Global Lighthouse Scores

View Lighthouse scores across the globe

URL	State	Device	Version	Request Time
https://shakeel.design	succeeded	Mobile	12.6.0	just now

Performance Scores

View Lighthouse Performance scores across all regions

	Score	FCP	LCP	TBT
US West (us-west1)	44 / 100	3.2s	17.1s	668ms
US East (us-east4)	62 / 100	3.4s	16.7s	278ms
Finland (europe-north1)	55 / 100	3.2s	17.2s	520ms
Germany (europe-west3)	66 / 100	3.2s	15.6s	102ms
Japan (asia-northeast1)	60 / 100	3.3s	16.4s	285ms
Australia (australia-southeast1)	62 / 100	3.2s	15.7s	257ms

Performance Metrics

Inspect each metric that leads to the performance score

	US West (us-west1)	US East (us-east4)	Finland (europe-north1)	Germany (europe-west3)	Japan (asia-northeast1)	Australia (australia-southeast1)
First Contentful Paint	3.2s	3.4s	3.2s	3.2s	3.3s	3.2s
Speed Index	4.3s	3.4s	3.2s	4.1s	3.8s	3.2s
Largest Contentful Paint	17.1s	16.7s	17.2s	15.6s	16.4s	15.7s
Total Blocking Time	668ms	278ms	520ms	102ms	285ms	257ms
Cumulative Layout Shift	0.15	0	0.02	0.07	0.07	0.07

Insights for US West

Regional details about your Lighthouse run

US West | Full Lighthouse Report

Transferred Assets

Review transferred assets and their size

Total Requests	Total Transferred Size	Total Resource Size
25	5.52 MB	6.11 MB

Lighthouse Audits

Review passed and failed audits for each Lighthouse score

Performance

44

Audits that have impact on the performance of your site

- Largest Contentful Paint

Largest Contentful Paint marks the time at which the largest text or image is painted. Learn more about the Largest Contentful Paint metric

- **Use efficient cache lifetimes**
  A long cache lifetime can speed up repeat visits to your page. Learn more
- **Layout shift culprits**
  Layout shifts occur when elements move absent any user interaction. Investigate the causes of layout shifts, such as elements being added, removed, or their fonts changing as the page loads
- **Document request latency**
  Your first network request is the most important. Reduce its latency by avoiding redirects, ensuring a fast server response, and enabling text compression
- **Font display**
  Consider setting font-display to swap or optional to ensure text is consistently visible. swap can be further optimized to mitigate layout shifts with font metric overrides
- **Improve image delivery**
  Reducing the download time of images can improve the perceived load time of the page and LCP. Learn more about optimizing image size
- **LCP request discovery**
  Optimize LCP by making the LCP image discoverable from the HTML immediately, and avoiding lazy-loading
- **Render blocking requests**
  Requests are blocking the page's initial render, which may delay LCP. Deferring or inlining can move these network requests out of the critical path
- **Eliminate render-blocking resources**
  Resources are blocking the first paint of your page. Consider delivering critical JS/CSS inline and deferring all non-critical JS/styles. Learn how to eliminate render-blocking resources
- **Properly size images**
  Serve images that are appropriately-sized to save cellular data and improve load time. Learn how to size images
- **Reduce unused JavaScript**
  Reduce unused JavaScript and defer loading scripts until they are required to decrease bytes consumed by network activity. Learn how to reduce unused JavaScript
- **Serve images in next-gen formats**
  Image formats like WebP and AVIF often provide better compression than PNG or JPEG, which means faster downloads and less data consumption.
  Learn more about modern image formats
- **Preconnect to required origins**
  Consider adding `preconnect` or `dns-prefetch` resource hints to establish early connections to important third-party origins. Learn how to preconnect to required origins
- **Avoid multiple page redirects**
  Redirects introduce additional delays before the page can be loaded. Learn how to avoid page redirects
- **Reduce the impact of third-party code**
  Third-party code can significantly impact load performance. Limit the number of redundant third-party providers and try to load third-party code after your page has primarily finished loading. Learn how to minimize third-party impact
- **Largest Contentful Paint element**
  This is the largest contentful element painted within the viewport. Learn more about the Largest Contentful Paint element
- **Avoid large layout shifts**
  These are the largest layout shifts observed on the page. Each table item represents a single layout shift, and shows the element that shifted the most. Below each item are possible root causes that led to the layout shift. Some of these layout shifts may not be included in the CLS metric value due to windowing. Learn how to improve CLS

- **First Contentful Paint**
  First Contentful Paint marks the time at which the first text or image is painted. Learn more about the First Contentful Paint metric
- **Total Blocking Time**
  Sum of all time periods between FCP and Time to Interactive, when task length exceeded 50ms, expressed in milliseconds. Learn more about the Total Blocking Time metric
- **Cumulative Layout Shift**
  Cumulative Layout Shift measures the movement of visible elements within the viewport. Learn more about the Cumulative Layout Shift metric
- **Speed Index**
  Speed Index shows how quickly the contents of a page are visibly populated. Learn more about the Speed Index metric
- **Optimize DOM size**

A large DOM can increase the duration of style calculations and layout reflows, impacting page responsiveness. A large DOM will also increase memory usage.
Learn how to avoid an excessive DOM size

- **Forced reflow**
  Many APIs, typically reading layout geometry, force the rendering engine to pause script execution in order to calculate the style and layout. Learn more about forced reflow and its mitigations

- **Network dependency tree**
  Avoid chaining critical requests by reducing the length of chains, reducing the download size of resources, or deferring the download of unnecessary resources to improve page load

- **Optimize viewport for mobile**
  Tap interactions may be delayed by up to 300 ms if the viewport is not optimized for mobile

- **Time to Interactive**
  Time to Interactive is the amount of time it takes for the page to become fully interactive. Learn more about the Time to Interactive metric

- **Max Potential First Input Delay**
  The maximum potential First Input Delay that your users could experience is the duration of the longest task. Learn more about the Maximum Potential First Input Delay metric

- **Defer offscreen images**
  Consider lazy-loading offscreen and hidden images after all critical resources have finished loading to lower time to interactive. Learn how to defer offscreen images

- **Minify CSS**
  Minifying CSS files can reduce network payload sizes. Learn how to minify CSS

- **Minify JavaScript**
  Minifying JavaScript files can reduce payload sizes and script parse time. Learn how to minify JavaScript

- **Reduce unused CSS**
  Reduce unused rules from stylesheets and defer CSS not used for above-the-fold content to decrease bytes consumed by network activity. Learn how to reduce unused CSS

- **Efficiently encode images**
  Optimized images load faster and consume less cellular data. Learn how to efficiently encode images

- **Enable text compression**
  Text-based resources should be served with compression (gzip, deflate or brotli) to minimize total network bytes. Learn more about text compression

- **Initial server response time was short**
  Keep the server response time for the main document short because all other requests depend on it. Learn more about the Time to First Byte metric

- **Use video formats for animated content**
  Large GIFs are inefficient for delivering animated content. Consider using MPEG4/WebM videos for animations and PNG/WebP for static images instead of GIF to save network bytes. Learn more about efficient video formats

- **Remove duplicate modules in JavaScript bundles**
  Remove large, duplicate JavaScript modules from bundles to reduce unnecessary bytes consumed by network activity.

- **Avoid serving legacy JavaScript to modern browsers**
  Polyfills and transforms enable legacy browsers to use new JavaScript features. However, many aren't necessary for modern browsers. Consider modifying your JavaScript build process to not transpile Baseline features, unless you know you must support legacy browsers. Learn why most sites can deploy ES6+ code without transpiling

- **Preload Largest Contentful Paint image**
  If the LCP element is dynamically added to the page, you should preload the image in order to improve LCP. Learn more about preloading LCP elements

- **Avoid enormous network payloads**
  Large network payloads cost users real money and are highly correlated with long load times. Learn how to reduce payload sizes

- **Serve static assets with an efficient cache policy**
  A long cache lifetime can speed up repeat visits to your page. Learn more about efficient cache policies

- **Avoids an excessive DOM size**
  A large DOM will increase memory usage, cause longer style calculations, and produce costly layout reflows. Learn how to avoid an excessive DOM size

- **JavaScript execution time**
  Consider reducing the time spent parsing, compiling, and executing JS. You may find delivering smaller JS payloads helps with this. Learn how to reduce Javascript execution time

- **Minimizes main-thread work**
  Consider reducing the time spent parsing, compiling and executing JS. You may find delivering smaller JS payloads helps with this. Learn how to minimize main-thread work

- **Ensure text remains visible during webfont load**
  Leverage the `font-display` CSS feature to ensure text is user-visible while webfonts are loading. Learn more about`font-display`
- **Largest Contentful Paint image was not lazily loaded**
  Above-the-fold images that are lazily loaded render later in the page lifecycle, which can delay the largest contentful paint. Learn more about optimal lazy loading
- **Uses passive listeners to improve scrolling performance**
  Consider marking your touch and wheel event listeners as `passive` to improve your page's scroll performance. Learn more about adopting passive event listeners
- **Avoid `document.write()`**
  For users on slow connections, external scripts dynamically injected via `document.write()` can delay page load by tens of seconds. Learn how to avoid document.write()
- **Image elements do not have explicit `width` and `height`**
  Set an explicit width and height on image elements to reduce layout shifts and improve CLS. Learn how to set image dimensions
- **Has a `<meta name="viewport">` tag with `width` or `initial-scale`**
  A `<meta name="viewport">` not only optimizes your app for mobile screen sizes, but also prevents a 300 millisecond delay to user input.
  Learn more about using the viewport meta tag

91 **Accessibility**
Opportunities to improve the accessibility for your visitors

100 **Best Practices**
Standards that your site should follow

100 **SEO**
Checks that ensure your page follows basic search engine advices