http://127.0.0.1:5500/employee-list.html

97

Performance

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

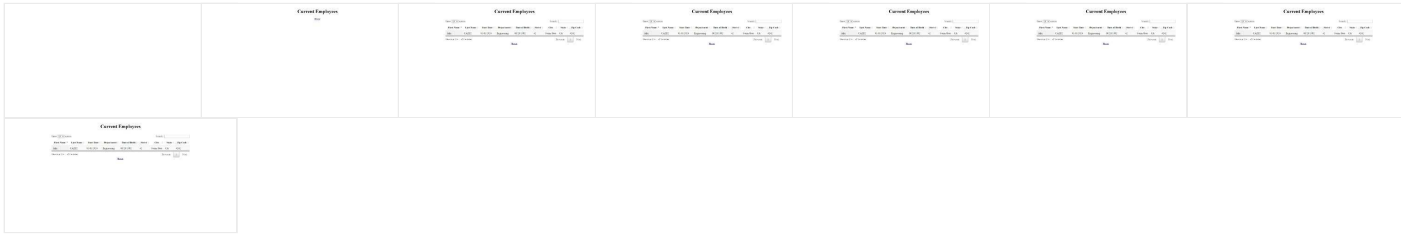
▲ 0–49 ■ 50–89 ● 90–100

METRICS

Expand view

- First Contentful Paint: 1.0 s
- Largest Contentful Paint: 1.0 s
- Total Blocking Time: 0 ms
- Cumulative Layout Shift: 0
- Speed Index: 1.0 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

URL	Transfer Size	Potential Savings
Google CDN Cdn	**30.9 KiB**	**830 ms**
…3.5.1/jquery.min.js (ajax.googleapis.com)	30.9 KiB	830 ms
datatables.net	**28.5 KiB**	**410 ms**
…js/jquery.dataTables.min.js (cdn.datatables.net)	28.5 KiB	410 ms

▲ Does not have a `<meta name="viewport">` tag with `width` or `initial-scale` No `<meta name="viewport">` tag found

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

■ Minify JavaScript — Potential savings of 34 KiB

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

If your build system minifies JS files automatically, ensure that you are deploying the production build of your application. You can check this with the React Developer Tools extension. Learn more.

URL	Transfer Size	Potential Savings
`chrome-extension://nngceckbapebfimnlniiiahkandclblb/content/bootstrap-autofill.js`	27.5 KiB	13.0 KiB
`chrome-extension://nngceckbapebfimnlniiiahkandclblb/content/notificationBar.js`	17.1 KiB	10.8 KiB
`chrome-extension://cjpalhdlnbpafiamejdnhcphjbkeiagm/js/contentscript.js`	15.1 KiB	7.9 KiB
`chrome-extension://bkdgdianpkfahpkmphgehigalpighjck/LetsGetColorBlindContent.js`	5.0 KiB	2.3 KiB

■ Avoid serving legacy JavaScript to modern browsers — Potential savings of 0 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
Milanote Web Clipper Chrome Extension	**0.3 KiB**

URL	Potential Savings
chrome-extension://mipimgcmndeggldjcbjfeogcpoafomhl/drawer.bundle.js	0.2 KiB
drawer.bundle.js:1 @babel/plugin-transform-classes	
drawer.bundle.js:1 @babel/plugin-transform-regenerator	
chrome-extension://mipimgcmndeggldjcbjfeogcpoafomhl/pinner.bundle.js	0.1 KiB
pinner.bundle.js:1 @babel/plugin-transform-classes	

Page prevented back/forward cache restoration — 1 failure reason

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 WebSocket cannot enter back/forward cache.	Pending browser support
/employee-list.html (127.0.0.1)	

Avoid large layout shifts — 1 element found

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

Element	CLS Contribution
 a	0.000

User Timing marks and measures — 1 user timing

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.

Use the React DevTools Profiler, which makes use of the Profiler API, to measure the rendering performance of your components. Learn more.

Name	Type	Start Time	Duration
__v3	Mark	0.00 ms	

Initial server response time was short — Root document took 0 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

If you are server-side rendering any React components, consider using `renderToPipeableStream()` or `renderToStaticNodeStream()` to allow the client to receive and hydrate different parts of the markup instead of all at once. Learn more.

URL	Time Spent
0.1 1st Party	**0 ms**
/employee-list.html (127.0.0.1)	0 ms

Avoids enormous network payloads — Total size was 67 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 (5)

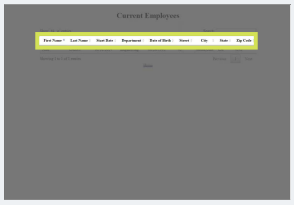
URL	Transfer Size
datatables.net	**31.6 KiB**
…js/jquery.dataTables.min.js (cdn.datatables.net)	28.5 KiB
…css/jquery.dataTables.min.css (cdn.datatables.net)	2.4 KiB
…images/sort_both.png (cdn.datatables.net)	0.3 KiB
…images/sort_asc.png (cdn.datatables.net)	0.3 KiB
Google CDN Cdn	**30.9 KiB**
…3.5.1/jquery.min.js (ajax.googleapis.com)	30.9 KiB

URL	Transfer Size
0.1 1st Party	**4.2 KiB**
/employee-list.html (127.0.0.1)	2.5 KiB
/employee-list.js (127.0.0.1)	1.0 KiB
/app.css (127.0.0.1)	0.7 KiB

Avoids an excessive DOM size — 52 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

Consider using a "windowing" library like `react-window` to minimize the number of DOM nodes created if you are rendering many repeated elements on the page. Learn more. Also, minimize unnecessary re-renders using `shouldComponentUpdate`, `PureComponent`, or `React.memo` and skip effects only until certain dependencies have changed if you are using the `Effect` hook to improve runtime performance.

Statistic	Element	Value
Total DOM Elements		52
Maximum DOM Depth	option	7
Maximum Child Elements	 tr	9

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: **682.465 ms**

Initial Navigation

- /employee-list.html (127.0.0.1)
  - …css/jquery.dataTables.min.css (cdn.datatables.net) **- 150.923 ms, 2.41 KiB**
  - /app.css (127.0.0.1) **- 8.426 ms, 0.67 KiB**
  - …3.5.1/jquery.min.js (ajax.googleapis.com) **- 641.14 ms, 30.91 KiB**
  - …js/jquery.dataTables.min.js (cdn.datatables.net) **- 162.26 ms, 28.54 KiB**

/employee-list.js (127.0.0.1) **- 6.446 ms, 1.02 KiB**

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 (4)

URL	Total CPU Time	Script Evaluation	Script Parse
0.1 1st Party	**241 ms**	**74 ms**	**94 ms**
/employee-list.html (127.0.0.1)	241 ms	74 ms	94 ms
Milanote Web Clipper Chrome Extension	**146 ms**	**90 ms**	**49 ms**
chrome-extension://mipimgcmndeggldjcbjfeogcpoafomhl/drawer.bundle.js	87 ms	58 ms	26 ms
chrome-extension://mipimgcmndeggldjcbjfeogcpoafomhl/pinner.bundle.js	60 ms	32 ms	23 ms
Google CDN Cdn	**88 ms**	**50 ms**	**2 ms**
…3.5.1/jquery.min.js (ajax.googleapis.com)	88 ms	50 ms	2 ms
Unattributable	**56 ms**	**6 ms**	**0 ms**
Unattributable	56 ms	6 ms	0 ms
Wappalyzer - Technology profiler Chrome Extension	**52 ms**	**12 ms**	**5 ms**
chrome-extension://gppongmhjkpfnbhagpmjfkannfbllamg/js/content.js	52 ms	12 ms	5 ms

Minimizes main-thread work — 0.8 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
Script Evaluation	312 ms
Script Parsing & Compilation	213 ms

Category	Time Spent
Other	146 ms
Style & Layout	46 ms
Parse HTML & CSS	35 ms
Garbage Collection	12 ms
Rendering	5 ms

Minimize third-party usage — Third-party code blocked the main thread for 0 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
datatables.net	**32 KiB**	**0 ms**
…js/jquery.dataTables.min.js (cdn.datatables.net)	29 KiB	0 ms
Google CDN Cdn	**31 KiB**	**0 ms**
…3.5.1/jquery.min.js (ajax.googleapis.com)	31 KiB	0 ms
Milanote Web Clipper Chrome Extension	**15 KiB**	**0 ms**
chrome-extension://mipimgcmndeggldjcbjfeogcpoafomhl/css/drawer.css	15 KiB	0 ms
Wappalyzer - Technology profiler Chrome Extension	**6 KiB**	**0 ms**
Angular DevTools Chrome Extension	**1 KiB**	**0 ms**

Largest Contentful Paint element — 1,020 ms

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

Element	
	h1

Element

Phase	% of LCP	Timing
TTFB	12%	120 ms
Load Delay	0%	0 ms
Load Time	0%	0 ms
Render Delay	88%	900 ms

Avoid long main-thread tasks — 2 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 (1)

URL	Start Time	Duration
Milanote Web Clipper Chrome Extension		**150 ms**
chrome-extension://mipimgcmndeggldjcbjfeogcpoafomhl/pinner.bundle.js	399 ms	150 ms
0.1 1st Party		**150 ms**
/employee-list.html (127.0.0.1)	193 ms	150 ms

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

PASSED AUDITS (23) 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

If your build system minifies CSS files automatically, ensure that you are deploying the production build of your application. You can check this with the React Developer Tools extension. Learn more.

- 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. 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

If you are not server-side rendering, split your JavaScript bundles with `React.lazy()`. Otherwise, code-split using a third-party library such as loadable-components.

- 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

If you are using React Router, minimize usage of the `<Redirect>` component for route navigations.

- 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

- Uses efficient cache policy on static assets — 0 resources found

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

- 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

- 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

Captured at Jan 24, 2024, 8:57 AM GMT+4
Initial page load

Emulated Desktop with Lighthouse 11.2.0
Custom throttling

Single page load

Using Chromium 120.0.0.0 with devtools