SkillNewton

Technical SEO Checklist

Technical SEO checks for robots.txt, sitemaps, canonicals, redirects, HTTPS, hreflang, JavaScript, Core Web Vitals, logs and schema.

This checklist covers the technical foundations that let search engines crawl, render and index a website efficiently. It is written for developers, SEO specialists and students who want a structured way to review a site's infrastructure. Work through it with Google Search Console, a site crawler and your browser's developer tools open. Fix anything that blocks indexing of important pages first, then move on to performance and efficiency improvements.

How to use it

1. **Crawl the site.** Run a full crawl with JavaScript rendering enabled so you can compare raw and rendered HTML.
2. **Open Search Console reports.** Keep the Page indexing, Sitemaps, Crawl stats and Core Web Vitals reports to hand.
3. **Check each section.** Mark each item as pass, fail or not applicable, with example URLs for every failure.
4. **Hand fixes to developers.** Write each fix as a clear ticket with the affected template, expected behaviour and how to test it.

robots.txt and crawl control

- [ ] Serve robots.txt at the root of each host
  It applies only to the protocol, host and port it is served on, so each subdomain needs its own file.
- [ ] Make sure important pages and resources are not disallowed
  Blocking CSS or JavaScript can stop Google rendering pages correctly.
- [ ] Do not use robots.txt to remove pages from the index
  Google does not support noindex in robots.txt; use a meta robots or X-Robots-Tag noindex instead.
- [ ] Reference the XML sitemap in robots.txt
  Add a `Sitemap:` line with the full sitemap URL.
- [ ] Check the robots.txt report in Search Console
  It shows which robots.txt files Google found, when they were fetched and any parse errors.
- [ ] Keep the file under 500 KiB
  Google ignores rules beyond its 500 KiB limit.

XML sitemaps

- [ ] Include only canonical, indexable URLs that return 200
  Remove redirects, 404s, noindexed and non-canonical URLs from the sitemap.
- [ ] Respect the size limits
  Each sitemap can hold up to 50,000 URLs or 50 MB uncompressed; use a sitemap index file for more.
- [ ] Keep lastmod accurate
  Google uses lastmod when it is consistently accurate and ignores priority and changefreq.
- [ ] Submit sitemaps in Search Console
  Check the Sitemaps report for fetch errors and the number of discovered URLs.

- [ ] Generate sitemaps automatically
  Let the CMS or build process update the sitemap when pages are added or removed.

Canonicals and duplicate URLs

- [ ] Add a self-referencing canonical to every indexable page
  Use an absolute URL in `<link rel="canonical">` in the head.
- [ ] Check canonicals point to 200, indexable URLs
  A canonical pointing to a redirect, 404 or noindexed page sends conflicting signals.
- [ ] Keep signals consistent
  Internal links, sitemaps, redirects and canonicals should all point to the same preferred URL.
- [ ] Compare declared and Google-selected canonicals
  URL Inspection shows both; a mismatch means Google sees stronger signals for another URL.
- [ ] Handle URL parameters
  Tracking, sorting and session parameters should canonicalise to the clean URL.

Status codes and redirects

- [ ] Return a real 404 or 410 for missing pages
  Pages that say "not found" but return 200 are reported as soft 404s.
- [ ] Use 301 or 308 for permanent moves
  Redirect each old URL to its closest equivalent, not everything to the homepage.
- [ ] Remove redirect chains and loops
  Redirect directly to the final URL; Googlebot follows up to 10 hops before giving up.
- [ ] Monitor 5xx server errors
  Frequent server errors can cause Google to slow crawling; check the Crawl stats report.
- [ ] Keep migration redirects in place for at least a year
  Google recommends keeping redirects for at least one year after a site move.

HTTPS and security

- [ ] Serve every page over HTTPS with a valid certificate
  Check for expiry dates and certificate warnings across subdomains.
- [ ] Redirect HTTP to HTTPS with a single 301
  Each HTTP URL should redirect straight to its HTTPS equivalent.
- [ ] Fix mixed content
  Images, scripts and stylesheets loaded over HTTP on HTTPS pages trigger browser warnings or are blocked.
- [ ] Consider HSTS
  The Strict-Transport-Security header tells browsers to use HTTPS only; roll it out carefully.
- [ ] Check the Security issues report in Search Console
  It flags hacked content, malware and social engineering detected by Google.

International SEO (hreflang)

- [ ] Add hreflang for every language or regional version
  Use ISO 639-1 language codes, optionally with an ISO 3166-1 alpha-2 region, for example `en-in`.

- [ ] Include return links
  Each version must link back to the others; hreflang without return links may be ignored.
- [ ] Include a self-reference and x-default
  List the page itself, and use `x-default` for a language selector or fallback page.
- [ ] Point hreflang only to canonical, 200 URLs
  Do not reference redirected, noindexed or non-canonical URLs.
- [ ] Use one method consistently
  Implement hreflang in HTML head, HTTP headers or the sitemap, not a mix that can conflict.

Rendering and JavaScript

- [ ] Compare raw HTML with the rendered HTML
  Use URL Inspection (View crawled page) to check that main content and links appear after rendering.
- [ ] Use real links with href attributes
  Google follows <a href> links; links that work only through onclick handlers may not be discovered.
- [ ] Prefer server-side rendering or static generation for key content
  Content available in the initial HTML is processed faster and more reliably.
- [ ] Do not rely on URL fragments for different content
  Google generally ignores everything after # when indexing.
- [ ] Return correct status codes from single-page apps
  Missing routes should return 404 or add a noindex, not show an error message with status 200.

Pagination and faceted navigation

- [ ] Link paginated pages with normal crawlable links
  Each page in the series should be reachable through links such as "Next" and numbered pages.
- [ ] Give each paginated page its own canonical
  Do not canonicalise page 2 onwards to page 1, as items listed there may be missed.
- [ ] Make infinite scroll crawlable
  Back infinite scroll with paginated URLs that load without JavaScript interaction.
- [ ] Control filter and sort URLs
  Decide which facets deserve indexing; canonicalise, noindex or block the rest to avoid crawl waste.

Core Web Vitals and performance

- [ ] LCP at or under 2.5 seconds
  Optimise server response, preload or prioritise the hero image and remove render-blocking resources.
- [ ] INP at or under 200 milliseconds
  Break up long tasks, defer non-critical JavaScript and audit third-party scripts.
- [ ] CLS at or under 0.1
  Set dimensions on media, reserve space for dynamic elements and avoid inserting content above existing content.
- [ ] Check field data, not just lab scores
  Search Console and PageSpeed Insights show real-user data from the Chrome UX Report at the 75th percentile.
- [ ] Use caching, compression and a CDN
  Enable Brotli or gzip, long cache lifetimes for static files and a CDN for global audiences.

Log files and structured data

- [ ] Analyse server logs for search engine bots
  See which URLs Googlebot requests, how often, and which important pages it rarely visits.
- [ ] Verify that bot traffic is genuine
  Confirm Googlebot with a reverse DNS lookup or Google's published IP ranges, as user agents can be faked.
- [ ] Find crawl waste
  Look for bots spending many requests on parameters, redirects or error pages.
- [ ] Validate structured data with the Rich Results Test
  Fix errors on each template and check the Enhancements reports in Search Console.
- [ ] Check structured data survives rendering
  If markup is injected with JavaScript, confirm it appears in the rendered HTML.

Online version, with the matching free guide: skillnewton.com/resources/templates/technical-seo-checklist/

Learn this hands-on: Technical SEO & Website Optimization – skillnewton.com/courses/technical-seo-course/