Todo-list-app documentation

1 . Todo-list-app Brief

Todo-list-app is an application for tasks list management. This app allows us to add a new task to todo-list, update the task, delete the task, and change the state of the task to completed or active. This app build with minimalist design and basic todo-list-app functionality.

This app is created using MVC design pattern in Javascript which separates data logic (Model), display functionality (View), and connect them with Controller.
Model : contains local storage object and prototypes function to manage (create, update, remove, count) it for todolist-app tasks data in browser. It defines data management of the app.
View : contains DOM element and its structure manipulation functionality that represent displays and user interactions.
Controller : contains prototypes function for controlling interactions between Model and View. Controller functions do listens to user actions from view, converting input from View, then calling Model to perform requested operations, and calling View to display results from Model.

2. Program Specification

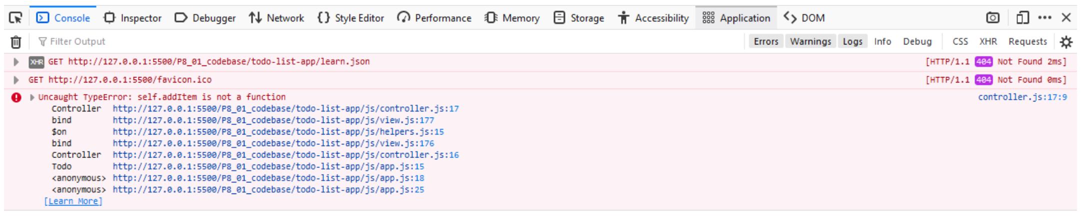
- index.html is the entry point of this todomvc application.
- Index.css is the css styles of this todomvc application.
- Base.css is the common style of this todomvc application.
- App.js creates the instance of todo application and initiate object needed like the storage, Model, View, and Controller Object.
- Store.js creates the data storage object using local session storage and manages it with functions.
- Helper.js contains helper functions for query selector, event listener wrapper, and handler attach or delegate.
- Template.js contains template function to escape characters, display list items, change button states.
- Controller.js creates controller objects that connect interaction between Model and View with its prototypes such as :
  - setView to loads and initialize the view,
  - showAll to displays all items in the todo-list,
  - showActive to renders uncompleted tasks,
  - showCompleted to renders completed tasks,
  - addItem to creates new todo task, saving it in the local storage by adding ID,
  - editItem to starts editing mode of todo task by matching with the correct ID,
  - editItemSave to successfully edits item and save the changing by using matched ID,
  - editItemCancel to cancels the item editing mode,

- removeItem to removes item from to-do-list and storage by using its ID as a parameter,
- removeCompletedItems to removes all completed tasks,
- toggleComplete to gives ID and updates the state of completeness of task in the storage,
- toggleAll to change the state of completeness of the tasks: on/off

- Model.js to create model objects and connects it with storage object and provide prototypes such as :
  - create to creates a new todo model and saves it in the storage,
  - read to finds and returns a model in storage, if the query isn't given, returns everything,
  - update to updates a model, every action based on unique ID,
  - remove to removes a model from storage,
  - removeAll to removes all data from storage,
  - getCount to counting active, completed and total tasks by finding the in the storage.

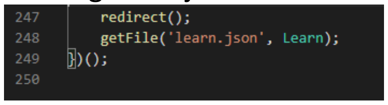
- View.js to create View Object to manipulate DOM attached to user interaction such as :
  - bind to takes a todo application event and registers the handler,
  - render to renders the given command with the options.

3. Manual Bugs Fix



```
XHR GET http://127.0.0.1:5500/P8_01_codebase/todo-list-app/learn.json          [HTTP/1.1 404 Not Found 2ms]
GET http://127.0.0.1:5500/favicon.ico                                          [HTTP/1.1 404 Not Found 0ms]
Uncaught TypeError: self.addItem is not a function                                       controller.js:17:9
    Controller  http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/controller.js:17
    bind        http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/view.js:177
    $on         http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/helpers.js:15
    bind        http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/view.js:176
    Controller  http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/controller.js:16
    Todo        http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/app.js:15
    <anonymous> http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/app.js:18
    <anonymous> http://127.0.0.1:5500/P8_01_codebase/todo-list-app/js/app.js:25
  [Learn More]
```

1. Typo mistake:
   - Controller.prototype.adddItem at js/controller.js(line 95)
     Fixed become :
     - Controller.prototype.addItem
2. Missing learn.json file when base.js try to get file on line 248



```
247         redirect();
248         getFile('learn.json', Learn);
249     })();
250
```

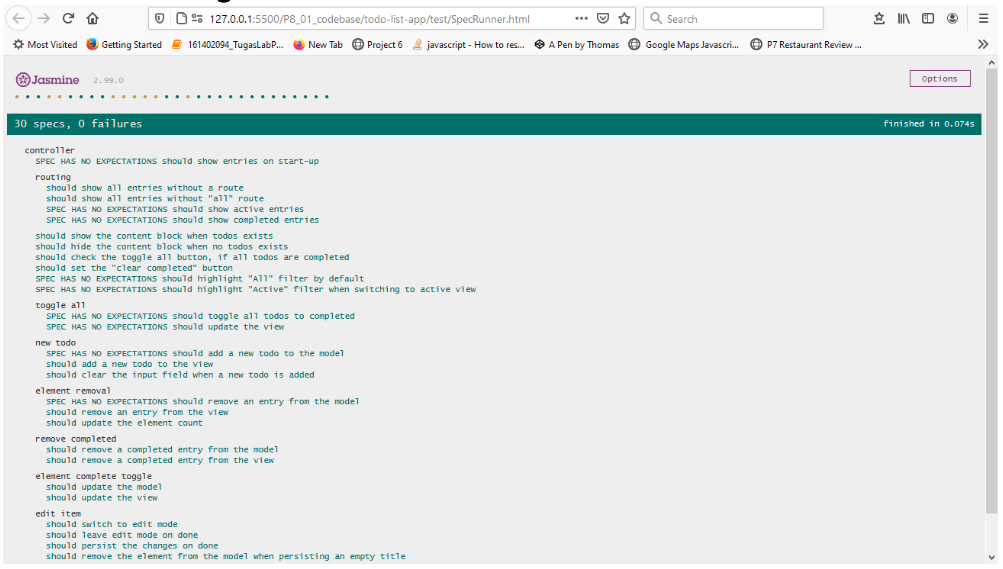
   Fixed become :
   - Create learn.json inside todo-list-app

3. Associated label to the input with class not id since a label can be associated only with exactly one form control. The same class can be used by multiple elements

therefore it will be problem later figuring out in which case, which element would the label be for?

- <input class="toggle-all" type="checkbox">
  <label for="toggle-all">Mark all as complete</label>
  Fixed become:
  - <input class="toggle-all" id=” toggle-all” type="checkbox">

4. Jasmine Unit Testing.



```
127.0.0.1:5500/P8_01_codebase/todo-list-app/test/SpecRunner.html

Jasmine 2.99.0                                                    Options

30 specs, 0 failures                                    finished in 0.074s

controller
  SPEC HAS NO EXPECTATIONS should show entries on start-up
  routing
    should show all entries without a route
    should show all entries without "all" route
    SPEC HAS NO EXPECTATIONS should show active entries
    SPEC HAS NO EXPECTATIONS 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
  SPEC HAS NO EXPECTATIONS should highlight "All" filter by default
  SPEC HAS NO EXPECTATIONS should highlight "Active" filter when switching to active view
  toggle all
    SPEC HAS NO EXPECTATIONS should toggle all todos to completed
    SPEC HAS NO EXPECTATIONS should update the view
  new todo
    SPEC HAS NO EXPECTATIONS 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
    SPEC HAS NO EXPECTATIONS 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
```

Additional test is required to already written one since some specs are still empty and have no expectation define as shown above. To do so new tests were added based on existing testing strategy. Newly created tests check following cases required:

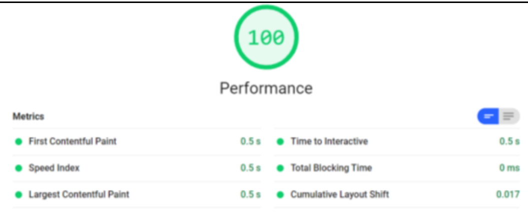
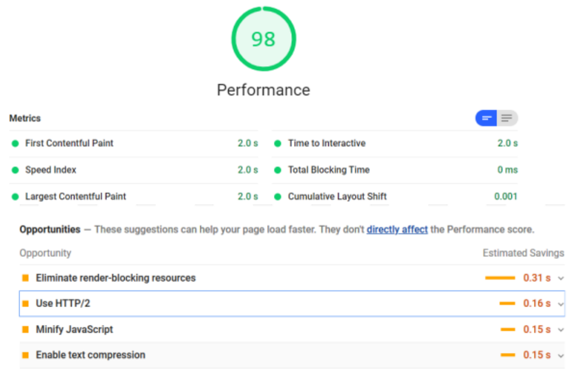
1. should show entries on start-up
2. should show all entries without "all" route
3. should show active entries
4. should show completed entries
5. should show the content block when todos exists
6. should highlight "All" filter by default
7. should toggle all todos to completed
8. should update the view
9. should add a new todo to the model
10. should remove an entry from the model

To run the jasmine unit testing open SpecRunner.html in test folder through any browser and you’ll find newly created test asked on the requirement.

5. Audit todo-list app

the audit of todo-list app was performed using Lighthouse developer tools in google chrome browser on windows towards overall performance score of the app in desktop and mobile device.

##results shows :

Desktop	Mobile
 100 Performance Metrics ● First Contentful Paint 0.5 s ● Time to Interactive 0.5 s ● Speed Index 0.5 s ● Total Blocking Time 0 ms ● Largest Contentful Paint 0.5 s ● Cumulative Layout Shift 0.017	 98 Performance Metrics ● First Contentful Paint 2.0 s ● Time to Interactive 2.0 s ● Speed Index 2.0 s ● Total Blocking Time 0 ms ● Largest Contentful Paint 2.0 s ● Cumulative Layout Shift 0.001 **Opportunities** — These suggestions can help your page load faster. They don't directly affect the Performance score. Opportunity / Estimated Savings: Eliminate render-blocking resources 0.31 s; Use HTTP/2 0.16 s; Minify JavaScript 0.15 s; Enable text compression 0.15 s

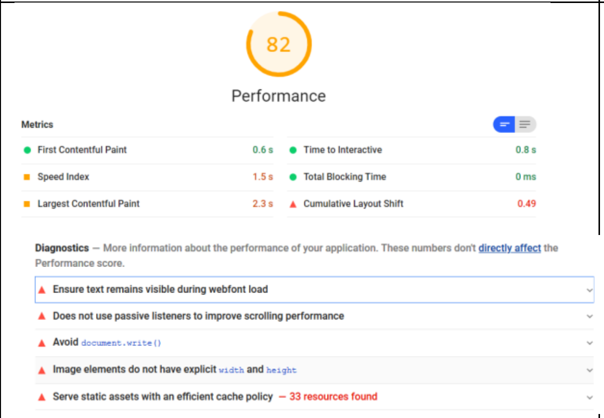
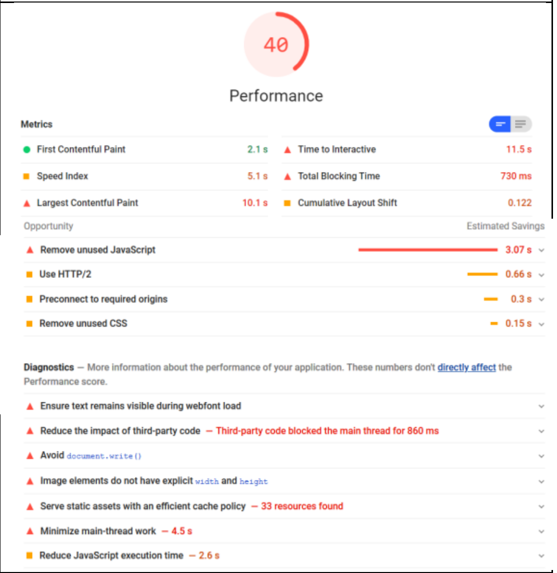
No.	Metrics	Desktop	Mobile
1.	First Contentful Paint : how long it takes the browser to render the first piece of DOM content after a user navigates to your page.	0.5s	2.0s
2.	Speed Index : measures how quickly content is visually displayed during page load. Video first capture then progression frame between.	0.5s	2.0s
3.	Largest Contentful Paint: measures the time from when the page starts loading to when the largest text block or image element is rendered on the screen	0.5s	2.0s
4.	Time to Interactive: measures how long it takes a page to become *fully* interactive. A page is considered fully interactive when: • The page displays useful content, which is measured by the First Contentful Paint, • Event handlers are registered for most visible page elements, and • The page responds to user interactions within 50 milliseconds.	0.5s	2.0s
5.	Total Blocking Time : measures the total amount of time that a page is blocked from responding to user	0ms	0ms

	input, such as mouse clicks, screen taps, or keyboard presses. The sum is calculated by adding the *blocking portion* of all long tasks between First Contentful Paint and Time to Interactive. Any task that executes for more than 50 ms is a long task. The amount of time after 50 ms is the blocking portion. For example, if Lighthouse detects a 70 ms long task, the blocking portion would be 20 ms.		
6.	Cumulative Layout Shift : measures the movement of visible elements within the viewport to avoid getting unexpected change to elements position due to unknown dom element get added dynamically to existing content of page.	0.017	0.001
7.	Overall Performance Score	100	98

6. Audit competitor todo-list app

Competitors webpage todolistme.net was examined as recommended to be compared with this todo-list app by using lighthouse google chrome developer tools as well.

##results shows :

Desktop	Mobile
	

Desktop:

82

Performance

Metrics

● First Contentful Paint	0.6 s	● Time to Interactive	0.8 s
■ Speed Index	1.5 s	● Total Blocking Time	0 ms
■ Largest Contentful Paint	2.3 s	▲ Cumulative Layout Shift	0.49

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

- ▲ Ensure text remains visible during webfont load
- ▲ Does not use passive listeners to improve scrolling performance
- ▲ Avoid `document.write()`
- ▲ Image elements do not have explicit `width` and `height`
- ▲ Serve static assets with an efficient cache policy — 33 resources found

Mobile:

40

Performance

Metrics

● First Contentful Paint	2.1 s	▲ Time to Interactive	11.5 s
■ Speed Index	5.1 s	▲ Total Blocking Time	730 ms
▲ Largest Contentful Paint	10.1 s	■ Cumulative Layout Shift	0.122

Opportunity	Estimated Savings
▲ Remove unused JavaScript	3.07 s
■ Use HTTP/2	0.66 s
■ Preconnect to required origins	0.3 s
■ Remove unused CSS	0.15 s

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

- ▲ Ensure text remains visible during webfont load
- ▲ Reduce the impact of third-party code — Third-party code blocked the main thread for 860 ms
- ▲ Avoid `document.write()`
- ▲ Image elements do not have explicit `width` and `height`
- ▲ Serve static assets with an efficient cache policy — 33 resources found
- ▲ Minimize main-thread work — 4.5 s
- ■ Reduce JavaScript execution time — 2.6 s

No.	Metrics	Desktop	Mobile
1.	First Contentful Paint : how long it takes the browser to render the first piece of DOM content after a user navigates to your page.	0.6s	2.1s
2.	Speed Index : measures how quickly content is visually displayed during page load. Video first capture then progression frame between.	1.5s	5.1s
3.	Largest Contentful Paint: measures the time from when the page starts loading to when the largest text block or image element is rendered on the screen	2.3s	10.1s
4.	Time to Interactive: measures how long it takes a page to become *fully* interactive. A page is considered fully interactive when: • The page displays useful content, which is measured by the First Contentful Paint, • Event handlers are registered for most visible page elements, and • The page responds to user interactions within 50 milliseconds.	0.8s	11.5s
5.	Total Blocking Time : measures the total amount of time that a page is blocked from responding to user input, such as mouse clicks, screen taps, or keyboard presses. The sum is calculated by adding the *blocking portion* of all long tasks between First Contentful Paint and Time to Interactive. Any task that executes for more than 50 ms is a long task. The amount of time after 50 ms is the blocking portion. For example, if Lighthouse detects a 70 ms long task, the blocking portion would be 20 ms.	0ms	730ms
6.	Cumulative Layout Shift : measures the movement of visible elements within the viewport to avoid getting unexpected change to elements position due to unknown dom element get added dynamically to existing content of page.	0.49	0.122
7.	Overall Performance Score	82	40

7. Competitor Comparison

The table below shows the results of comparison between two web application described before. To highlight the performance of the todo-list-app it was recommended to compare its results to the competitor webpage: todolistme.net application:

No.	Metrics	Todo-list app		Todolistme.net	
		Desktop	Mobile	Desktop	Mobile
1.	First Contentful Paint	0.5s	2.0s	0.6s	2.1s
2.	Speed Index	0.5s	2.0s	1.5s	5.1s
3.	Largest Contentful Paint	0.5s	2.0s	2.3s	10.1s
4.	Time to Interactive	0.5s	2.0s	0.8s	11.5s
5.	Total Blocking Time	0ms	0ms	0ms	730ms
6.	Cumulative Layout Shift	0.017	0.001	0.49	0.122
7.	Overall Performance Score	100	98	82	40

8. Summary

Based on Audit result from lighthouse google chrome developer tools. The summary of audit is shown bellow:

Todo-list-app

No.	Pros (+)	Cons (-)
1.	Show great performance with overall performance score 98 to 100	Basic Todo-list-app Functionality
2.	Fast load and operate	Needs text compression with gzip to minimize total network bytes
3.	Minimalist Design	Needs minifying javascript files and CSS file to reduce payload sizes and script parse time using webPack plugin.
4.	Code are well commented and Based On MVC model makes the code more readable	
5.	Based on google's opinion through lighthouse metrics. This app give better user experience since each metric give good values (green)	

Todolistme.net

No.	Pros (+)	Cons (-)
1.	Contains more complete todolist-app functionality where user can : • registration, • synchronizing, • printing, • commercials, • multiple lists, • sorting, • drag and drop reordering.	Show moderate to bad performance with overall performance score 82 to 40
2.	Javacsript and CSS are optimized with minified so it already reduce payload sizes and script parse time.	Slow Loading
3.	Images are optimized with image formats like JPEG 2000, JPEG XR, and WebP so it provides better compression than PNG or JPEG, which means faster downloads and less cellular data consumption.	Has high CLS score that impacts on unexpected layout shifts due to inexplicitly set image elements width and height
4.	Text compression is enabled made total network bytes data consumption is minimized	number of redundant third-party providers (google double click ads, google analytics, google fonts, other api) and its code scripts injected delays loading and increase data transfer impact slow process.
5.		Needs to serve static assets with an efficient cache policy to serve long cache lifetime and speed up page

Github Wiki: can be found in https://github.com/sintaanjelina/P8_sinta/wiki