Introduction

Hello. I’m Alex. This is me learning what Design Patterns and Style Guides are, and my journey to making my own. The journey began when the project was assigned as coursework as part of my ‘Web Design for Industry’ course at University, and ends with me publishing this article.

Research

For research I looked at a host of Style Guides and Pattern Libraries hosted on the website <styleguides.io>, from which I drew inspiration about what typically gets covered in different guides, how to layout the library and what other components I could include. I created a list in my readme file of what sites I found particularly helpful and why, and tried to incorporate any lessons I learned in my own work.

* <http://voiceandtone.com/> - Inspiration for how the navigation bar is along the side of the screen which means the user doesn’t have to scroll back to the top of the page to navigate elsewhere in the site
* <https://google.github.io/styleguide/cppguide.html> - Super in-depth no confusion as to what developers should do to comply with the google standards, including guidelines as to how code should be written
* <http://ux.mailchimp.com/patterns/> - Useful reference point for use of the grid system, and used to determine what sort of buttons could be implemented.

I also researched the [Atomic Web Design model](http://bradfrost.com/blog/post/atomic-web-design/), developed by Brad Frost. While I found the approach to web development interesting, I am only approaching competence using BEM, and so I didn’t want to try any new development techniques while still learning the basics of web development.

Design Process

I designed the library using the [Sprint model](http://www.gv.com/sprint/), a method whereby I would identify a need/ requirement, design a solution, implement it and evaluate the outcome to determine any area of improvement. This model ties closely to my testing plan, where I implement components step by step and constantly that they work, and fix them when they don’t.

Test Plan/ Process

Preliminary planning identified the following areas of testing:

* Browser support and Page sizing
* Functionality
* User feedback

Browser support is the performance of the website on web browsers such as Google Chrome, Mozilla Firefox, Microsoft Edge and Internet Explorer, and can be done after every change or when I think I’m done with a component. Paging sizing, or scalability is tested as the page is developed, as part of the Sprint model. I will be constantly adjusting the sizes of components to better fit different screen sizes. As already mentioned, functionality is constantly tested as I build the website, and any failures or drawbacks will be mentioned during the documentation part of this blog. User feedback will be gathered at the end of the project, and will heavily focus on how easy the user found the site to use, and their opinion on colour schemes and other cosmetic factors.

Specifications

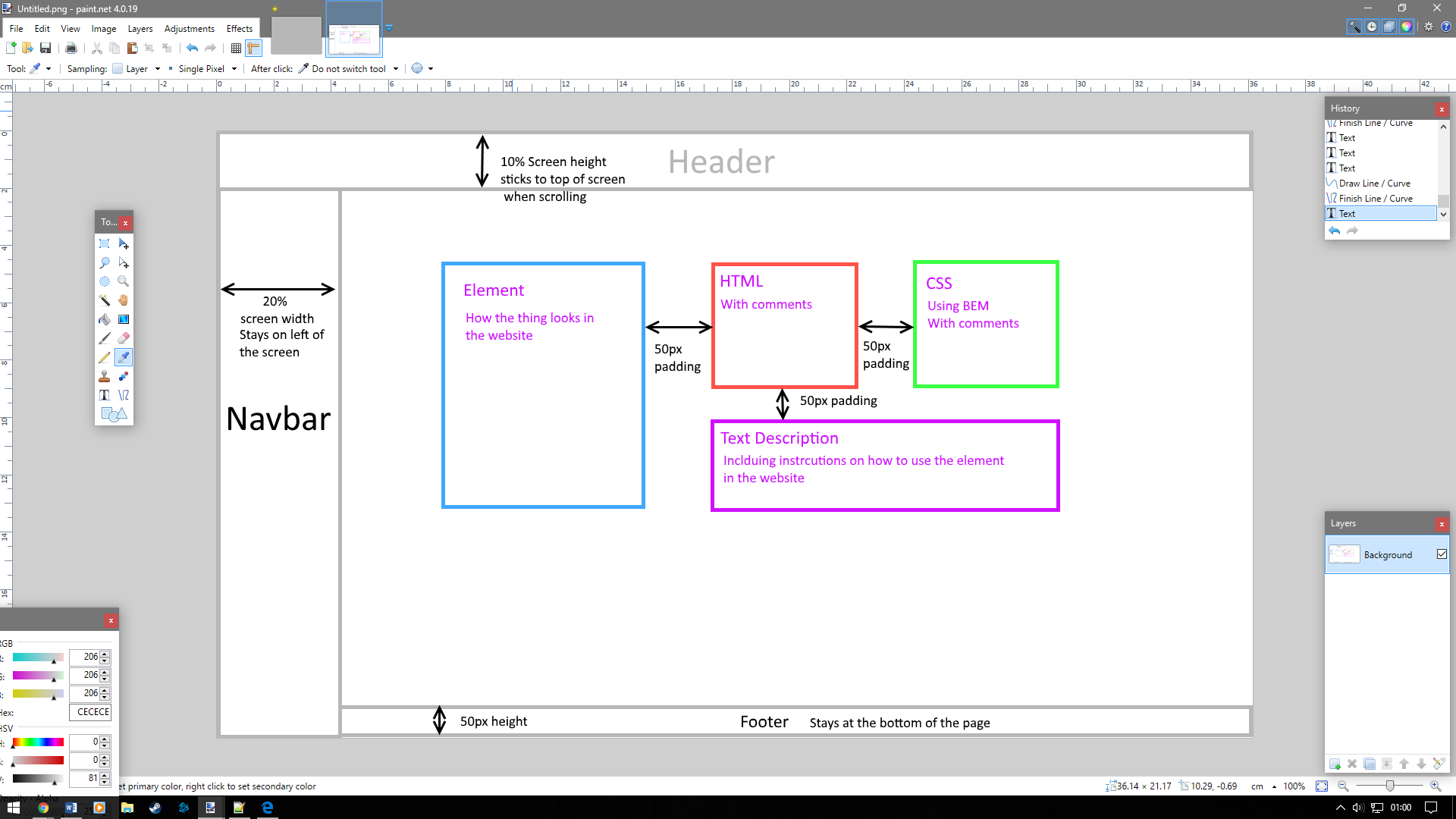
The project brief specified the following elements as essential, with a working example of how the element works in practice, with documentation and code snippets:

1. Navigation Bar
2. Header Element (carousel or just static)
3. A range of six button types
4. Jumbotron or Call to Action
5. Footer possibly incorporating social media icon links etc.
6. Component of your choice

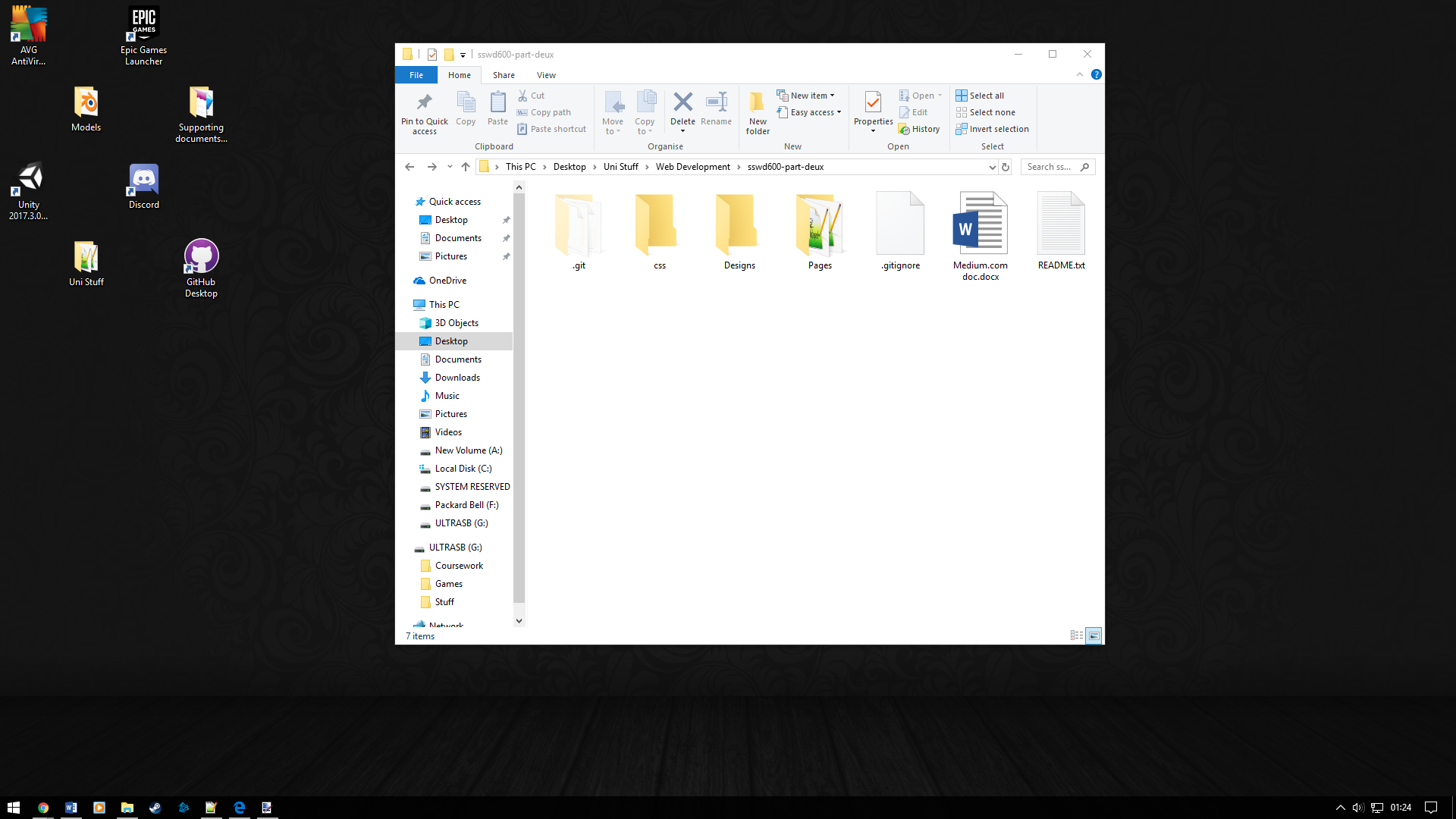
Additionally the brief called for a 2,000 word blog post on [Medium.com](https://medium.com/), which you are currently reading.

Pre-Processing and Planning

The first stage in my design process was doing a quick mock-up of what I wanted the site to look like, using paint.NET, focussing on the layout of the website itself more than colours and fonts. It was important for me to know how I would be laying the site out before I started, to stopping myself wasting development time tweaking the order of different components on the site. At this stage I also made rough drafts of the six buttons I needed to make, as well as designing the Call to Action.

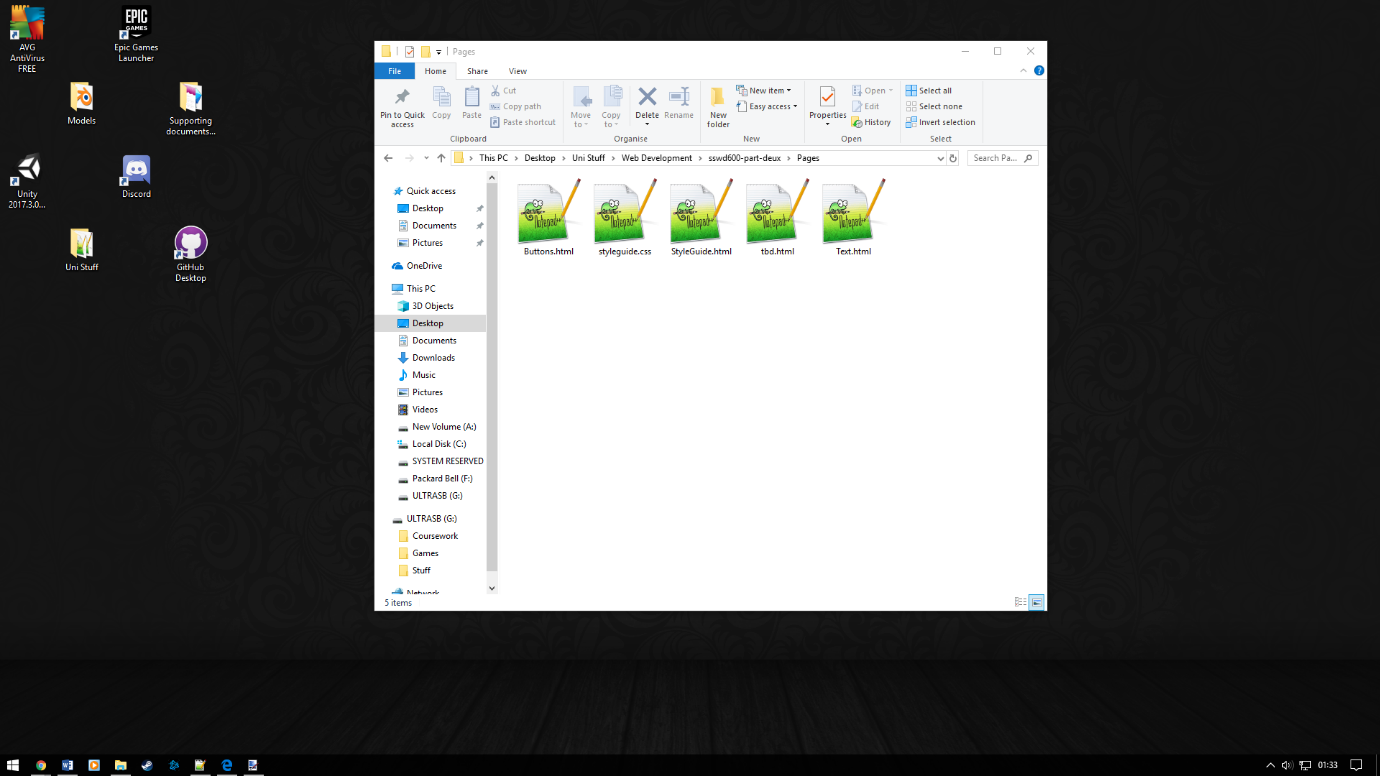


Documentation

The first step in creating my design library was setting up the project folder. With this done, I linked the folder to my GitHUb

Hub account using the GitHub desktop application.

The next step, as documented in Pre-Processing stage was to create the actual pages for the site. Based on the needs specified in the brief, I figured I only needed four pages: a general Style Guide page to contain the Navigation Bar, Header, Footer and guidelines to follow when writing code for this site; a page for Buttons which would also contain the Call-to-Action; a page to describe various font sizes and colours called Text; and a page for the component of my choice, which was given the placeholder name “tbd”. Lastly I created the css file for the entire website.



I started coding with the Navigation Bar. I did this using an unordered list to be displayed along the side of the site, to be the full height of whatever screen it was on at 200px wide. Here the first issue occurred: the menu had some small padding along the top and bottom despite the css file not having any. I fixed this by setting the top and left margins to -10px, which aligned the edge of the bar with the edge of the screen.

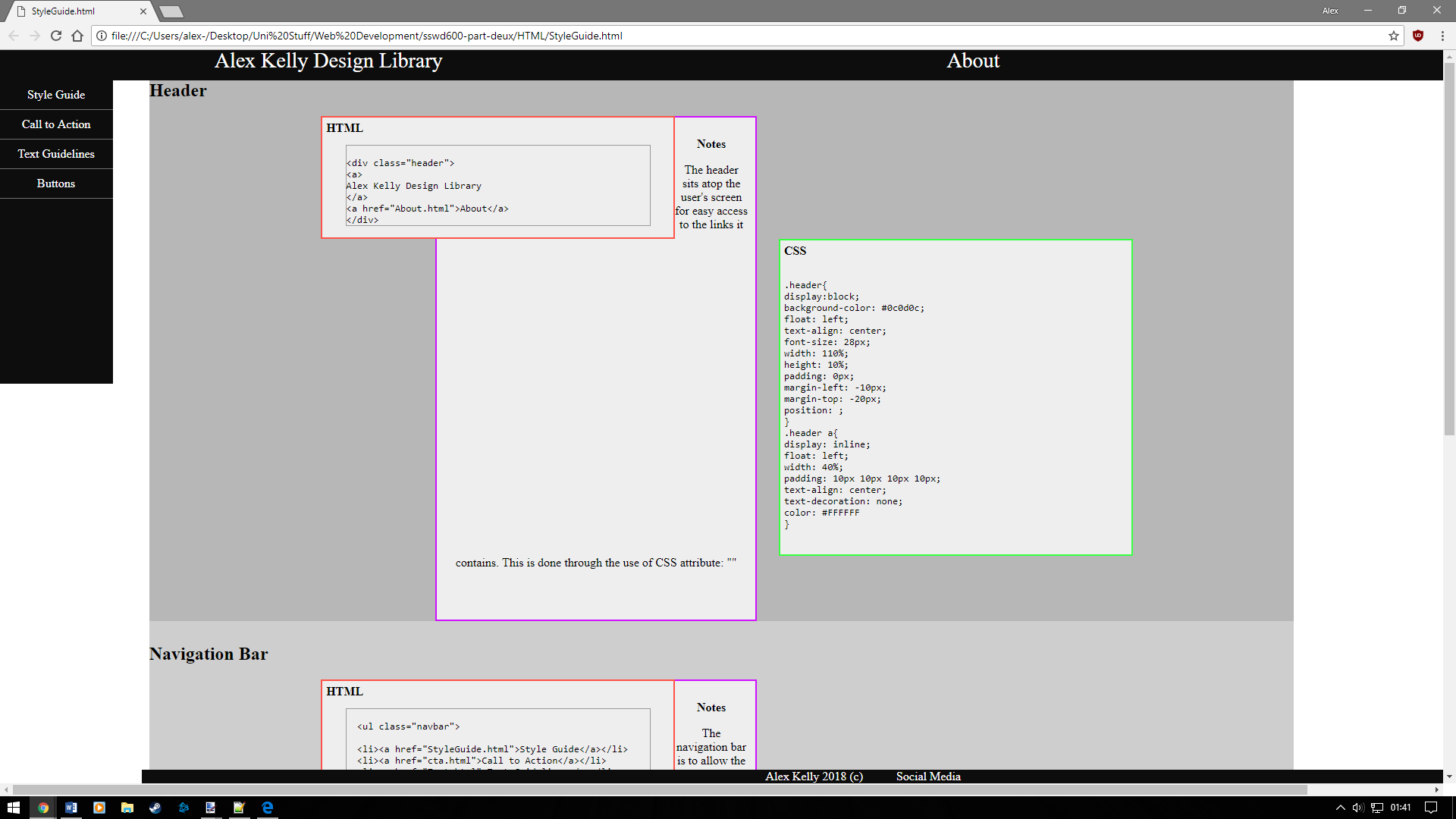
I used a similar technique to create the Header and Footer. The Header I set to stick to the top of the screen in the same way as the navigation bar does, because I personally prefer websites that do that, and because I wanted to have a link to an about page for the website, which was based off of the readme made for the GitHub repository. The Footer I added a watermark and an area to place social media links later on, focusing instead on adding the code snippets for the three elements I had created.



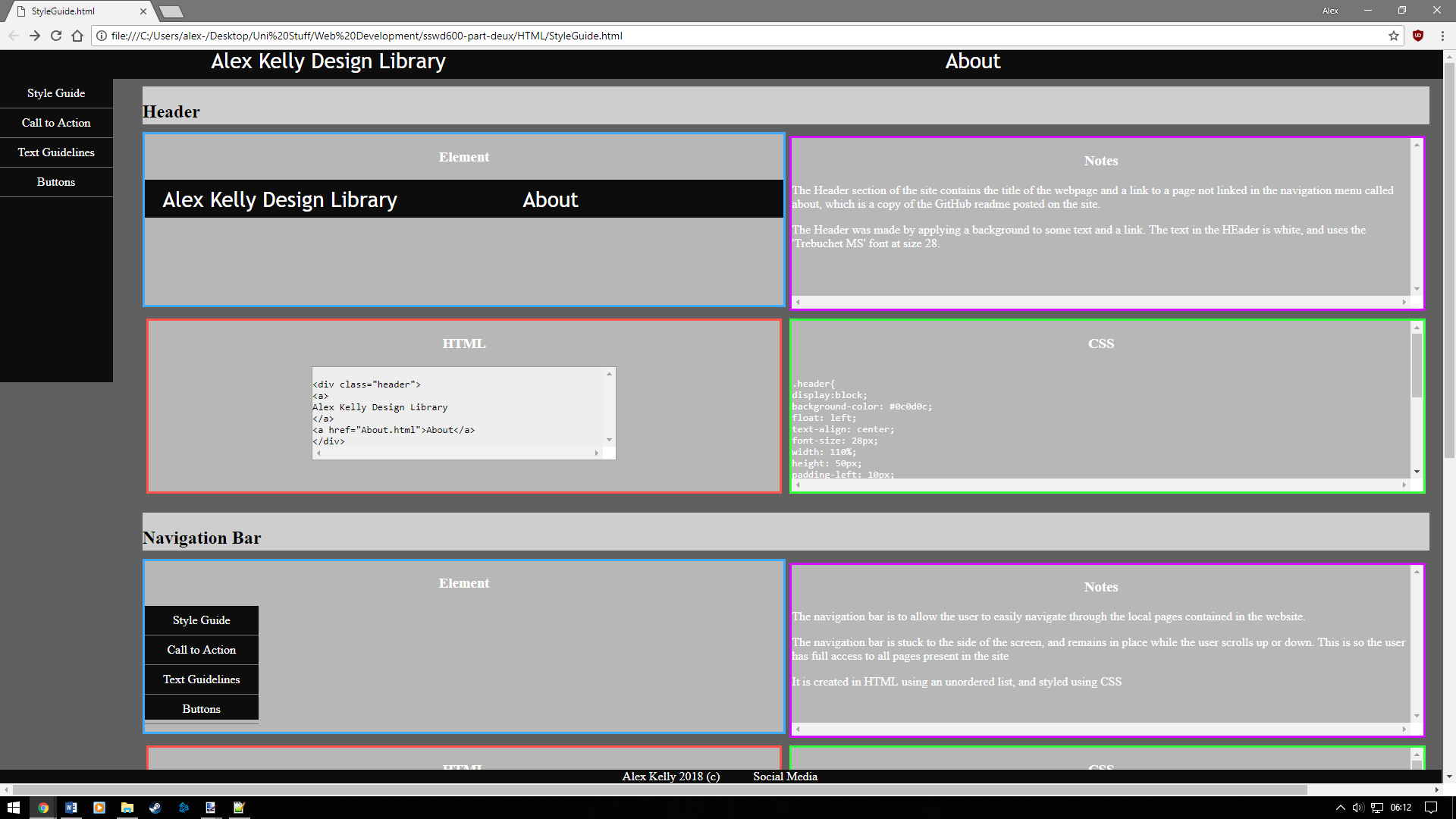
I created a set of five containers:

* A pair of containers to hold elements and their descriptors. These containers were identical aside from their name and background colour.
* A container to hold the HTML snippet with a h2 header
* A container for CSS with a h2 header and different border
* A container to explain how the element was made, and how it should be used in the site.

Unfortunately in their current state, where each container was set to ‘float: left;’, the latter three containers were unable to be aligned together according to the design, as the ‘notes’ container would fix its top border to the top of the ‘HTML’ container. Googling the problem led to two solutions: fiddling with margins to force the containers to align properly or restructuring the site into a CSS.grid format. I opted for the latter option, because it would be better practice to rebuild the containers in a more elegant style than forcing alignments through some manipulating margins.



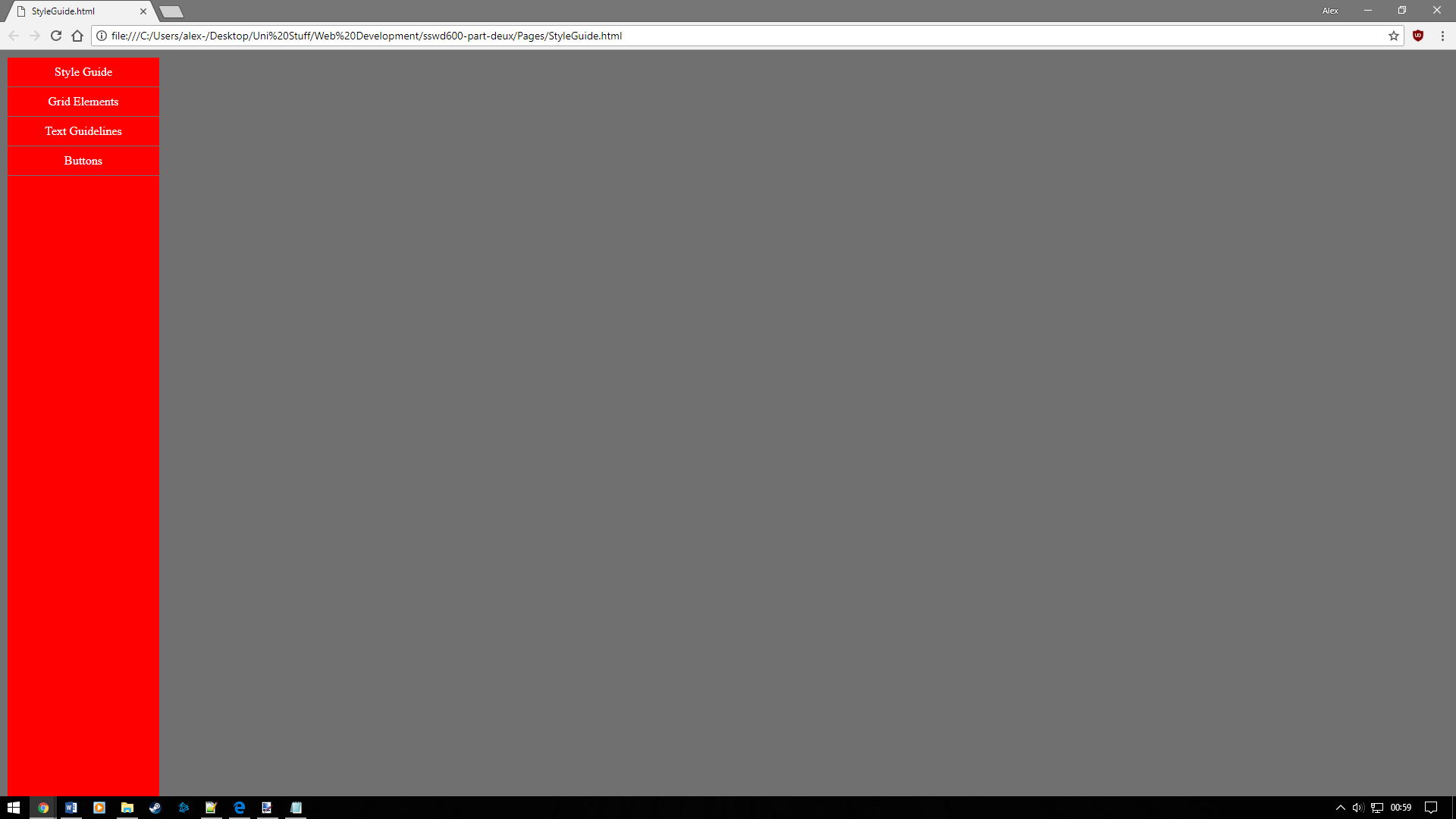
I opted to use a two-by-three grid structure to display elements, with the top row representing the name of whatever element I was showcasing, and the other four coordinates to display the name of each thing and the content, be that a representation of an element, its HTML or CSS code or the base description of that element.



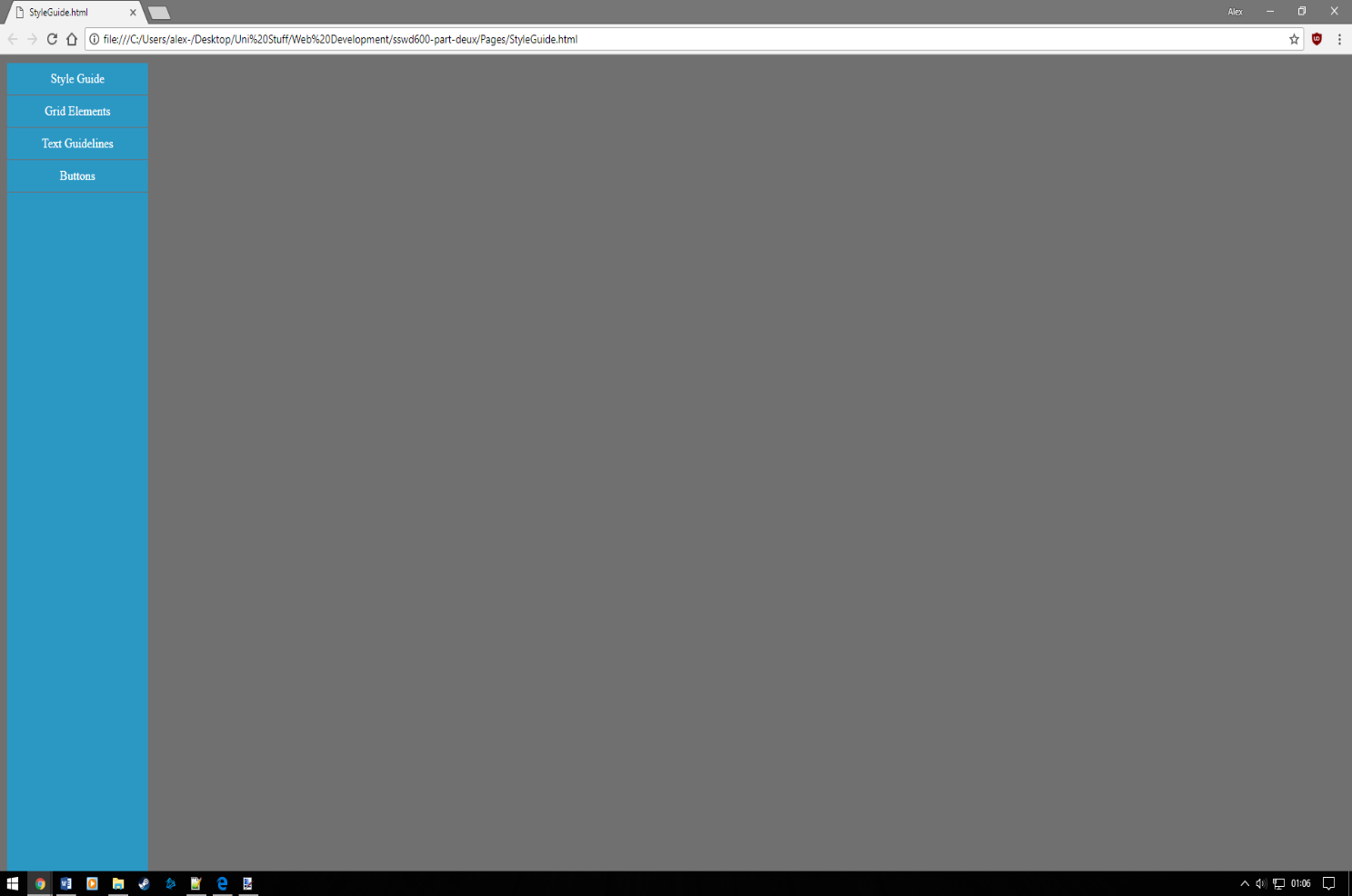
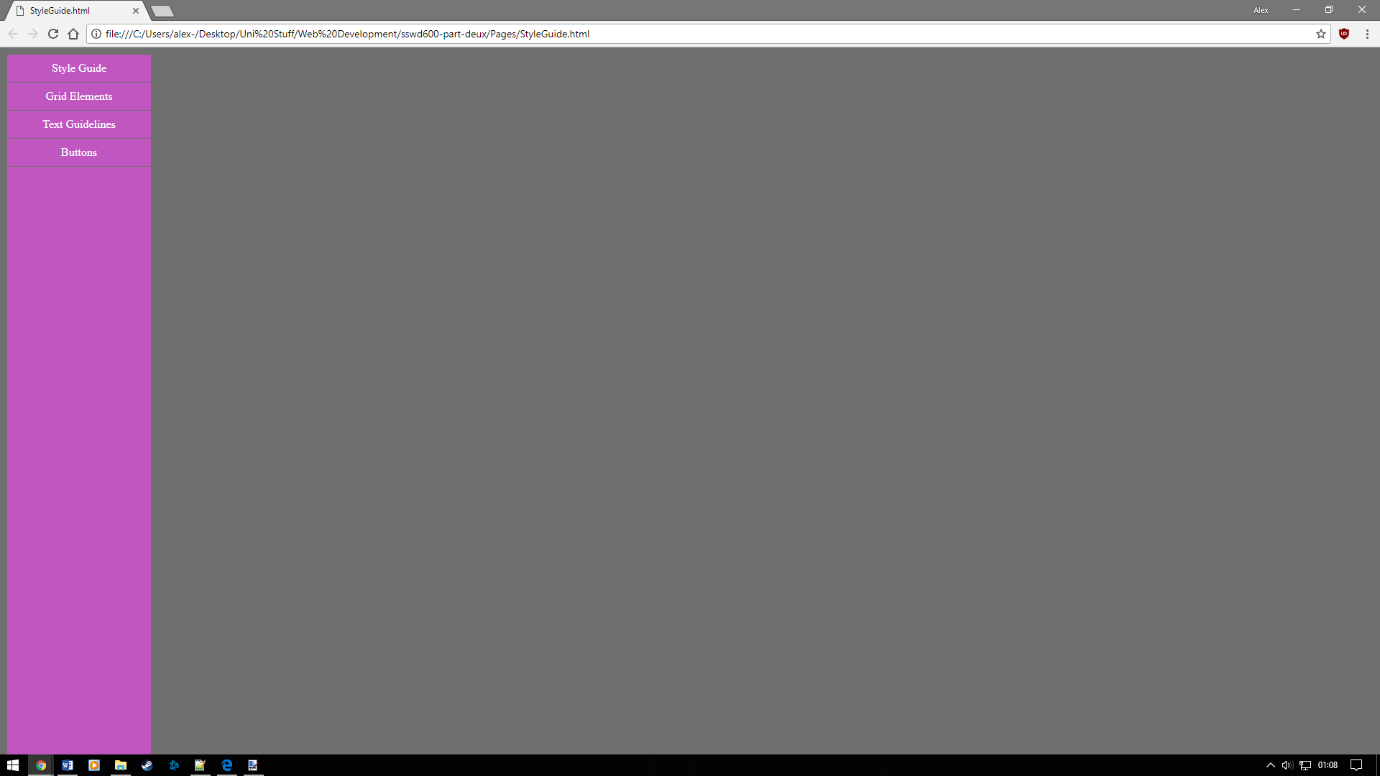
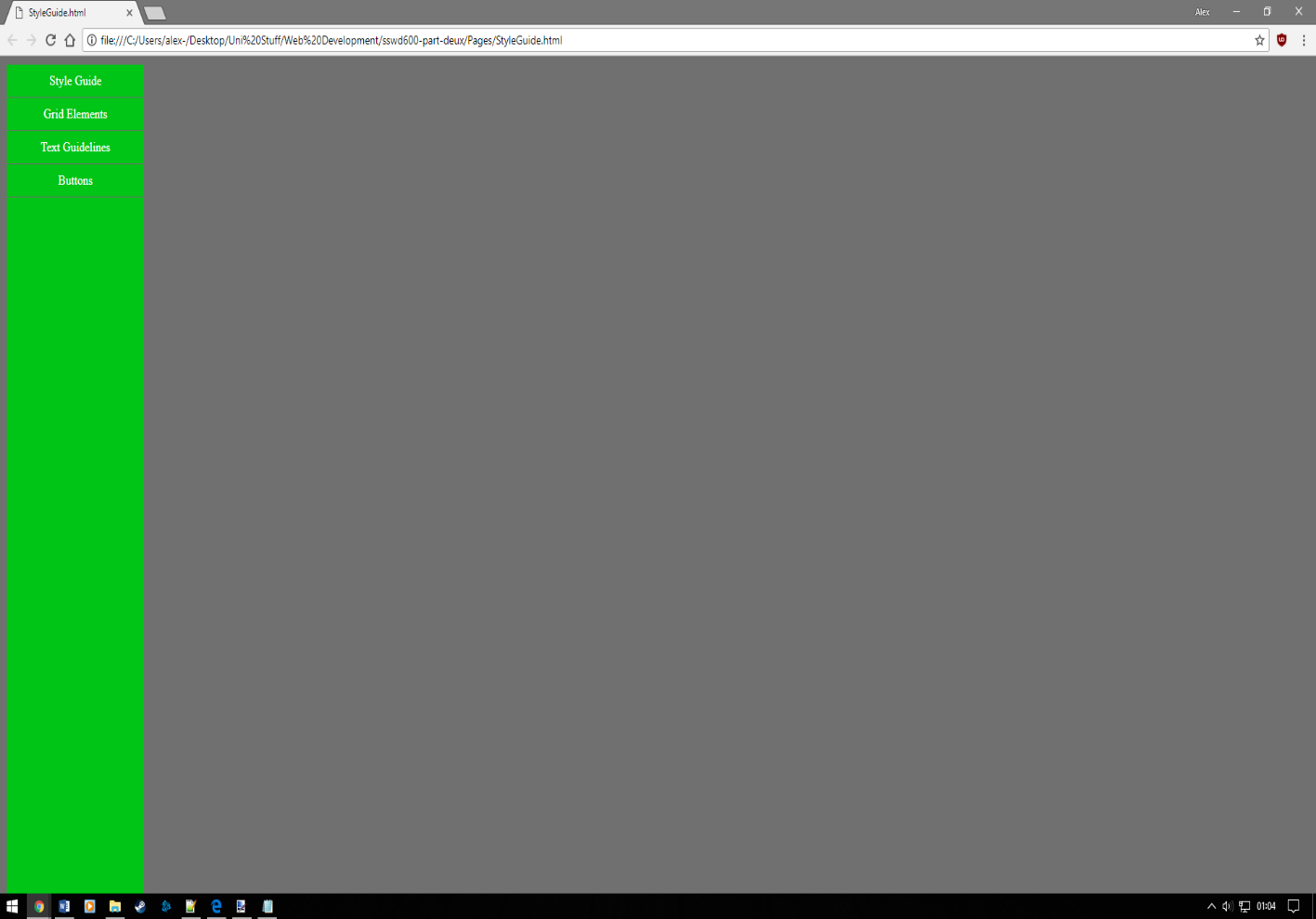
With a better template in place, I moved on to creating my buttons. Through researching various pattern libraries on stylesguide.io, I decided to make buttons to log in and out of a website, two to navigate left or right through a presentation or slideshow, and two buttons that linked to Twitter and my GitHub repository respectively. Creating the latter two buttons was the most difficult, because I couldn’t just add a ‘href=”website.com”’ link within the button, due to this function not being supported in HTML5 and such being bad practice, having to instead create a small form element that would link to the sites when clicked.

The last piece of coding I did was the Call to Action/ Own choice component. I chose to design a button that would enable a user to sign up to a website, while also having a checkbox to not subscribe to a newsletter.

Experimentation



The initial build of the site was based on having a dark grey background, with a red navigation bar with a blue header white text. This plan was quickly scrapped when put into practice, because I didn’t like how the colours clashed. I also considered how the other elements in the library would look, because I felt that having too much red in the web page would feel, not aggressive per se but definitely less welcoming than I would like.



I experimented with the above colour schemes, deciding on the green, primarily because green is commonly used in a positive sense, and still retains the calming feel that the blue does.

Conclusion

I am fairly happy with the way website turned out. I somehow managed to get all the components to work and feel moderately responsive, and have given myself a head start for future improvement. I am still unsure about the final colour scheme, having never found a set of colours that felt just right, which is why I didn’t include a section on colours in the navigation bar. I also feel that if I had planned out how to display the different snippets and elements, I wouldn’t have had to redo as much work as I did. On the other hand, doing so gave me more reusable elements I could use to speed up development of the rest of the site, so it could be argued that the hassle evened itself out.

I mention in my readme file that I use the BEM naming convention in this site, but I’m not entirely sure I did to the best the standard can be used for. Because of the way I re-used the elements I created, and the uniformity I wanted to minimise the workflow, I never used the full ‘modifier’ part of BEM.

The brief called for me to investigate a web component of my own choice and implement, but I confess to getting so focussed on early colour schemes and layouts that I left that component too late, and chose to implement the relatively simple ‘checkbox’ component. I feel less guilty however, when I think of how I used it as part of my Call to Action component, as part of a system to entice viewers to the site to subscribe to a fictional newsletter. Additionally, the brief called for frequent updates to GitHub. However, due to misunderstanding the software I only ever pushed my very beginning project to the dedicated repository before the hand-in date, which is my own fault for not checking the pushes were going through.

Testing the site in different browsers revealed an issue with Internet Explorer 9 and below, which apparently doesn’t support CSS grids. Fortunately, the layout

I feel the final output could have been improved by utilising some jQuery in the site, but due to my inexperience with HTML and CSS, I never got around to learning how code jQuery and implement it in the site. However, because I want to get better at web development it is definitely a topic on my radar to research.