Progressive Web App	Performance	Accessibility	Best Practices
0	26	94	75

0 Progressive Web App

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

✖ 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.

⚠ The page body should render some content if its scripts are not available.

✖ Uses HTTPS: 3 insecure requests found

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.

▼ Insecure URLs:

192.168.0.103:3000/
192.168.0.103:3000/assets/styles.css
192.168.0.103:3000/dist/bundle.js

✖ 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

Satisfied if the Time To Interactive duration is shorter than 10 seconds, as defined by the PWA Baseline Checklist. Network throttling is required (specifically: RTT latencies >= 150 RTT are expected).

Audit error: Failed to execute 'atob' on 'Window': The string to be decoded is not correctly encoded.

User can be prompted to Install the Web App

While users can manually add your site to their homescreen, the prompt (aka app install banner) will proactively prompt the user to install the app if the various requirements are met and the user has moderate engagement with your site.

Failures: No manifest was fetched, Site does not register a Service Worker, Manifest start_url is not cached by a Service Worker.

Configured for a custom splash screen

A default splash screen will be constructed for your app, but satisfying these requirements guarantee a high-quality splash screen that transitions the user from tapping the home screen icon to your app's first paint

Failures: No manifest was fetched.

Address bar matches brand colors

The browser address bar can be themed to match your site. A `theme-color` meta tag will upgrade the address bar when a user browses the site, and the manifest theme-color will apply the same theme site-wide once it's been added to homescreen.

Failures: No manifest was fetched, No `<meta name="theme-color">` tag found.

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.

No viewport meta tag found

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.

The viewport size is 981px, whereas the window size is 412px.

Manual checks to verify

These audits 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.

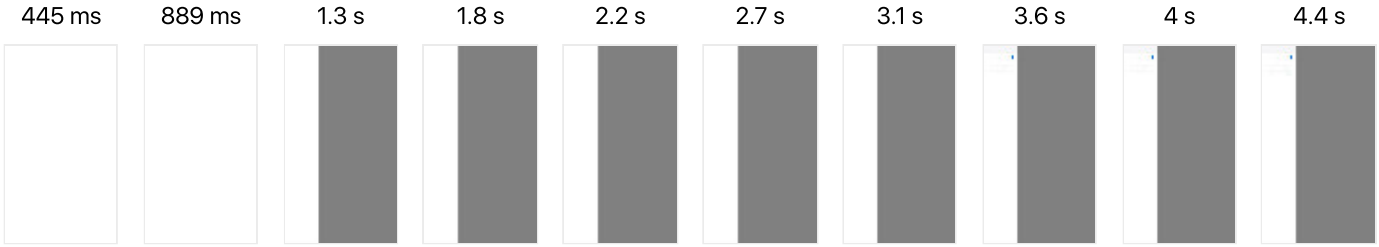
Performance

26

These encapsulate your app's performance.

Metrics

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



First meaningful paint: 3183.5ms

First meaningful paint measures when the primary content of a page is visible. Learn more.

First Interactive (beta):

Audit error: Failed to execute 'atob' on 'Window': The string to be decoded is not correctly encoded.

Consistently Interactive (beta):

Audit error: Failed to execute 'atob' on 'Window': The string to be decoded is not correctly encoded.

99 Perceptual Speed Index: 1085 (target: < 1,250)

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

0 Estimated Input Latency

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 score is higher than Lighthouse's target score, users may perceive your app as laggy. Learn more.

Audit error: Failed to execute 'atob' on 'Window': The string to be decoded is not correctly encoded.

Opportunities

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

Reduce render-blocking stylesheets 789 ms

Link elements are blocking the first paint of your page. Consider inlining critical links and deferring non-critical ones. Learn more.

▼ View Details

URL	Size (KB)	Delayed Paint By (ms)

...css/bootstrap.min.css	0 KB	789ms
/assets/styles.css	0 KB	580ms

Offscreen images

Audit error: Failed to execute 'atob' on 'Window': The string to be decoded is not correctly encoded.

Diagnostics

More information about the performance of your application.

Critical Request Chains: 4

The Critical Request Chains below show you what resources are required for first render of this page. Improve page load by reducing the length of chains, reducing the download size of resources, or deferring the download of unnecessary resources. Learn more.

Longest chain: **4,120.2ms** over **1** requests, totalling **0KB**

▼ View critical network waterfall:

Initial Navigation

- / (192.168.0.103)
  - ...css/bootstrap.min.css (maxcdn.bootstrapcdn.com) **- 787.2ms, 0KB**
  - /assets/styles.css (192.168.0.103) **- 579.8ms, 0KB**
  - /dist/bundle.js (192.168.0.103) **- 1,608.8ms, 0KB**
  - ...css/bootstrap.min.css (maxcdn.bootstrapcdn.com) **- 521.3ms, 0KB**
- ...shared/tpc-check.html (noojglkidnpfjbincgijbaiedldjfbhh) **- 81ms, 0KB**

User Timing marks and measures: 12

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

▼ View Details

Name	Type	Time
nanobus:DOMContentLoaded	Measure	0.455 ms
nanobus:render	Measure	0.95 ms
nanobus:shows:load	Measure	44.885 ms
choo:render	Measure	57.784 ms
nanobus:shows:load-start	Mark	2,975.338 ms
nanobus:shows:load-end	Mark	3,020.018 ms

Name	Type	Time
nanobus:DOMContentLoaded-start	Mark	3,024.528 ms
nanobus:DOMContentLoaded-end	Mark	3,024.786 ms
nanobus:render-start	Mark	4,371.053 ms
nanobus:render-end	Mark	4,372.009 ms
choo:renderStart	Mark	4,376.222 ms
choo:renderEnd	Mark	4,434.011 ms

View 7 passed items

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.

Optimize images

Optimized images take less time to download and save cellular data. The identified images could have smaller file sizes when compressed as JPEG (q=85). Learn more about image optimization.

Serve images as WebP

WebP images take less time to download and save cellular data. Learn more about image optimization.

Enable text compression

Text-based responses should be served with compression (gzip, deflate or brotli) to minimize total network bytes. Learn more.

100 Avoids enormous network payloads: Total size was 193 KB (target: < 1,600 KB)

Network transfer size costs users real dollars and is highly correlated with long load times. Try to find ways to reduce the size of required files.

▼ View Details

URL	Total Size	Transfer Time
/dist/bundle.js	171 KB	930ms
...css/bootstrap.min.css	20 KB	110ms

URL	Total Size	Transfer Time
/	1 KB	0ms
/assets/styles.css	0 KB	0ms
...css/bootstrap.min.css	0 KB	0ms
/injected.js	0 KB	0ms
...shared/tpc-check.html	0 KB	0ms
...img/buffer-hover-icon@2x.png	0 KB	0ms

100 Avoids an excessive DOM size: 62 nodes (target: < 1,500 nodes)

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, cause longer style calculations, and produce costly layout reflows. Learn more.

▼ View details

Total DOM Nodes	DOM Depth	Maximum Children
62	10	5
target: < 1,500 nodes	target: < 32	target: < 60 nodes

94

Accessibility

These checks highlight opportunities to improve the accessibility of your app.

Elements Describe Contents Well

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

✖ Form elements have associated labels.

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

▼ View failing elements

```
<input type="text" name="title" id="title" value="" class="form-control form-control-sm">
<input type="number" step="1" min="0" name="season" id="season" value="0" class="form-
control form-control-sm">
<input type="number" step="1" min="0" name="episode" id="episode" value="0" class="form-
control form-control-sm">
```

Page Specifies Valid Language

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

✖ `<html>` element has a `[lang]` attribute.

The `lang` attribute is useful for multilingual screen reader users who may prefer a language other than the default.

▼ View failing elements

```
<html>
```

View 8 passed items

Elements Use Attributes Correctly

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

✔ `[accesskey]` values are unique.

`accesskey` attributes allow the user to quickly activate or focus part of the page.Using the same `accesskey` more than once could lead to a confusing experience.

✔ `<audio>` elements contain a `<track>` element with `[kind="captions"]`.

Captions convey information such as identifying who is speaking, dialogue, and non-speech information. This can help deaf or hearing impaired users access meaningful content.

✔ 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.

✔ `<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.

✔ 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.

✔ Cells in a `<table>` element that use the `[headers]` attribute only refer 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.

✔ `<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.

ARIA Attributes Follow Best Practices

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

- ✔ `[aria-*]` attributes match their roles.

  Each ARIA `role` supports a specific subset of `aria-*` attributes. Mismatching these invalidates the `aria-*` attributes. Learn more.

- ✔ `[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.

- ✔ `[role]`s that require child `[role]`s contain them.

  Some ARIA parent roles require specific roles on their children to perform their accessibility function.

- ✔ `[role]`s are contained by their required parent element.

  Some ARIA roles require specific roles on their parent element to perform their accessibility function.

- ✔ `[role]` values are valid.

  ARIA roles require specific values to perform their accessibility function.

- ✔ `[aria-*]` attributes have valid values.

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

- ✔ `[aria-*]` attributes are valid and not misspelled.

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

Elements Have Discernable Names

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

- ✔ 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 more.

- ✔ Links have a discernable name.

  Link text (and alternate text for images, when used as links) that is discernible, not duplicated, and focusable improves the navigating experience for screen reader users.

Elements Describe Contents Well

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

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

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

- ✔ Document has a `<title>` element.

  Screen reader users use page titles to get an overview of the contents of the page.

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

  Screen reader users rely on a frame title to describe the contents of the frame.

- ✔ Presentational `<table>` elements avoid using `<th>`, `<caption>` or the `[summary]` attribute.

  The presence of `<th>`, `<caption>` or the `summary` attribute on a presentational table may produce a confusing experince for a screen reader user as these elements usually indicates a data table.

- ✔ `<object>` elements have `[alt]` text.

  Screen readers cannot translate non-text content. Adding alt text to `<object>` elements will help a screen reader convey the meaning to a user.

- ✔ `<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.

- ✔ `<video>` elements contain a `<track>` element with `[kind="description"]`.

  Audio descriptions provide relevant information for videos that dialogue cannot, such as facial expressions and scenes.

Color Contrast Is Satisfactory

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous 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

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

- ✔ `<dl>`'s 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.

- ✔ Definition list items are wrapped in `<dl>` elements.

  Definition list items (<dt> and/or <dd>) wrapped in parent <dl> elements enable screen readers to properly announce content.

- ✔ `[id]` attributes on the page are unique.

  Unique `id=""` attributes help ensure assistive technologies do not overlook elements with the same id.

- ✔ 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 aides screen reader output.

- ✔ 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

Page Specifies Valid Language

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

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

  Specifying a valid BCP 47 language helps screen readers announce text properly.

- ✔ `[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.

Meta Tags Used Properly

Screen readers and other assistive technologies require annotations to understand otherwise ambiguous content.

- ✔ 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

- ✔ `[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.

Best Practices

75

We've compiled some recommendations for modernizing your web app and avoiding performance pitfalls. These audits do not affect your score but are worth a look.

Uses HTTPS: 3 insecure requests found

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.

▼ Insecure URLs:

192.168.0.103:3000/
192.168.0.103:3000/assets/styles.css
192.168.0.103:3000/dist/bundle.js

Uses HTTP/2 for its own resources: 3 requests were not handled over h2

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

▼ View Details

URL	Protocol
http://192.168.0.103:3000/	http/1.1
http://192.168.0.103:3000/assets/styles.css	http/1.1
http://192.168.0.103:3000/dist/bundle.js	http/1.1

Manifest's `short_name` won't be truncated when displayed on homescreen

Make your app's `short_name` less than 12 characters to ensure that it's not truncated on homescreens. Learn more.

View 9 passed items

Avoids Application Cache

Application Cache has been deprecated by Service Workers. Consider implementing an offline solution using the Cache Storage API.

Avoids WebSQL DB

Web SQL is deprecated. Consider using IndexedDB instead. 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 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.

Generated by **Lighthouse** 2.0.0 on Jun 5, 2017, 11:54 PM EDT | File an issue