https://klohne.github.io/Projet-5/

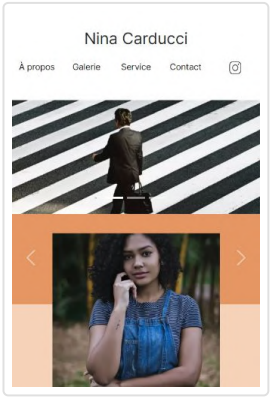
Performance	Accessibility	SEO
97	100	100

97

Performance

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

▲ 0–49 50–89 90–100



METRICS Expand view

First Contentful Paint
1.7 s

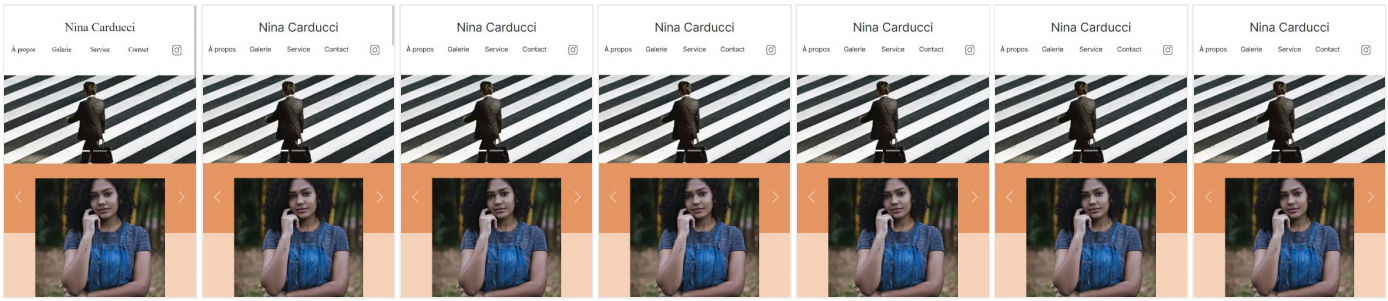
Largest Contentful Paint
2.4 s

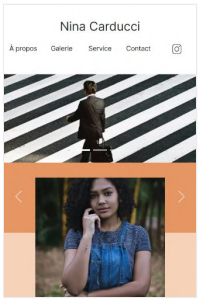
Total Blocking Time
20 ms

Cumulative Layout Shift
0.002

Speed Index
1.7 s

View Treemap





Show audits relevant to: All FCP LCP TBT CLS

DIAGNOSTICS

▲ Eliminate render-blocking resources — Potential savings of 670 ms

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. FCP LCP

☑ Show 3rd-party resources (2)

URL	Transfer Size	Potential Savings
JSDelivr CDN Cdn	**26.2 KiB**	**970 ms**
…css/bootstrap.min.css (cdn.jsdelivr.net)	26.2 KiB	970 ms
Google Fonts Cdn	**1.3 KiB**	**790 ms**
/css2?family=… (fonts.googleapis.com)	1.3 KiB	790 ms
GitHub Utility 1st Party	**22.9 KiB**	**450 ms**
…bootstrap/bootstrap.bundle.min.js (klohne.github.io)	22.9 KiB	450 ms

▲ Reduce unused CSS — Potential savings of 25 KiB

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. FCP LCP

URL	Transfer Size	Potential Savings
JSDelivr CDN Cdn	**26.2 KiB**	**25.1 KiB**
…css/bootstrap.min.css (cdn.jsdelivr.net)	26.2 KiB	25.1 KiB

Serve static assets with an efficient cache policy — 9 resources found

A long cache lifetime can speed up repeat visits to your page. Learn more about efficient cache policies.

URL	Cache TTL	Transfer Size
GitHub Utility 1st Party		**133 KiB**
…max%20466/aaron-pau….webp (klohne.github.io)	10m	41 KiB
…700/ryoji-iwa….webp (klohne.github.io)	10m	34 KiB
…bootstrap/bootstrap.bundle.min.js (klohne.github.io)	10m	23 KiB
…max%20466/nina.webp (klohne.github.io)	10m	22 KiB
…max%20466/ali-morsh….webp (klohne.github.io)	10m	9 KiB
…assets/maugallery.min.js (klohne.github.io)	10m	2 KiB
…assets/style.min.css (klohne.github.io)	10m	1 KiB
…images/instagram.webp (klohne.github.io)	10m	1 KiB
…assets/scripts.js (klohne.github.io)	10m	0 KiB

Avoid serving legacy JavaScript to modern browsers — Potential savings of 8 KiB

Polyfills and transforms enable legacy browsers to use new JavaScript features. However, many aren't necessary for modern browsers. For your bundled JavaScript, adopt a modern script deployment strategy using module/nomodule feature detection to reduce the amount of code shipped to modern browsers, while retaining support for legacy browsers. Learn how to use modern JavaScript TBT

URL	Potential Savings
`chrome-extension://omghfjlpggmjjaagoclmmobgdodcjboh/timezoneChange.js`	8.2 KiB
timezoneChange.js:1 `Date.now`	

Page prevented back/forward cache restoration — 3 failure reasons

Many navigations are performed by going back to a previous page, or forwards again. The back/forward cache (bfcache) can speed up these return navigations. Learn more about the bfcache

Failure reason	Failure type
Pages with an in-flight network request are not currently eligible for back/forward cache. /Projet-5/ (klohne.github.io)	Actionable
Back/forward cache is disabled due to extensions using messaging API. /Projet-5/ (klohne.github.io)	Pending browser support
Back/forward cache is disabled by flags. Visit chrome://flags/#back-forward-cache to enable it locally on this device. /Projet-5/ (klohne.github.io)	Not actionable

Minimize third-party usage — Third-party code blocked the main thread for 40 ms

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. TBT

Third-Party	Transfer Size	Main-Thread Blocking Time
jQuery CDN Cdn	**30 KiB**	**36 ms**
/jquery-3.4.1.min.js (code.jquery.com)	30 KiB	36 ms
Google Fonts Cdn	**62 KiB**	**0 ms**
…v13/UcC73FwrK….woff2 (fonts.gstatic.com)	46 KiB	0 ms
…v13/rnCu-xNNw….woff2 (fonts.gstatic.com)	15 KiB	0 ms
JSDelivr CDN Cdn	**26 KiB**	**0 ms**
…css/bootstrap.min.css (cdn.jsdelivr.net)	26 KiB	0 ms

JavaScript execution time — 0.4 s

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. TBT

Show 3rd-party resources (2)

URL	Total CPU Time	Script Evaluation	Script Parse
GitHub Utility 1st Party	**680 ms**	**88 ms**	**49 ms**
/Projet-5/ (klohne.github.io)	610 ms	46 ms	39 ms

URL	Total CPU Time	Script Evaluation	Script Parse
…bootstrap/bootstrap.bundle.min.js (klohne.github.io)	71 ms	42 ms	10 ms
jQuery CDN Cdn	**453 ms**	**241 ms**	**11 ms**
/jquery-3.4.1.min.js (code.jquery.com)	453 ms	241 ms	11 ms
Unattributable	**268 ms**	**14 ms**	**0 ms**
`Unattributable`	268 ms	14 ms	0 ms
JSDelivr CDN Cdn	**51 ms**	**0 ms**	**0 ms**
…css/bootstrap.min.css (cdn.jsdelivr.net)	51 ms	0 ms	0 ms

Minimizes main-thread work — 1.5 s

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 TBT

Category	Time Spent
Other	601 ms
Style & Layout	362 ms
Script Evaluation	354 ms
Script Parsing & Compilation	62 ms
Parse HTML & CSS	56 ms
Rendering	46 ms

Avoid long main-thread tasks — 7 long tasks found

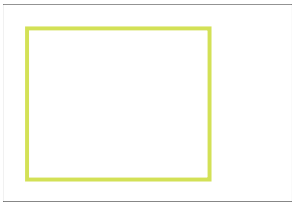
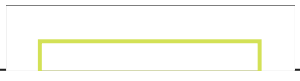
Lists the longest tasks on the main thread, useful for identifying worst contributors to input delay. Learn how to avoid long main-thread tasks TBT

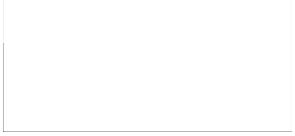
Show 3rd-party resources (3)

URL	Start Time	Duration
GitHub Utility 1st Party		**366 ms**
/Projet-5/ (klohne.github.io)	953 ms	126 ms
/Projet-5/ (klohne.github.io)	778 ms	107 ms
/Projet-5/ (klohne.github.io)	885 ms	68 ms
…bootstrap/bootstrap.bundle.min.js (klohne.github.io)	2,132 ms	65 ms
jQuery CDN Cdn		**132 ms**
/jquery-3.4.1.min.js (code.jquery.com)	2,430 ms	82 ms
/jquery-3.4.1.min.js (code.jquery.com)	2,512 ms	50 ms
JSDelivr CDN Cdn		**51 ms**
…css/bootstrap.min.css (cdn.jsdelivr.net)	2,081 ms	51 ms

Avoid large layout shifts — 5 elements found

These DOM elements contribute most to the CLS of the page. Learn how to improve CLS CLS

Element	CLS Contribution
 h3.about-me__introduction	0.001
h2.about-me__title	0.001
h1.name	0.000
 h1.name	0.000

Element	CLS Contribution
 a	0.000

Avoids an excessive DOM size — 133 elements

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

Statistic	Element	Value
Total DOM Elements		133
Maximum DOM Depth	div.mg-prev	9
Maximum Child Elements	 div.gallery-items-row.row	9

Initial server response time was short — Root document took 30 ms

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. FCP LCP

URL	Time Spent
GitHub Utility 1st Party	**30 ms**
/Projet-5/ (klohne.github.io)	30 ms

Avoids enormous network payloads — Total size was 255 KiB

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

Show 3rd-party resources (4)

URL	Transfer Size
GitHub Utility 1st Party	**132.8 KiB**
…max%20466/aaron-pau….webp (klohne.github.io)	41.4 KiB
…700/ryoji-iwa….webp (klohne.github.io)	33.7 KiB
…bootstrap/bootstrap.bundle.min.js (klohne.github.io)	22.9 KiB
…max%20466/nina.webp (klohne.github.io)	22.0 KiB
…max%20466/ali-morsh….webp (klohne.github.io)	8.5 KiB
/Projet-5/ (klohne.github.io)	4.2 KiB
Google Fonts Cdn	**60.6 KiB**
…v13/UcC73FwrK….woff2 (fonts.gstatic.com)	46.0 KiB
…v13/rnCu-xNNw….woff2 (fonts.gstatic.com)	14.6 KiB
jQuery CDN Cdn	**30.2 KiB**
/jquery-3.4.1.min.js (code.jquery.com)	30.2 KiB
JSDelivr CDN Cdn	**26.2 KiB**
…css/bootstrap.min.css (cdn.jsdelivr.net)	26.2 KiB

Avoid chaining critical requests — 5 chains found

The Critical Request Chains below show you what resources are loaded 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 how to avoid chaining critical requests. FCP LCP

Maximum critical path latency: **416.351 ms**

Initial Navigation

- /Projet-5/ (klohne.github.io)
  - …css/bootstrap.min.css (cdn.jsdelivr.net) **- 90.706 ms, 26.18 KiB**
  - …assets/style.min.css (klohne.github.io) **- 40.06 ms, 1.39 KiB**
  - /css2?family=… (fonts.googleapis.com)
    - …v13/UcC73FwrK….woff2 (fonts.gstatic.com) **- 56.133 ms, 46.00 KiB**

…v13/rnCu-xNNw….woff2 (fonts.gstatic.com) **- 59.148 ms, 14.58 KiB**

…bootstrap/bootstrap.bundle.min.js (klohne.github.io) **- 49.239 ms, 22.91 KiB**

Largest Contentful Paint element — 2,360 ms

This is the largest contentful element painted within the viewport. Learn more about the Largest Contentful Paint element LCP

Element
img.d-block.w-100.h-auto

Phase	% of LCP	Timing
TTFB	27%	630 ms
Load Delay	1%	20 ms
Load Time	16%	380 ms
Render Delay	57%	1,340 ms

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

PASSED AUDITS (24) Hide

Properly size images

Serve images that are appropriately-sized to save cellular data and improve load time. Learn how to size images.

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. FCP LCP

Minify JavaScript

Minifying JavaScript files can reduce payload sizes and script parse time. Learn how to minify JavaScript. FCP LCP

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. LCP

Efficiently encode images

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

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.

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. FCP LCP

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. FCP LCP

Avoid multiple page redirects

Redirects introduce additional delays before the page can be loaded. Learn how to avoid page redirects. FCP LCP

Preload key requests

Consider using `<link rel=preload>` to prioritize fetching resources that are currently requested later in page load. Learn how to preload key requests. FCP LCP

Use HTTP/2

HTTP/2 offers many benefits over HTTP/1.1, including binary headers and multiplexing. Learn more about HTTP/2.

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 LCP

Remove duplicate modules in JavaScript bundles

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

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. LCP

User Timing marks and measures

Consider instrumenting your app with the User Timing API to measure your app's real-world performance during key user experiences. Learn more about User Timing marks.

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 about `font-display`. FCP LCP

Lazy load third-party resources with facades

Some third-party embeds can be lazy loaded. Consider replacing them with a facade until they are required. Learn how to defer third-parties with a facade. TBT

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. LCP

Element
img.d-block.w-100.h-auto

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.

Avoids `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().

Avoid non-composited animations

Animations which are not composited can be janky and increase CLS. Learn how to avoid non-composited animations CLS

Image elements 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 CLS

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. TBT

Accessibility

These checks highlight opportunities to improve the accessibility of your web app. Automatic detection can only detect a subset of issues and does not guarantee the accessibility of your web app, so manual testing is also encouraged.

ADDITIONAL ITEMS TO MANUALLY CHECK (10) Hide

- Interactive controls are keyboard focusable

  Custom interactive controls are keyboard focusable and display a focus indicator. Learn how to make custom controls focusable.

- Interactive elements indicate their purpose and state

  Interactive elements, such as links and buttons, should indicate their state and be distinguishable from non-interactive elements. Learn how to decorate interactive elements with affordance hints.

- The page has a logical tab order

  Tabbing through the page follows the visual layout. Users cannot focus elements that are offscreen. Learn more about logical tab ordering.

- Visual order on the page follows DOM order

  DOM order matches the visual order, improving navigation for assistive technology. Learn more about DOM and visual ordering.

- 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 how to avoid focus traps.

- 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 how to direct focus to new content.

- 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 about landmark elements.

- Offscreen content is hidden from assistive technology

  Offscreen content is hidden with display: none or aria-hidden=true. Learn how to properly hide offscreen content.

- Custom controls have associated labels

  Custom interactive controls have associated labels, provided by aria-label or aria-labelledby. Learn more about custom controls and labels.

- Custom controls have ARIA roles

  Custom interactive controls have appropriate ARIA roles. Learn how to add roles to custom controls.

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

PASSED AUDITS (20) Hide

- [aria-*] attributes match their roles

  Each ARIA role supports a specific subset of aria-* attributes. Mismatching these invalidates the aria-* attributes. Learn how to match ARIA attributes to their roles.

- [aria-hidden="true"] is not present on the document <body>

  Assistive technologies, like screen readers, work inconsistently when aria-hidden="true" is set on the document <body>. Learn how aria-hidden affects the document body.

- [aria-*] attributes have valid values

  Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid values. Learn more about valid values for ARIA attributes.

- [aria-*] attributes are valid and not misspelled

Assistive technologies, like screen readers, can't interpret ARIA attributes with invalid names. Learn more about valid ARIA attributes.

Buttons 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 how to make buttons more accessible.

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 about the `alt` attribute.

Input buttons have discernible text.

Adding discernable and accessible text to input buttons may help screen reader users understand the purpose of the input button. Learn more about input buttons.

`[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 about the viewport meta tag.

`[aria-hidden="true"]` elements do not contain focusable descendents

Focusable descendents within an `[aria-hidden="true"]` element prevent those interactive elements from being available to users of assistive technologies like screen readers. Learn how `aria-hidden` affects focusable elements.

Background and foreground colors have a sufficient contrast ratio

Low-contrast text is difficult or impossible for many users to read. Learn how to provide sufficient color contrast.

Document has a `<title>` element

The title gives screen reader users an overview of the page, and search engine users rely on it heavily to determine if a page is relevant to their search. Learn more about document titles.

`<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 about the `lang` attribute.

`<html>` element has a valid value for its `[lang]` attribute

Specifying a valid BCP 47 language helps screen readers announce text properly. Learn how to use the `lang` attribute.

Form elements have associated labels

Labels ensure that form controls are announced properly by assistive technologies, like screen readers. Learn more about form element labels.

Links 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 how to make links accessible.

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 about proper list structure.

List items (`<li>`) are contained within `<ul>`, `<ol>` or `<menu>` parent elements

Screen readers require list items (<li>) to be contained within a parent <ul>, <ol> or <menu> to be announced properly. Learn more about proper list structure.

Heading elements appear in a sequentially-descending order

Properly ordered headings that do not skip levels convey the semantic structure of the page, making it easier to navigate and understand when using assistive technologies. Learn more about heading order.

Values assigned to `role=""` are valid ARIA roles.

ARIA `roles` enable assistive technologies to know the role of each element on the web page. If the `role` values are misspelled, not existing ARIA `role` values, or abstract roles, then the purpose of the element will not be communicated to users of assistive technologies. Learn more about ARIA roles.

Image elements do not have `[alt]` attributes that are redundant text.

Informative elements should aim for short, descriptive alternative text. Alternative text that is exactly the same as the text adjacent to the link or image is potentially confusing for screen reader users, because the text will be read twice. Learn more about the `alt` attribute.

NOT APPLICABLE (40) Hide

`[accesskey]` values are unique

Access keys let users quickly focus a part of the page. For proper navigation, each access key must be unique. Learn more about access keys.

- `button`, `link`, and `menuitem` elements have accessible names

  When an element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to make command elements more accessible.

- Elements with `role="dialog"` or `role="alertdialog"` have accessible names.

  ARIA dialog elements without accessible names may prevent screen readers users from discerning the purpose of these elements. Learn how to make ARIA dialog elements more accessible.

- ARIA input fields have accessible names

  When an input field doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about input field labels.

- ARIA `meter` elements have accessible names

  When a meter element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to name `meter` elements.

- ARIA `progressbar` elements have accessible names

  When a `progressbar` element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to label `progressbar` elements.

- `[role]`s have all required `[aria-*]` attributes

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

- Elements with an ARIA `[role]` that require children to contain a specific `[role]` have all required children.

  Some ARIA parent roles must contain specific child roles to perform their intended accessibility functions. Learn more about roles and required children elements.

- `[role]`s are 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 about ARIA roles and required parent element.

- `[role]` values are valid

  ARIA roles must have valid values in order to perform their intended accessibility functions. Learn more about valid ARIA roles.

- Elements with the `role=text` attribute do not have focusable descendents.

  Adding `role=text` around a text node split by markup enables VoiceOver to treat it as one phrase, but the element's focusable descendents will not be announced. Learn more about the `role=text` attribute.

- ARIA toggle fields have accessible names

  When a toggle field doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about toggle fields.

- ARIA `tooltip` elements have accessible names

  When a tooltip element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn how to name `tooltip` elements.

- ARIA `treeitem` elements have accessible names

  When a `treeitem` element doesn't have an accessible name, screen readers announce it with a generic name, making it unusable for users who rely on screen readers. Learn more about labeling `treeitem` elements.

- The page contains a heading, skip link, or landmark region

  Adding ways to bypass repetitive content lets keyboard users navigate the page more efficiently. Learn more about bypass blocks.

- `<dl>`'s contain only properly-ordered `<dt>` and `<dd>` groups, `<script>`, `<template>` or `<div>` elements.

  When definition lists are not properly marked up, screen readers may produce confusing or inaccurate output. Learn how to structure definition lists correctly.

- Definition list items are 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 how to structure definition lists correctly.

- `[id]` attributes on active, focusable elements are unique

  All focusable elements must have a unique `id` to ensure that they're visible to assistive technologies. Learn how to fix duplicate `ids`.

- ARIA IDs are unique

  The value of an ARIA ID must be unique to prevent other instances from being overlooked by assistive technologies. Learn how to fix duplicate ARIA IDs.

- No form fields have multiple labels

  Form fields with multiple labels can be confusingly announced by assistive technologies like screen readers which use either the first, the last, or all of the labels. Learn how to use form labels.

- `<frame>` or `<iframe>` elements have a title

  Screen reader users rely on frame titles to describe the contents of frames. Learn more about frame titles.

- `<html>` element has an `[xml:lang]` attribute with the same base language as the `[lang]` attribute.

  If the webpage does not specify a consistent language, then the screen reader might not announce the page's text correctly. Learn more about the `lang` attribute.

- `<input type="image">` elements 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 about input image alt text.

- Links are distinguishable without relying on color.

  Low-contrast text is difficult or impossible for many users to read. Link text that is discernible improves the experience for users with low vision. Learn how to make links distinguishable.

- The document does not use `<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 about the refresh meta tag.

- `<object>` elements have alternate text

  Screen readers cannot translate non-text content. Adding alternate text to `<object>` elements helps screen readers convey meaning to users. Learn more about alt text for `object` elements.

- Select elements have associated label elements.

  Form elements without effective labels can create frustrating experiences for screen reader users. Learn more about the `select` element.

- Skip links are focusable.

  Including a skip link can help users skip to the main content to save time. Learn more about skip links.

- No element has 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 about the `tabindex` attribute.

- Tables have different content in the summary attribute and `<caption>`.

The summary attribute should describe the table structure, while `<caption>` should have the onscreen title. Accurate table mark-up helps users of screen readers. Learn more about summary and caption.

- Cells in a `<table>` element that use the `[headers]` attribute refer to table cells within the 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 about the `headers` attribute.

- `<th>` elements and elements with `[role="columnheader"/"rowheader"]` 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 about table headers.

- `[lang]` attributes have a valid value

Specifying a valid BCP 47 language on elements helps ensure that text is pronounced correctly by a screen reader. Learn how to use the `lang` attribute.

- `<video>` elements 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 about video captions.

- All heading elements contain content.

A heading with no content or inaccessible text prevent screen reader users from accessing information on the page's structure. Learn more about headings.

- Identical links have the same purpose.

Links with the same destination should have the same description, to help users understand the link's purpose and decide whether to follow it. Learn more about identical links.

- Touch targets have sufficient size and spacing.

Touch targets with sufficient size and spacing help users who may have difficulty targeting small controls to activate the targets. Learn more about touch targets.

- Elements with visible text labels have matching accessible names.

Visible text labels that do not match the accessible name can result in a confusing experience for screen reader users. Learn more about accessible names.

Tables use `<caption>` instead of cells with the `[colspan]` attribute to indicate a caption.

Screen readers have features to make navigating tables easier. Ensuring that tables use the actual caption element instead of cells with the `[colspan]` attribute may improve the experience for screen reader users. Learn more about captions.

`<td>` elements in a large `<table>` have one or more table headers.

Screen readers have features to make navigating tables easier. Ensuring that <td> elements in a large table (3 or more cells in width and height) have an associated table header may improve the experience for screen reader users. Learn more about table headers.

SEO

These checks ensure that your page is following basic search engine optimization advice. There are many additional factors Lighthouse does not score here that may affect your search ranking, including performance on Core Web Vitals. Learn more about Google Search Essentials.

ADDITIONAL ITEMS TO MANUALLY CHECK (1)

Hide

Structured data is valid

Run the Structured Data Testing Tool and the Structured Data Linter to validate structured data. Learn more about Structured Data.

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

PASSED AUDITS (12)

Hide

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. TBT

Document has a `<title>` element

The title gives screen reader users an overview of the page, and search engine users rely on it heavily to determine if a page is relevant to their search. Learn more about document titles.

Document has a meta description

Meta descriptions may be included in search results to concisely summarize page content. Learn more about the meta description.

Page has successful HTTP status code

Pages with unsuccessful HTTP status codes may not be indexed properly. Learn more about HTTP status codes.

Links have descriptive text

Descriptive link text helps search engines understand your content. Learn how to make links more accessible.

Links are crawlable

Search engines may use `href` attributes on links to crawl websites. Ensure that the `href` attribute of anchor elements links to an appropriate destination, so more pages of the site can be discovered. Learn how to make links crawlable

Page isn't blocked from indexing

Search engines are unable to include your pages in search results if they don't have permission to crawl them. Learn more about crawler directives.

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 about the `alt` attribute.

Document has a valid `hreflang`

hreflang links tell search engines what version of a page they should list in search results for a given language or region. Learn more about `hreflang`.

Document uses legible font sizes — 100% legible text

Font sizes less than 12px are too small to be legible and require mobile visitors to "pinch to zoom" in order to read. Strive to have >60% of page text ≥12px. Learn more about legible font sizes.

Source	Selector	% of Page Text	Font Size
`Legible text`		100.00%	≥ 12px

Document avoids plugins

Search engines can't index plugin content, and many devices restrict plugins or don't support them. Learn more about avoiding plugins.

Tap targets are sized appropriately — 100% appropriately sized tap targets

Interactive elements like buttons and links should be large enough (48x48px), or have enough space around them, to be easy enough to tap without overlapping onto other elements. Learn more about tap targets.

NOT APPLICABLE (2)

Hide

robots.txt is valid

If your robots.txt file is malformed, crawlers may not be able to understand how you want your website to be crawled or indexed. Learn more about robots.txt.

Document has a valid `rel=canonical`

Canonical links suggest which URL to show in search results. Learn more about canonical links.

Captured at Feb 9, 2024, 7:14 PM GMT+1	Emulated Moto G Power with Lighthouse 11.3.0	Single page load
Initial page load	Slow 4G throttling	Using Chromium 121.0.0.0 with devtools