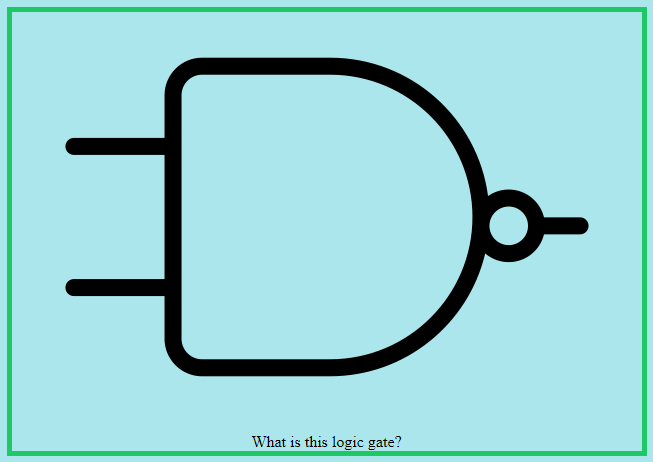
Changes:

When creating the design, the background was white, during the making of the website I discovered black text on a white background is hard to see for people with poor vision. I opted to add a light blue background that took the edge off the bright white but still fitted with the original green element design.

#1



#1 – Showing a green border around a picture

I decided the border around images should be green to keep a consistent feel down the page even when you have scrolled past the nav bar.

To help the user understand where they are on the website, I added a header with the page name to every page.

#2



#2 – Showing black border on highlighted button

I realised a colour change to identify navigational buttons may not be enough for people who struggle seeing colours, so I added a border that appears around the button when hovered over.

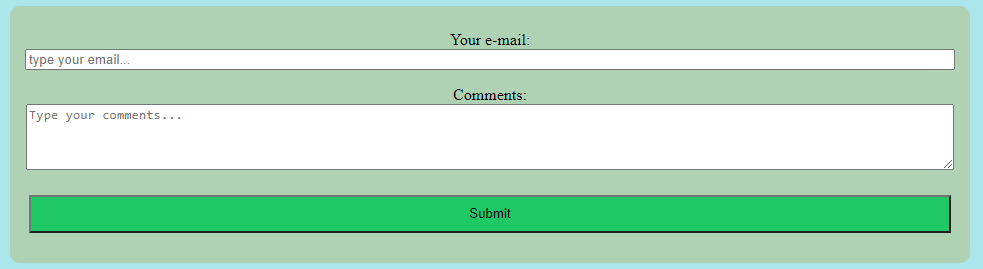
I decided the third chart should be a boxplot on the productivity page.

Opposed to the design’s site map, on the quiz page all questions where placed on one page for efficiency – they did not need their own pages.

Organisation:

Each page was created with a template that had the navigation bar and footer written so that all pages would appear the same. All pages use the main site\_design.css to manipulate the common elements such as the nav bar and footer but also the common elements that I wished to keep similar across the website such as image figure borders and button style.

#3



#3 – Showing a unique element that was designed in a unique CSS file

Each page has its own CSS file for unique element style, these could have been placed in the main CSS file, but I preferred the smaller document for individual styles as rules may get lost or forgotten in the large file.

Each unique CSS file had code to set the current page’s nav button darker to highlight the current page. Some files only have this and those could be moved to the main CSS file, but I prefer the structure in case I want to add a unique CSS rule to a page.

To start with all pages had a linked JavaScript file for the nav buttons but I removed this in favour of anchors as buttons. I felt JavaScript should only be used when needed and it was not needed in this case.

Now, only the quiz page and productivity page have JavaScript files that are unique and linked to just them. The quiz JavaScript allows the user to click their answer – I felt this was more satisfying than CSS hover. The productivity JavaScript Has functions for drawing three different charts and creating a table from the data in the file.

The different types of files are separated into folders just of that file type.

Optimisation:

#4



#4 – Showing an example of a svg image

I have kept the number of images to a minimum and have used no videos as these are the largest file size so take longest to load. I used a svg file for the Ruby logos which produce clearer images and have smaller file sizes than png.

Security:

#3 – Also shows form that is referenced in security section

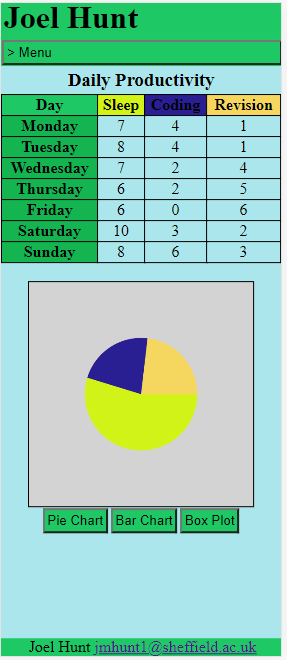
I am using a form on the contact page; I cannot guarantee the security of this service. For this reason, no vitally secret information is collected. Only the user’s comments text and the e-mail that is used to send the message. The website is likely not run-on https as that would require an SSL certificate so there could be a man in the middle attack to steal the forms data. For this reason, I wished to avoid disclosure of the user’s personal information.

Debugging:

I ran my pages through a validator and found I had </h2> instead of </h3> for one of my headings, this has since been corrected. This would have been a mistake from adding similar titles to every page to improve location understanding.

Testing:

#5



I have used Chrome’s Responsive Device Mode to view the website at different size to make sure the elements change from mobile to desktop. I have also accessed the website on my phone using VS code live server and all text seems readable to me and the images change to the appropriate mobile layout.

#5 – Showing the use of Chrome’s Responsive Device Mode to represent a Galaxy Fold. Shows how elements change to be an appropriate size for a handheld device.