Results for: https://barakplasma.github.io/Performant-Photo-Gallery/
Jun 1, 2018, 3:21 PM GMT+3 • Runtime settings

| Performance | Progressive Web App | Accessibility | Best Practices | SEO |
|---|---|---|---|---|
| 95 | 82 | 100 | 94 | 89 |

## Performance

These encapsulate your web app's current performance and opportunities to improve it.

95

### Metrics

These metrics encapsulate your web app's performance across a number of dimensions.


- First meaningful paint — 1,250 ms
  First meaningful paint measures when the primary content of a page is visible. Learn more.
- First Interactive (beta) — 2,790 ms
  First Interactive marks the time at which the page is minimally interactive. Learn more.
- Consistently Interactive (beta) — 2,790 ms
  Consistently Interactive marks the time at which the page is fully interactive. Learn more.
- Perceptual Speed Index: 3,700 — 71
  Speed Index shows how quickly the contents of a page are visibly populated. Learn more.
- Estimated Input Latency: 17 ms — 100
  The score above is an estimate of how long your app takes to respond to user input, in milliseconds. There is a 90% probability that a user encounters this amount of latency, or less. 10% of the time a user can expect additional latency. If your latency is higher than 50 ms, users may perceive your app as laggy. Learn more.

### Opportunities

These are opportunities to speed up your application by optimizing the following resources.

- Offscreen images — 2,350 ms, 430 KB

  Consider lazy-loading offscreen and hidden images to improve page load speed and time to interactive. Learn more.

  View Details

| URL | Original | Potential Savings |
|---|---|---|
| /get/e83cb10…_640.jpg (pixabay.com) | 123 KB | 123 KB (100%) |
| /get/e830b70…_640.jpg (pixabay.com) | 74 KB | 74 KB (100%) |
| /get/e83cb10…_640.jpg (pixabay.com) | 54 KB | 54 KB (100%) |
| /get/e837b30…_640.jpg (pixabay.com) | 50 KB | 50 KB (100%) |
| /get/e835b50…_640.jpg (pixabay.com) | 49 KB | 49 KB (100%) |
| /get/ef3cb10…_640.jpg (pixabay.com) | 44 KB | 44 KB (100%) |
| /get/ea36b90…_640.jpg (pixabay.com) | 46 KB | 36 KB (77%) |

- Serve images in next-gen formats — 1,420 ms, 261 KB

  Image formats like JPEG 2000, JPEG XR, and WebP often provide better compression than PNG or JPEG, which means faster downloads and less data consumption. Learn more.

  View Details

| URL | Original | Potential Savings |
|---|---|---|
| /get/ea36b90…_640.jpg (pixabay.com) | 99 KB | 41 KB (41%) |
| /get/e83cb10…_640.jpg (pixabay.com) | 123 KB | 37 KB (30%) |
| /get/e135b00…_640.jpg (pixabay.com) | 98 KB | 34 KB (35%) |
| /get/e830b70…_640.jpg (pixabay.com) | 74 KB | 29 KB (40%) |
| /get/e83cb10…_640.jpg (pixabay.com) | 54 KB | 27 KB (51%) |
| /get/e837b30…_640.jpg (pixabay.com) | 50 KB | 26 KB (52%) |
| /get/e835b50…_640.jpg (pixabay.com) | 49 KB | 22 KB (45%) |
| /get/ef3cb10…_640.jpg (pixabay.com) | 44 KB | 22 KB (51%) |
| /get/ea36b90…_640.jpg (pixabay.com) | 46 KB | 21 KB (46%) |

- Reduce render-blocking stylesheets — 570 ms

  External stylesheets are blocking the first paint of your page. Consider delivering critical CSS via `<style>` tags and deferring non-critical styles. Learn more.

  View Details

| URL | Size (KB) | Delayed Paint By (ms) |
|---|---|---|

| URL | Size (KB) | Delayed Paint By (ms) |
|---|---|---|
| …css/main.f3863000.css (barakplasma.github.io) | 0.54 KB | 572 ms |

## Diagnostics

More information about the performance of your application.

- Uses inefficient cache policy on static assets: 11 assets found — 71
  A long cache lifetime can speed up repeat visits to your page. Learn more.
  - View Details

| URL | Cache TTL | Size (KB) |
|---|---|---|
| …js/main.5f0522a5.js (barakplasma.github.io) | 10 m | 83 KB |
| …css/main.f3863000.css (barakplasma.github.io) | 10 m | 1 KB |
| /get/e83cb10…_640.jpg (pixabay.com) | 1 d | 124 KB |
| /get/ea36b90…_640.jpg (pixabay.com) | 1 d | 100 KB |
| /get/e135b00…_640.jpg (pixabay.com) | 1 d | 99 KB |
| /get/e830b70…_640.jpg (pixabay.com) | 1 d | 75 KB |
| /get/e83cb10…_640.jpg (pixabay.com) | 1 d | 54 KB |
| /get/e837b30…_640.jpg (pixabay.com) | 1 d | 50 KB |
| /get/e835b50…_640.jpg (pixabay.com) | 1 d | 50 KB |
| /get/ea36b90…_640.jpg (pixabay.com) | 1 d | 47 KB |
| /get/ef3cb10…_640.jpg (pixabay.com) | 1 d | 44 KB |

- Uses an excessive DOM size: 2,125 nodes — 97
  Browser engineers recommend pages contain fewer than ~1,500 DOM nodes. The sweet spot is a tree depth < 32 elements and fewer than 60 children/parent element. A large DOM can increase memory usage, cause longer style calculations, and produce costly layout reflows. Learn more.
  - View details

| Total DOM Nodes | DOM Depth | Maximum Children |
|---|---|---|
| 2,125 | 9 | 1,040 |
| target: < 1,500 nodes | target: < 32 | target: < 60 nodes |

- Critical Request Chains: 2
  The Critical Request Chains below show you what resources are issued with a high priority. Consider reducing the length of chains, reducing the download size of resources, or deferring the download of unnecessary resources to improve page load. Learn more.
  Longest chain: **2,067.6ms** over **2** requests, totalling **82.77 KB**
  - View critical network waterfall:

*Initial Navigation*
- /Performant-Photo-Gallery/ (barakplasma.github.io)
  - …css/main.f3863000.css (barakplasma.github.io) **- 571.9ms, 0.54 KB**
  - …js/main.5f0522a5.js (barakplasma.github.io) **- 1,016.7ms, 82.77 KB**

## 15 Passed Audits

- Reduce render-blocking scripts
  Script elements are blocking the first paint of your page. Consider inlining critical scripts and deferring non-critical ones. Learn more.
- Properly size images
  Serve images that are appropriately-sized to save cellular data and improve load time. Learn more.
- Minify CSS
  Minifying CSS files can reduce network payload sizes.Learn more.
- Minify JavaScript
  Minifying JavaScript files can reduce payload sizes and script parse time. Learn more.
- Unused CSS rules
  Remove unused rules from stylesheets to reduce unnecessary bytes consumed by network activity. Learn more
- Optimize images
  Optimized images load faster and consume less cellular data. Learn more.
- Enable text compression
  Text-based responses should be served with compression (gzip, deflate or brotli) to minimize total network bytes. Learn more.
- Keep server response times low (TTFB): 560 ms
  Time To First Byte identifies the time at which your server sends a response. Learn more.
- Avoids page redirects: 0 ms — ✓
  Redirects introduce additional delays before the page can be loaded. Learn more.
- Preload key requests: 0 ms — 100
  Consider using <link rel=preload> to prioritize fetching late-discovered resources sooner Learn more.
- Avoids enormous network payloads: Total size was 728 KB — 100
  Large network payloads cost users real money and are highly correlated with long load times. Learn more.
  - View Details

| URL | Total Size | Transfer Time |
|---|---|---|
| /get/e83cb10…_640.jpg (pixabay.com) | 124 KB | 680 ms |
| /get/ea36b90…_640.jpg (pixabay.com) | 100 KB | 540 ms |
| /get/e135b00…_640.jpg (pixabay.com) | 99 KB | 540 ms |
| …js/main.5f0522a5.js (barakplasma.github.io) | 83 KB | 450 ms |
| /get/e830b70…_640.jpg (pixabay.com) | 75 KB | 410 ms |
| /get/e83cb10…_640.jpg (pixabay.com) | 54 KB | 300 ms |
| /get/e837b30…_640.jpg (pixabay.com) | 50 KB | 280 ms |
| /get/e835b50…_640.jpg (pixabay.com) | 50 KB | 270 ms |

| URL | Total Size | Transfer Time |
|---|---|---|
| /get/ea36b90…_640.jpg (pixabay.com) | 47 KB | 260 ms |
| /get/ef3cb10…_640.jpg (pixabay.com) | 44 KB | 240 ms |

User Timing marks and measures: 0
Consider instrumenting your app with the User Timing API to create custom, real-world measurements of key user experiences. Learn more.

JavaScript boot-up time: 1,110 ms
Consider reducing the time spent parsing, compiling and executing JS. You may find delivering smaller JS payloads helps with this.

View Details

| URL | Script Evaluation | Script Parsing & Compile |
|---|---|---|
| …js/main.5f0522a5.js (barakplasma.github.io) | 362 ms | 1 ms |
| /js/ractive.js (cdbfbhnfamigibakdbmmnihbpjldojgl) | 44 ms | 52 ms |
| /vendor.min.js (ejfdniofoaeobdncklagejfcnfajnpgc) | 0 ms | 82 ms |
| /bubble_compiled.js (aapbdbdomjkkjkaonfhkkikfgjllcleb) | 39 ms | 34 ms |
| /js/main.js (cdbfbhnfamigibakdbmmnihbpjldojgl) | 48 ms | 1 ms |
| /js/pagewrap.bundle.js (lmhkpmbekcpmknklioeibfkpmmfibljd) | 40 ms | 2 ms |
| /js/content.bundle.js (lmhkpmbekcpmknklioeibfkpmmfibljd) | 21 ms | 10 ms |
| /lib/settings.js (dbepggeogbaibhgnhhndojpepiihcmeb) | 16 ms | 2 ms |
| /content/content.js (bfbameneiokkgbdmiekhjnmfkcnldhhm) | 11 ms | 7 ms |
| /lib/keyboard_utils.js (dbepggeogbaibhgnhhndojpepiihcmeb) | 11 ms | 1 ms |
| /js/contentscript.js (klbibkeccnjlkjkiokjodocebajanakg) | 11 ms | 1 ms |
| /content_scripts/vimium_frontend.js (dbepggeogbaibhgnhhndojpepiihcmeb) | 9 ms | 3 ms |
| /js/page-action.js (ajiejmhbejpdgkkigpddefnjmgcbkenk) | 10 ms | 1 ms |

Main thread work breakdown: 1,540 ms
Consider reducing the time spent parsing, compiling and executing JS.You may find delivering smaller JS payloads helps with this.

View Details

| Category | Work | Time spent |
|---|---|---|
| Script Evaluation | Evaluate Script | 663 ms |
| Script Evaluation | Run Microtasks | 10 ms |
| Style & Layout | Layout | 234 ms |
| Style & Layout | Recalculate Style | 69 ms |
| Script Parsing & Compile | Compile Script | 263 ms |
| Parsing HTML & CSS | Parse HTML | 115 ms |
| Parsing HTML & CSS | Parse Stylesheet | 1 ms |
| Garbage collection | Minor GC | 60 ms |
| Garbage collection | DOM GC | 34 ms |
| Garbage collection | Major GC | 12 ms |
| Compositing | Update Layer Tree | 58 ms |
| Compositing | Composite Layers | 9 ms |
| Paint | Paint | 14 ms |

All text remains visible during webfont loads
Leverage the font-display CSS feature to ensure text is user-visible while webfonts are loading. Learn more.

## Progressive Web App

These checks validate the aspects of a Progressive Web App, as specified by the baseline PWA Checklist.

82

### 2 Failed Audits

User will not be prompted to Install the Web App
Browsers can proactively prompt users to add your app to their homescreen, which can lead to higher engagement. Learn more.
Failures: Manifest does not have icons at least 192px.

Is not configured for a custom splash screen
A themed splash screen ensures a high-quality experience when users launch your app from their homescreens. Learn more.
Failures: Manifest does not have icons at least 512px.

### 9 Passed Audits

Registers a service worker
The service worker is the technology that enables your app to use many Progressive Web App features, such as offline, add to homescreen, and push notifications. Learn more.

Responds with a 200 when offline
If you're building a Progressive Web App, consider using a service worker so that your app can work offline. Learn more.

Contains some content when JavaScript is not available
Your app should display some content when JavaScript is disabled, even if it's just a warning to the user that JavaScript is required to use the app. Learn more.

Uses HTTPS
All sites should be protected with HTTPS, even ones that don't handle sensitive data. HTTPS prevents intruders from tampering with or passively listening in on the communications between your app and your users, and is a prerequisite for HTTP/2 and many new web platform APIs. Learn more.

Redirects HTTP traffic to HTTPS
If you've already set up HTTPS, make sure that you redirect all HTTP traffic to HTTPS. Learn more.

Page load is fast enough on 3G
A fast page load over a 3G network ensures a good mobile user experience. Learn more.

- Address bar matches brand colors ✓
  The browser address bar can be themed to match your site. Learn more.
- Has a `<meta name="viewport">` tag with `width` or `initial-scale` ✓
  Add a viewport meta tag to optimize your app for mobile screens. Learn more.
- Content is sized correctly for the viewport ✓
  If the width of your app's content doesn't match the width of the viewport, your app might not be optimized for mobile screens. Learn more.

### Additional items to manually check

These checks are required by the baseline PWA Checklist but are not automatically checked by Lighthouse. They do not affect your score but it's important that you verify them manually.

- Site works cross-browser
  To reach the most number of users, sites should work across every major browser. Learn more.
- Page transitions don't feel like they block on the network
  Transitions should feel snappy as you tap around, even on a slow network, a key to perceived performance. Learn more.
- Each page has a URL
  Ensure individual pages are deep linkable via the URLs and that URLs are unique for the purpose of shareability on social media. Learn more.

## Accessibility

100

These checks highlight opportunities to improve the accessibility of your web app. Only a subset of accessibility issues can be automatically detected so manual testing is also encouraged.

### 9 Passed Audits

#### Elements Use Attributes Correctly

These are opportunities to improve the configuration of your HTML elements.

- Image elements have `[alt]` attributes ✓
  Informative elements should aim for short, descriptive alternate text. Decorative elements can be ignored with an empty alt attribute.Learn more.

#### Elements Describe Contents Well

These are opportunities to make your content easier to understand for a user of assistive technology, like a screen reader.

- Document has a `<title>` element ✓
  Screen reader users use page titles to get an overview of the contents of the page. Learn more.

#### Color Contrast Is Satisfactory

These are opportunities to improve the legibility of your content.

- Background and foreground colors have a sufficient contrast ratio ✓
  Low-contrast text is difficult or impossible for many users to read. Learn more.

#### Elements Are Well Structured

These are opportunities to make sure your HTML is appropriately structured.

- `[id]` attributes on the page are unique ✓
  The value of an id attribute must be unique to prevent other instances from being overlooked by assistive technologies. Learn more.
- Lists contain only `<li>` elements and script supporting elements (`<script>` and `<template>`). ✓
  Screen readers have a specific way of announcing lists. Ensuring proper list structure aids screen reader output. Learn more.
- List items (`<li>`) are contained within `<ul>` or `<ol>` parent elements ✓
  Screen readers require list items (`<li>`) to be contained within a parent `<ul>` or `<ol>` to be announced properly. Learn more.

#### Page Specifies Valid Language

These are opportunities to improve the interpretation of your content by users in different locales.

- `<html>` element has a `[lang]` attribute ✓
  If a page doesn't specify a lang attribute, a screen reader assumes that the page is in the default language that the user chose when setting up the screen reader. If the page isn't actually in the default language, then the screen reader might not announce the page's text correctly. Learn more.
- `<html>` element has a valid value for its `[lang]` attribute ✓
  Specifying a valid BCP 47 language helps screen readers announce text properly. Learn more.

#### Meta Tags Used Properly

These are opportunities to improve the user experience of your site.

- `[user-scalable="no"]` is not used in the `<meta name="viewport">` element and the `[maximum-scale]` attribute is not less than 5. ✓
  Disabling zooming is problematic for users with low vision who rely on screen magnification to properly see the contents of a web page. Learn more.

### 26 Not Applicable Audits

#### Elements Use Attributes Correctly

These are opportunities to improve the configuration of your HTML elements.

- `[accesskey]` values are not unique
  Access keys let users quickly focus a part of the page. For proper navigation, each access key must be unique. Learn more.
- `<audio>` elements are missing a `<track>` element with `[kind="captions"]`.
  Captions make audio elements usable for deaf or hearing-impaired users, providing critical information such as who is talking, what they're saying, and other non-speech information. Learn more.
- `<input type="image">` elements do not have `[alt]` text
  When an image is being used as an `<input>` button, providing alternative text can help screen reader users understand the purpose of the button. Learn more.
- Some elements have a `[tabindex]` value greater than 0
  A value greater than 0 implies an explicit navigation ordering. Although technically valid, this often creates frustrating experiences for users who rely on assistive technologies. Learn more.
- Cells in a `<table>` element that use the `[headers]` attribute refers to other cells of that same table.
  Screen readers have features to make navigating tables easier. Ensuring `<td>` cells using the `[headers]` attribute only refer to other cells in the same table may improve the experience for screen reader users. Learn more.
- `<th>` elements and elements with `[role="columnheader"/"rowheader"]` do not have data cells they describe.
  Screen readers have features to make navigating tables easier. Ensuring table headers always refer to some set of cells may improve the experience for screen reader users. Learn more.

#### ARIA Attributes Follow Best Practices

These are opportunities to improve the usage of ARIA in your application which may enhance the experience for users of assistive technology, like a screen reader.

- `[aria-*]` attributes do not match their roles
  Each ARIA `role` supports a specific subset of `aria-*` attributes. Mismatching these invalidates the `aria-*` attributes. Learn more.
- `[role]`s do not have all required `[aria-*]` attributes

Some ARIA roles have required attributes that describe the state of the element to screen readers. Learn more.

- Elements with `[role]` that require specific children `[role]`s, are missing.
  Some ARIA parent roles must contain specific child roles to perform their intended accessibility functions. Learn more.
- `[role]`s are not contained by their required parent element
  Some ARIA child roles must be contained by specific parent roles to properly perform their intended accessibility functions. Learn more.
- `[role]` values are not valid
  ARIA roles must have valid values in order to perform their intended accessibility functions. Learn more.
- `[aria-*]` attributes do not have valid values
  Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid values. Learn more.
- `[aria-*]` attributes are not valid or misspelled
  Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid names. Learn more.

### Elements Have Discernible Names

These are opportunities to improve the semantics of the controls in your application. This may enhance the experience for users of assistive technology, like a screen reader.

- Buttons do not have an accessible name
  When a button doesn't have an accessible name, screen readers announce it as "button", making it unusable for users who rely on screen readers. Learn more.
- Links do not have a discernible name
  Link text (and alternate text for images, when used as links) that is discernible, unique, and focusable improves the navigation experience for screen reader users. Learn more.

### Elements Describe Contents Well

These are opportunities to make your content easier to understand for a user of assistive technology, like a screen reader.

- The page does not contain a heading, skip link, or landmark region
  Adding ways to bypass repetitive content lets keyboard users navigate the page more efficiently. Learn more.
- `<frame>` or `<iframe>` elements do not have a title
  Screen reader users rely on frame titles to describe the contents of frames. Learn more.
- Form elements do not have associated labels
  Labels ensure that form controls are announced properly by assistive technologies, like screen readers. Learn more.
- Presentational `<table>` elements do not avoid using `<th>`, `<caption>` or the `[summary]` attribute.
  A table being used for layout purposes should not include data elements, such as the th or caption elements or the summary attribute, because this can create a confusing experience for screen reader users. Learn more.
- `<object>` elements do not have `[alt]` text
  Screen readers cannot translate non-text content. Adding alt text to `<object>` elements helps screen readers convey meaning to users. Learn more.
- `<video>` elements do not contain a `<track>` element with `[kind="captions"]`.
  When a video provides a caption it is easier for deaf and hearing impaired users to access its information. Learn more.
- `<video>` elements do not contain a `<track>` element with `[kind="description"]`.
  Audio descriptions provide relevant information for videos that dialogue cannot, such as facial expressions and scenes. Learn more.

### Elements Are Well Structured

These are opportunities to make sure your HTML is appropriately structured.

- `<dl>`'s do not contain only properly-ordered `<dt>` and `<dd>` groups, `<script>` or `<template>` elements.
  When definition lists are not properly marked up, screen readers may produce confusing or inaccurate output. Learn more.
- Definition list items are not wrapped in `<dl>` elements
  Definition list items (`<dt>` and `<dd>`) must be wrapped in a parent `<dl>` element to ensure that screen readers can properly announce them. Learn more.

### Page Specifies Valid Language

These are opportunities to improve the interpretation of your content by users in different locales.

- `[lang]` attributes do not have a valid value
  Specifying a valid BCP 47 language on elements helps ensure that text is pronounced correctly by a screen reader. Learn more.

### Meta Tags Used Properly

These are opportunities to improve the user experience of your site.

- The document uses `<meta http-equiv="refresh">`
  Users do not expect a page to refresh automatically, and doing so will move focus back to the top of the page. This may create a frustrating or confusing experience. Learn more.

## Additional items to manually check

These items address areas which an automated testing tool cannot cover. Learn more in our guide on conducting an accessibility review.

- The page has a logical tab order
  Tabbing through the page follows the visual layout. Users cannot focus elements that are offscreen. Learn more.
- Interactive controls are keyboard focusable
  Custom interactive controls are keyboard focusable and display a focus indicator. Learn more.
- The user's focus is directed to new content added to the page
  If new content, such as a dialog, is added to the page, the user's focus is directed to it. Learn more.
- User focus is not accidentally trapped in a region
  A user can tab into and out of any control or region without accidentally trapping their focus. Learn more.
- Custom controls have associated labels
  Custom interactive controls have associated labels, provided by aria-label or aria-labelledby. Learn more.
- Custom controls have ARIA roles
  Custom interactive controls have appropriate ARIA roles. Learn more.
- Visual order on the page follows DOM order
  DOM order matches the visual order, improving navigation for assistive technology. Learn more.
- Offscreen content is hidden from assistive technology
  Offscreen content is hidden with display: none or aria-hidden=true. Learn more.
- Headings don't skip levels
  Headings are used to create an outline for the page and heading levels are not skipped. Learn more.
- HTML5 landmark elements are used to improve navigation
  Landmark elements (<main>, <nav>, etc.) are used to improve the keyboard navigation of the page for assistive technology. Learn more.

1 Failed Audits

- Manifest's `short_name` will be truncated when displayed on homescreen ✗
  Make your app's `short_name` fewer than 12 characters to ensure that it's not truncated on homescreens. Learn more.

15 Passed Audits

- Avoids Application Cache ✓
  Application Cache is deprecated. Learn more.
- Avoids WebSQL DB ✓
  Web SQL is deprecated. Consider using IndexedDB instead. Learn more.
- Uses HTTPS ✓
  All sites should be protected with HTTPS, even ones that don't handle sensitive data. HTTPS prevents intruders from tampering with or passively listening in on the communications between your app and your users, and is a prerequisite for HTTP/2 and many new web platform APIs. Learn more.
- Uses HTTP/2 for its own resources ✓
  HTTP/2 offers many benefits over HTTP/1.1, including binary headers, multiplexing, and server push. Learn more.
- 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.
- Avoids Mutation Events in its own scripts ✓
  Mutation Events are deprecated and harm performance. Consider using Mutation Observers instead. Learn more.
- Avoids `document.write()` ✓
  For users on slow connections, external scripts dynamically injected via `document.write()` can delay page load by tens of seconds. Learn more.
- Opens external anchors using `rel="noopener"` ✓
  Open new tabs using `rel="noopener"` to improve performance and prevent security vulnerabilities. Learn more.
- Avoids requesting the geolocation permission on page load ✓
  Users are mistrustful of or confused by sites that request their location without context. Consider tying the request to user gestures instead. Learn more.
- Avoids front-end JavaScript libraries with known security vulnerabilities ✓
  Some third-party scripts may contain known security vulnerabilities that are easily identified and exploited by attackers.
- Avoids requesting the notification permission on page load ✓
  Users are mistrustful of or confused by sites that request to send notifications without context. Consider tying the request to user gestures instead. Learn more.
- Avoids deprecated APIs ✓
  Deprecated APIs will eventually be removed from the browser. Learn more.
- Allows users to paste into password fields ✓
  Preventing password pasting undermines good security policy. Learn more
- No browser errors logged to the console ✓
  Errors logged to the console indicate unresolved problems. They can come from network request failures and other browser concerns.
- Displays images with correct aspect ratio ✓
  Image display dimensions should match natural aspect ratio.

## SEO

89

These checks ensure that your page is optimized for search engine results ranking. There are additional factors Lighthouse does not check that may affect your search ranking. Learn more.

Content Best Practices
Format your HTML in a way that enables crawlers to better understand your app's content.

- Document does not have a meta description ✗
  Meta descriptions may be included in search results to concisely summarize page content. Learn more.

8 Passed Audits

- Mobile Friendly
  Make sure your pages are mobile friendly so users don't have to pinch or zoom in order to read the content pages. Learn more.
  - Has a `<meta name="viewport">` tag with `width` or `initial-scale` ✓
    Add a viewport meta tag to optimize your app for mobile screens. Learn more.
  - Document uses legible font sizes ✓
    Font sizes less than 16px are too small to be legible and require mobile visitors to "pinch to zoom" in order to read. Strive to have >75% of page text ≥16px. Learn more.
    - View Details

| Source | Selector | % of Page Text | Font Size |
|---|---|---|---|
| Legible text | | 100.00% | ≥ 16px |

- Content Best Practices
  Format your HTML in a way that enables crawlers to better understand your app's content.
  - Document has a `<title>` element ✓
    Screen reader users use page titles to get an overview of the contents of the page. Learn more.
  - Links have descriptive text ✓
    Descriptive link text helps search engines understand your content. Learn more.
  - Document has a valid `hreflang` ✓
    hreflang allows crawlers to discover alternate translations of the page content. Learn more.
  - Document avoids plugins ✓
    Most mobile devices do not support plugins, and many desktop browsers restrict them.
- Crawling and Indexing
  To appear in search results, crawlers need access to your app.
  - Page has successful HTTP status code ✓
    Pages with unsuccessful HTTP status codes may not be indexed properly. Learn more.
  - Page isn't blocked from indexing ✓
    The "Robots" directives tell crawlers how your content should be indexed. Learn more.

1 Not Applicable Audits

- Content Best Practices
  Format your HTML in a way that enables crawlers to better understand your app's content.

- Document has a valid `rel=canonical`
  Canonical links suggest which URL to show in search results. Read more in Use canonical URLs.

## Additional items to manually check

Run these additional validators on your site to check additional SEO best practices.

- Page is mobile friendly
  Take the Mobile-Friendly Test to check for audits not covered by Lighthouse, like sizing tap targets appropriately. Learn more.
- Structured data is valid
  Run the Structured Data Testing Tool and the Structured Data Linter to validate structured data. Learn more.