

CSWD – Report

|  |
| --- |
| Owen Cannon  Computing  S1628221  OCANNO200@CALEDONIAN.AC.UK |
|  |

Table of Contents

[Introduction 3](#_Toc500730682)

[Part 1 3](#_Toc500730683)

[Folder Structure 3](#_Toc500730684)

[Data Structure 3](#_Toc500730685)

[Metadata Structure 4](#_Toc500730686)

[Part 2 5](#_Toc500730687)

[Part 3 7](#_Toc500730688)

[Frameworks 7](#_Toc500730689)

[Angular 8](#_Toc500730690)

[Vue 8](#_Toc500730691)

[Realization 9](#_Toc500730692)

[JavaScript 9](#_Toc500730693)

[Browser Support 9](#_Toc500730694)

[JQuery 9](#_Toc500730695)

[APIs 10](#_Toc500730696)

[History API 10](#_Toc500730697)

[File API 10](#_Toc500730698)

[Plugins 10](#_Toc500730699)

[FileSaver.js 10](#_Toc500730700)

[JQueryUI 11](#_Toc500730701)

[UniteGallery 11](#_Toc500730702)

[Issues 11](#_Toc500730703)

[Clock 11](#_Toc500730704)

[File Display 11](#_Toc500730705)

[Problems We Couldn’t Fix 12](#_Toc500730706)

[Conclusion 12](#_Toc500730707)

[References 13](#_Toc500730708)

[Appendices 14](#_Toc500730709)

# Introduction

As part of the Client-Side Web Development (CSWD) module at Glasgow Caledonian University we have been asked to create a report that will support our main piece of coursework (the submitted website). This report will contain the following sections.

1. Description of folder, data and metadata structures and why we used them.
2. Description of testing performed.
3. Reflective piece looking back on what we have learned.

# Part 1

This part of the report will detail the folder, data and metadata structures used in the website and why we decided to make use of them.

## Folder Structure

The original folder structure came from the grayscale bootstrap theme we imported from start bootstrap. This theme made use of a modularly structured file hierarchy that suited our needs fine and as such we decided to keep this structure and add to it. Each key part of the website was contained in its own folder e.g. all JavaScript files were contained in a folder called “js”, our main stylesheet was contained in a folder called “css”, files integrated into the website were contained in a folder called “files”, the bootstrap theme images are contained in a folder called “img”, the SASS stylesheets are contained in a folder called “scss”, all larger third party libraries used with the exception of jQueryUI are included in the “vendor” folder, and finally all other files e.g. the text documents, PHP script and actual webpage exist in the main directory known as “Client Side Web Coursework”.

## Data Structure

Following on from this the data structure follows the same set of ideals every file has a singular purpose e.g. “loadfiles.js” loads in all the files the user should be able to see and “passwordchecker.php” checks the password. This is the ideal way to handle a data structure although it is more time consuming, we could have all the scripts contained in the “index.html” page but this would become a nightmare to maintain in the future and would promote code that looks visually unappealing. All the files used are stored locally on the user’s PC, we decided this was ideal as although there are alternatives none of them can keep the data “alive” forever. The data is then pulled out and displayed in a visually appealing way using both ajax and the jQuery library UniteGallary.

## Metadata Structure

The metadata used in our website takes many forms including: General comments, every module in our website is commented this improves readability by outside sources and maintenance. Uploaded images and videos (uploaded via the “preview” buttons) give the user the name of the file and its file size both can be used to describe the uploaded item. The image gallery gives a name and description to the image that is currently being viewed this adds information, the video gallery also has a name and author which adds information. The file table displays the names of the files contained in the userstuff files folder, this table can be searched using these names to filter the results. The final pieces of metadata contained in our website are in the quiz which contains a description and author.

# Part 2

This part will document the testing we performed on the website.

The testing we used on the website falls into “black box” testing category. We performed integration tests as we integrated modules and once the website was completed a full system test was carried out the documentation of this system test can be seen below (this test was performed in Chrome, Firefox, Edge and the newest version of IE.

|  |  |  |  |
| --- | --- | --- | --- |
| App Currently being used: | Expected Result | Actual Result | General Comments |
| Do the navigation elements direct to the correct places | Yes | Yes |  |
| Does clicking the down arrow on the main screen navigate to the about section | Yes | Yes |  |
| Does the clock on the navbar update correctly | Yes | Yes |  |
| Do the hrefs that navigate away from the website function correctly | Yes | Yes |  |
| Clicking admin login displays the login modal | Yes | Yes |  |
| I cannot submit the login form unless a username and password of 5 characters in length has been entered | Yes | Yes |  |
| Inputting the wrong username and password displays this to the user and DOES NOT give them access to admin features | Yes | Yes |  |
| Inputting the correct username and password displays this to the user and DOES give them access to admin features | Yes | Yes |  |
| Clicking the add items button displays the add items modal | Yes | Yes |  |
| Admin can upload text and html documents here and edit/save them, as well as type and save new documents? | Yes | Yes |  |
| Clicking the preview image button displays the image previewer modal | Yes | Yes |  |
| Uploading an image here displays it to the user | Yes | Yes |  |
| Clicking the preview video box displays the video previewer modal | Yes | Yes |  |
| Uploading a video here displays it to the user | Yes | Yes |  |
| Clicking to play this video plays it | Yes | Yes |  |
| Clicking the todo list button displays the todo list | Yes | Yes |  |
| Adding an item to the todo list adds it to the list | Yes | No | Not sure why this doesn’t work |
| Editting a todo list item edits it | Yes | Yes |  |
| Deleting a todo list item deletes it from the list | Yes | Yes |  |
| Marking an item as complete adds it to the complete section | Yes | Yes |  |
| Marking an item as incomplete adds it to the incomplete section | Yes | Yes |  |
| Clicking view items displays the items | Yes | Yes |  |
| The image and video gallery navigation and usability options function | Yes | Yes |  |
| Searching for a file in the file table filters the table to only show relevant results | Yes | Yes |  |
| Clicking a pdf file opens it in the pdf viewer iframe | Yes | Yes |  |
| Clicking any other file will download it | Yes | Yes |  |
| Clicking the start quiz button displays the quiz | Yes | Yes |  |
| The show and hide buttons in the view items section show and hide the appropriate content | Yes | Yes |  |
| History API updates the page url so the user can’t back out of the SPA unless it is on their first visit to the page | Yes | Yes |  |
| Insert images into the user stuff folder so they are displayed in the gallery | Yes | Partly | The first image put in the folder will be displayed however only the first will be displayed, the folder also needs at least one image, or the rest of the data will not be displayed |

# Part 3

This part will serve as the reflective piece of this report. In this section we will document what was learned in the process of making this website.

## Frameworks

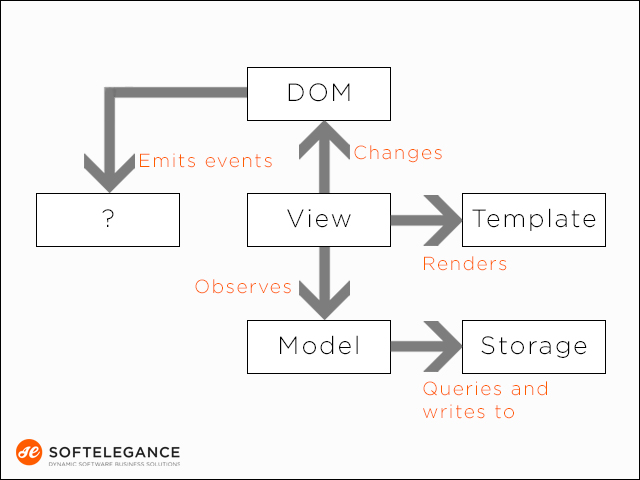
When first starting the coursework, I decided to consider JavaScript frameworks, these are "an application framework written in JavaScript”(En.wikipedia.org, 2017) they are different from JavaScript libraries such as jQuery because frameworks define the entire application design. Frameworks defining application design goes as far as changing how users interact with code, instead of user code calling a framework the framework calls and uses the user code. According to 1stWebDesigner blog (2017) the overall benefits of frameworks that make them appeal are as follows:

* Advantages
  + Efficiency
    - Frameworks make it possible to complete tasks that would normally require hundreds of lines of code in a few minutes with pre-built functions.
  + Cost
    - Most frameworks are free.
  + Security
    - Most large frameworks come with prepacked security features that will aid your website
  + Support
    - Every framework comes packed with supporting documentation to get developers started.
* Disadvantages
  + Developers learn the framework not the language which is argued to make them dependant of the framework to perform any work.

The main appeal of frameworks in the context of this piece of coursework is that they are all advertised as fast reliable solutions to create responsive Single Page Applications (SPAs). After learning this I started considering frameworks including Angular and Vue which we will document a few words about below.

### Angular

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly (Docs.angularjs.org, 2017). Some people even go as far as to say HTML would be more angular like if it was designed for applications. Below is a diagram showing the architecture of Angular (diagram 1)



*Diagram 1: AngularJS ‘Architecture’ (Novosiolov, 2017)*

The main advantage Angular would have in our coursework is due to how it handles dependencies. Angular handles dependencies in a way that allows SPAs dynamic loading to a present a very “native application” feel (Wintellect, 2017).

### Vue

While researcher angular I stumbled upon a newer framework known as vue.js, it claims to contain all the best parts of other big frameworks with its website stating the following, “Vue is a **progressive framework** for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable” (Vuejs.org, 2017). Vue has been renowned in recent years as being the best framework for creating single page applications therefore it interested me.

### Realization

I was ready to start creating my website using vue.js when I reread the coursework to notice it stated no frameworks. Although this was saddening the knowledge I gained researching them still aids in my client-side web development skills and I will consider using frameworks in the future.

## JavaScript

The website we are handing in was originally created using basic JavaScript however due to problems listed below we changed to jQuery, because of this change the only module that still uses basic JavaScript is “todo.js” which is the to-do lists functionality.

### Browser Support

While testing the basic JavaScript website it functioned perfectly in chrome however, when taken into other browsers we started to notice problems. In Firefox the clock we have displayed on the navigation was almost always displaying the incorrect time for some reason we could not work out, this was corrected when we changed to a jQuery implementation (more is written about the clock below). In Edge and IE we experienced issues with our AJAX calls, this was due to the fact we forgot Microsoft browsers handled these differently, Chrome and Firefox both use the following (var xhr = new XMLHttpRequest( );) whereas Microsoft browsers make use of (var xhr = new ActiveXObject("Microsoft.XMLHTTP");) we could have added support for both of them into our application by using the following (var xhr = ( typeof window.XMLHttpRequest == "function" ) ? new XMLHttpRequest( ) : new ActiveXObject("Microsoft.XMLDOM");) however we instead decided to remake the website using jQuery as it normalizes AJAX calls across browsers. Once we switch to the current jQuery solution the website functioned the same across multiple browsers, while performing these tests we also noticed a lot of small UI features that appeared differently in browsers however there was no glaring issues that made the website unusable.

## JQuery

After the basic JavaScript website, we built had major functionality problems we decided to switch to jQuery, jQuery is the most used JavaScript library. The main advantages of jQuery are the following.

* Advantages
  + JQuery is lightweight compared to frameworks.
  + JQuery has many plugins available.
  + JQuery tends to be easier to learn.
  + JQuery normalizes AJAX calls cross browser.

The main reasons we decided to redo the website with jQuery is that it normalizes AJAX calls across browsers and we personally feel jQuery code is more readable and maintainable.

## APIs

We made use of a few Application Programming Interfaces (APIs) during the creation of our website we will document what these were and how well we implemented them below.

### History API

This API gives developers the ability to modify a website’s URL without a full-page refresh (CSS-Tricks, 2017). This was extremely useful in our SPA as without it if the user was to press the browsers back button they would be ejected from the site (because everything is displayed on the same page) however with it they are redirected back to the last URL that the history API updated from, this is useful as it gives the feel of navigation where there is none. It would have been ideal if we could save state and then restore state upon pops however due to coding inabilities we could not accomplish this, as a result we instead decided to implement show and hide buttons to filter content as needed (it should be noted the history API works fine with the modals it just does not make any content pulled in by AJAX calls disappears).

### File API

Our website makes heavy use of the File API, the file API makes it possible to display, handle, save and download files in the browser and is supported by every current large browser. We made use of the File API for our preview image and preview video module here the API allows the user to upload an image or video, this will then be displayed to them and the name / file size will be stated. Due to how we have decided to store data the file API was unable to save files for us as such we have made use of a jQuery plugin stated below.

## Plugins

The website we created makes use of two large jQuery plugins these are detailed below.

### FileSaver.js

We originally attempt to use LocalStorage for storing our data however due to how our browser handles data our files never stayed for a long time to be useful. After our failure to implement LocalStorage we looked at indexedDB but failed to get it to save data. After this we looked at LocalForage which is very similar to IndexedDB however this also wiped data before we had a chance to use it in any meaningful way. After these failed we started storing our files on the user’s PC and using AJAX to pull them into the site however, in doing it this way we couldn’t save new files to the user’s PC from the website due to security concerns of the file API. This is where FileSaver.js is used in our website it enables the “saveAs()” functionality of the file API in browsers that don’t support it (all modern browsers do not support this functionality due to the security concerns of viruses being plugged into the user’s PC, however due to the fact that the website is a client-side application this concern is of no issue to us). With this plugin we can allow our user to upload, edit, create and save files via our website the user can then have these files displayed by saving the in the appropriate folder (userstuff/files or userstuff/images).

### JQueryUI

Our website makes use of the jQueryUI plugin this plugin is described as a curated set of user interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript Library (jquery.org, 2017). We made use of this plugin for our dialogue popup messages that give the user more information about the website. Before considering the best way to deliver dialogue popups I had never heard of this, so it was an interesting plugin to research.

### UniteGallery

UniteGallery is a plugin that lets users display their images and videos in responsive galleries, these galleries have appropriate media controls to interact with them. We made use of this plugin to display our images and videos in an appealing way.

## Issues

While developing this website we encounter a few problems that we will document below.

### Clock

When we first created the clock, it was not displaying zeroes as such the clock did not function the way we wanted it to, the solution to this was to add the following line of code to script (if (i < 10) {i = "0" + i};) this adds a zero to numbers less than 10 making our clock a true digital clock.

### File Display

While creating the AJAX code that pulls our files and displays them in a table we originally had it pulling all files in the folder. It was observed while doing this that hidden files where being pulled out of the folder, these files where of no use to our users so we added the following filter to the code (if (val.match(/\.(pdf|doc|docx|txt|html|js|css|rar|7zip|xml|json|tut|zip)$/))) this will only display results that have one of the following file extensions meaning the users will only see files we intend them to see.

### Problems We Couldn’t Fix

Due to the limitations of the UniteGallery plugin we could only pull in one of the users uploaded files, all other files are disregarded by the website. Another problem we couldn’t fix is that while attempting to add items to our to do list they do not display.

# Conclusion

In conclusion during the duration of this coursework we learned a lot about web development and have a greater understanding into the inner workings of single page applications.

# References

En.wikipedia.org. (2017). *JavaScript framework*. [online] Available at: https://en.wikipedia.org/wiki/JavaScript\_framework [Accessed 10 Dec. 2017].

1stWebDesigner. (2017). *Web Frameworks: Pros And Cons Of Using Frameworks*. [online] Available at: https://1stwebdesigner.com/web-frameworks/ [Accessed 10 Dec. 2017].

Docs.angularjs.org. (2017). *AngularJS*. [online] Available at: https://docs.angularjs.org/guide/introduction [Accessed 10 Dec. 2017].

Novosiolov, A. (2017). *AngularJS advantages and limitations | SoftElegance's Blog*. [online] Blog.softelegance.com. Available at: http://blog.softelegance.com/angularjs/angularjs-advantages-and-limitations/ [Accessed 10 Dec. 2017].

Wintellect. (2017). *10 Reasons Web Developers Should Learn AngularJS - Wintellect*. [online] Available at: https://www.wintellect.com/10-reasons-web-developers-should-learn-angularjs/ [Accessed 10 Dec. 2017].

Vuejs.org. (2017). *Introduction — Vue.js*. [online] Available at: https://vuejs.org/v2/guide/ [Accessed 10 Dec. 2017].

CSS-Tricks. (2017). *Using the HTML5 History API | CSS-Tricks*. [online] Available at: https://css-tricks.com/using-the-html5-history-api/ [Accessed 11 Dec. 2017].

jquery.org, j. (2017). *jQuery UI*. [online] Jqueryui.com. Available at: https://jqueryui.com/ [Accessed 11 Dec. 2017].