ToDo List App Technical Documentation

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

Chapter 2

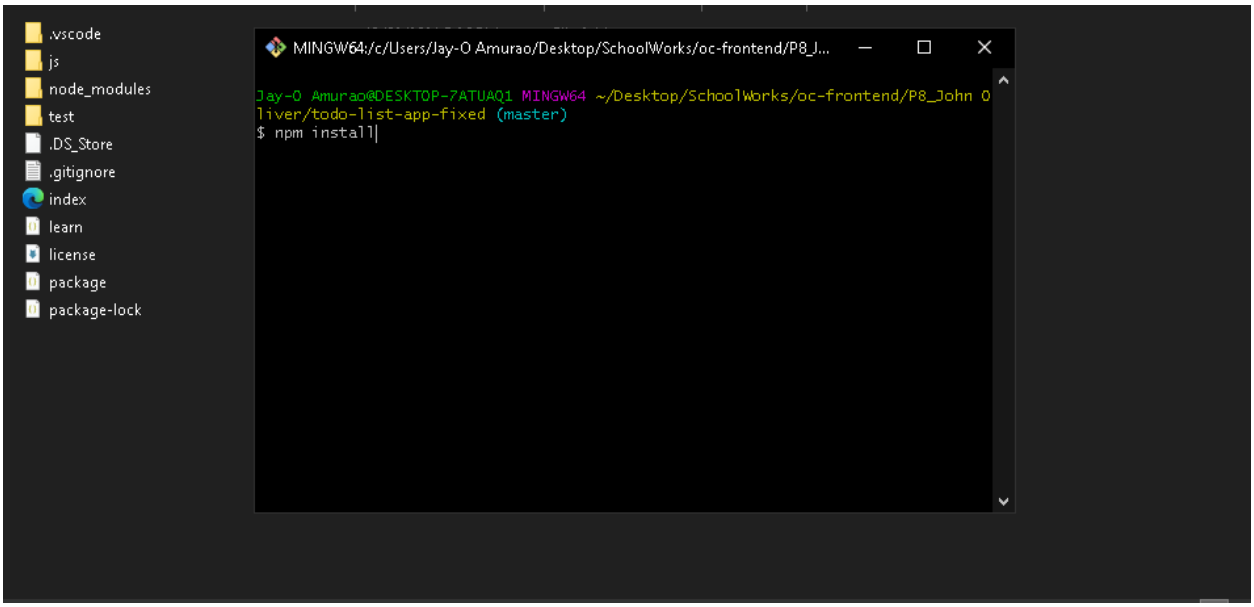
This chapter contains the installation process, Starting the application, Using the application, Software Design pattern, Bugs/Error fixing, and the Test results.

Installation

Before installing you must first have the following software installed in order to have this project on your local machine. You must have installed ***NodeJS*** (A JavaScript engine), and ***Git***, an open-source version control software.

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].

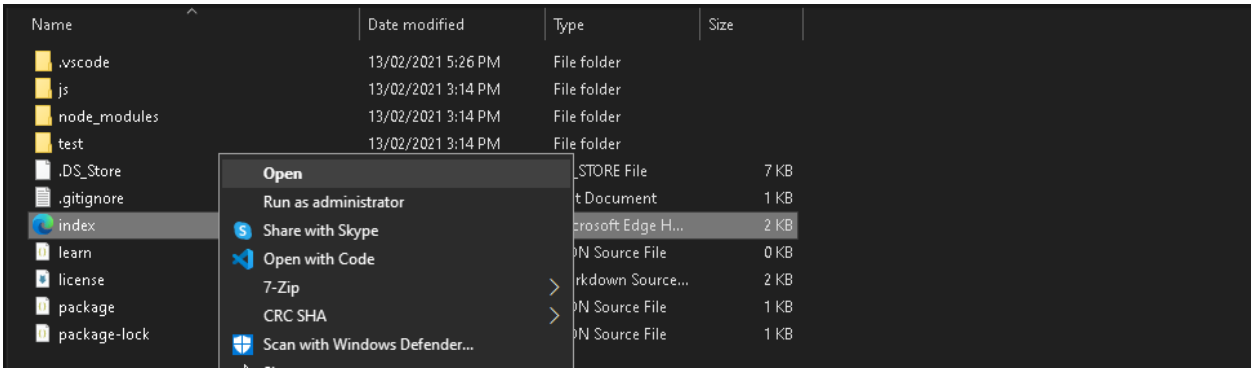
Once downloaded, open your terminal/command line in order for you to navigate in the project ***P8_John Oliver*** folder then to the ***todo-list-app-fixed*** folder. As you are in the ***todo-list-app-fixed*** folder, in your terminal, type the command ***npm install*** in order to install all dependencies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



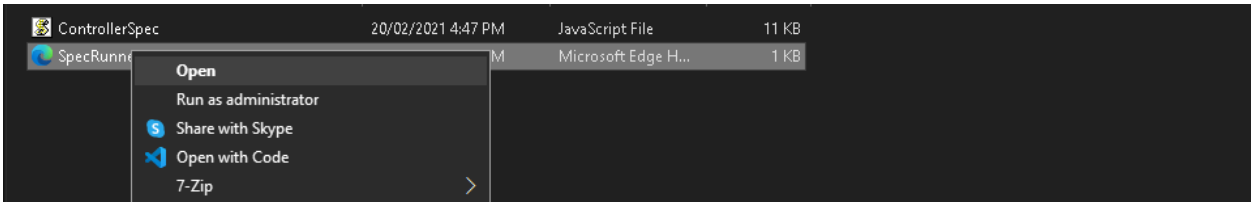
Once everything is installed, we will now proceed on how to start the application.

Starting the application

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



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 run 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 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.

Bugs/Error fixing

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

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

  Misspelled function name

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

  Changed to

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

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

  Unnecessary line of code

```
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

- **index.html** line 16

  Added id element name ***toggle-all***

```
<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 start *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

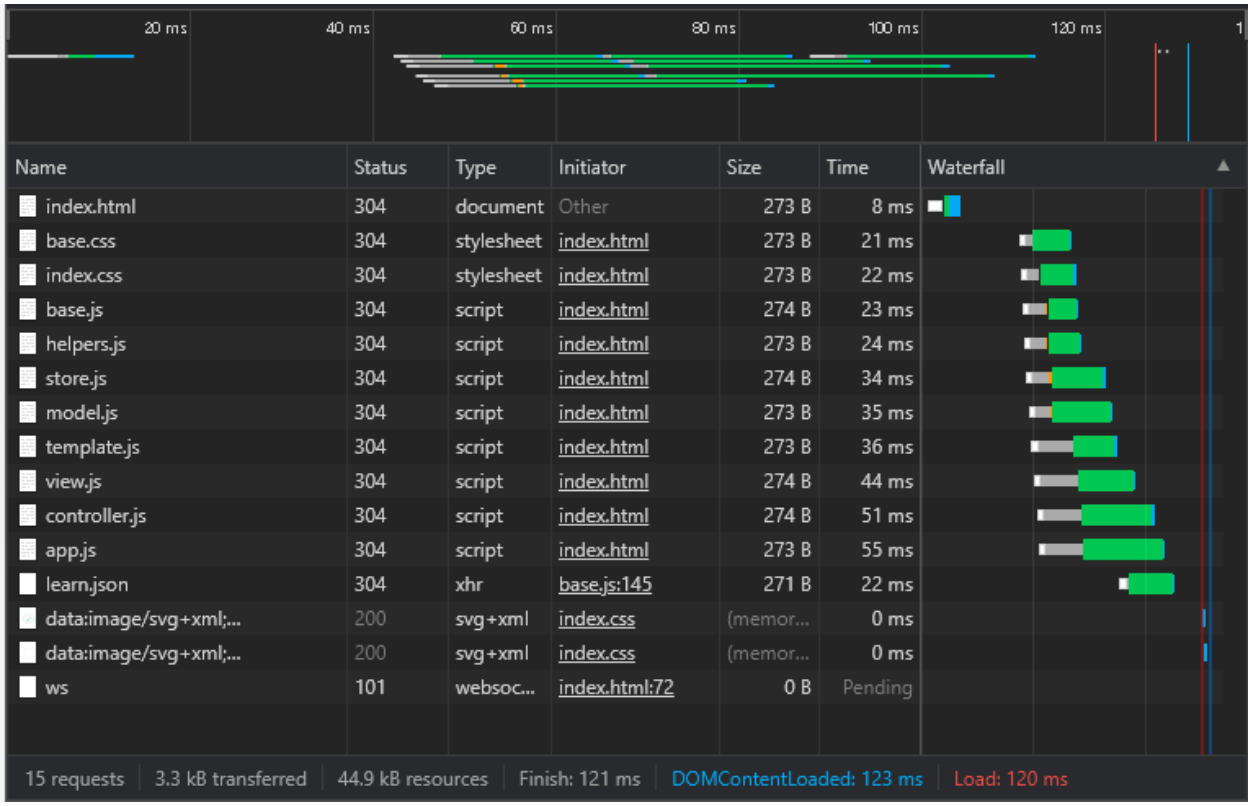
```
Jasmine 2.99.0                                                    Options
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

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
```

Performance Audit

ToDo List App (Our Application audit)

Network



Name	Status	Type	Initiator	Size	Time
index.html	304	document	Other	273 B	8 ms
base.css	304	stylesheet	index.html	273 B	21 ms
index.css	304	stylesheet	index.html	273 B	22 ms
base.js	304	script	index.html	274 B	23 ms
helpers.js	304	script	index.html	273 B	24 ms
store.js	304	script	index.html	274 B	34 ms
model.js	304	script	index.html	273 B	35 ms
template.js	304	script	index.html	273 B	36 ms
view.js	304	script	index.html	274 B	44 ms
controller.js	304	script	index.html	274 B	51 ms
app.js	304	script	index.html	273 B	55 ms
learn.json	304	xhr	base.js:145	271 B	22 ms
data:image/svg+xml;...	200	svg+xml	index.css	(memor...	0 ms
data:image/svg+xml;...	200	svg+xml	index.css	(memor...	0 ms
ws	101	websoc...	index.html:72	0 B	Pending

15 requests | 3.3 kB transferred | 44.9 kB resources | Finish: 121 ms | DOMContentLoaded: 123 ms | Load: 120 ms

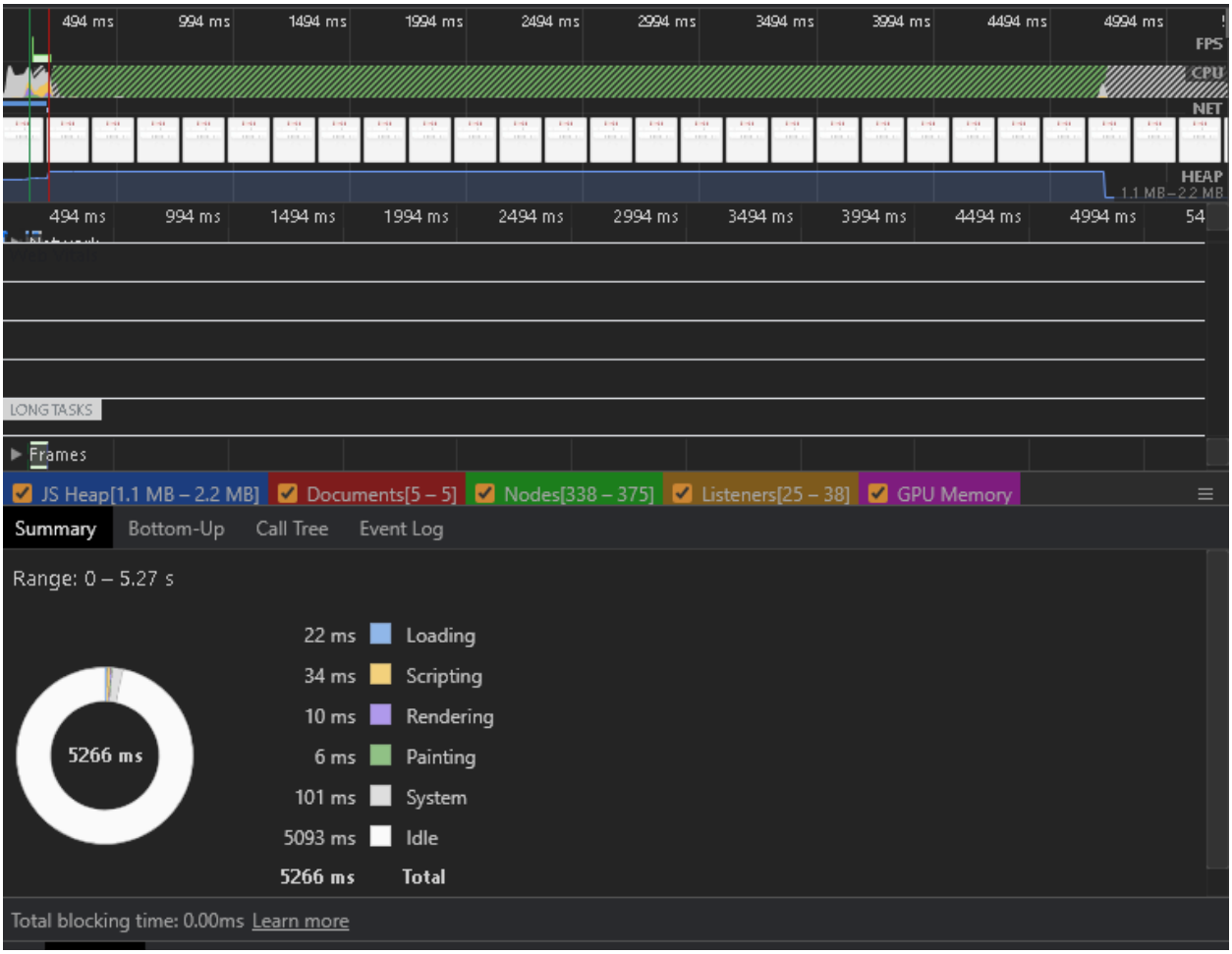
Requests sent: 15

Resources transferred: 44.9 kB

Load time: 120 ms

DOMContentLoaded: 123 ms

Performance



Loading: 22 ms

Scripting: 34 ms

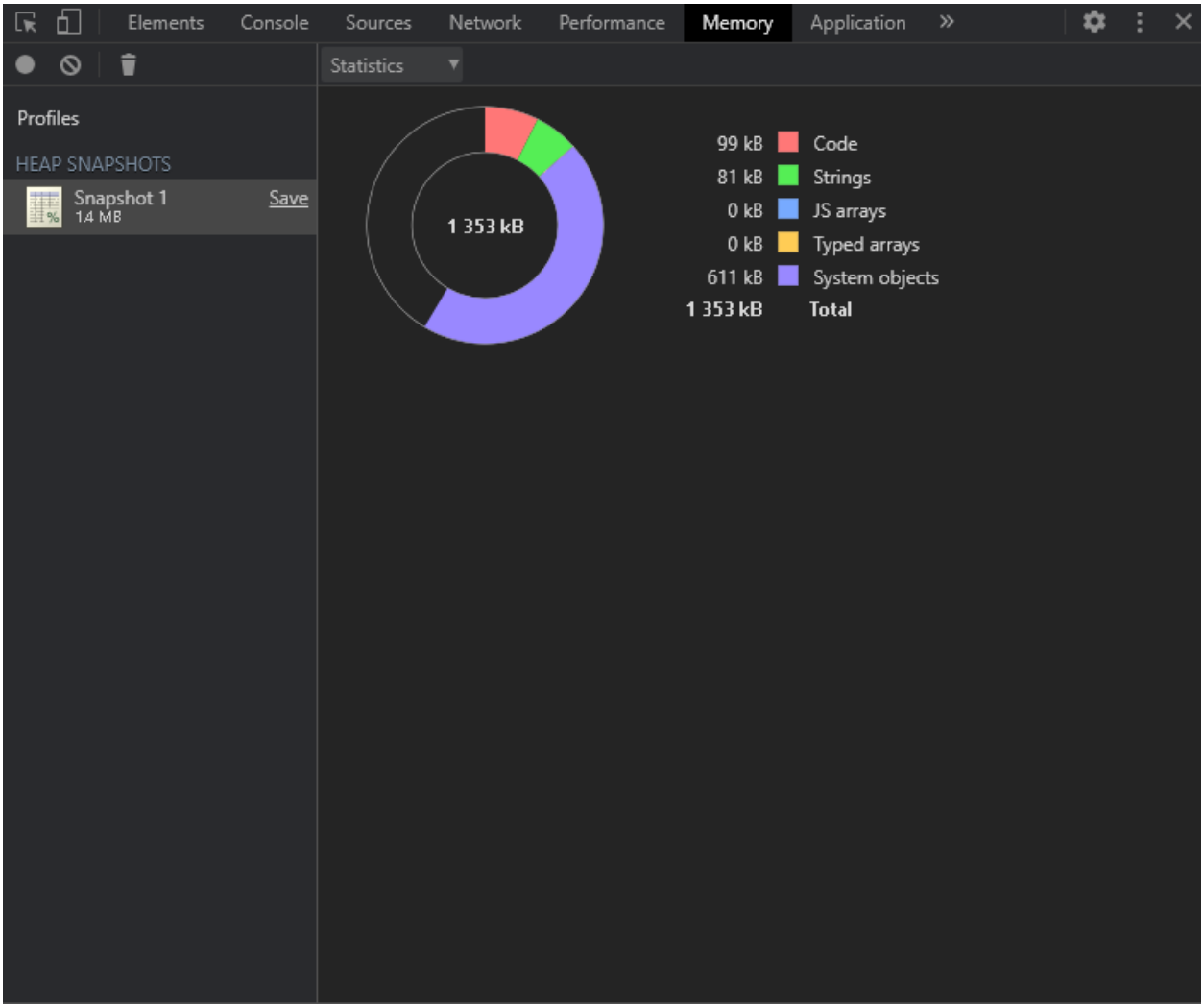
Rendering: 10 ms

Painting: 6 ms

System: 101 ms

Idle: 5093 ms

Memory



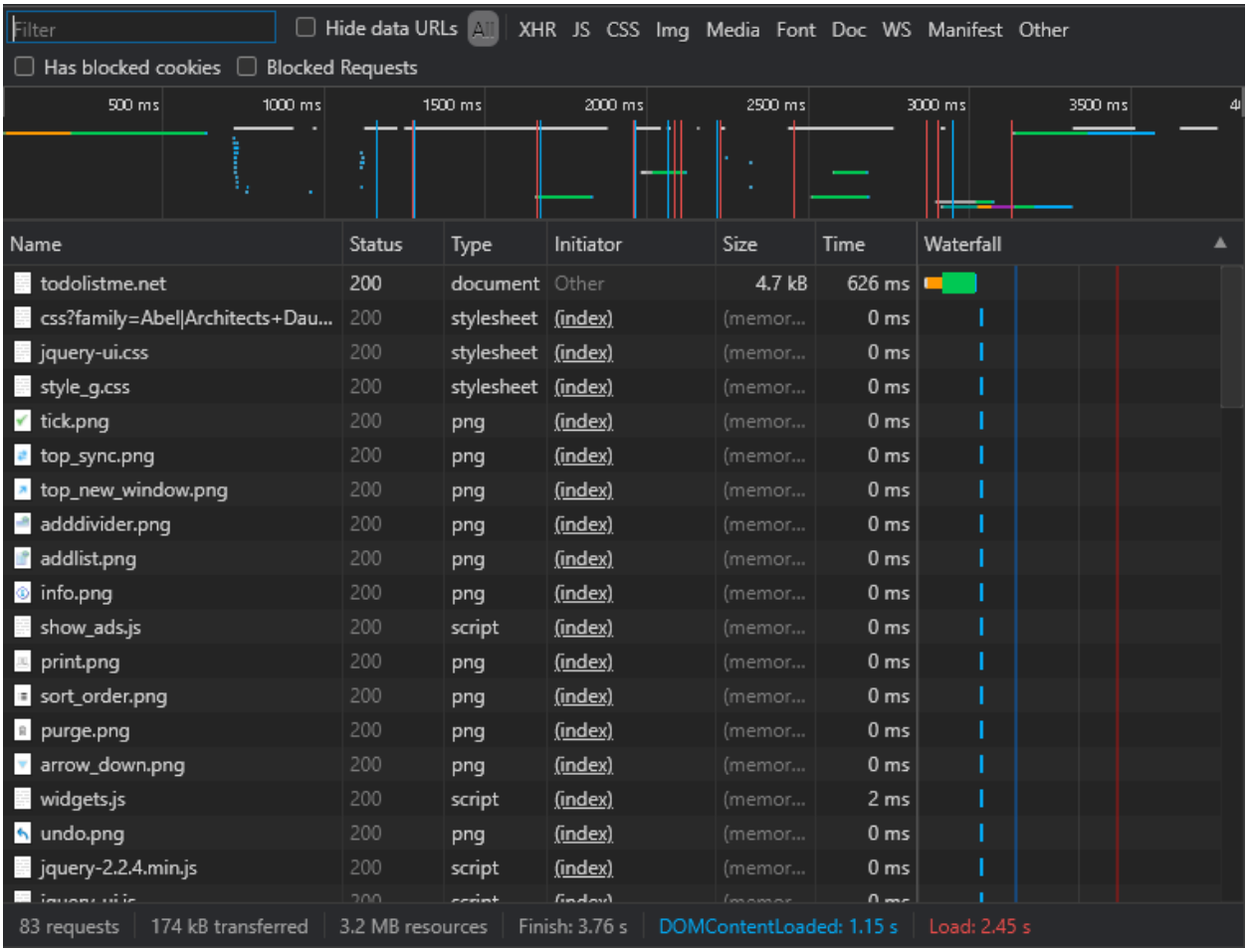
Total Memory usage: 1.3 MB

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



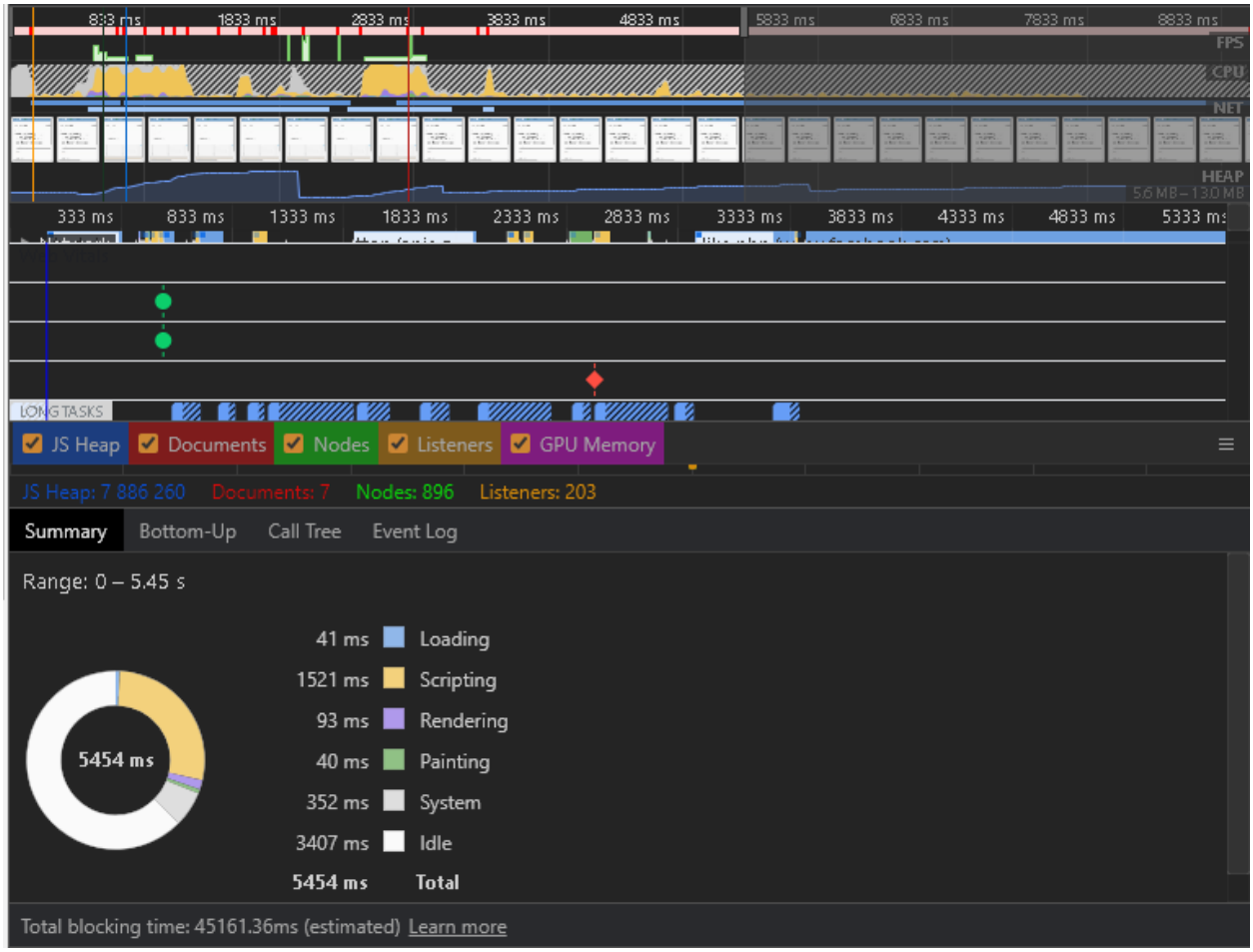
Requests sent: 83

Resources transferred: 3.2 MB

Load time: 2.45 s

DOMContentLoaded: 1.15 ms

Performance



Loading: 41 ms

Scripting: 1521 ms

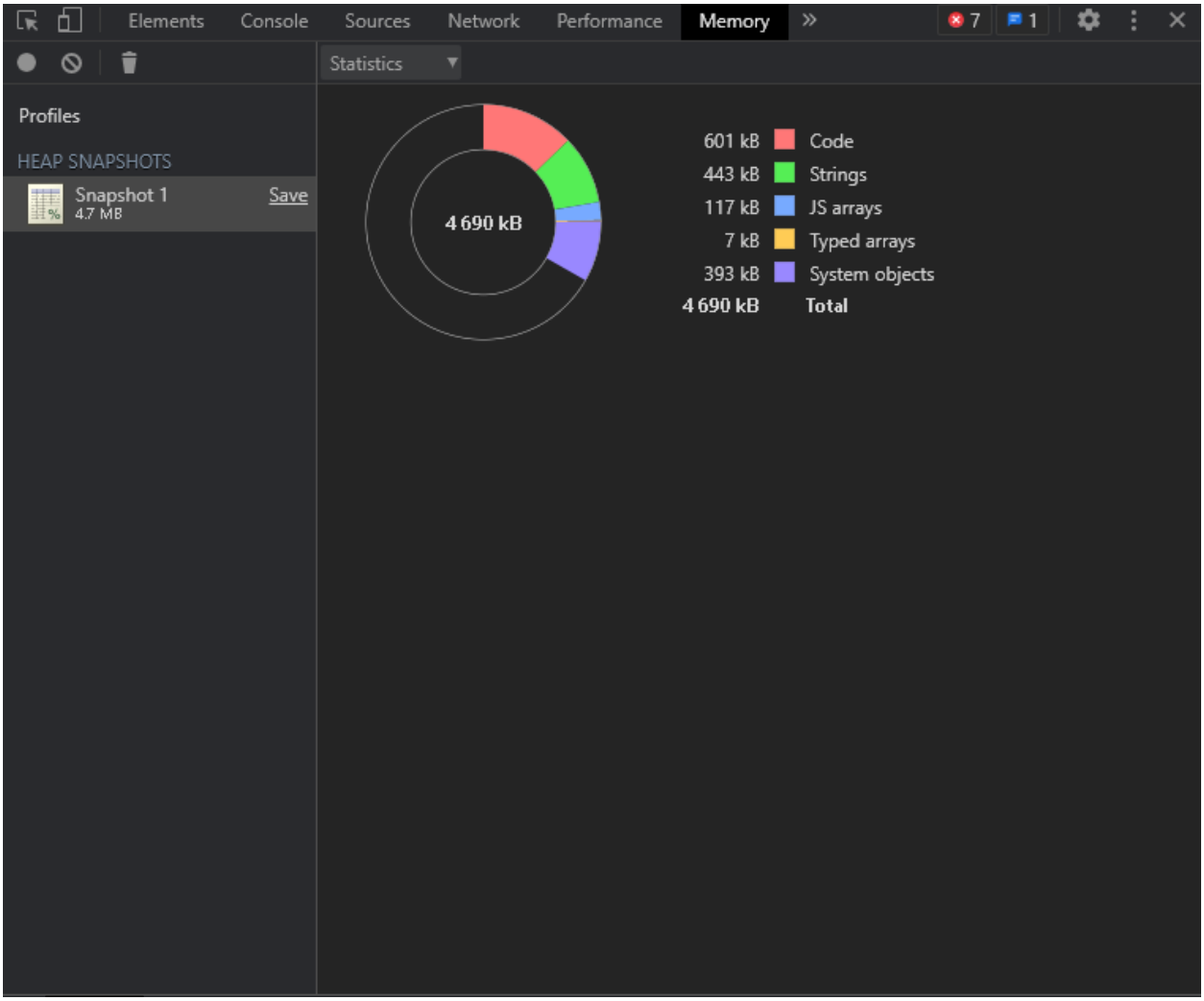
Rendering: 93 ms

Painting: 40 ms

System: 352 ms

Idle: 3407 ms

Memory

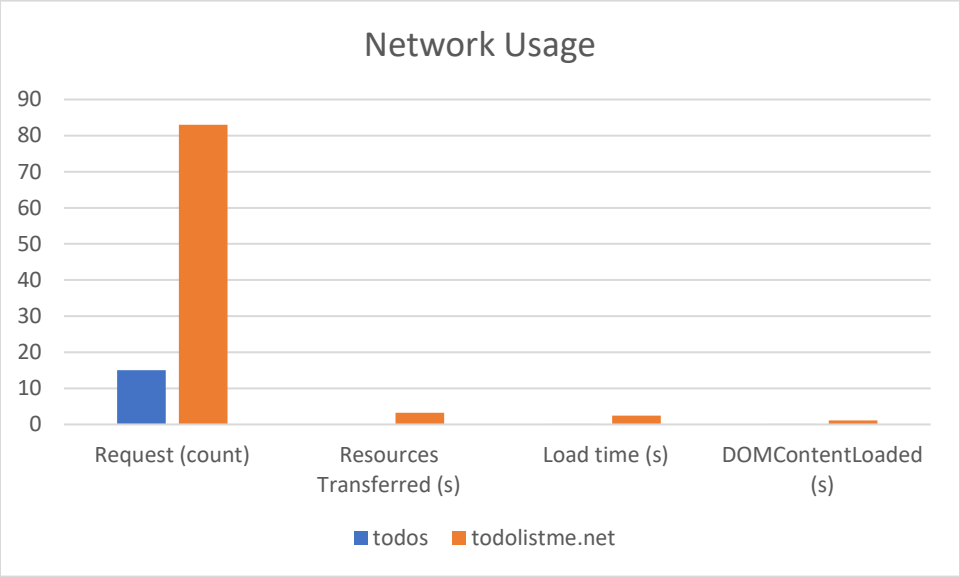


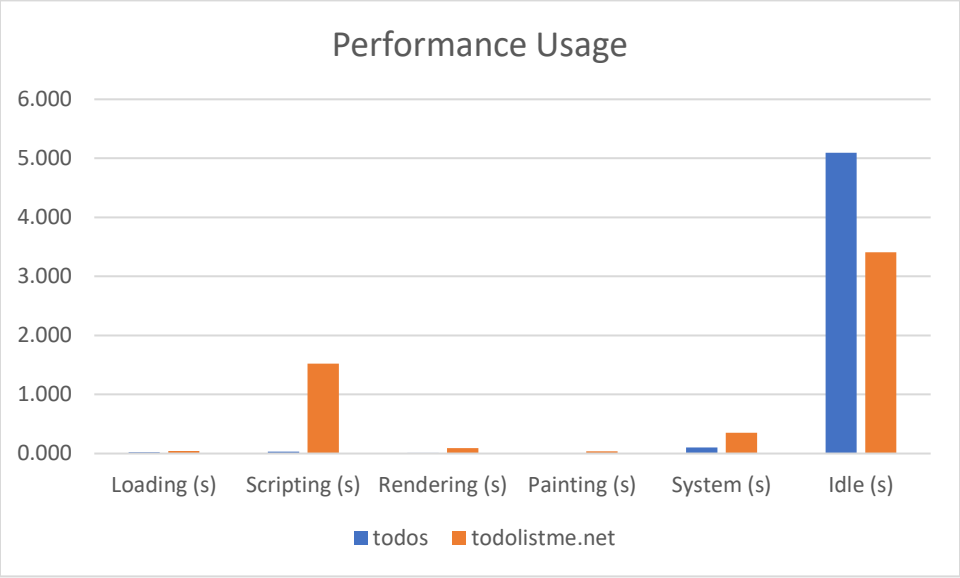
Total Memory usage: 4.69 MB

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

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