MIDNIGHT WEBS

FREE RESOURCE · PERFORMANCE CHECKLIST

Website Speed & Core Web Vitals Checklist

Google scores page experience on three numbers: how fast the main content appears, how quickly the page reacts, and how much it jumps around. This checklist shows how to read those numbers, find what is slowing your pages and keep them from getting worse.

Updated November 2026 | 12 min read | 39 checks

WHO SHOULD USE IT	WHAT IT COVERS	TARGETS
Anyone responsible for a site that feels slow or fails PageSpeed Insights	6 areas, from measurement to monitoring	LCP ≤ 2.5 s · INP ≤ 200 ms · CLS ≤ 0.1, at the 75th percentile of mobile visits

START HERE

Fix the largest image on your most visited page first. On most small-business sites, one oversized hero image is the biggest single cause of a slow Largest Contentful Paint.

TICK THE BOX WHEN DONE · MUST DO = HIGHEST IMPACT

01 Measure the pages that matter

Test the templates that carry visits and revenue. A perfect score on a page nobody opens changes nothing.

- [ ] MUST DO **Learn the three targets**
  Largest Contentful Paint (LCP) ≤ 2.5 s, Interaction to Next Paint (INP) ≤ 200 ms, Cumulative Layout Shift (CLS) ≤ 0.1. Google assesses them at the 75th percentile, meaning three quarters of real visits must meet each target. INP replaced First Input Delay in March 2024.

- [ ] MUST DO **Start with field data**
  In PageSpeed Insights, read the "Discover what your real users are experiencing" block first. It comes from the Chrome UX Report, covers the last 28 days and is what Google actually uses.

- [ ] **Use lab data to find causes**
  Lighthouse (in Chrome DevTools or PageSpeed Insights) points to specific files and elements. Lab scores vary between runs, so test three times and use the median.

- [ ] **Test each template, not just the homepage**
  Home, category, product or service, article and checkout. Pages on the same template tend to share the same problems.

- [ ] **Record a baseline**
  Date, URL, device and the three numbers, saved in a spreadsheet. Without a before, an after is only an opinion.

- [ ] **Separate mobile from desktop**
  Mobile is the stricter test, and Google indexes the mobile version first.

HOW TO CHECK

Check the Core Web Vitals report in Search Console. It groups your URLs into Good, Needs improvement and Poor, so you can see which template to fix.

02 Fix images and Largest Contentful Paint

On most sites the LCP element is an image, and images are the heaviest files on the page.

- [ ] MUST DO **Find your LCP element**
  In Chrome DevTools, Performance panel, or PageSpeed Insights diagnostics. Fix that specific element first; speeding up other images will not help.

- [ ] MUST DO **Do not lazy-load the LCP image**
  Use `fetchpriority="high"` on it and remove `loading="lazy"`. Lazy-loading everything below the visible area is right; applying it to the hero image delays it.

- [ ] **Use modern formats**
  AVIF or WebP are usually much smaller than JPEG at similar quality. Keep a JPEG fallback only if you must support very old browsers.

- [ ] **Serve the right size for each screen**
  Use `srcset` and sizes so a phone downloads a 600 px-wide image rather than a 3000 px original.

- [ ] **Always set width and height**
  So the browser reserves space before the image loads, which prevents layout shift.

- [ ] **Aim for a lean hero image**
  As a working target, keep the LCP image below about 200 KB on mobile; compress until you can see a difference, then step back one notch.

- [ ] **Use a CDN or image service**
  To resize, convert and cache automatically, so editors cannot upload a 6 MB photo by accident.

HOW TO CHECK

Re-run PageSpeed Insights after the change and confirm the LCP element and its time have improved. A waterfall in WebPageTest shows when the image actually starts downloading.

03 Reduce JavaScript and third-party weight

JavaScript is the main reason pages feel unresponsive, and slow responsiveness is what INP measures.

- [ ] MUST DO **List every third-party script and its owner**
  Analytics, chat, ads, pixels, A/B tools, consent banners, heatmaps. Remove anything that nobody has looked at in the last quarter.

- [ ] MUST DO **Load non-essential scripts late**
  Use `defer` or `async`, and load chat widgets or video embeds only when the visitor interacts. A static preview image ("facade") instead of a YouTube iframe saves hundreds of kilobytes.

- [ ] **Break up long tasks**
  Any task over 50 ms blocks the page from responding. Find them in the Performance panel and split the work or move it off the main thread.

- [ ] **Remove unused code**
  Check coverage in DevTools. Many sites ship a whole library for a single small feature.

- [ ] **Audit your tag manager**
  One container can load dozens of tags on every page. Restrict tags to the pages that need them.

- [ ] **Test with CPU throttling**
  Use 4x slowdown in DevTools to approximate a mid-range phone; your laptop hides the problem.

HOW TO CHECK

Click a menu, a filter or "add to cart" right after the page loads. If there is a visible lag, INP is probably in trouble; confirm it in the field data.

04 Fonts, CSS and layout stability

Text should appear at once and stay where it is. Late fonts, late CSS and late banners are the usual causes of jumping content.

- [ ] MUST DO **Reserve space for everything that loads late**
  Ads, embeds, cookie banners, newsletter bars and images. Use fixed dimensions or the CSS `aspect-ratio` property so nothing pushes content down.
- [ ] **Self-host fonts in WOFF2**
  It removes an extra connection to another domain and gives you control over caching.
- [ ] **Limit font files and weights**
  Two families and three weights is plenty. Subset to the languages you actually use.
- [ ] **Use `font-display: swap` and preload the key file**
  Preload only the one or two fonts used above the fold, with the `crossorigin` attribute.
- [ ] **Reduce font-swap shift**
  Match the fallback font's metrics with `size-adjust` and related descriptors, so the swap barely moves the text.
- [ ] **Keep critical CSS small**
  Inline what is needed for the first screen and load the rest without blocking rendering. Remove CSS from frameworks you no longer use.
- [ ] **Prefer inline SVG over icon fonts**
  A few SVGs are lighter and render without waiting for a font.

HOW TO CHECK

Load the page on a throttled connection and watch for movement. In Chrome DevTools, the Layout Shift regions option highlights exactly what moved.

05 Hosting, caching and server response

A fast front end cannot rescue a slow server. Time to First Byte (TTFB) delays everything that follows.

- [ ] MUST DO **Measure TTFB where your customers are**
  A good TTFB is under about 0.8 s, and poor above 1.8 s. A server in the wrong continent adds delay before any code runs.
- [ ] MUST DO **Cache static files for a long time**
  Fingerprinted CSS, JS, fonts and images can use `Cache-Control: public, max-age=31536000, immutable`. Repeat visits then download almost nothing.
- [ ] **Enable Brotli or Gzip compression**
  It typically cuts text files (HTML, CSS, JS, SVG) by more than half.

- [ ] **Put a CDN in front of the site**
  It serves files from a location near the visitor and absorbs traffic spikes.
- [ ] **Cache HTML carefully**
  Safe for public pages when you can purge it on every update; never for carts, accounts or checkout.
- [ ] **Remove redirect chains**
  Each redirect adds a full round trip. Link straight to the final address, including the https:// and trailing-slash form.
- [ ] **Review slow queries and heavy plugins**
  If TTFB is high only on dynamic pages, the database or a plugin is the likelier cause than the network.

HOW TO CHECK

In DevTools Network tab, click the main document and check Waiting for server response. Repeat from a phone on mobile data.

06 Test on real devices and monitor after launch

Performance drifts. New images, scripts and features undo good work unless someone owns the numbers.

- [ ] MUST DO **Test on a mid-range phone and a slow connection**
  Chrome DevTools offers Slow 4G and CPU throttling, but a real mid-range Android phone is more honest.
- [ ] MUST DO **Set a performance budget**
  For example, LCP image under 200 KB, JavaScript under 300 KB compressed, no more than a set number of third-party tags. Agree it with whoever adds features.
- [ ] **Measure real users continuously**
  The web-vitals JavaScript library sends LCP, INP and CLS from real visits to your analytics, so you do not wait for the 28-day report.
- [ ] **Add checks to your release process**
  Lighthouse CI or a scheduled PageSpeed test can warn when a change makes a key page slower.
- [ ] **Review the slowest pages monthly**
  Fix the largest cause first, record the result, repeat.
- [ ] **Re-test after changes to templates, themes or plugins**
  A single plugin update can add 200 KB of script.

HOW TO CHECK

Compare this month's Search Console Core Web Vitals report with last month's. The count of Poor URLs should be falling, not rising.

Sources and further reading

web.dev: Web Vitals (thresholds)
https://web.dev/articles/vitals

web.dev: Interaction to Next Paint
https://web.dev/articles/inp

web.dev: Optimize Largest Contentful Paint
https://web.dev/articles/optimize-lcp

web.dev: Time to First Byte
https://web.dev/articles/ttfb

Need a hand with this?

Midnight Webs designs, secures and maintains websites for small businesses. Tell us what you need and we will scope it.

midnightwebs.com/estimate

Read online: midnightwebs.com/resources/website-speed-core-web-vitals-checklist