs may be stricter, but may not weaken this spec without an explicit architecture decision.

0. Source Evidence and Existing Standards Read

This spec closes a cross-product standards gap surfaced by the 2026-07-14 to 2026-07-20 frontend incident chain:

MC	Incident class	Standard added here
#105793	Bilko stage Entra CIAM refresh-cookie regression after hard navigation	Auth/session invariants, cookie-domain checks, CI auth replay
#106006	Next build failed on invalid page export; prior CI had passed with effectively same web code	`next build` is a required gate; dependency/lockfile drift controls
#106020	Custom-domain E2E residual refresh-cookie failure: CI-vs-local delta	custom-domain E2E, test cookie isolation, env-delta evidence

MC	Incident class	Standard added here
#106087	Long-lived tab after deploy kept stale bundle; button did nothing until reload	deploymentId/version-skew guard and chunk-load recovery
#106088	Turnstile config used invalid size value	vendor-config validation and browser console gate

Existing documents read and incorporated:

- `~/system/specs/FRONTEND-BLUEPRINT.md` — Next.js 15, React 19, strict TypeScript, next-intl, state/testing baseline.
- `~/system/specs/ALAI-UNIVERSAL-BLUEPRINT.md` — cross-product invariants and security headers.
- `~/system/specs/ungameable-testing-methodology.md` — tests must interact, assert, and leave evidence.
- `~/ALAI/products/Bilko/docs/frontend/FRONTEND-ARCHITECTURE.md` — product-level i18n, error boundary, performance, environment baseline.
- `~/ALAI/products/Bilko/docs/frontend/DESIGN-SYSTEM.md` — component/visual system context.
- `~/ALAI/products/Bilko/docs/frontend/STATE-MANAGEMENT.md` and `FORMS.md` — current frontend state/form gaps.
- `~/ALAI/products/Bilko/docs/frontend/ACCESSIBILITY-AUDIT.md` — WCAG and interaction constraints.

1. Non-Negotiable Frontend Invariants

1. **`tsc` is not a build.** A PR is not frontend-build-clean until the framework production build passes (`next build` for Next.js).
2. **HTML shell is not cacheable.** Serve route HTML/app shell with `Cache-Control: no-store` or a product-approved equivalent that always revalidates before use.
3. **Hashed chunks are immutable.** Static hashed JS/CSS/image chunks may be `public, max-age=31536000, immutable` only when their filename contains content hash or framework build hash.
4. **Every deployed frontend exposes a build identity.** UI and API must agree on a deploy/build identifier so long-lived tabs can detect skew.
5. **Every user action has loading, success, empty, error, and retry behavior where relevant.** Dead clicks are bugs, even if the API/backend is healthy.
6. **Auth is validated on the public/custom domain, not only localhost or raw cloud host.** Cookies, redirects, SameSite, Secure, Domain, Path, and CORS differ by domain.
7. **No production mock data.** If real data cannot be fetched, render an explicit empty/error state, not fake records.
8. **No frontend claim is verified by HTTP 200 alone.** Playwright/browser evidence is required for user-facing flows.

2. Build Identity and Version-Skew Guard

2.1 Required build metadata

Each frontend build must expose a comparable build identity to the client. Two mechanisms are both valid implementations of this invariant — pick one per product, do not require both:

(a) Structured metadata endpoint/asset — a stable JSON payload:

```
{
  "product": "bilko",
  "environment": "stage",
  "gitSha": "full-or-short-sha-from-ci",
  "deploymentId": "next-build-id-or-ci-run-id",
  "builtAt": "2026-07-20T18:00:00Z"
}
```

Recommended paths:

- Next.js: `/build-meta.json` in `public/`, generated in CI before `next build`.
- API-backed apps: API `/api/v1/health` also returns compatible `gitSha` / `deploymentId`.

(b) Response header + build-time-baked client value — a single opaque build identifier (commit SHA or CI run ID) set as a response header (e.g. `x-build-id`) on all routes, compared against the same value baked into the client bundle at build time (e.g. Next.js `env:` config, inlined into `NEXT_PUBLIC_*`). Cheaper to implement (no JSON endpoint, a `HEAD` request is enough to read the comparison value) and is what Bilko currently has shipped and tested (MC #106087) — see `apps/web/next.config.js`'s `x-build-id` header + `NEXT_PUBLIC_BUILD_ID`, and `apps/web/lib/hooks/use-version-skew-guard.ts`. Bilko does **not** currently have a `/build-meta.json` file; do not treat (a) as the only compliant option when auditing an existing product against this spec.

2.2 Client-side skew detection

Every authenticated or long-lived shell must:

1. Read current build identity on boot.
2. Poll or revalidate on window focus and after route changes.
3. Detect changed `deploymentId` / `gitSha`.
4. Show a visible toast/banner: **“New version available — refresh to continue.”**
5. Provide a button that calls `window.location.reload()`.

6. On dynamic import or chunk load failure (`ChunkLoadError`, script 404), show the same refresh UX and log to Sentry.

Do not silently reload during form entry unless the product has explicit unsaved-change protection.

2.3 Acceptance tests

CI must include at least one version-skew test per product:

- Simulate current `/build-meta.json` changing after page load.
- Verify toast/banner appears.
- Click refresh CTA and verify `location.reload` path is invoked or page reloads.
- Simulate a chunk-load error where possible and verify the same recovery UX.

3. Error, Retry, Loading, Empty, and Disabled-State UX

3.1 Standard state model

Every async view/mutation implements these states explicitly:

State	UI requirement
Loading	skeleton or spinner with accessible label; action controls disabled when duplicate submission would be unsafe
Success	visible state change, toast, navigation, or updated data row
Empty	human-readable empty state with next action where applicable
Error	human-readable message; no raw stack traces; includes request correlation ID when available
Retry	visible retry control for transient network/server errors

3.2 Retry rules

- GET/query failures: allow user retry and optionally one automatic retry with backoff.
- POST/PUT/PATCH/DELETE: no blind automatic retry unless the operation is idempotent by key. Use client-generated idempotency keys for financial or document mutations.
- Auth 401: attempt exactly one refresh/session repair before redirecting to login.
- Validation 400/422: field-level errors; do not show generic “Something went wrong” only.

- Rate limit 429: show wait/backoff message.

3.3 Dead-click prevention

Every button/link that triggers a side effect must have at least one observable outcome:

- URL change, modal open/close, toast, disabled/loading state, data update, or field-level error.
- Playwright must click core CTAs and assert the outcome. Checking that a button exists is not enough.

4. MSAL / Entra / Auth Pattern

This section applies to Entra External ID / MSAL products and all products with browser-auth cookies.

4.1 Configuration invariants

- MSAL authority, client ID, redirect URI, post-logout redirect URI, and known authorities are environment-specific and documented in `.env.example`.
- Public `NEXT_PUBLIC_*` auth variables are passed at build time for Next.js Docker builds, matching `FRONTEND-BLUEPRINT.md` §3.
- Redirect URIs must use the product public/custom domain for stage/prod E2E, not raw cloud host unless the product explicitly supports both.

4.2 Cookie/session invariants

Refresh/session cookies must be verified on the live domain:

- `Secure` on HTTPS.
- `HttpOnly` for refresh/session tokens not read by JS.
- `SameSite=None` only when cross-site flow requires it; otherwise `Lax` preferred.
- `Domain` must match the browser origin strategy. Custom-domain E2E must not inject cookies for only the raw cloud host.
- `Path` is explicit and broad enough for session refresh routes.

4.3 Client auth flow rules

- **Interactive-action enabled/disabled state MUST derive from the exact same async-ready signal that gates the SDK's own provider mount, never a separate synchronous proxy condition.** This is the checkable form of "no competing login

redirect/session refresh races": if the condition gating a button's `disabled` prop and the condition gating whether the auth provider is actually mounted can independently be true/false, the race exists. A reviewer verifies this by comparing the two conditions directly in the diff (MC #105793 — a click landing in the gap between "provider not yet mounted" and "button already enabled" reached the SDK's documented no-provider stub context, whose interactive methods reject a promise the click handler discarded, producing a silent, permanent dead click).

- **A permanent (not transient) async-init failure MUST surface a visible error and a manual retry affordance — never leave the action silently disabled forever.** See §3.1's state model; a permanently-failed SDK init is the harder variant of the "Error" row and needs the explicit callout here because a naive implementation will not automatically avoid the trap below. (MC #105983 — Parisa PR184 non-blocking finding: the correct fail-closed gate above still left the button disabled forever with no explanation on a genuine permanent failure, e.g. CIAM metadata unreachable.)
- **A retry affordance MUST guard against duplicate concurrent retry attempts using state distinct from the failure flag itself — not the same flag, and not the generic mutation-idempotency rule in §3.2 (which is about API calls, not SDK re-initialization).** The failure flag is typically reset to "not failed" the instant a retry begins, before the retried attempt resolves or rejects — if the duplicate-click guard reads that same flag, a second click during the retry window is not blocked, because the flag is transiently `false` while the retry is in flight. Track a separate "retry in progress" state that only clears once the retried attempt actually settles.
- Silent token/session repair may run once per failing navigation; after that show a real error or redirect to login.
- Logout clears product app state, MSAL cache, in-memory access token, and server session cookie where applicable.
- **E2E auth fixtures must be isolated per test worker and per domain via the test's own code (`test.use({ storageState: ... })`, explicit `context.clearCookies()`/fresh-context construction) — not merely documented as a CLI-invocation contract in a comment.** A comment describing how CI is expected to invoke the test runner (e.g. "run with `--no-deps`") is not verifiable by the test file itself and will silently drift the moment the pipeline YAML changes without every spec's header being re-read. Treat "isolation contract stated only in prose" as equivalent to "no isolation" for review purposes (MC #106020 — a spec's header comment claimed a `--no-deps` contract that the actual CI invocation did not honor; a shared `setup` project injected a cookie into the persisted `storageState` before the spec ran, producing a CI-only failure that never reproduced locally because local runs happened to use the flag the comment described but CI did not).

4.4 Required auth E2E

For every auth-enabled product stage gate:

1. Login on public/custom domain.

2. Verify dashboard or protected landing renders authenticated state.
3. Hard-navigate to at least one protected deep link.
4. Verify user remains authenticated and is not redirected to login.
5. Capture `Set-Cookie` and request cookie-domain evidence for session/refresh endpoints when debugging auth failures.
6. Logout and verify protected route no longer renders authenticated data.

5. Cache Policy Norm

5.1 Required headers

Asset class	Required cache behavior
HTML/app shell/routes	`Cache-Control: no-store` or revalidate-equivalent approved by deploy owner
`/_next/static/*` hashed chunks	`Cache-Control: public, max-age=31536000, immutable`
Public hashed assets	immutable only when filename is content-hashed
`build-meta.json`	`Cache-Control: no-store`
Service worker	avoid by default; if used, must have a documented update strategy and tests

5.2 CI header check

Deploy verification must assert headers on the public URL:

- HTML route has no-store/revalidate behavior.
- At least one loaded JS chunk has immutable caching.
- `build-meta.json` is not cached.
- No localhost/LAN URL appears in user-facing output or docs for CEO/client handoff.

6. i18n Standard — next-intl

Baseline remains `FRONTEND-BLUEPRINT.md` §9.

Additional engineering conventions:

1. Translation keys use namespaces by product domain: `navigation.dashboard`, `invoices.createButton`, `errors.networkError`.
2. User-visible strings in JSX are banned once a product is declared multi-language.

3. Backend returns locale-independent values: ISO dates, enum codes, numeric amounts, currency codes.
4. Frontend formats with `next-intl` / `Intl.NumberFormat` / `Intl.DateTimeFormat`.
5. Error responses use stable error codes; frontend maps codes to localized copy.
6. CI runs a missing-key check for every supported locale.
7. E2E covers at least default locale plus one non-English locale for navigation labels and date/amount formatting on market-critical pages.

7. CSP and Browser Security Baseline

7.1 Required security headers

At minimum on stage/prod public domains:

```
Content-Security-Policy: default-src 'self'; base-uri 'self'; object-src 'none'; frame-ancestors 'none'; script-src 'self' 'nonce-{per-request-nonce}' 'strict-dynamic'; style-src 'self' 'unsafe-inline'; img-src 'self' data: blob: https:; font-src 'self' data:; connect-src 'self' https:; form-action 'self'; upgrade-insecure-requests
Strict-Transport-Security: max-age=31536000; includeSubDomains
X-Content-Type-Options: nosniff
X-Frame-Options: DENY
Referrer-Policy: strict-origin-when-cross-origin
Permissions-Policy: camera=(), microphone=(), geolocation=()
```

Notes:

- Prefer nonces for inline scripts. Avoid `unsafe-eval` in stage/prod.
- `style-src 'unsafe-inline'` may be temporarily tolerated for framework/style tooling, but products should move toward nonce/hash-based styles where feasible.
- Third-party vendors such as Turnstile, analytics, Sentry, and fonts must be explicitly listed per product. No wildcard vendor domains without a security review.
- CSP report-only may be used during rollout, but enforcement must be the target for beta/prod.

7.2 Vendor config validation

Any third-party widget must have a typed/config-validated wrapper. Example classes:

- Turnstile `size` allowed values must be `normal`, `compact`, or `flexible`; invalid values fail lint/unit test.

- Analytics/Sentry DSNs and environment names must be validated at boot and surfaced as warnings/errors in stage.

Classify any third-party-widget console error into one of two buckets before treating it as a defect (2026-06-17 lesson — a Turnstile `600010` error was chased as a real bug for ~2 agent rounds before being identified as a known false-positive class; the sitekey/domain config was already correct the whole time):

- **(a) Behavioral/detection-class errors** — the widget's own bot-detection, challenge, or session-validation logic legitimately behaves differently under headless/automated browser contexts than under a real user's browser (Turnstile's `600010` is the documented example). These are NOT necessarily real bugs. Require a live-browser repro before treating as a defect — a headless probe alone is not sufficient evidence either way, in either direction (a headless-clean run does not prove a working real-user flow, and a headless error does not prove a broken one).
- **(b) Configuration/parameter-class errors** — the widget rejects a literal value passed to its own public API (an enum member that doesn't exist, a malformed ID, a missing required field). These fail identically regardless of automation context, because the SDK's own input-validation code rejects the value before any environment-dependent logic runs (e.g. "Invalid value for parameter size... got invisible" — `invisible` has never been a valid Turnstile `size` value, in any context). Fix directly against the SDK's documented parameter contract; still confirm the fix in a real browser afterward.

The error message itself is usually the tell (a message naming a parameter and its accepted values is bucket (b); a message naming a challenge/session/verification failure code is bucket (a)) — when in doubt, verify in a real browser rather than guessing which bucket applies.

8. Test Pyramid and CI Gate

8.1 Required frontend gates

Gate	Required command/class	Why
Typecheck	`tsc --noEmit` or framework equivalent	Catches static TS errors only
Lint	ESLint + product rules	Catches exports, hooks, a11y, i18n/mocks patterns
Unit/component	Vitest/React Testing Library	Validates state machines and wrappers
Production build	`next build`	Required because Next validates App Router exports/build-only behavior
Browser smoke	Playwright on built/deployed app	Verifies user-visible runtime

Gate	Required command/class	Why
Critical E2E	Playwright flows by feature class	Auth, forms, navigation, mutations, deep links
Console/network gate	Playwright listeners	Fails on app console errors and unexpected 4xx/5xx
Header/cache/CSP gate	curl/Playwright response inspection	Prevents cache/security regressions
Bundle/perf/a11y	Lighthouse or equivalent	Prevents slow/inaccessible regressions

8.2 `tsc` is not enough

`tsc` can pass while `next build` fails because Next.js validates route module exports, server/client boundaries, metadata rules, and framework build semantics. Therefore:

- PR cannot be green with `tsc` only.
- Main/promote cannot run unless the exact commit passed `next build` using the lockfile used in deploy.
- CI must install dependencies from lockfile (`npm ci`, `pnpm install --frozen-lockfile`, or equivalent). No floating install for production build.

8.3 Browser evidence rule

Every user-facing fix must leave machine evidence:

- screenshot or trace for the interacted flow,
- console error log,
- network failure summary,
- exact public URL and commit/build ID.

HTTP 200 without DOM/action assertions is not evidence.

9. PR Review Checklist

Use this checklist in every frontend PR review. A reviewer may mark non-applicable items as `N/A` only with a one-line reason.

Scope. This checklist applies to the code the PR actually adds or changes, not retroactively to the entire product codebase. A pre-existing condition elsewhere in the product that the PR does not touch or extend is not grounds to block that PR — check Appendix B (Known Deviations) first; if the gap is listed there with an open tracking MC, the PR is not blocked on it unless the PR's own diff makes the deviation worse (adds new code that further depends on the deviated-from behavior). A

gap not yet listed in Appendix B should be flagged and added there, not silently waved through and not used to block an unrelated PR.

Build and dependency discipline

- ☐ Lockfile changed only when dependency change is intentional.
- ☐ Dependency install in CI uses frozen lockfile / `npm ci`.
- ☐ `tsc` passes.
- ☐ Lint passes.
- ☐ Production framework build passes (`next build`).
- ☐ No invalid App Router page/layout exports.

Runtime UX

- ☐ Loading, empty, error, and retry states are implemented for every async area touched.
- ☐ Buttons/links clicked in tests have observable outcomes.
- ☐ Mutations prevent duplicate unsafe submission.
- ☐ Error copy is human-readable and does not expose stack traces.

Auth/session

- ☐ Auth redirect/session flow tested on public/custom domain.
- ☐ Hard navigation to protected deep link remains authenticated after login.
- ☐ Logout clears client and server session state.
- ☐ Cookie attributes are correct for the deployed domain strategy.

Cache/versioning

- ☐ Build metadata is present and no-store.
- ☐ Version-skew detection exists for long-lived shells or is tracked as an explicit product debt.
- ☐ HTML no-store/revalidate and chunks immutable headers verified on deployed/public URL.

i18n/a11y/security

- [ ] No new hardcoded user-visible strings in multi-language products.
- [ ] Locale-aware date/amount formatting used.
- [ ] Keyboard navigation and focus states remain valid.
- [ ] CSP/security headers are not weakened.
- [ ] Third-party widget config uses allowed values and is tested.

Evidence

- [ ] PR includes command output for typecheck/lint/build/tests.
- [ ] PR includes browser evidence for user-facing changes.
- [ ] Known browser console warnings are named; unexpected console errors fail.

10. CI Gate Definition

Minimum required gate for every ALAI product frontend PR:

```
# dependency discipline
npm ci || pnpm install --frozen-lockfile

# static gates
npm run typecheck
npm run lint
npm run test:unit

# framework build gate — mandatory
npm run build

# browser smoke on built artifact or deployed preview
npm run test:e2e:smoke
```

Additional required stage/promote gate:

```
# public URL verification
curl -sI "$PUBLIC_WEB_URL"
curl -s "$PUBLIC_WEB_URL/build-meta.json"
npm run test:e2e:auth-critical
npm run test:e2e:core-flows
npm run test:e2e:headers
```

A product may use different script names, but must map to the classes above in `docs/frontend/ci-gate.md`.

11. Product Adoption Requirements

Each product must add or update:

1. `docs/frontend/engineering.md` — product deviations and ownership.
2. `docs/frontend/ci-gate.md` — exact CI commands and public URL checks.
3. `docs/frontend/auth.md` — if auth-enabled, cookie/redirect/session domain map.
4. A build-identity mechanism per §2.1 — either `public/build-meta.json` or an equivalent header + baked-client-value scheme (see §2.1 for both).
5. Playwright smoke covering navigation, one critical form/action, auth if relevant, and console/network gate.

12. Definition of Done for Frontend User-Facing Work

Scope. Same as §9: applies to the code the task actually changed. A pre-existing gap tracked in Appendix B (Known Deviations) does not block a task's own done-ness unless that task's diff extends or depends on the deviated-from behavior.

A frontend task is not done until all are true:

1. Relevant code/docs changed in the correct product worktree.
2. `BUILD-BLUEPRINT.md` or product blueprint was read before code edits.
3. Typecheck, lint, tests, and production framework build pass or failures are explicitly scoped as unrelated with evidence.
4. Browser test interacted with the affected UI and asserted outcome.
5. Public/deployed URL verified when the task is deploy/user-facing.
6. Evidence files exist before MC ready/done.
7. Independent reviewer/validator checked the exact diff for M/H or risky work.

Appendix A — Reviewer Short Form

Reviewer verdict format:

```
Frontend Spec v1 review: PASS | PARTIAL | BLOCKED
Build gate: pass/fail + command
Browser gate: pass/fail + URL/evidence
Auth/cache/CSP impacted: yes/no
Required follow-ups before merge/promote: ...
```

Appendix B — Known Deviations

This spec describes a target state. Some sections were written by generalizing from a single incident/product without an existing implementation elsewhere; some already-shipped code predates this spec and has not yet been brought into compliance. Both are tracked here rather than left as unstated gaps — a reviewer applying §9/§12 to an EXISTING product should check this table before blocking a PR on a pre-existing condition unrelated to that PR's diff (see §9/§12 note below).

Product	Section	Gap	Tracking MC
Bilko	§7.1 CSP baseline (`script-src` nonce/no-`unsafe-eval`)	Production `next.config.js` currently ships `script-src 'self' 'unsafe-eval' 'unsafe-inline'` (own `TODO (production hardening)` comment already present in source)	#106097
Bilko	§6.6 (`CI runs a missing-key check for every supported locale`)	No such CI check currently exists; missing-locale-key coverage was verified manually (per-file JSON well-formedness only) during this week's fixes	#106098
Bilko	§8.1 Console/network gate (listed as universally required)	Only 6 of 37 `apps/e2e/tests/*.spec.ts` files currently implement console-error listening	#106099
Bilko	§3.2 (`Use client-generated idempotency keys for financial or document mutations`)	Zero references to idempotency keys in `apps/web/lib/api.ts` today — architecture decision needed on whether/how to implement, not yet resolved	#106100

Product	Section	Gap	Tracking MC
Bilko	§2 Build Identity and Version-Skew Guard (whole section)	Implemented and tested (`apps/web/lib/hooks/use-version-skew-guard.ts` + `VersionSkewGuard` component, mechanism (b) from §2.1) but **not yet merged to `main`** — MC #106087 status is `paused`, pending priority review, not a live incident (natural reload already resolves it for most users)	#106087

When a task touches a product/section pair listed here, the deviation does not block that task's own PR (see §9/§12) but should not be silently widened either — do not add new code that further relies on the deviated-from behavior (e.g. do not add a new inline `<script>` that depends on `unsafe-eval` while #106097 is open).

Appendix C — Known Anti-Patterns

- Treating `tsc` as a production build.
- Header-only deploy verification (`curl 200`) for user-facing changes.
- E2E tests that only check body length or URL existence.
- Cookie injection for raw host while tests run on custom domain.
- E2E isolation stated only as a comment describing a CLI-invocation contract (e.g. "run with `--no-deps`") instead of being enforced in the test's own code — the comment is not verifiable by the test file and will drift from the actual CI YAML unnoticed.
- App Router page files exporting arbitrary constants/functions.
- Floating package installs that let Next/framework behavior drift between CI runs.
- Immutable caching on un-hashed HTML or build metadata.
- Silent chunk-load failure with no refresh UX.
- Invalid third-party widget enum values caught only by browser console.

Revision #2
Created 2026-07-20 18:06:27 UTC by John
Updated 2026-07-20 19:44:11 UTC by John