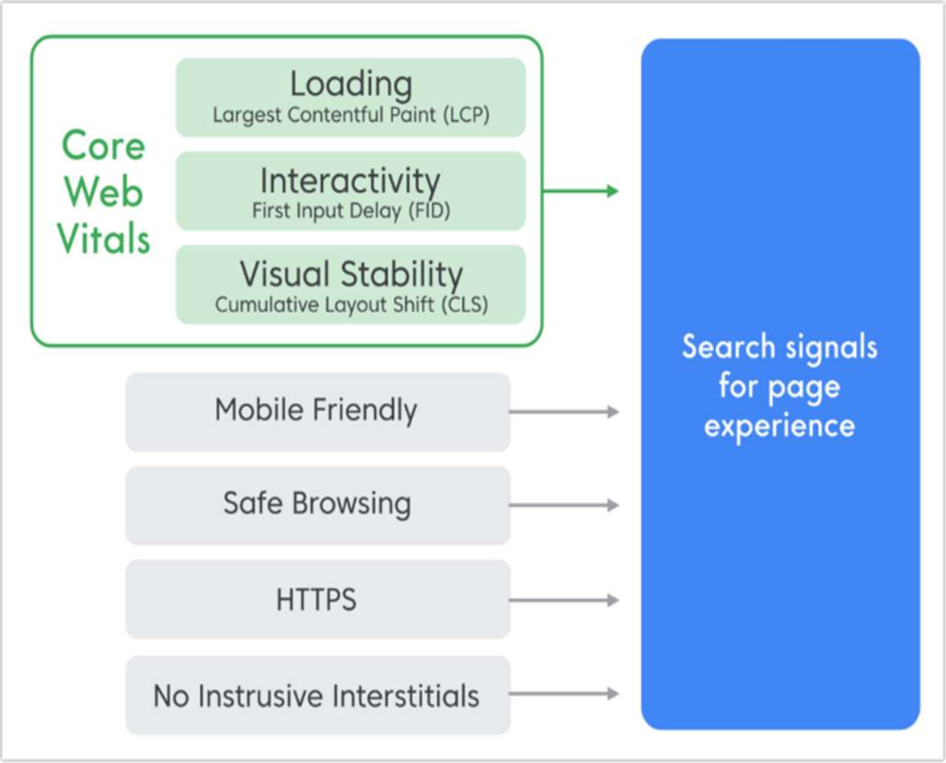
Introduction to Web Vitals:

- Google's Web Vitals Initiative
- Unified guidance for web page quality signals.
- Simplifies performance-measuring tools.
- Focuses on **Core Web Vitals** for improved user experience.

Core Web Vitals:

- Subset of Web Vitals applicable to all web pages.
- Measurable by all site owners and integrated into Google tools.
- Each of the Core Web Vitals represents a distinct facet of the user experience


Core Web Vitals
Loading
Largest Contentful Paint (LCP)
Interactivity
First Input Delay (FID)
Visual Stability
Cumulative Layout Shift (CLS)
Mobile Friendly
Safe Browsing
HTTPS
No Instrusive Interstitials
Search signals for page experience

Google's Page Experience combines Core Web Vitals + Web Search signals.

Current Core Web Vitals Metrics:

- Largest Contentful Paint (LCP)
- First Input Delay (FID)
- Cumulative Layout Shift (CLS)

Core Web Vitals Lifecycle Phases:

- Consists of three phases: Experimental, Pending, Stable.
- Each phase signifies the readiness and status of metrics.

Experimental
Pending
Stable

The stages of the Core Web Vitals lifecycle.

1- Experimental Phase

- Metrics undergoing testing and community feedback.
- Subject to significant changes based on evaluation.

2- Pending Phase

- Metrics that have passed testing and feedback.
- Have a defined timeline for stabilization.

3- Stable Phase

- Current essential Core Web Vitals for great user experiences.
- Includes metrics like Largest Contentful Paint (LCP), Cumulative Layout Shift (CLS), and Input Delay (INP).

Largest Contentful Paint

- Stable Core Web Vital metric for perceived load speed.
- Indicates when a page's main content is likely loaded during the page load timeline.

Purpose:

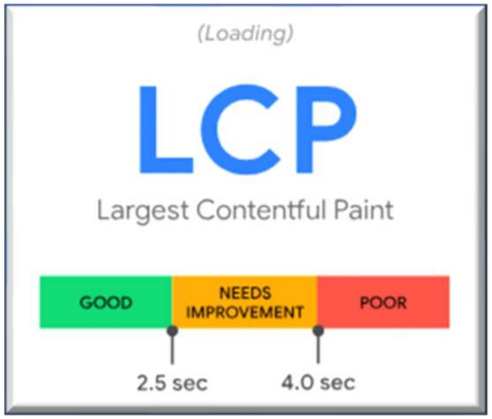
- Ensures a fast LCP to reassure users of a useful page.
- Resolves historical challenges of accurately measuring main content load times.

Challenges with Previous Metrics:

- Traditional metrics like load or DOMContentLoaded don't align with user-visible content.
- Newer metrics like First Contentful Paint (FCP) capture only initial loading moments.

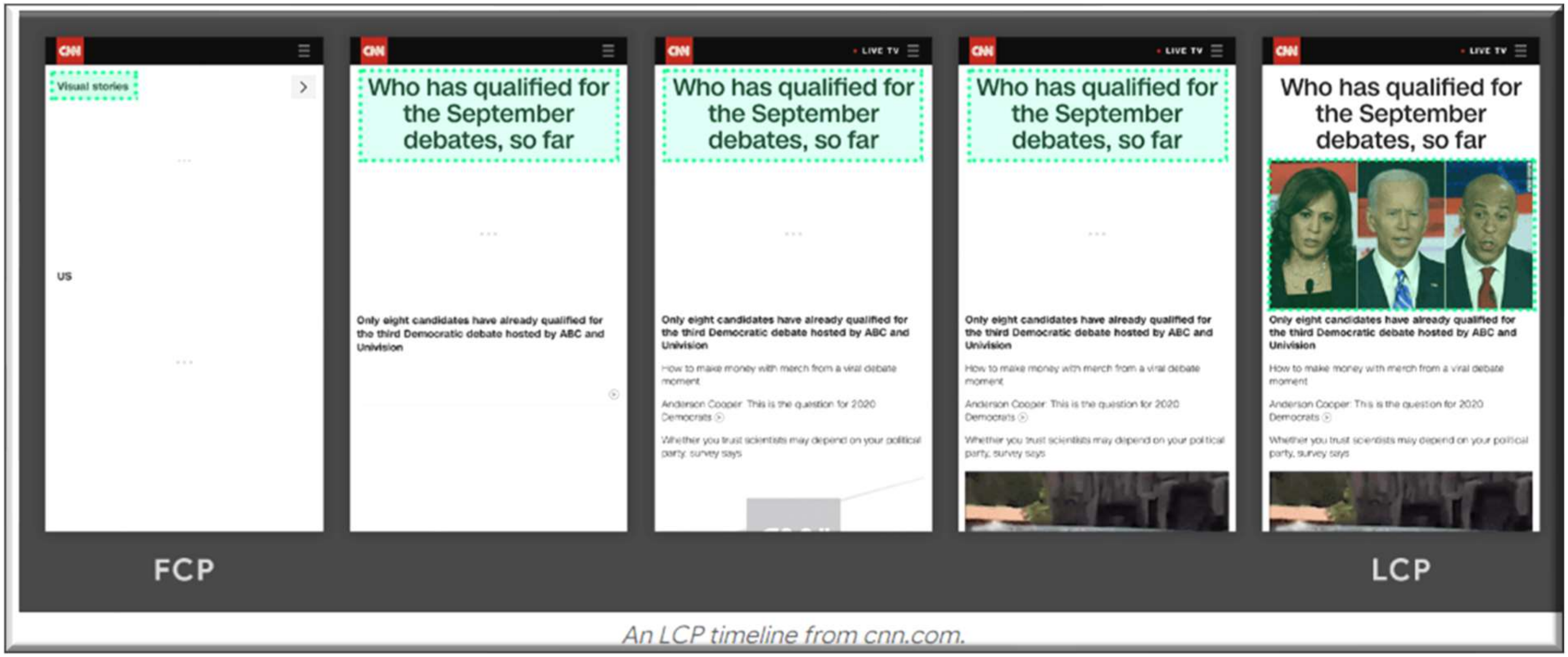
Element Considerations:

- Includes <img>, <image> inside <svg>, <video>, and text-containing block-level elements.
- Intentionally limited set to reduce complexity.



When is LCP Reported?

- Browser dispatches a PerformanceEntry of type largest-contentful-paint when the first frame is painted.
- Subsequent PerformanceEntry is dispatched whenever the largest contentful element changes during subsequent frame renderings.



Visual stories

US

Who has qualified for the September debates, so far

Only eight candidates have already qualified for the third Democratic debate hosted by ABC and Univision

How to make money with merch from a viral debate moment

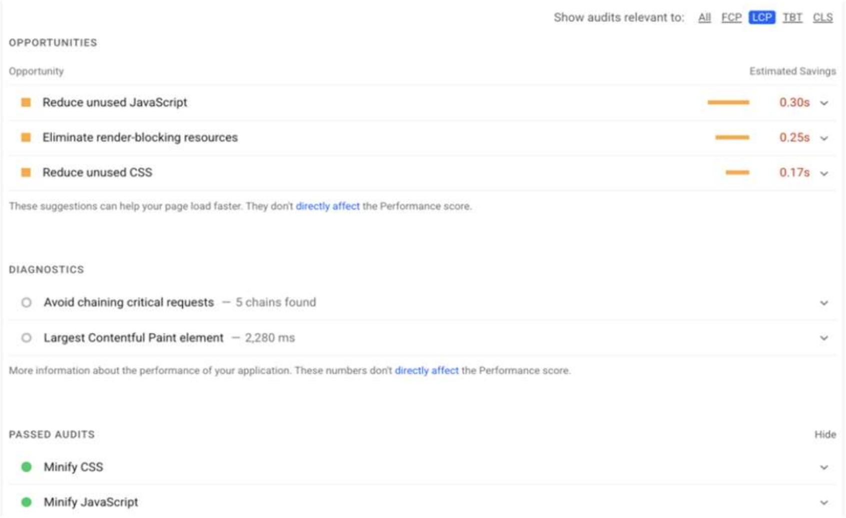
Anderson Cooper: This is the question for 2020 Democrats

Whether you trust scientists may depend on your political party, survey says

An LCP timeline from cnn.com.

Optimization Strategies in LCP

- **Reduce TTFB**: Minimize server response times, reduce redirects, and optimize caching to expedite initial content delivery.
- **Eliminate Resource Load Delay**: Ensure critical resources are discoverable and start loading early using preload hints or prioritization.
- **Optimize Resource Load Time**: Compress images, use modern formats (e.g., WebP), and leverage CDNs to minimize download times.
- **Reduce Element Render Delay**: Inline critical CSS, defer non-essential JavaScript, and avoid render-blocking resources for faster rendering.



Show audits relevant to: All FCP LCP TBT CLS

OPPORTUNITIES

Opportunity	Estimated Savings
Reduce unused JavaScript	0.30s
Eliminate render-blocking resources	0.25s
Reduce unused CSS	0.17s

These suggestions can help your page load faster. They don't directly affect the Performance score.

DIAGNOSTICS

○ Avoid chaining critical requests — 5 chains found

○ Largest Contentful Paint element — 2,280 ms

More information about the performance of your application. These numbers don't directly affect the Performance score.

PASSED AUDITS Hide

● Minify CSS

● Minify JavaScript

Lighthouse diagnostics and suggestions for improving LCP.

Cumulative Layout Shift

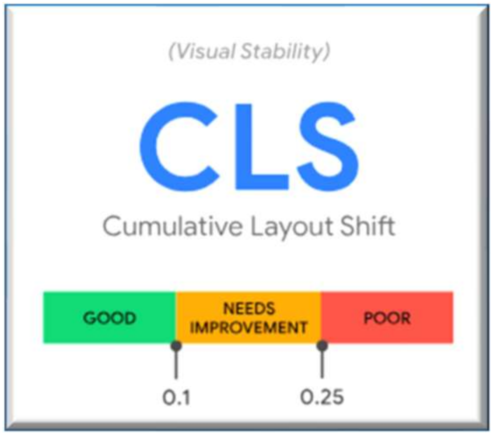
- Cumulative Layout Shift (CLS) is a stable Core Web Vital metric that measures visual stability.
- It quantifies unexpected layout shifts experienced by users, ensuring a delightful page experience.

Impact of Unexpected Layout Shifts:

- Disrupts user experience by causing loss of reading position or accidental clicks due to sudden text or element movements.
- Can lead to serious usability issues and user frustration.

Causes of Layout Shifts:

- Asynchronous loading of resources or dynamically added DOM elements.
- Elements like images, videos, or fonts rendering differently than expected.
- Dynamic resizing of third-party ads or widgets.



Cumulative Layout Shift (CLS) scenario that can disrupt user experience:

A sudden shift in layout makes the user confirm a large order they intended to cancel.

Optimization Strategies in CLS

Reserve Space for Content:

- Use CSS **aspect-ratio** or **min-height** to reserve space for dynamically loaded elements.

Delay Injected Content:

- Load ads, iframes, and dynamic content after the initial layout or when user interactions trigger them.

Preload Critical Resources:

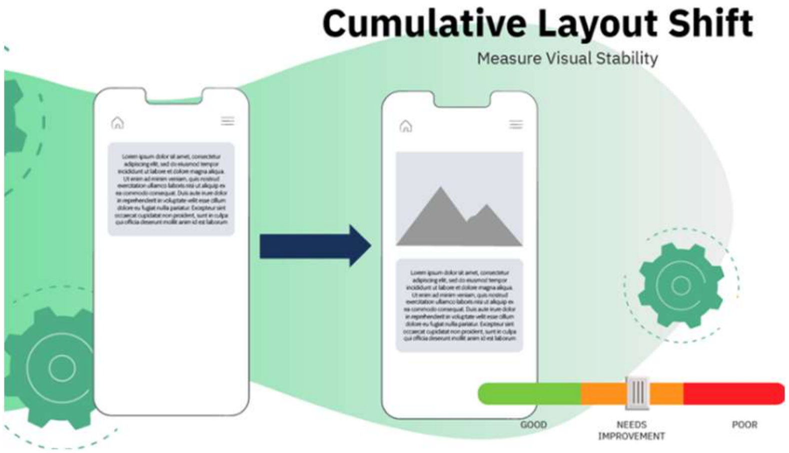
- Prioritize loading critical resources like web fonts and images to minimize shifts.

Use Responsive Image Techniques:

- Employ **srcset** and **<picture>** elements for responsive images to avoid unexpected layout changes.

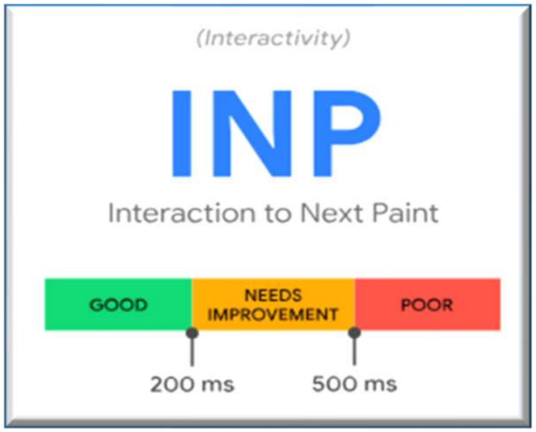
Avoid Animations Causing Layout Shifts:

- Use transform animations (**translate**, **scale**, **rotate**) instead of properties like **top** and **left** that trigger layout changes.



Interaction to Next Paint:

- INP is a stable Core Web Vital metric that evaluates page responsiveness using data from the Event Timing API.
- It observes latency for all click, tap, and keyboard interactions, reporting the longest duration while ignoring outliers.
- A low INP signifies consistent and quick responsiveness to user interactions.



Responsiveness Importance:

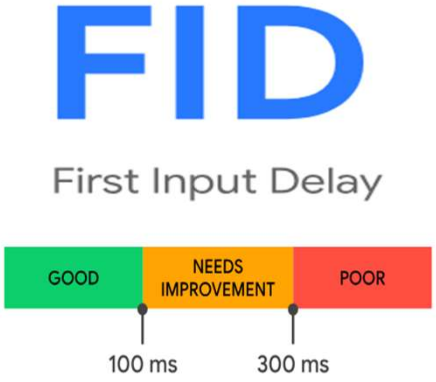
- Good responsiveness ensures the page swiftly responds to interactions, providing immediate visual feedback to users.
- Visual feedback confirms actions like adding items to a cart, opening menus, or authenticating logins.

First Input Delay:

- FID measures the delay between a user's first interaction with a page (e.g., click, tap, or custom control use) and when the browser can process the event.

Importance of FID:

- FID reflects a user's initial impression of a site's responsiveness during interaction.
- A good FID score of 100 milliseconds or less is optimal for ensuring a positive user experience.



First Input Delay an important metric to track

- FID is simply a real user metric and one of the most compelling web performance indicators.
- From an SEO perspective, it's now official that First Input Delay will begin to affect your website's rankings.

Optimization Strategies in FID

1. **Identify Heavy JavaScript**:

   •Excessive JavaScript execution on the main thread can cause delays.

   •Use Chrome DevTools to find Long Tasks (>50ms) that block user interactions.

2. **Break Up Long Tasks**:

   •Split long-running JavaScript into smaller, asynchronous tasks.

   •Reduces input delay and improves FID.

3. **Optimize Interaction Readiness**:

   •Optimize first-party script loading to avoid delays.

   •Minimize JavaScript size and execution times using progressive loading and code splitting.

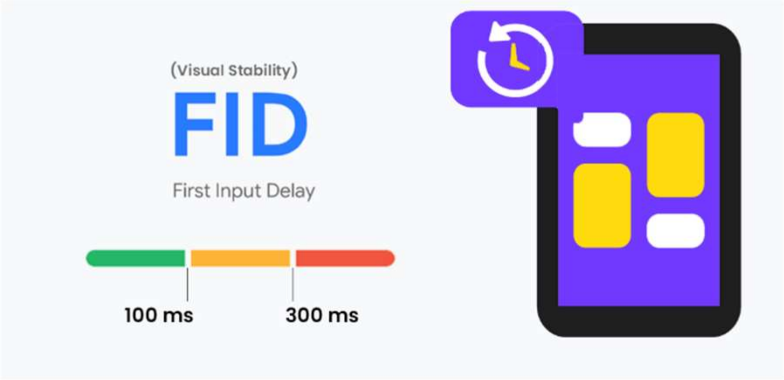
4. **Manage Data Fetching**:

   •Minimize reliance on cascading data fetches to reduce interaction latency.

5. **Handle Third-Party Scripts**:

   •Prioritize loading critical scripts first.

   •Manage third-party scripts to prevent network congestion.



6. Reduce JavaScript Execution Time:

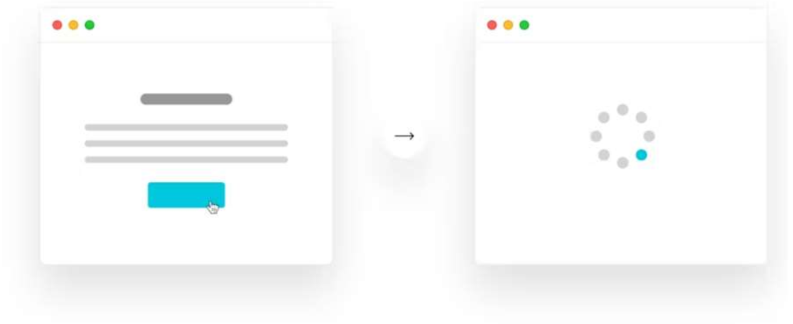
- Defer unused JavaScript to avoid render-blocking scripts.
- Minimize unused polyfills to optimize JavaScript performance.

7.Code Splitting:

- Implement code-splitting to load necessary JavaScript for critical-path content.
- Use dynamic imports for lazy-loading non-essential code.

8. Polyfill Optimization:

- Use **@babel/preset-env** to include necessary polyfills based on targeted browsers.
- Utilize the **module/nomodule** pattern for separate bundles targeting modern and legacy browsers.



First Input Delay (FID) is measured from the time a user interaction occurred to when the event handler was finally invoked.

Speed Index Measures:

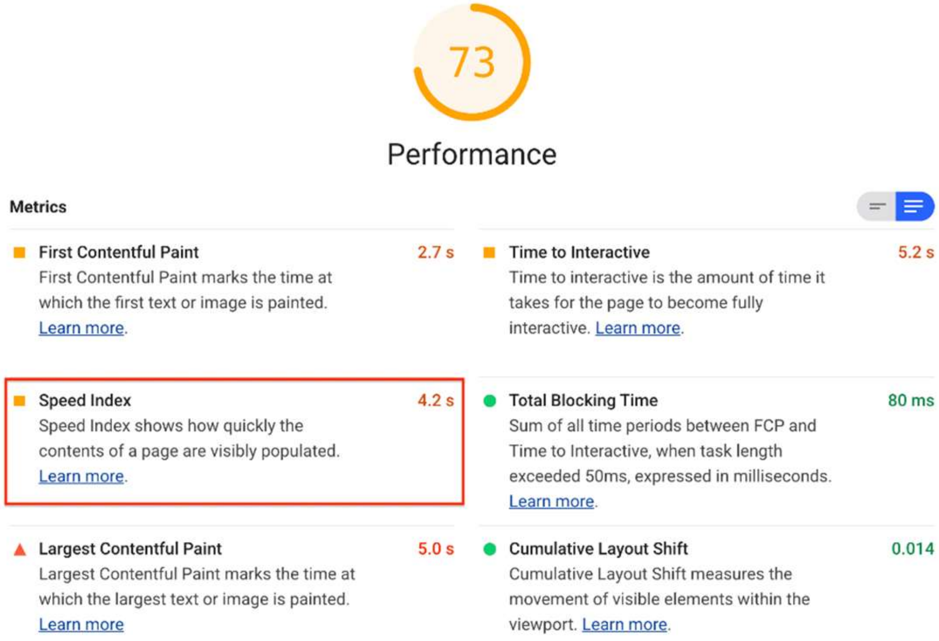
- Speed Index quantifies how quickly content visually appears during page load by analyzing frame progression in a video capture of the loading process.

Determining Speed Index Score:

- Your Speed Index score compares your page's speed with real-world benchmarks from the HTTP Archive.
- Interpretation:
  - 0–3.4 seconds: Green (fast)
  - 3.4–5.8 seconds: Orange (moderate)
  - Over 5.8 seconds: Red (slow)

Improving Speed Index:

  - Minimize main thread work.
  - Reduce JavaScript execution time.
  - Ensure text remains visible during webfont loading.



73

Performance

Metrics

■ First Contentful Paint 2.7 s
First Contentful Paint marks the time at which the first text or image is painted. Learn more.

■ Time to Interactive 5.2 s
Time to interactive is the amount of time it takes for the page to become fully interactive. Learn more.

■ Speed Index 4.2 s
Speed Index shows how quickly the contents of a page are visibly populated. Learn more.

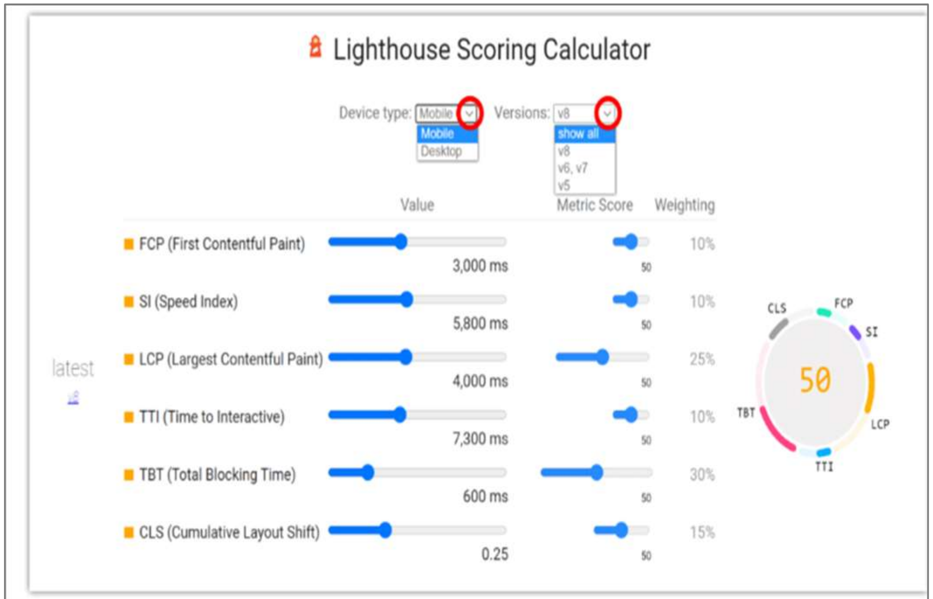
● Total Blocking Time 80 ms
Sum of all time periods between FCP and Time to Interactive, when task length exceeded 50ms, expressed in milliseconds. Learn more.

▲ Largest Contentful Paint 5.0 s
Largest Contentful Paint marks the time at which the largest text or image is painted. Learn more

● Cumulative Layout Shift 0.014
Cumulative Layout Shift measures the movement of visible elements within the viewport. Learn more.

Lighthouse Performance scoring details

1. **Performance Score Fluctuations**:
   - Changes in Performance score can stem from factors like A/B tests, network conditions, or device differences.
2. **Weighted Performance Score**:
   - The Performance score is a weighted average of key metrics that affect user perception:
     - First Contentful Paint: 10%
     - Speed Index: 10%
     - Largest Contentful Paint: 25%
     - Total Blocking Time: 30%
     - Cumulative Layout Shift: 25%
3. **Improving Performance Score**:
   - Use the Lighthouse scoring calculator to set performance thresholds.
   - Act on Opportunities and Diagnostics from the Lighthouse report for specific improvement suggestions.



Lighthouse Scoring Calculator

Device type: Mobile Versions: v8

	Value	Metric Score	Weighting
FCP (First Contentful Paint)	3,000 ms	50	10%
SI (Speed Index)	5,800 ms	50	10%
LCP (Largest Contentful Paint)	4,000 ms	50	25%
TTI (Time to Interactive)	7,300 ms	50	10%
TBT (Total Blocking Time)	600 ms	50	30%
CLS (Cumulative Layout Shift)	0.25	50	15%

Lighthouse scoring calculator allows you to select device type and scoring version.

4. Metric Score Calculation:

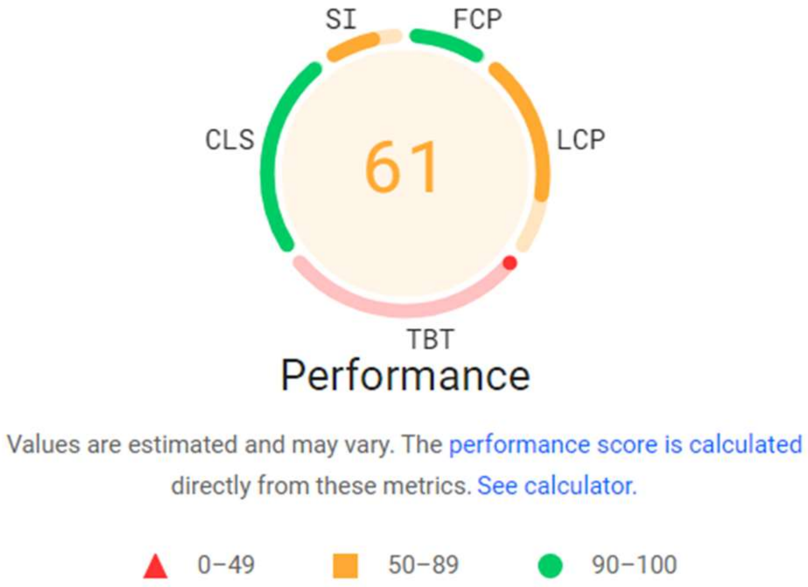
- Lighthouse converts raw metric values to a 0-100 score based on a log-normal distribution from real website performance data.

5. Desktop vs Mobile Scores:

- Lighthouse scores are based on real-world performance data, ensuring accurate representation for both desktop and mobile.

6. Color-coded Scores:

- Scores are color-coded for easy interpretation:
  - 0-49 (red): Poor
  - 50-89 (orange): Needs Improvement
  - 90-100 (green): Good



Values are estimated and may vary. The performance score is calculated directly from these metrics. See calculator.