ToDo List App Technical Documentation

Table of Contents

Chapter 1

This chapter contains what the application can do and its workaround. It is also indicated in the chapter on the overview of this application.

ToDo List App is web application which allows user to do the following:

- Add new To-Do activity
- Edit/Modify a particular To-Do activity
- Delete/Remove a particular/all To-Do activities
- Ability to mark particular activity/all activity as either active or completed
- Display all, active, and completed To-Do activities

This web application uses and takes advantage of MVC (Model-View-Controller) architecture which takes advantage of separation of code concerns.

MVC patterns separate the input, processing, and output of an application. This model divided into three interconnected parts called the model, the view, and the controller. All of the three above given components are built to handle some specific development aspects of any web or .net application development.

In the MVC application development, the controller receives all requests for the application and then instructs the model to prepare any information required by the view. The view uses that data prepared by the controller to bring the final output

Chapter 2

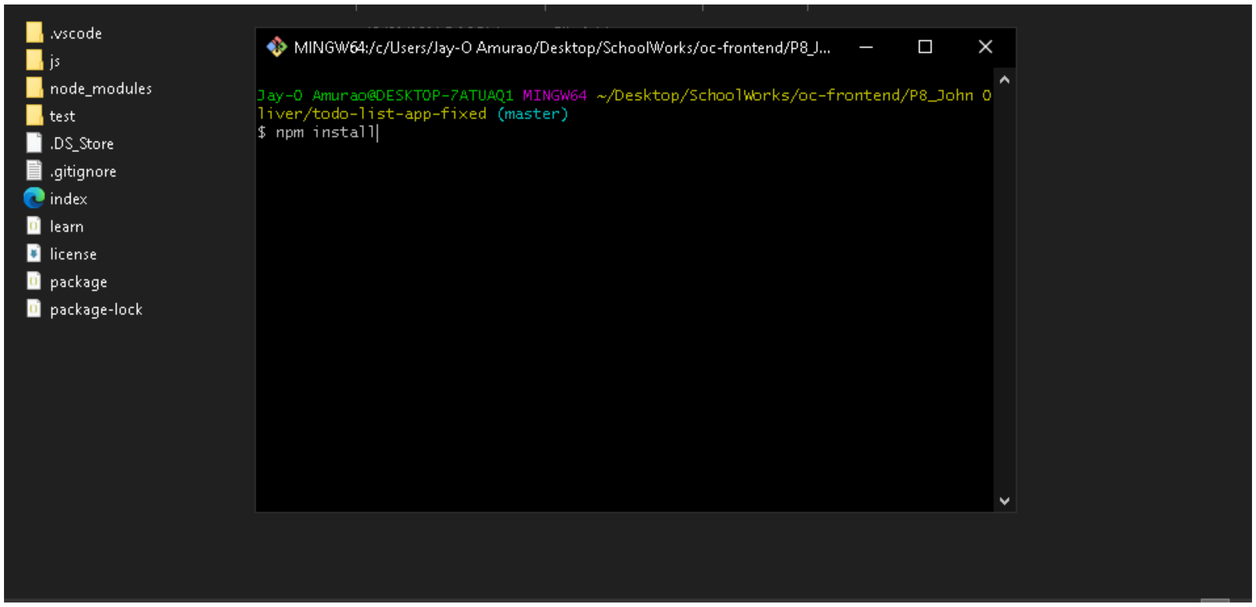
This chapter contains the installation process, Starting the application, Using the application, and Software Design pattern. Also, in this chapter, the devices and software were used to make the application work is a ***Windows 10 1909 (build 18363.1379)*** and ***Microsoft Edge browser Version 89.0.774.45***

Installation

Before installing the web application, you must first have the following software installed in order to have this app on your local machine. You must have installed ***NodeJS*** (A JavaScript engine), and ***Git***, an open-source version control software. Follow the instructions provided from the documentation of the respective software needed to run this application.

To install this application (ToDo List App), you must have download from this link -> [https://github.com/joliveramu/oc-frontend/tree/master/P8_John%20Oliver].

Once downloaded, open your ***command line (In this documentation, Git bash terminal)*** in order for you to navigate in the project ***P8_John Oliver*** folder then to the ***todo-list-app-fixed*** folder. As you are in the ***todo-list-app-fixed*** folder, in your ***command prompt***, type the command ***npm install*** in order to install all dependencies that this application needs.

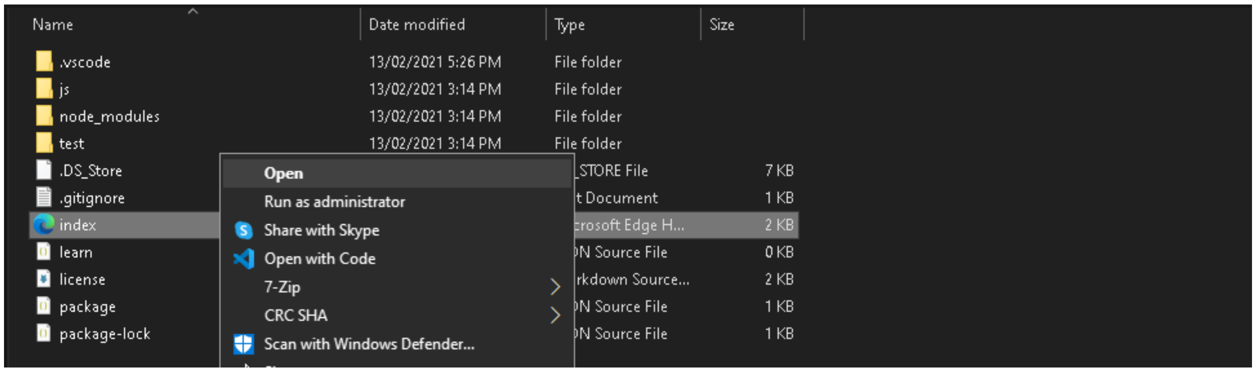


```
Jay-O Amurao@DESKTOP-7ATUAQ1 MINGW64 ~/Desktop/SchoolWorks/oc-frontend/P8_John O
liver/todo-list-app-fixed (master)
$ npm install
```

Once everything is installed, we will now proceed on how to start the application.

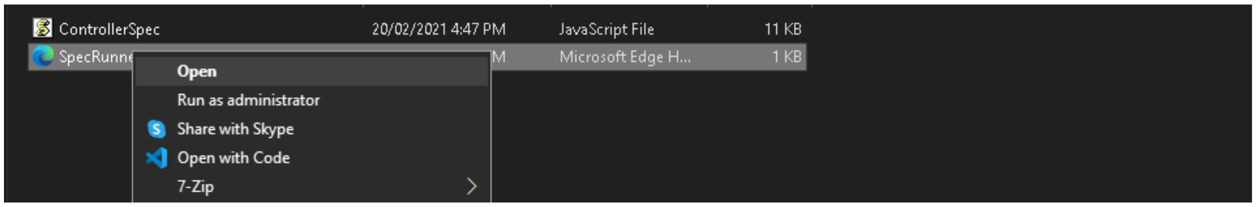
Starting the application

To start this application, you may simply ***open*** the ***index.html*** in your browser of preference.

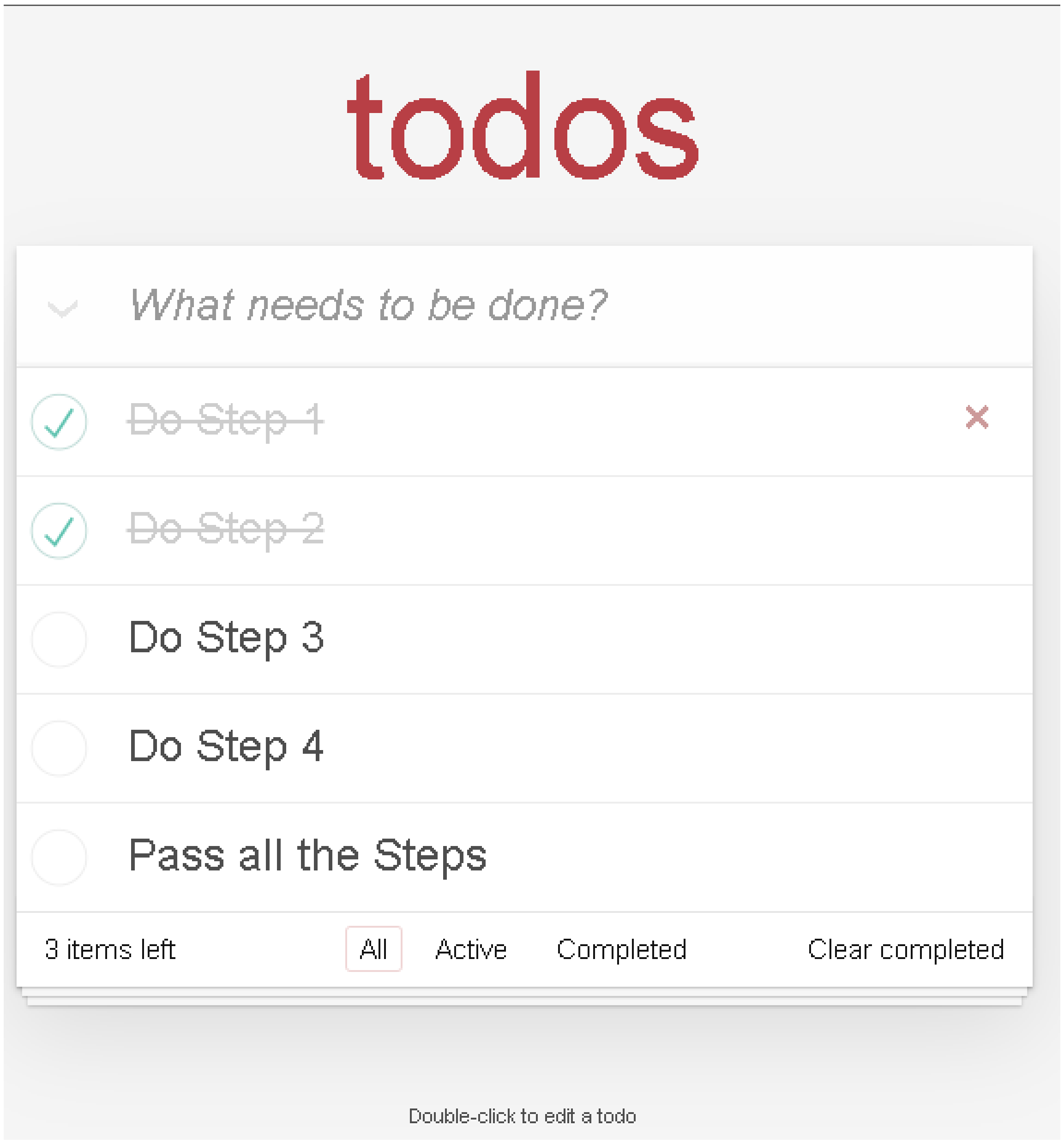


Name	Date modified	Type	Size
.vscode	13/02/2021 5:26 PM	File folder	
js	13/02/2021 3:14 PM	File folder	
node_modules	13/02/2021 3:14 PM	File folder	
test	13/02/2021 3:14 PM	File folder	
.DS_Store		_STORE File	7 KB
.gitignore		t Document	1 KB
index		crosoft Edge H...	2 KB
learn		ON Source File	0 KB
license		rkdown Source...	2 KB
package		ON Source File	1 KB
package-lock		ON Source File	1 KB

In order for you to view the unit testing results, inside the ***todo-list-app-fixed*** folder, navigate to the ***test*** folder and look for the ***SpecRunner.html***. You may ***open*** the web page in any browser of your preference.



Using the application



- To add new To-do activity, simply click onto field with text ***What needs to be done?*** type name of new To-do activity and press enter.
- To change name of existing To-do activity, double click the name of a particular activity and press enter after finish editing.

- To change status of an active To-do activity to completed or active, simply click the circular checkbox before the name of a particular To-do activity
- To change status of all active To-do activities to all completed or all active to-do activities, click the dropdown icon besides the ***What needs to be done?*** field.
- In order to see how many activities are still active, there’s a notification bar below (Beside All, Active, and Completed navigation bar)
- This filter buttons (All, Active, and Completed) will let you change view to display all To-do activities or only active or completed To-do activities.
- To delete one active or completed To-do activity, hover over its name and click this X icon once it appears.
- To delete all completed To-do activities, click the ***Clear completed***.

Software Design Pattern

This application uses a MVC architecture where **MVC** stands for **Model-View-Controller.** The Model, View, and Controller are different entities from each other.

1. ***MODEL***

   Model is responsible for managing the data of the application. It receives user input from the controller. He is responsible for CRUD (create, read, update and delete) operations. Our model is using local storage to save our todos.

2. ***VIEW***

   View is a presentation of the model in a particular format - in our case displaying All, Active or Completed todos (**route**). It also let user to interact with displayed data.

3. ***CONTROLLER***

   Controller responds to the user input from View and performs interactions with Model. It also reading data from Model and passing it to the view.

By using MVC architecture, our application works like Single Page Application (SPA) - that means user can interact with todos without reloading webpage.

Chapter 3

This chapter contains the bugs/error fixing, and tests done for the web application.

Bugs/Error fixing

This part contains the bugs/error fixing encountered through the application.

- **js/Controller.js** line 95

  Misspelled function name. With the misspelled name, the function for the addItem is missing and could not be recognized when the add function for list is being called.

```
Controller.prototype.adddItem = function (title) {
```

Changed to

```
Controller.prototype.addItem = function (title) {
```

- **js/Controller.js** line 169

  Unnecessary line of code. We don't have to log a lot of activities when we are about to deploy it for production. Using console logging is for debugging and development purposes only.

```
items.forEach(function(item) {
    if (item.id === id) {
        console.log("Element with ID: " + id + " has been removed.");
      }
    });
```

- **js/Store.js** line 89

  This line of code may produce an ID conflict in the near future. There are chances that the ID that would be generated here produce a duplicate.

```
/* Original Code Edit 1 */
    Generate an ID
    var newId = "";
    var charset = "0123456789";

    for (var i = 0; i < 6; i++) {
      newId += charset.charAt(Math.floor(Math.random() * charset.length));
    }
```

Changed to

```
// Generate an ID
   // Displays date in numerical format
   var newId = Date.now();
```

Which makes the generated ID progressive

Note: *In this scale of application, it is good to use this approach since we are not dealing with a lot of data to enter however, there are other approach that we can make this much better.*

- **index.html** line 16

  Added id element name ***toggle-all*****.** The select all functionality would not work if the id name *toggle-all* is not declared.

```
<input class="toggle-all" type="checkbox"/>
```

Changed to

```
<input id="toggle-all" class="toggle-all" type="checkbox"/>
```

Tests

The testing was done using the Jasmine JS framework. *Jasmine JS* is an open-source testing framework for JavaScript.

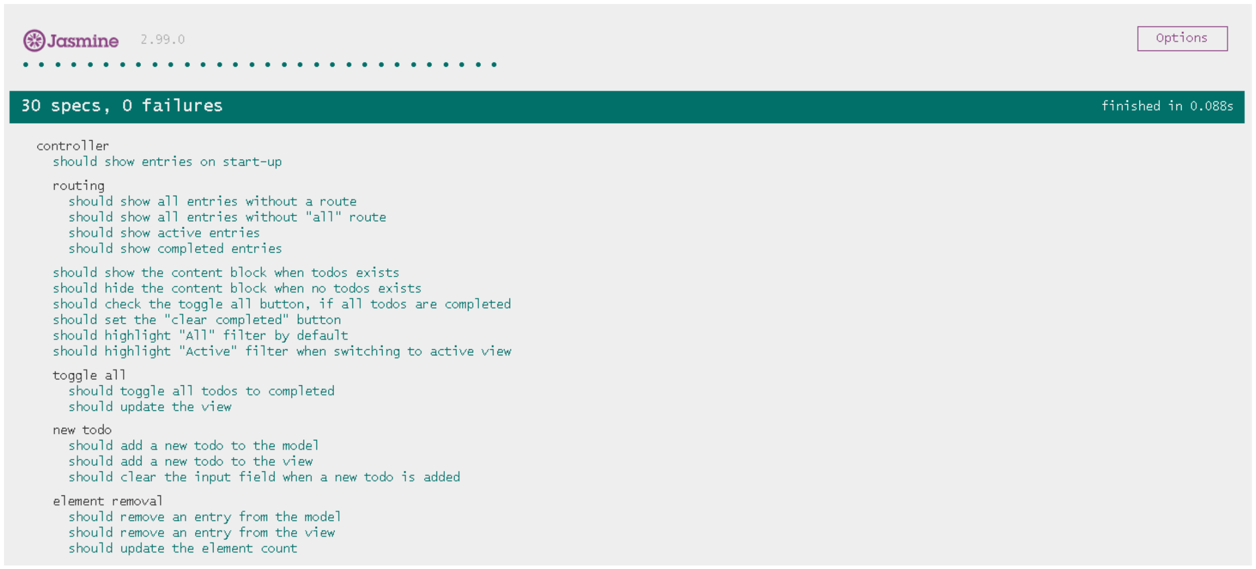
To start test simply open the ***SpecRunner.html*** file located in application main directory in ***test*** folder.

To create or modify tests open and modify ***ControllerSpec.js*** file located in the same directory where you have created the project.

Below are the Nine (9) additional tests that has been added to the existing tests in the *test/ControllerSpec.js*:

- should show entries on start-up
- should show active entries
- should show completed entries
- should highlight "All" filter by default
- should highlight "Active" filter when switching to active view
- should toggle all todos to completed
- should update the view
- should add a new todo to the model
- should remove an entry from the model

Screenshot of the Successful tests added



```
30 specs, 0 failures                                    finished in 0.088s

controller
  should show entries on start-up
  routing
    should show all entries without a route
    should show all entries without "all" route
    should show active entries
    should show completed entries
  should show the content block when todos exists
  should hide the content block when no todos exists
  should check the toggle all button, if all todos are completed
  should set the "clear completed" button
  should highlight "All" filter by default
  should highlight "Active" filter when switching to active view
  toggle all
    should toggle all todos to completed
    should update the view
  new todo
    should add a new todo to the model
    should add a new todo to the view
    should clear the input field when a new todo is added
  element removal
    should remove an entry from the model
    should remove an entry from the view
    should update the element count
```

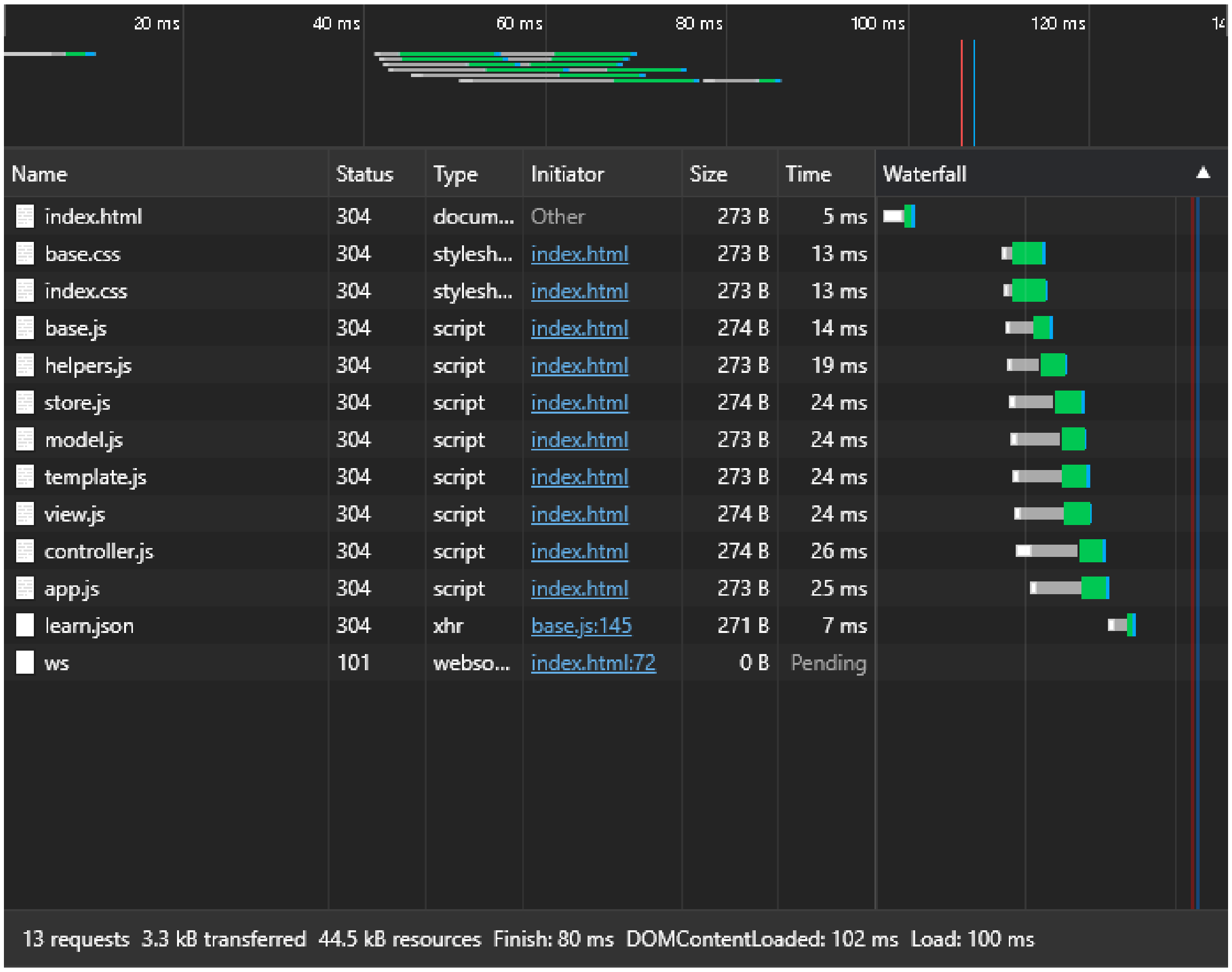
```
remove completed
  should remove a completed entry from the model
  should remove a completed entry from the view

element complete toggle
  should update the model
  should update the view

edit item
  should switch to edit mode
  should leave edit mode on done
  should persist the changes on done
  should remove the element from the model when persisting an empty title
  should remove the element from the view when persisting an empty title
  should leave edit mode on cancel
  should not persist the changes on cancel
```

Chapter 4

This chapter contains the audits of the Network, Performance, and Memory usage audit of the two competing web applications. Also, in this chapter, it describes the comprehensive ratings for each of application.

ToDo List App (Our Application)

Network



Name	Status	Type	Initiator	Size	Time	Waterfall
index.html	304	docum...	Other	273 B	5 ms	
base.css	304	stylesh...	index.html	273 B	13 ms	
index.css	304	stylesh...	index.html	273 B	13 ms	
base.js	304	script	index.html	274 B	14 ms	
helpers.js	304	script	index.html	273 B	19 ms	
store.js	304	script	index.html	274 B	24 ms	
model.js	304	script	index.html	273 B	24 ms	
template.js	304	script	index.html	273 B	24 ms	
view.js	304	script	index.html	274 B	24 ms	
controller.js	304	script	index.html	274 B	26 ms	
app.js	304	script	index.html	273 B	25 ms	
learn.json	304	xhr	base.js:145	271 B	7 ms	
ws	101	webso...	index.html:72	0 B	Pending	

13 requests 3.3 kB transferred 44.5 kB resources Finish: 80 ms DOMContentLoaded: 102 ms Load: 100 ms

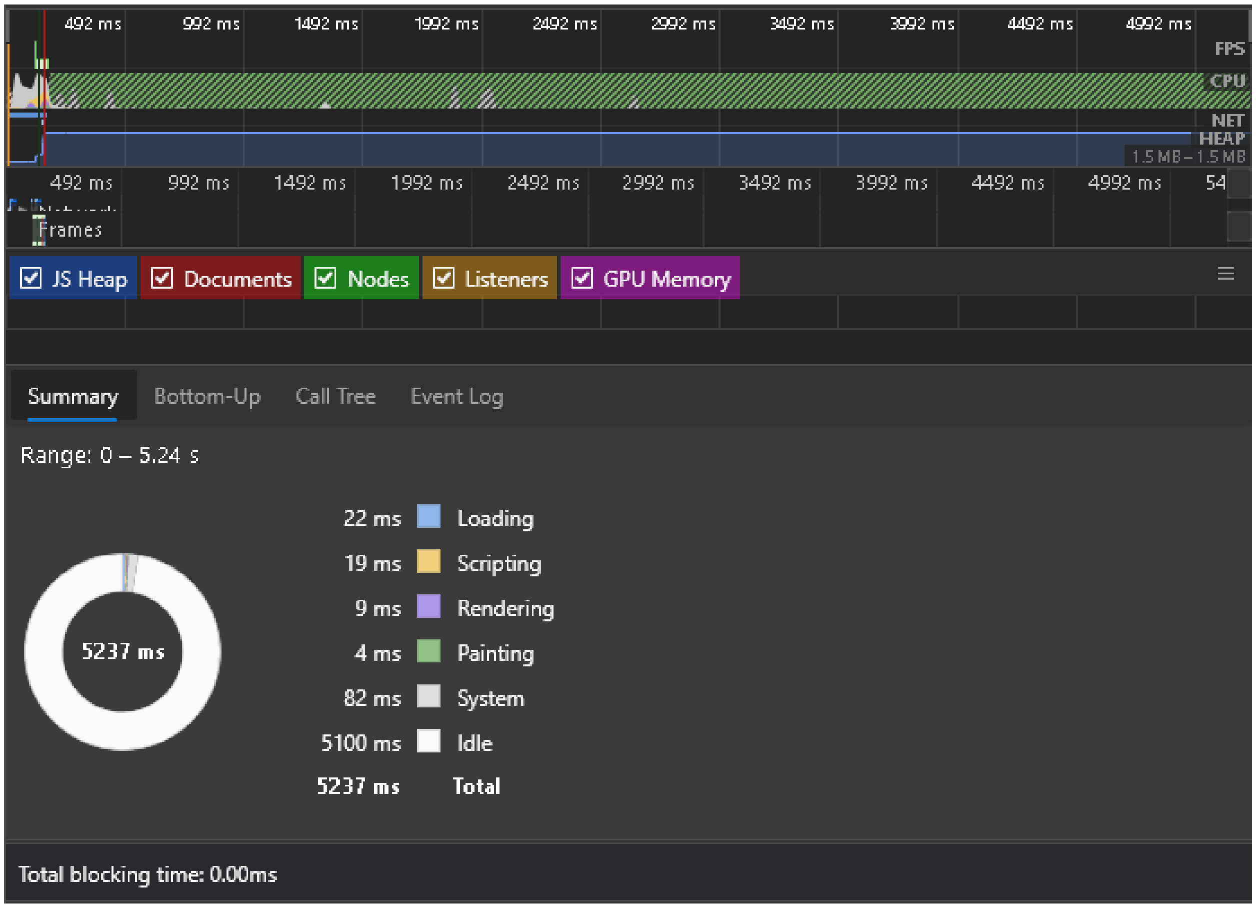
Requests sent: 13

Resources transferred: 44.5 kB

Load time: 100 ms

DOMContentLoaded: 102 ms

Performance



Range: 0 – 5.24 s

22 ms	🟦	Loading
19 ms	🟨	Scripting
9 ms	🟪	Rendering
4 ms	🟩	Painting
82 ms	⬜	System
5100 ms	⬜	Idle
5237 ms		**Total**

Total blocking time: 0.00ms

Loading: 22 ms

Scripting: 19 ms

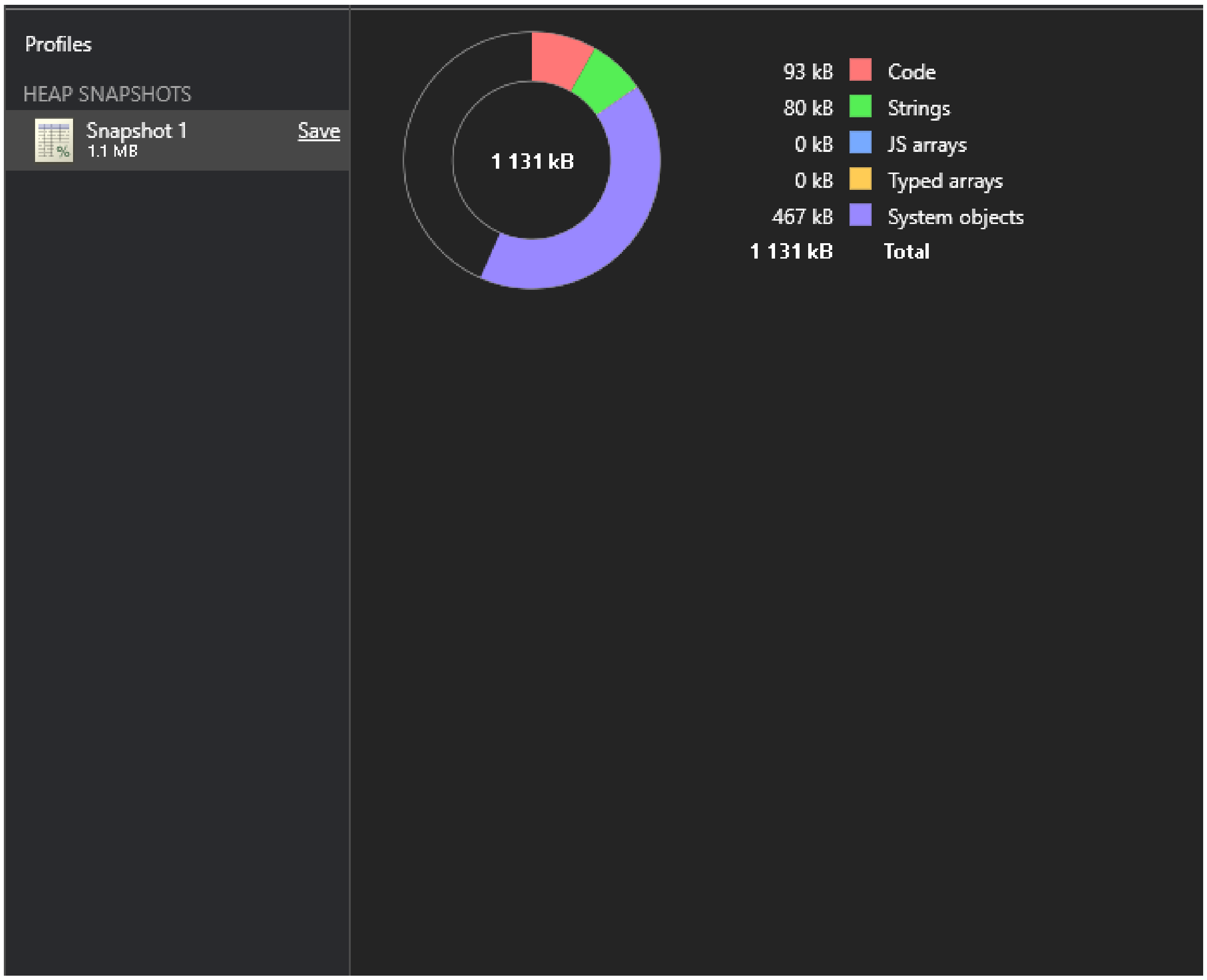
Rendering: 9 ms

Painting: 4 ms

System: 82 ms

Idle: 5100 ms

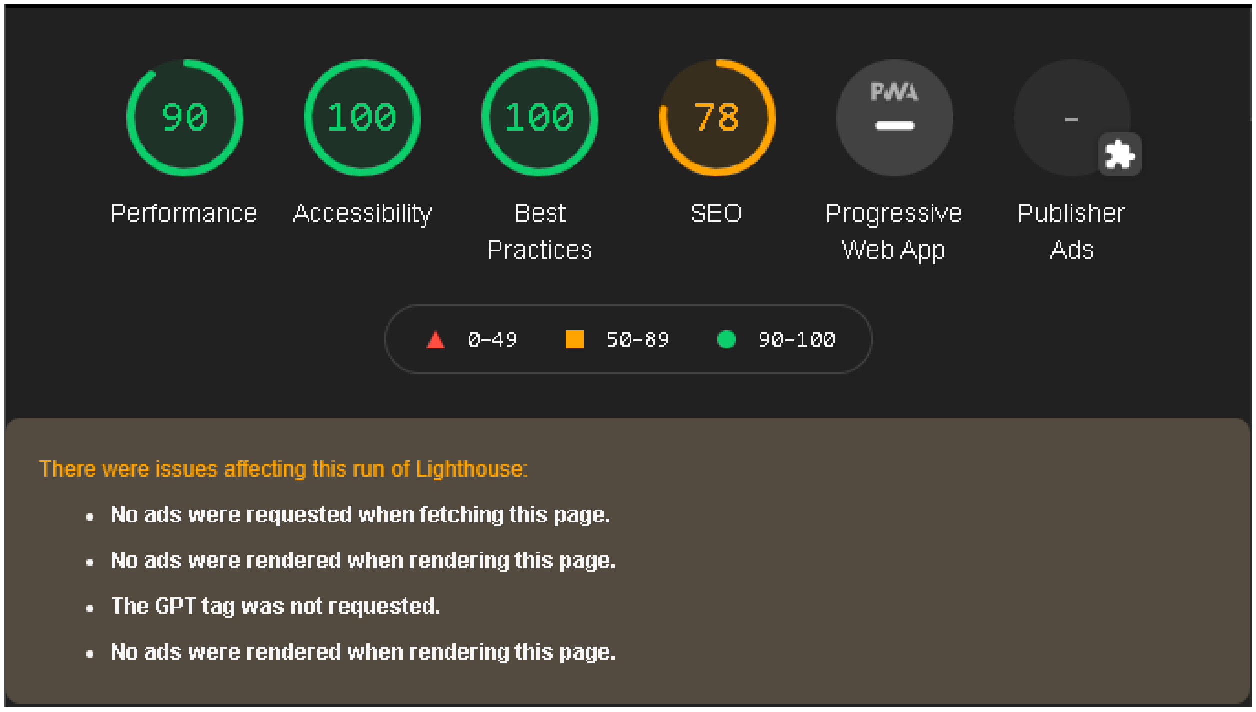
Memory



93 kB	🟥	Code
80 kB	🟩	Strings
0 kB	🟦	JS arrays
0 kB	🟨	Typed arrays
467 kB	🟪	System objects
1 131 kB		Total

Total Memory usage: 1.1 MB

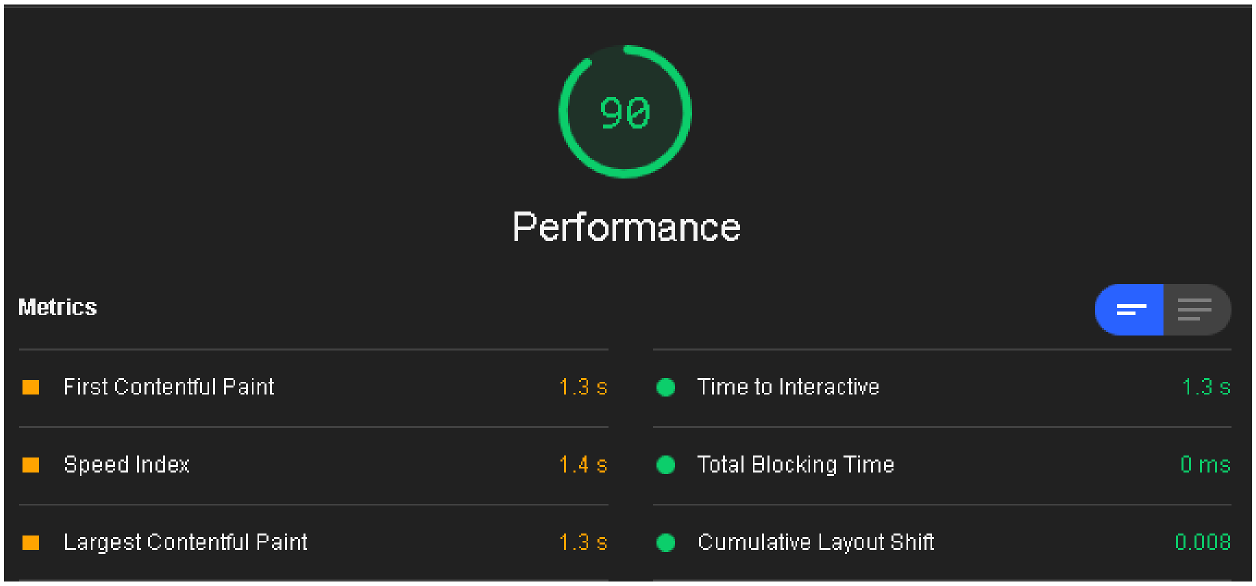
Overall Performance Audit



There were issues affecting this run of Lighthouse:

- No ads were requested when fetching this page.
- No ads were rendered when rendering this page.
- The GPT tag was not requested.
- No ads were rendered when rendering this page.

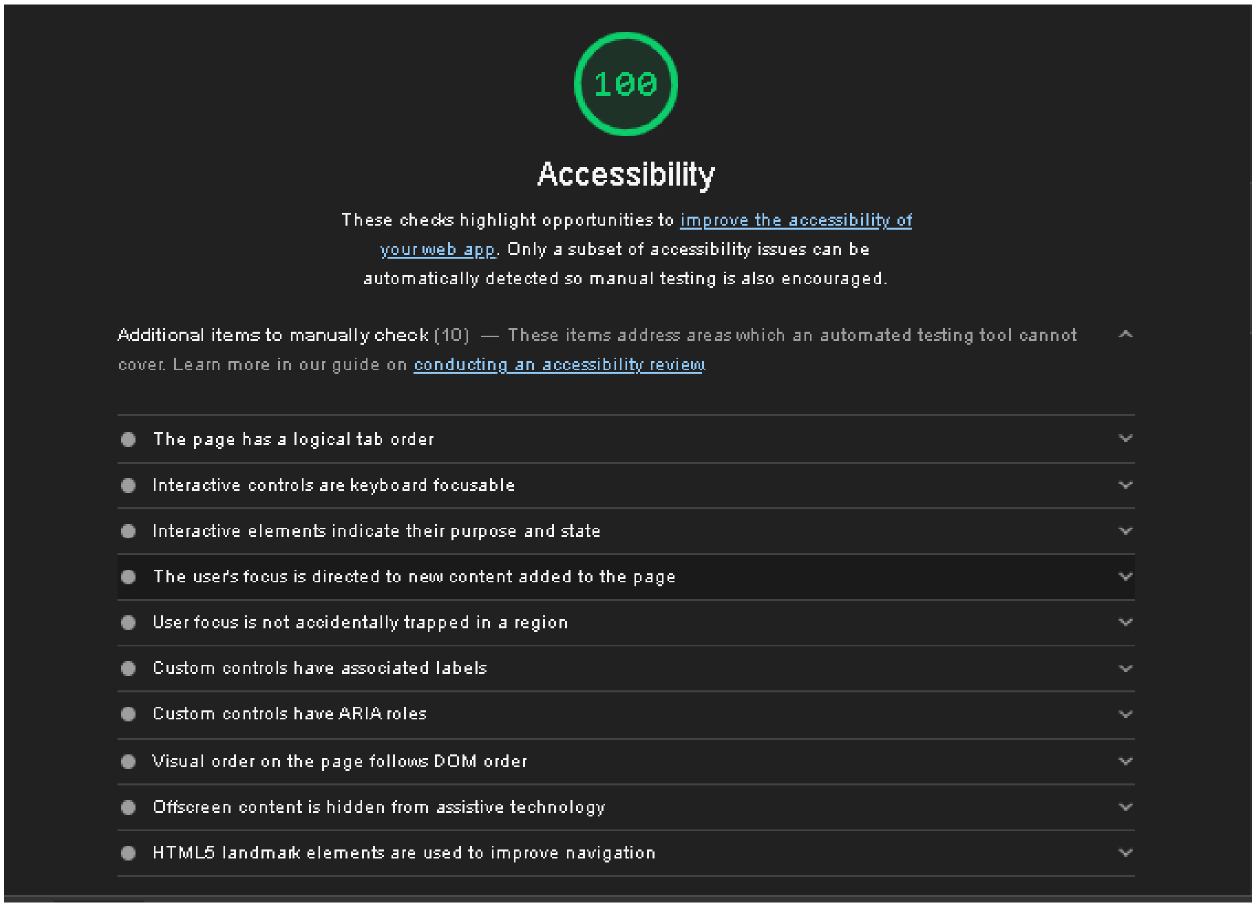
Performance



Metrics

■	First Contentful Paint	1.3 s	●	Time to Interactive	1.3 s
■	Speed Index	1.4 s	●	Total Blocking Time	0 ms
■	Largest Contentful Paint	1.3 s	●	Cumulative Layout Shift	0.008

Accessibility



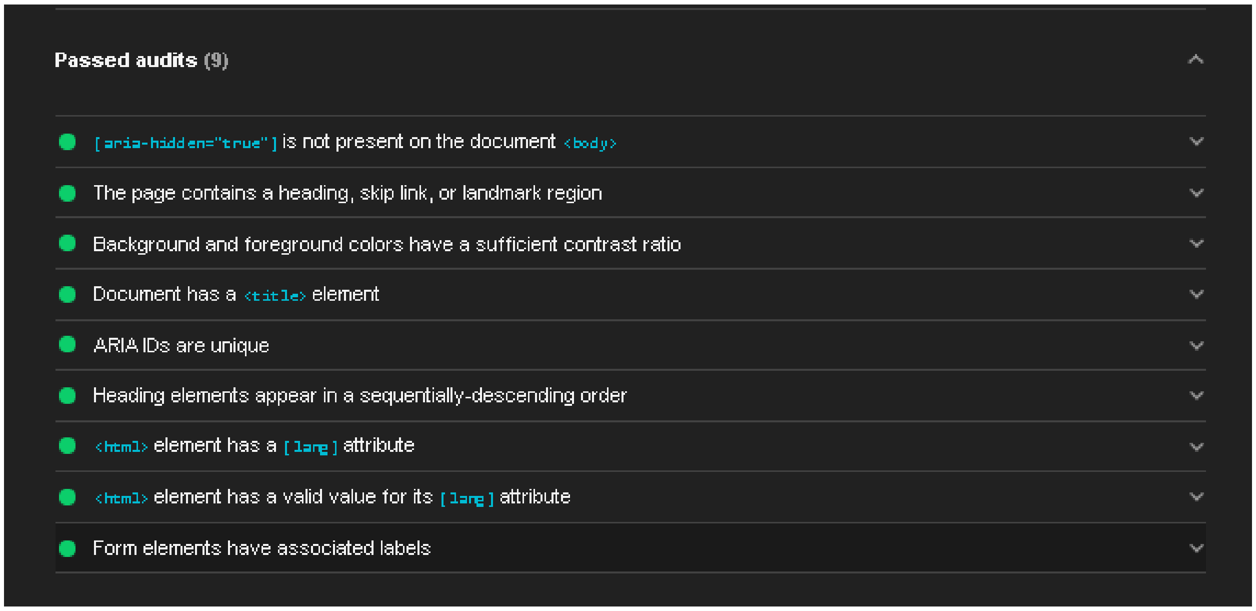
100

Accessibility

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.

Additional items to manually check (10) — These items address areas which an automated testing tool cannot cover. Learn more in our guide on conducting an accessibility review

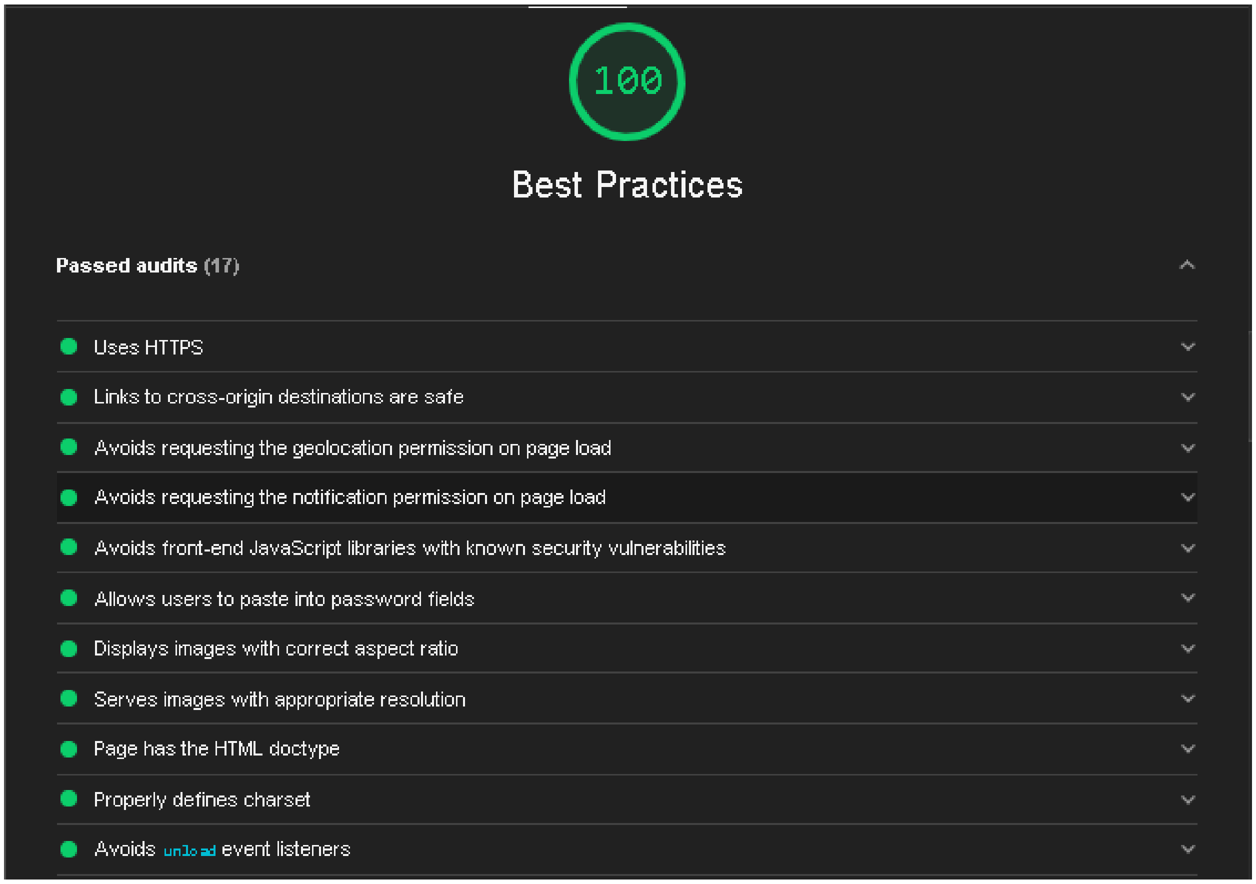
- The page has a logical tab order
- Interactive controls are keyboard focusable
- Interactive elements indicate their purpose and state
- The user's focus is directed to new content added to the page
- User focus is not accidentally trapped in a region
- Custom controls have associated labels
- Custom controls have ARIA roles
- Visual order on the page follows DOM order
- Offscreen content is hidden from assistive technology
- HTML5 landmark elements are used to improve navigation



Passed audits (9)

- `[aria-hidden="true"]` is not present on the document `<body>`
- The page contains a heading, skip link, or landmark region
- Background and foreground colors have a sufficient contrast ratio
- Document has a `<title>` element
- ARIA IDs are unique
- Heading elements appear in a sequentially-descending order
- `<html>` element has a `[lang]` attribute
- `<html>` element has a valid value for its `[lang]` attribute
- Form elements have associated labels

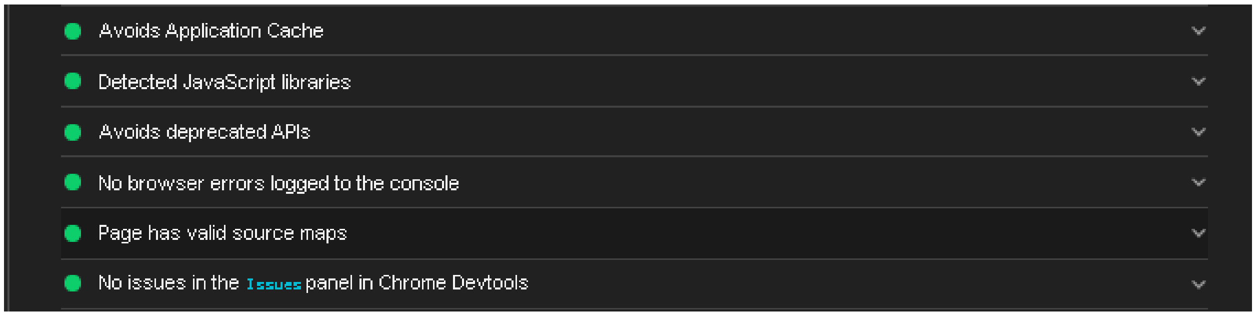
Best Practices


100

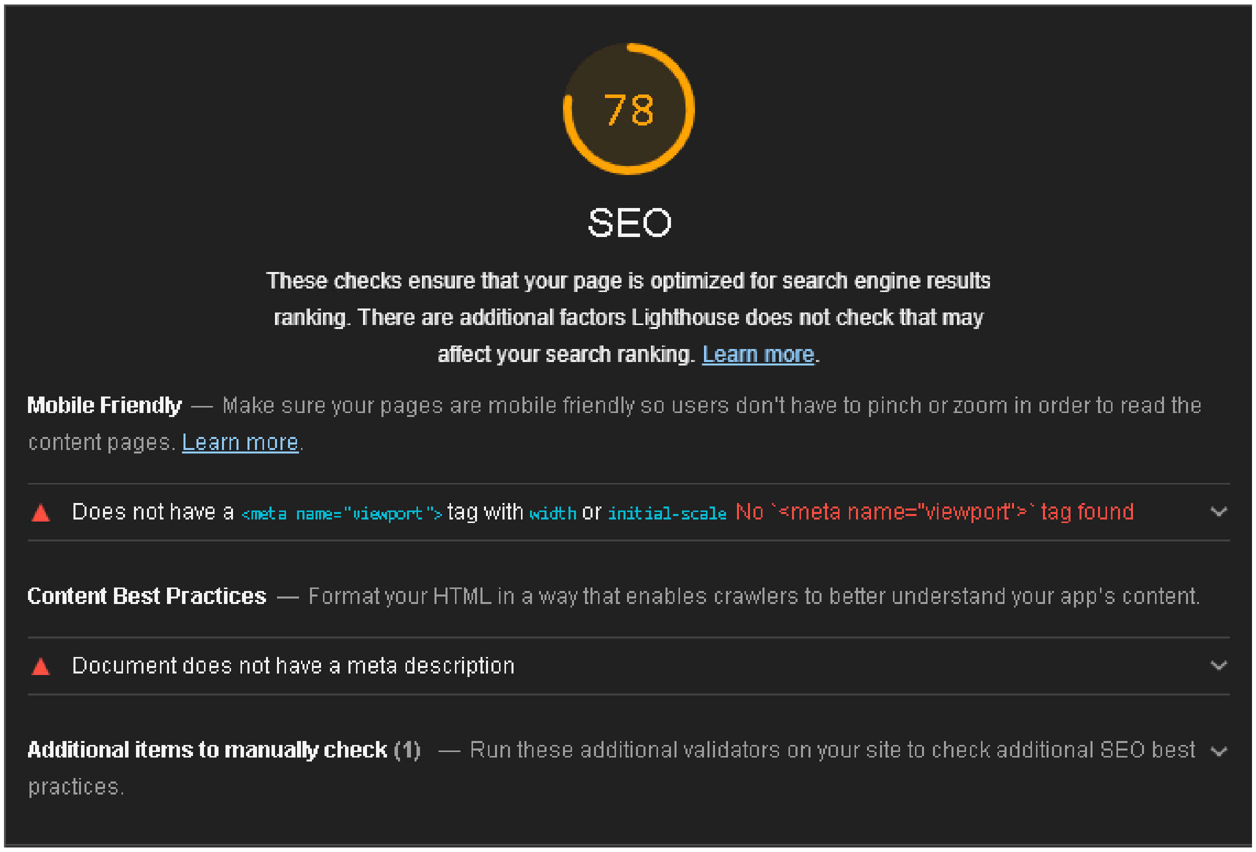
Best Practices

Passed audits (17)

- Uses HTTPS
- Links to cross-origin destinations are safe
- Avoids requesting the geolocation permission on page load
- Avoids requesting the notification permission on page load
- Avoids front-end JavaScript libraries with known security vulnerabilities
- Allows users to paste into password fields
- Displays images with correct aspect ratio
- Serves images with appropriate resolution
- Page has the HTML doctype
- Properly defines charset
- Avoids `unload` event listeners


- Avoids Application Cache
- Detected JavaScript libraries
- Avoids deprecated APIs
- No browser errors logged to the console
- Page has valid source maps
- No issues in the `Issues` panel in Chrome Devtools

SEO



78

SEO

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.

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.

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

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

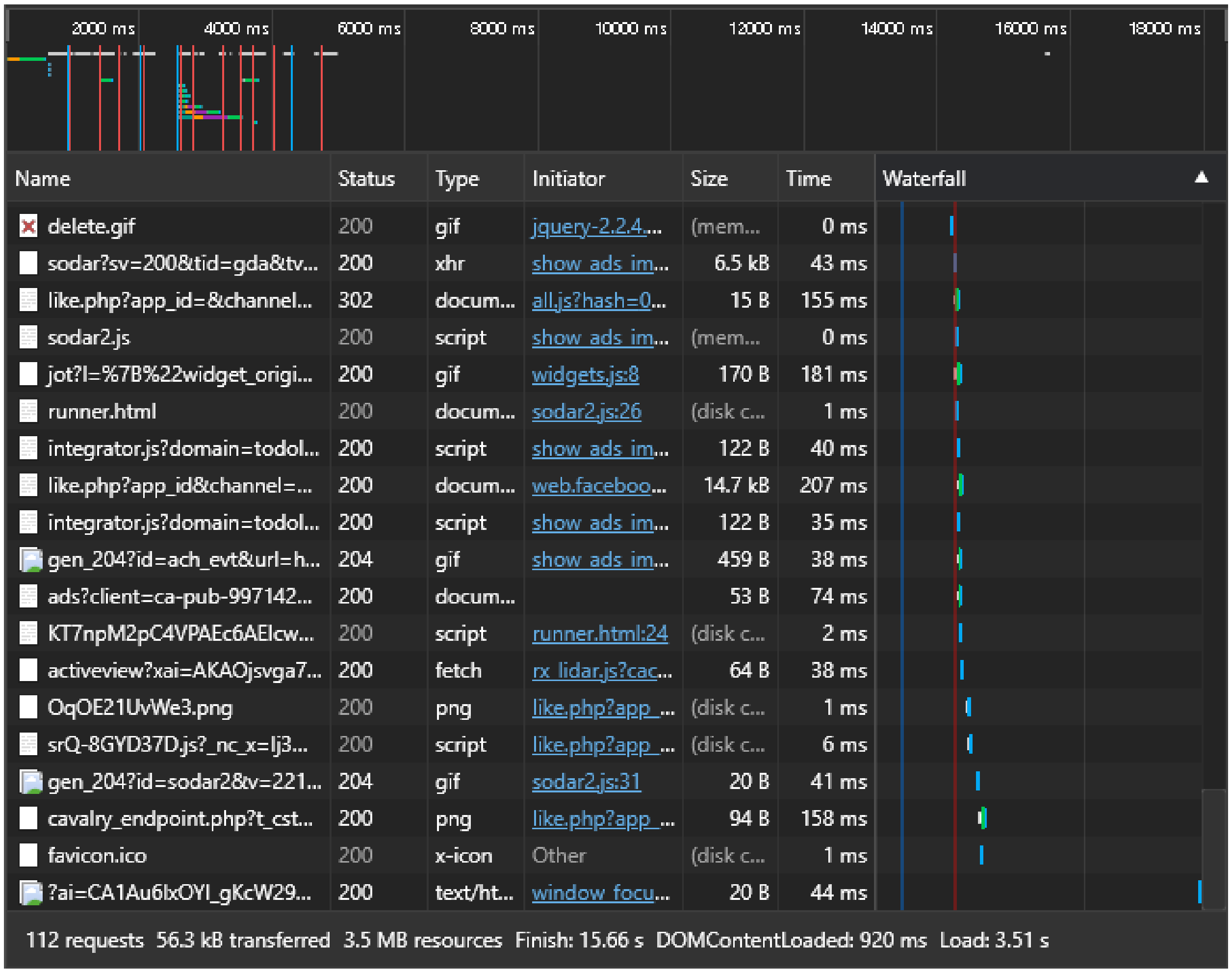
Additional items to manually check (1) — Run these additional validators on your site to check additional SEO best practices.

Suggestions

- Must use a color that will make user feel the app more readable and appealing to use
- Must add helpers (Guides in forms) with that, users can find this application self-learning without even a need of manual
- Must use the standard HTML5 metaviewport tag to make the site adaptive to any of screen sizes prioritizing mobile screens
- Must use standard and readable font sizes
- When in mobile view, icon sizes must be at least 48 x 48 px in size and use a border to determine that such icons will have a role in the web application.

Todolistme (Competitor website audit ***(http://todolistme.net/)***)

Network



Name	Status	Type	Initiator	Size	Time	Waterfall
delete.gif	200	gif	jquery-2.2.4....	(mem...	0 ms	
sodar?sv=200&tid=gda&tv...	200	xhr	show_ads_im...	6.5 kB	43 ms	
like.php?app_id=&channel...	302	docum...	all.js?hash=0...	15 B	155 ms	
sodar2.js	200	script	show_ads_im...	(mem...	0 ms	
jot?l=%7B%22widget_origi...	200	gif	widgets.js:8	170 B	181 ms	
runner.html	200	docum...	sodar2.js:26	(disk c...	1 ms	
integrator.js?domain=todol...	200	script	show_ads_im...	122 B	40 ms	
like.php?app_id&channel=...	200	docum...	web.faceboo...	14.7 kB	207 ms	
integrator.js?domain=todol...	200	script	show_ads_im...	122 B	35 ms	
gen_204?id=ach_evt&url=h...	204	gif	show_ads_im...	459 B	38 ms	
ads?client=ca-pub-997142...	200	docum...		53 B	74 ms	
KT7npM2pC4VPAEc6AElcw...	200	script	runner.html:24	(disk c...	2 ms	
activeview?xai=AKAOjsvga7...	200	fetch	rx_lidar.js?cac...	64 B	38 ms	
OqOE21UvWe3.png	200	png	like.php?app_...	(disk c...	1 ms	
srQ-8GYD37D.js?_nc_x=Ij3...	200	script	like.php?app_...	(disk c...	6 ms	
gen_204?id=sodar2&v=221...	204	gif	sodar2.js:31	20 B	41 ms	
cavalry_endpoint.php?t_cst...	200	png	like.php?app_...	94 B	158 ms	
favicon.ico	200	x-icon	Other	(disk c...	1 ms	
?ai=CA1Au6lxOYl_gKcW29...	200	text/ht...	window_focu...	20 B	44 ms	

112 requests 56.3 kB transferred 3.5 MB resources Finish: 15.66 s DOMContentLoaded: 920 ms Load: 3.51 s

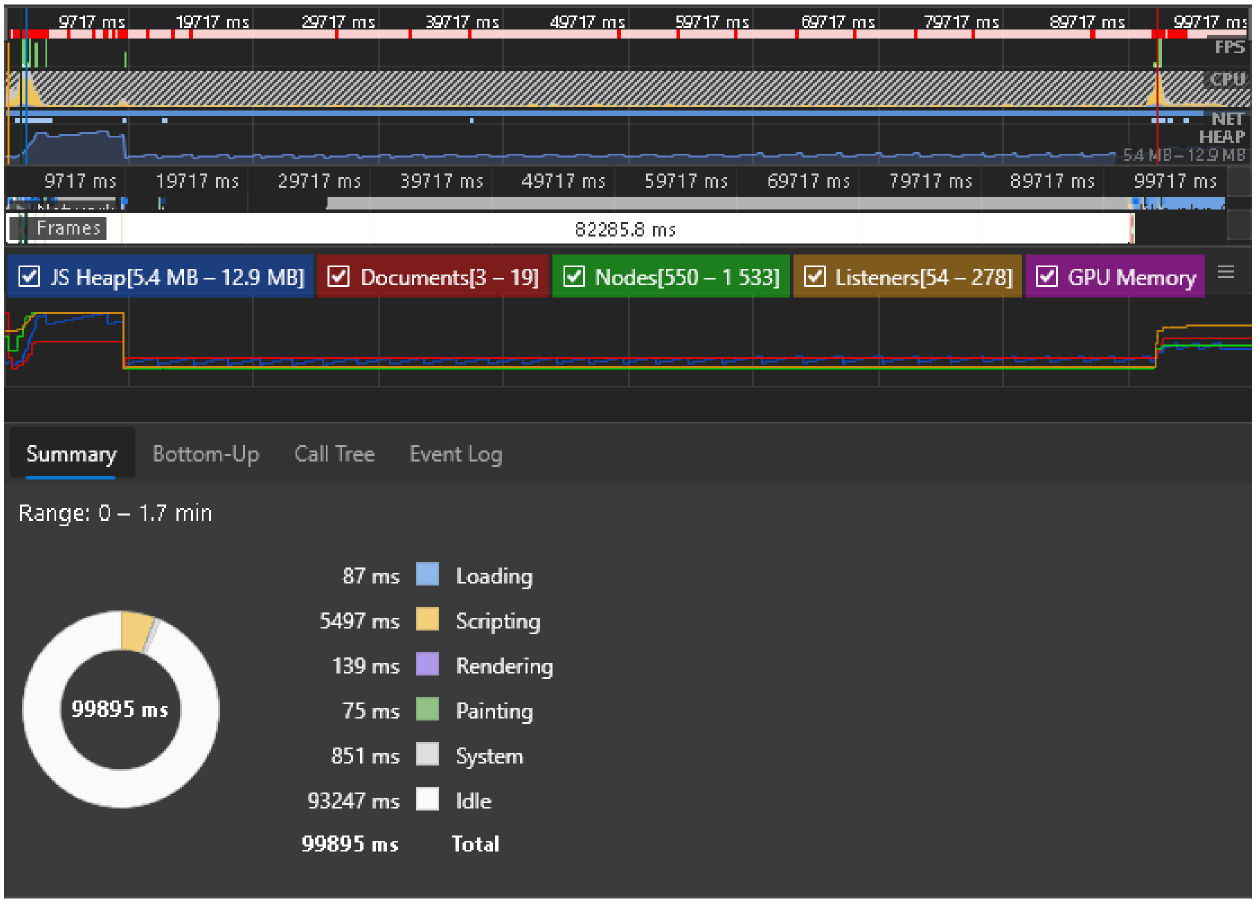
Requests sent: 112

Resources transferred: 3.5 MB

Load time: 3.51 s

DOMContentLoaded: 920 ms

Performance



Range: 0 – 1.7 min

87 ms	🟦	Loading
5497 ms	🟨	Scripting
139 ms	🟪	Rendering
75 ms	🟩	Painting
851 ms	⬜	System
93247 ms	⬜	Idle
99895 ms		**Total**

Loading: 87 ms

Scripting: 5497 ms

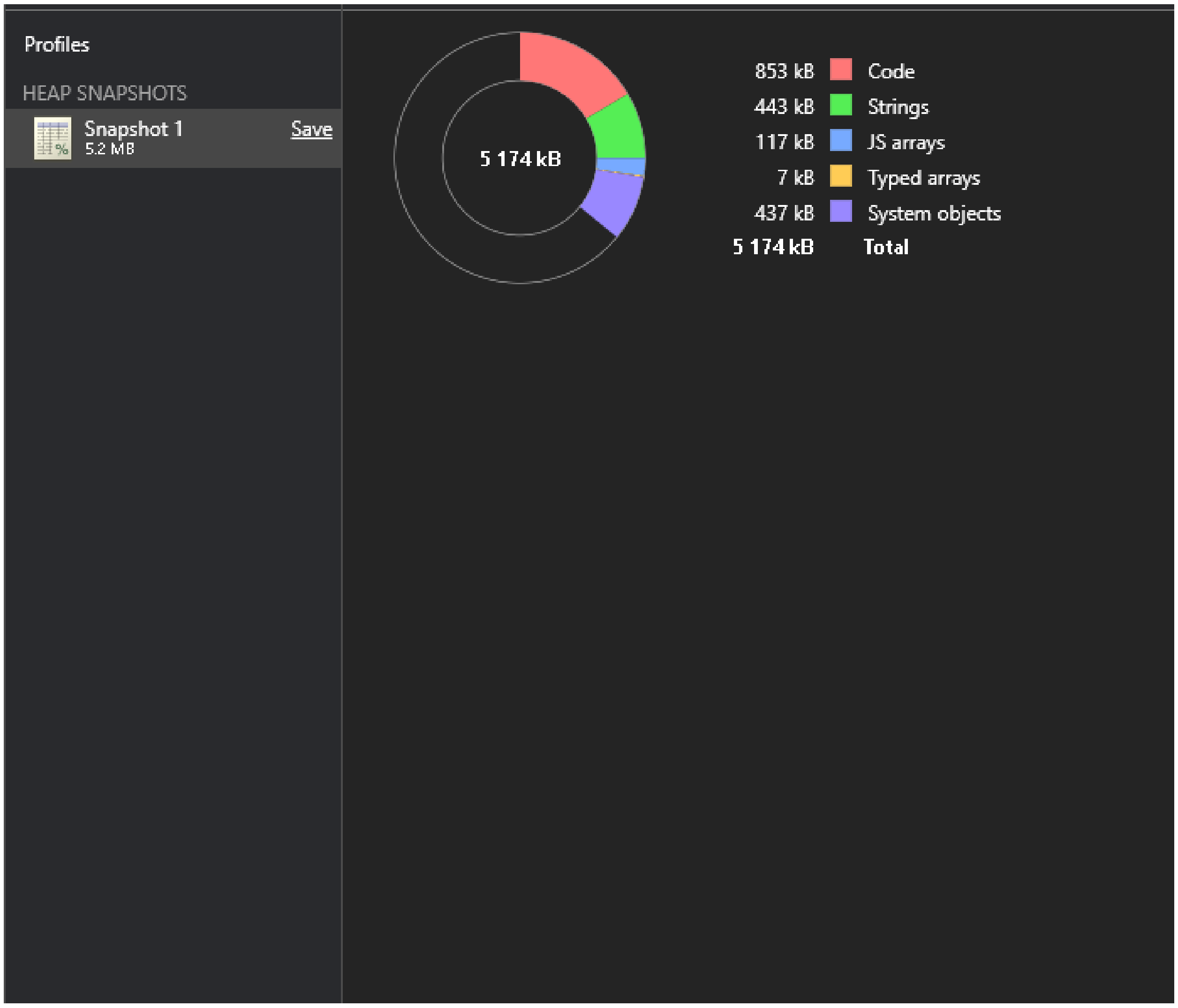
Rendering: 139 ms

Painting: 75 ms

System: 851 ms

Idle: 93247 ms

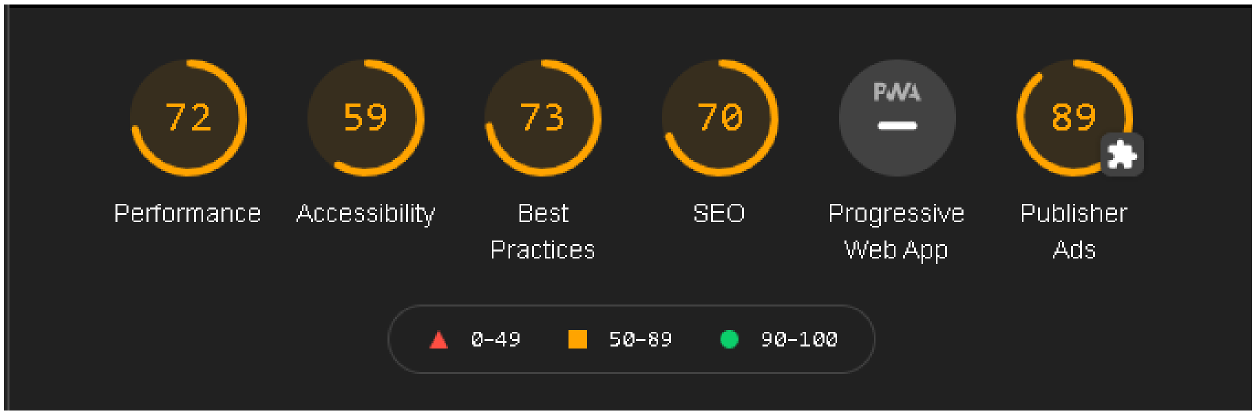
Memory



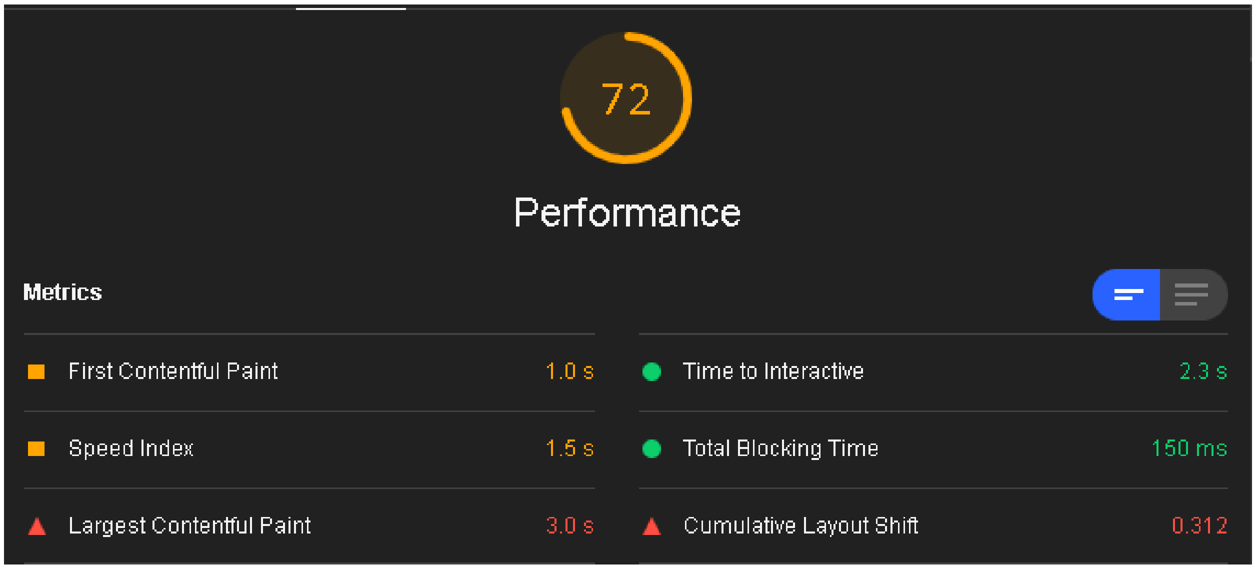
853 kB	🟥	Code
443 kB	🟩	Strings
117 kB	🟦	JS arrays
7 kB	🟨	Typed arrays
437 kB	🟪	System objects
5 174 kB		Total

Total Memory usage: 5.2 MB

Overall Performance Audit



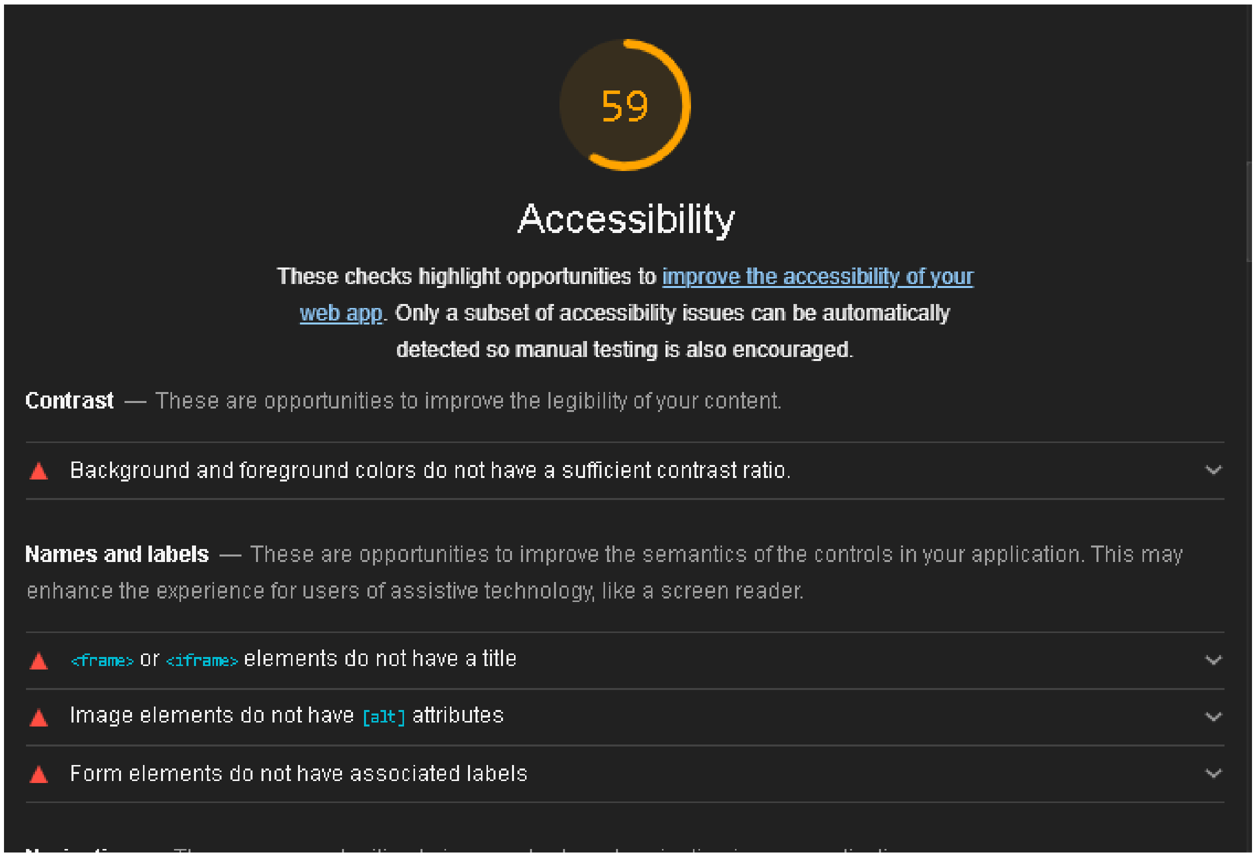
Performance



Metrics

■	First Contentful Paint	1.0 s	●	Time to Interactive	2.3 s
■	Speed Index	1.5 s	●	Total Blocking Time	150 ms
▲	Largest Contentful Paint	3.0 s	▲	Cumulative Layout Shift	0.312

Accessibility



59

Accessibility

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.

Contrast — These are opportunities to improve the legibility of your content.

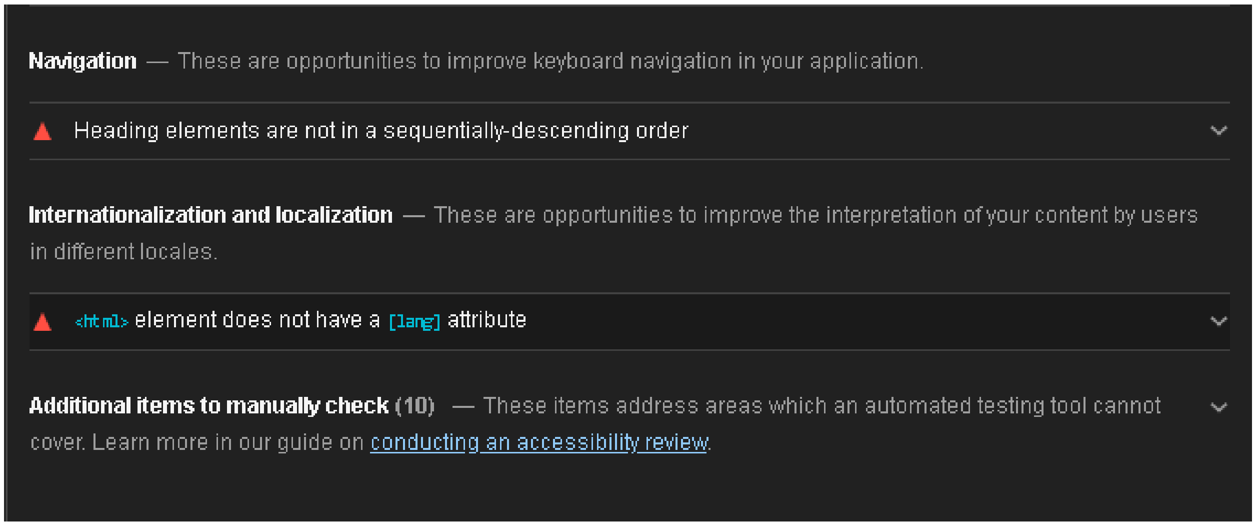
▲ Background and foreground colors do not have a sufficient contrast ratio.

Names and labels — 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.

▲ `<frame>` or `<iframe>` elements do not have a title

▲ Image elements do not have `[alt]` attributes

▲ Form elements do not have associated labels



Navigation — These are opportunities to improve keyboard navigation in your application.

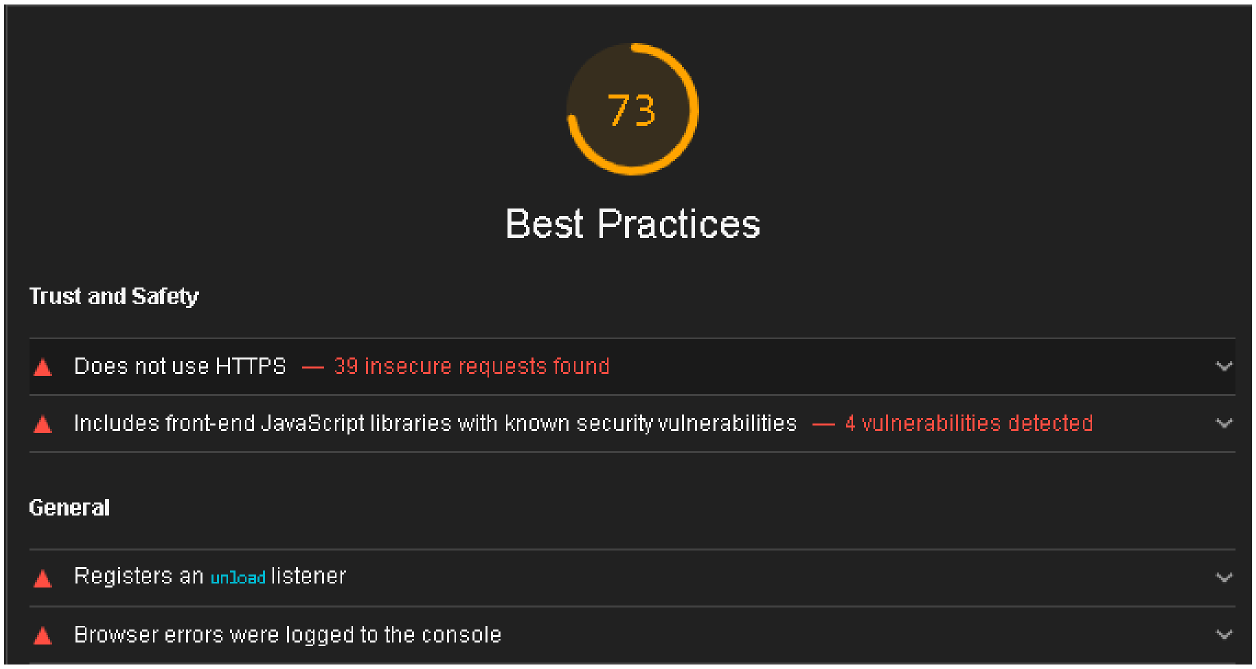
▲ Heading elements are not in a sequentially-descending order

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

▲ `<html>` element does not have a `[lang]` attribute

Additional items to manually check (10) — These items address areas which an automated testing tool cannot cover. Learn more in our guide on conducting an accessibility review.

Best Practices


73

Best Practices

Trust and Safety

▲ Does not use HTTPS — 39 insecure requests found

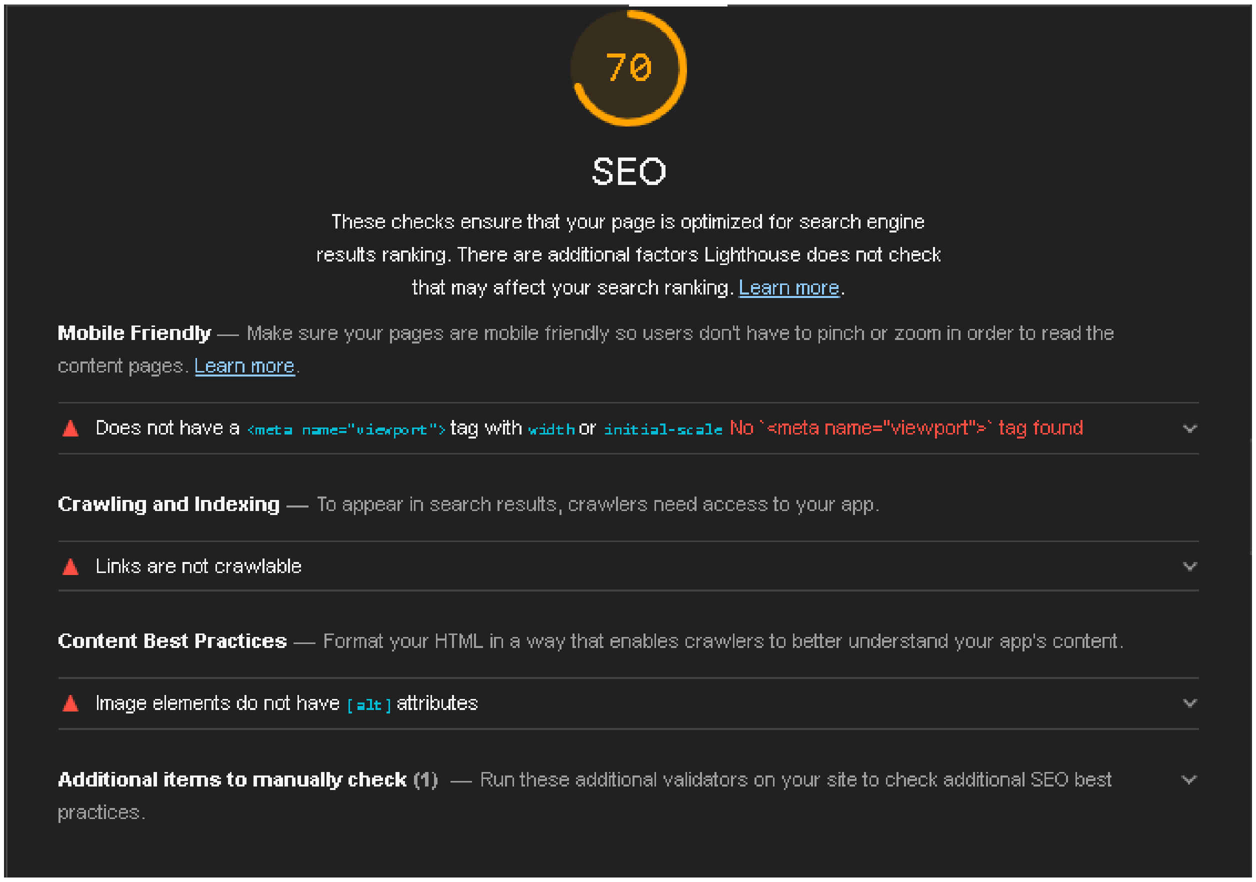
▲ Includes front-end JavaScript libraries with known security vulnerabilities — 4 vulnerabilities detected

General

▲ Registers an `unload` listener

▲ Browser errors were logged to the console

SEO


70

SEO

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.

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.

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

Crawling and Indexing — To appear in search results, crawlers need access to your app.

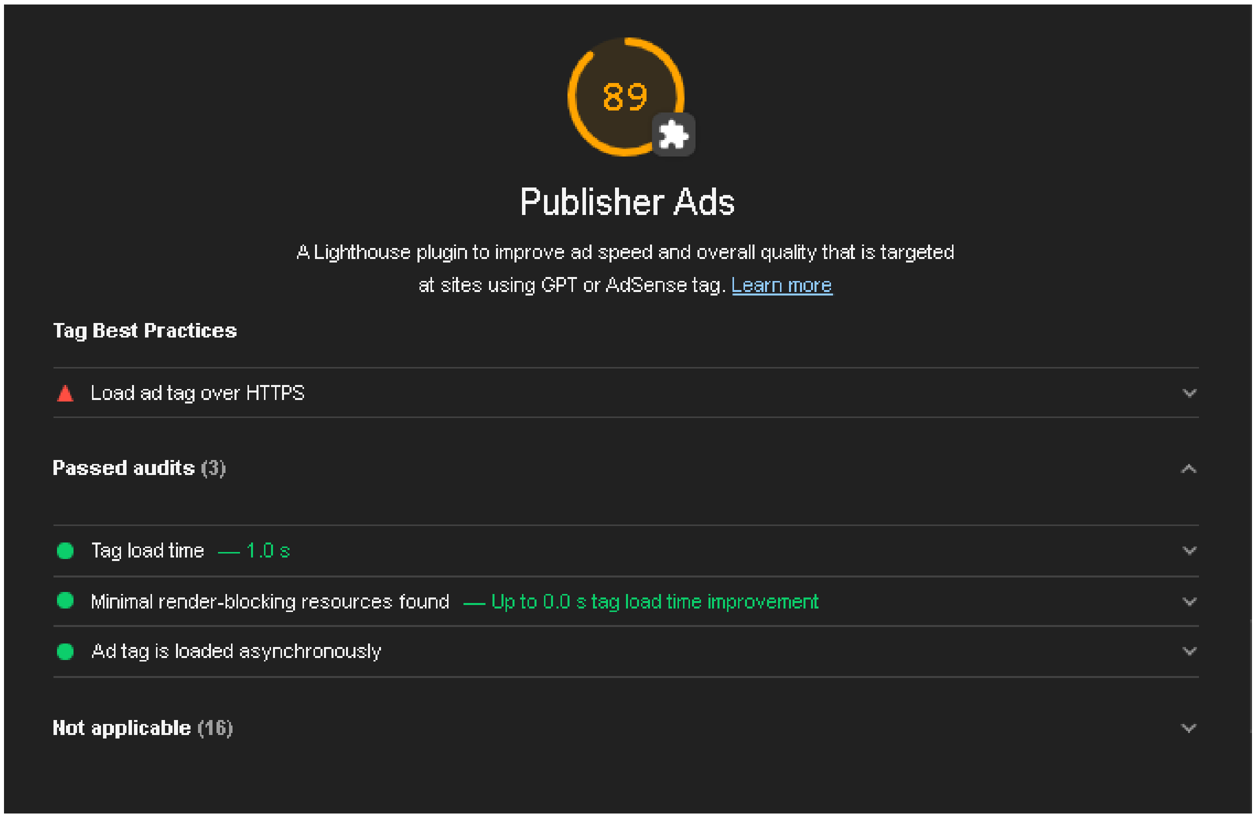
▲ Links are not crawlable

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

▲ Image elements do not have `[alt]` attributes

Additional items to manually check (1) — Run these additional validators on your site to check additional SEO best practices.

Publisher Ads



89

Publisher Ads

A Lighthouse plugin to improve ad speed and overall quality that is targeted at sites using GPT or AdSense tag. Learn more

Tag Best Practices

▲ Load ad tag over HTTPS

Passed audits (3)

● Tag load time — 1.0 s

● Minimal render-blocking resources found — Up to 0.0 s tag load time improvement

● Ad tag is loaded asynchronously

Not applicable (16)

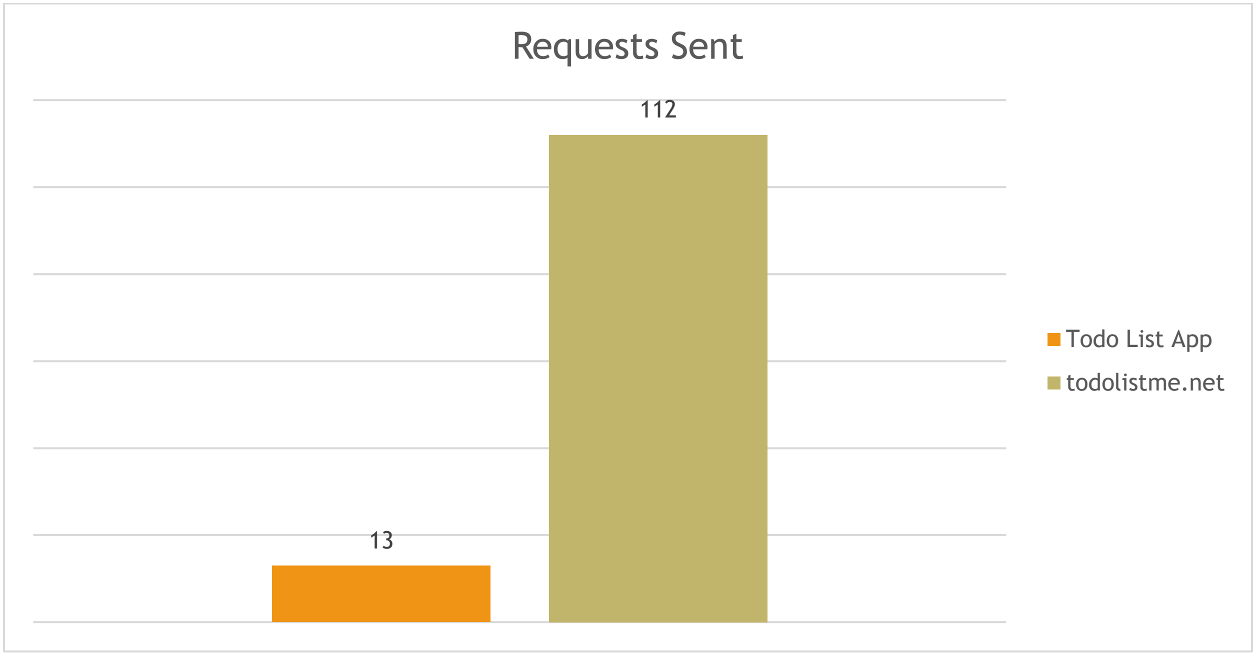
Suggestions

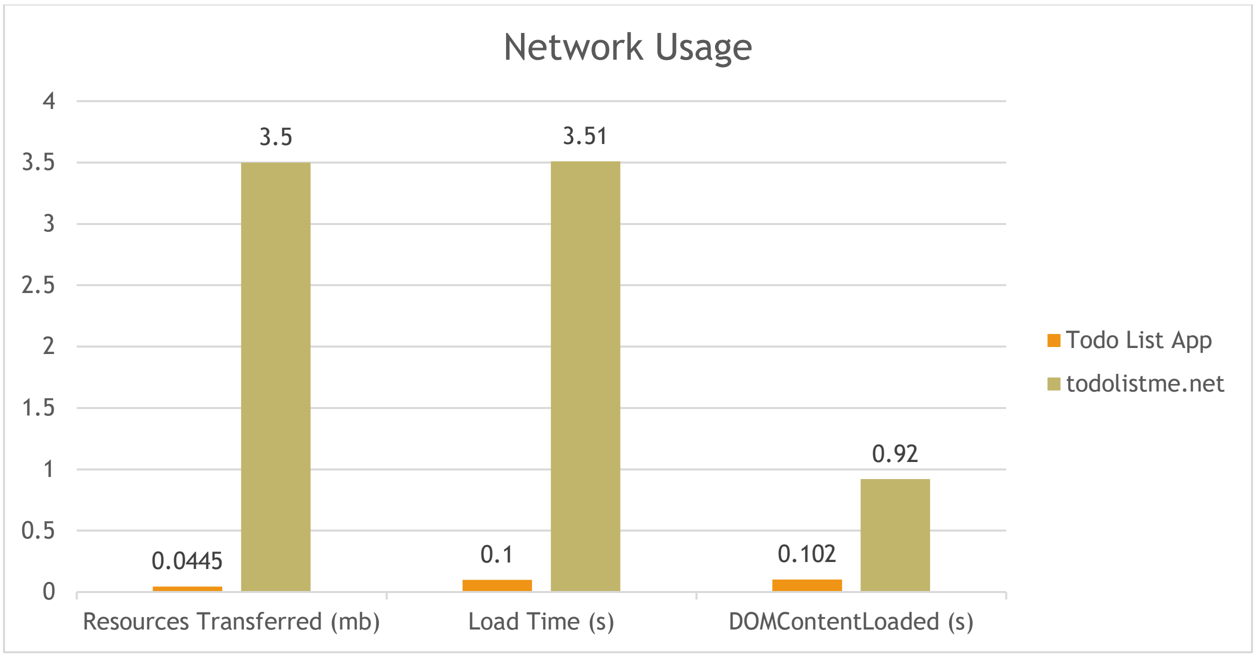
- Must be mobile-friendly in terms of screen size adaptability
- Should use https in order to make the web app secure
- Must have redesigned the whole interface. It is quite confusing to see the controls in their current placement
- Should have a minimalist design. There are a lot of unnecessary controls that should not be placed in their web app
- Must remove the ads as it will add loading time in loading the page
- Must use a simple color scheme that is readable enough for every user
- Should use a standard arial font instead of the times new roman is it is really painful to read times roman fonts
- It is XSS (Cross-site scripting) injectable. Must sanitize entered values prior to submitting in the list
- Must use a modern design. The design is way past its time. It looks like a design from the year late 1990s – 2005-ish

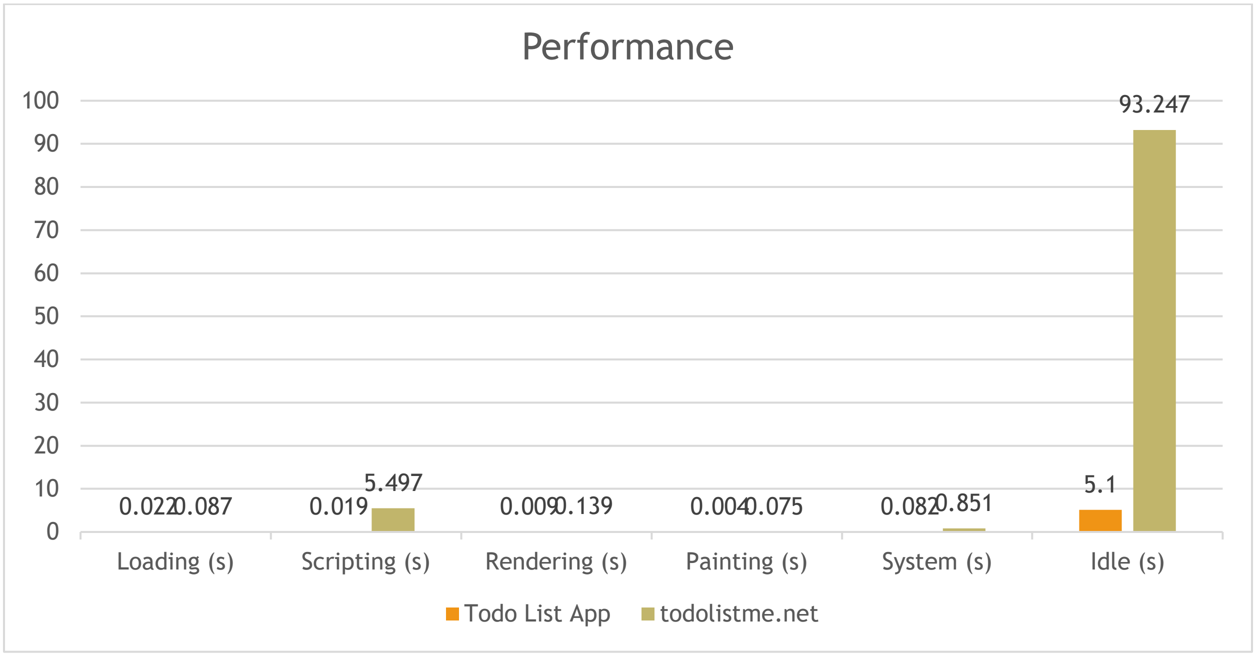
Chapter 5

This chapter contains the comparison of audit between the two competing web application. We have also included edge of the application against each other. Since most of the results came in milliseconds, I have converted it into seconds, with that, we can see how much time it consumes when they work.

Comparison of the Audit







Our Todo List app

Advantage

- Short loading time

- Less memory usage
- Clean and readable code (uses MVC-architecture)
- Code has comments which can guide reader what does code does
- Design is simple

Disadvantage

- Limited functionality
- The storage used for storing data can be exploited and, can be removed once cache/cookies are cleared from the browser

Todo List Me App (Competitor)

Advantage

- Have list sorting ability
- Can print lists
- Has drag-and-drop functionalities
- Can do multiple lists/categories
- Remote saving of data

Disadvantage

- Higher network usage
- Ads affected the loading time
- High memory usage

- Code is not well-written (It is XSS injectable)
- Long response of performing functionalities