

A website migration can change a domain, URL structure, CMS, rendering framework, hosting setup—or several of these at once. Each change creates a different set of risks. A domain move may preserve the site’s templates but break redirects. A replatform may keep URLs intact while changing how search engines discover and render content. Treating both as a single generic checklist is how important problems slip through.

A useful migration audit is forensic, specific to the implementation, and connected to developer-ready work. It establishes what search engines can access before the move, identifies the behavior that must be preserved, and verifies the result after launch. This guide covers the audit areas, implementation controls, and acceptance criteria that matter for domain migration SEO and replatform SEO.

Start by defining the migration and its acceptance criteria

Before auditing URLs, clarify what is changing. Document the old and new domains, URL patterns, [SEO audit service package](#) CMS, JavaScript framework, rendering method, hosting or CDN changes, market versions, and launch sequence. Also record whether the migration includes a redesign, content consolidation, navigation changes, or a change to analytics and consent tools. Combining these changes increases the number of possible failure points.



Agree on acceptance criteria before development begins. “SEO-ready” is not an acceptance criterion. Define observable conditions that can be tested in staging and production. For example:

- Every in-scope legacy URL resolves to its approved final destination in one redirect hop, or has a documented reason for a different outcome.
- Priority pages return the intended status code, have a self-referencing canonical, and are indexable unless the migration plan says otherwise.
- Rendered HTML contains the primary content, links, title, and canonical required for search engines and users.
- XML sitemaps contain only canonical, indexable URLs returning a successful response.
- Analytics events and conversions are validated in GA4 before and after launch.

Set owners and evidence requirements for each criterion. A passing screenshot is not enough if the underlying response code or rendered DOM is wrong.

Forensic technical SEO audit versus checklist audit

A checklist audit asks whether standard elements are present: redirects, canonicals, sitemaps, robots directives, and analytics. That is useful for coverage, especially when teams need a shared minimum standard. It does not necessarily reveal how the site behaves at scale or why a search engine may fail to process it.

A forensic technical SEO audit investigates the evidence behind the risk. It compares crawl data with server logs, checks actual rendered output, samples URL patterns, and traces individual URLs through redirects and canonicalization. It asks questions such as: Are important URLs being crawled? Which templates generate crawl traps? Does the rendered page expose the same canonical as the initial HTML? Are Googlebot requests reaching the origin, or being served a cached error?

Use both approaches. The checklist provides repeatable coverage; forensic analysis identifies site-specific failure modes. Services and platforms such as Four Dots, SEO-Audits.com, and Reportz.io may be part of a team's audit, reporting, or workflow toolkit. Regardless of provider, require findings to include evidence, affected URL examples, implementation details, and a testable outcome—not just a score or a list of generic recommendations.

Build a trustworthy baseline before the move

The old site is the reference point for preserving organic value. Do not rely on a single crawl or sitemap export. Build an inventory from multiple sources, then reconcile the differences.

- Export known URLs from XML sitemaps, analytics landing-page reports, Search Console, the CMS, and backlink data.
- Crawl the site using representative user agents and record status codes, indexability, canonicals, titles, headings, internal links, and pagination signals.
- Identify high-value pages using organic entrances, conversions, links, impressions, and business importance.
- Capture templates and behaviors for product, category, article, location, search, and other page types.
- Record robots.txt rules, sitemap locations, hreflang annotations, structured data, and any special directives.

Normalize URL variations before comparing inventories. Decide how to handle protocol, hostname, trailing slash, case, parameters, and default pages. A redirect map that treats these inconsistently can create chains, loops, or duplicate destinations.

Domain migration SEO: map every valuable URL

For a domain move, the central task is preserving intent and relevance from old URLs to their closest equivalent on the new domain. A blanket redirect to the new homepage is not a substitute for mapping. It discards page-level signals and can create a poor user experience.



Audit area	What to inspect	Acceptance criteria
Redirect mapping	Old URL, destination, status, hop count, and page equivalence	In-scope URLs reach the intended final URL directly, with no loops or unnecessary chains
Canonicalization	Protocol, hostname, slash rules, parameters, and canonical tags	Each indexable destination resolves consistently and points to its preferred URL
Internal links	Navigation, content links, image links, and structured data URLs	Internal links point directly to the new destination rather than relying on redirects
Search engine signals	Robots directives, XML sitemaps, and supported domain-change processes	New properties and sitemaps are verified; old URLs remain available to redirect

Test the redirect map against the full URL inventory, not only a handful of examples. Include query strings where they affect content or tracking, and specify whether parameters should be retained, removed, or normalized. Test both HTTP and HTTPS, with and without “www” if applicable. Keep redirects in place long term; do not remove them once rankings appear stable.

Replatform SEO: audit templates, rendering, and crawl paths

Replatform SEO requires more than checking whether URLs still return 200. A new CMS or frontend can change page output, internal linking, pagination, metadata, structured data, and the way crawlers discover content. Compare equivalent page types between the old and new environments. A page-level comparison is useful, but template-level validation is essential because one defect can affect thousands of URLs.

JavaScript rendering and crawl behavior

For JavaScript-heavy sites, inspect both the initial HTML response and the rendered DOM. Confirm whether key content and links are present before JavaScript runs, what rendering depends on client-side APIs, and whether essential content appears reliably after rendering. Test representative routes with the same rendering behavior expected from search engines, and investigate differences in blocked scripts, hydration failures, delayed API responses, and client-side route handling.

Pay particular attention to links generated only after user interaction, infinite scroll, filters, and pagination. Search engines need discoverable crawl paths; a control that works for a person does not automatically create crawlable URLs. Decide which filtered or parameterized URLs should be crawlable and indexable, then enforce that decision consistently through links, canonicals, robots directives, and sitemap inclusion.

Server logs and crawl budget allocation

Server logs show requests that a crawler actually made, while a crawl shows what an audit tool could discover. Use logs to compare Googlebot activity before and after the migration, including requested paths, response codes, response times, and user-agent verification where available. Segment by hostname, template, and status code so that a spike in requests to obsolete URLs or parameter traps is visible.

Look for repeated requests to redirect chains, soft 404s, blocked resources, duplicate URLs, and low-value parameter combinations. Check whether important new URLs are being crawled and whether server or CDN behavior differs for verified bots. Crawl budget is not a reason to block useful content indiscriminately; the goal is to make crawling efficient by reducing waste and making priority pages easy to discover.

Turn findings into developer-ready specifications

Every high-priority finding should become a ticket that a developer can implement and a tester can verify. Avoid tasks such as “fix canonicals” or “improve crawlability.” Include the affected template or URL set, current behavior, expected behavior, implementation notes, dependencies, priority, and evidence needed for closure.

Ticket fieldExample detail IssueCategory pages emit a canonical to the old domain ScopeAll category pages using the commerce category template Expected behaviorCanonical uses the final HTTPS URL on the new hostname Implementation noteUpdate the CMS canonical helper and verify server-rendered output Acceptance testSample URLs return 200; HTML and rendered DOM expose the same canonical

Specify where the fix belongs. Is it in the CMS, edge redirects, application router, server rendering layer, or CDN configuration? A technically correct recommendation that ignores the platform may be unimplementable. Include staging test steps and a named owner, and track blockers in the team’s normal ticketing process.

Pre-launch and launch-day controls

Use a staging environment that matches production behavior as closely as possible. Password protection or access controls are usually safer than relying only on robots.txt to prevent staging URLs from appearing in search. Before launch, run the following checks:

1. Crawl staging and compare priority URLs and templates with the approved migration inventory.
2. Validate status codes, redirect destinations, canonicals, robots directives, hreflang, and structured data.
3. Check rendered content and internal links on JavaScript-dependent templates.
4. Confirm sitemap generation, robots.txt rules, analytics configuration, consent behavior, and conversion tracking.
5. Test redirects in bulk and manually inspect high-value, linked, and frequently visited URLs.
6. Prepare rollback conditions, monitoring owners, and communication channels. Keep the launch room active in Slack.

At launch, verify DNS and TLS, confirm that the intended site is accessible, and check a controlled sample of URLs immediately. Then run automated crawls and inspect server logs as data becomes available. Submit updated sitemaps and use applicable search engine change-of-address workflows for domain moves. Do not remove old-domain redirects or make multiple unplanned URL changes during the same monitoring window.

Post-launch monitoring: catch what breaks after launch

The migration is not complete when the new site loads. Monitor technical signals daily at first, then adjust frequency as behavior stabilizes. Compare trends with the baseline and annotate the launch date in reporting. In GA4, check landing pages, conversions, referral behavior, and any changes in session attribution. Pair analytics with Search Console and logs: analytics measures user activity, while logs and crawl data help explain crawler access.

- Watch for increases in 4xx and 5xx responses, redirect hops, and unexpected noindex or canonical changes.
- Track indexed and crawled URLs, sitemap processing, and the appearance of new-domain pages.
- Compare organic landing pages and conversions by template, market, and device—not only sitewide totals.
- Review log samples for crawl shifts, repeated requests to old URLs, and failures on key assets or APIs.
- Re-crawl priority templates and redirect samples after every significant deployment.

Set escalation thresholds in advance. For example, decide who investigates if priority pages return errors, if a template begins emitting the wrong canonical, or if organic landing-page data changes sharply. Separate normal migration volatility from defects with evidence, and avoid making reactive changes without tracing the affected URL pattern.

Make the audit actionable

A strong migration audit is not a PDF that ends with “monitor rankings.” It is a working set of URL inventories, risk findings, implementation tickets, test cases, and post-launch checks. The best migration process combines checklist coverage with forensic evidence, accounts for the CMS and rendering architecture, and assigns clear ownership from discovery through verification.

For domain migration SEO, protect URL-level relevance with accurate redirects and consistent signals. For replatform SEO, verify that templates, rendered content, and crawl paths preserve the site’s discoverability. In both cases, define acceptance criteria early, test them against production behavior, and keep monitoring after launch. That is how a migration moves from a hopeful deployment to a controlled technical change.