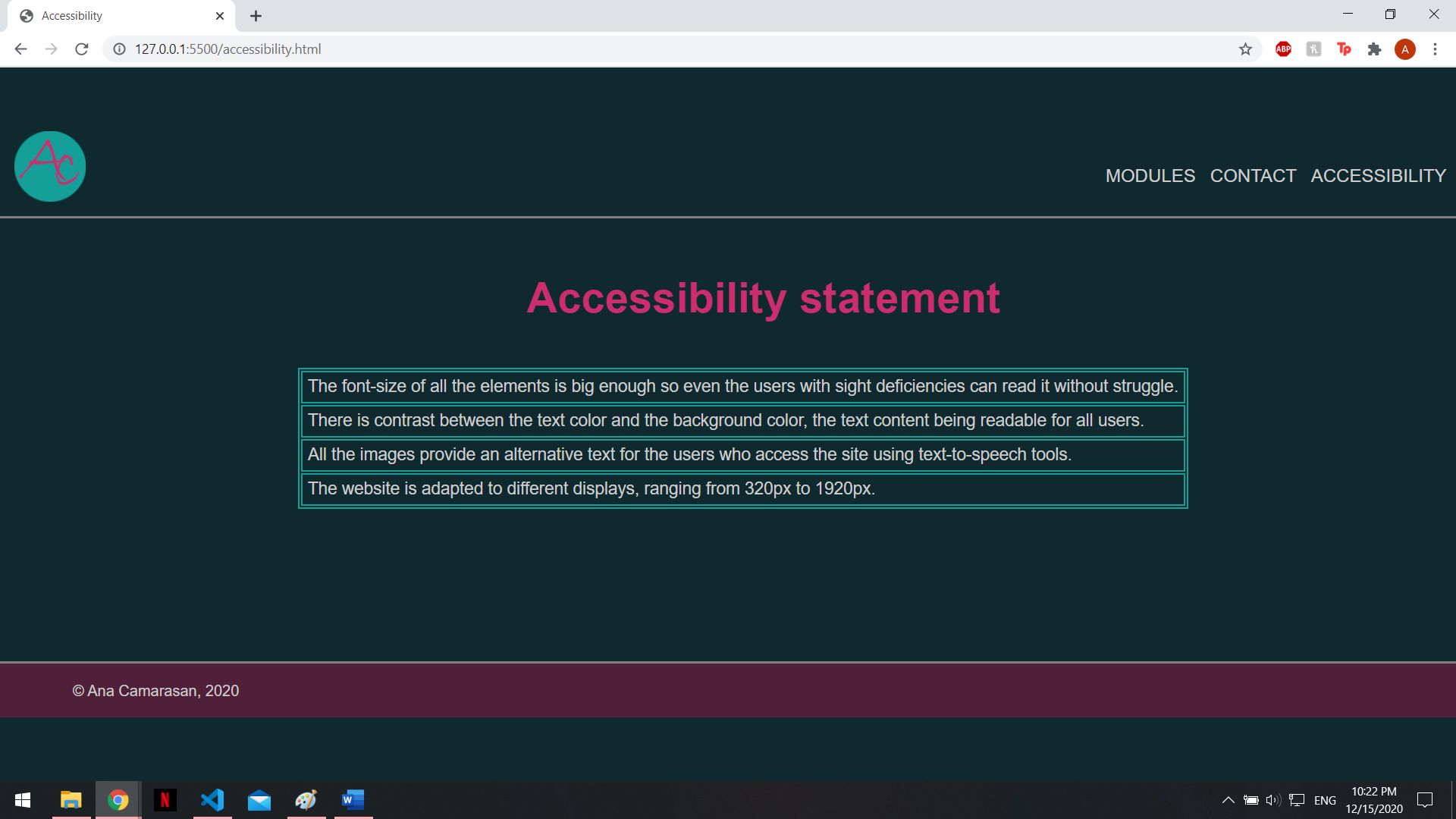
**Development and Testing**

**Changes**

The design remained almost the same as the one in the mock-ups provided in the other document. The only major change was on the accessibility page where I opted for a table as a way to display the statement. I made this decision after researching a couple websites about how accessibility statements should look like.



**Figure 1. The accessibility page with the new design**

**Organization**

* **Templates**

I created a template containing the header and the footer which I copy-pasted on every page. That was the start point for all the pages, the common elements that contributed to the website’s design. The navigation on the site is easier, the menu being all the time displayed at the top of all the pages.

* **File structure**

The file structure is a simple and intuitive one. The .html documents are in the main project folder. The .css file is in the css folder, the .js documents are in the js folder and all the images are in the img folder.

**Figure 2. The folders and files structure**

* **Debugging tools**

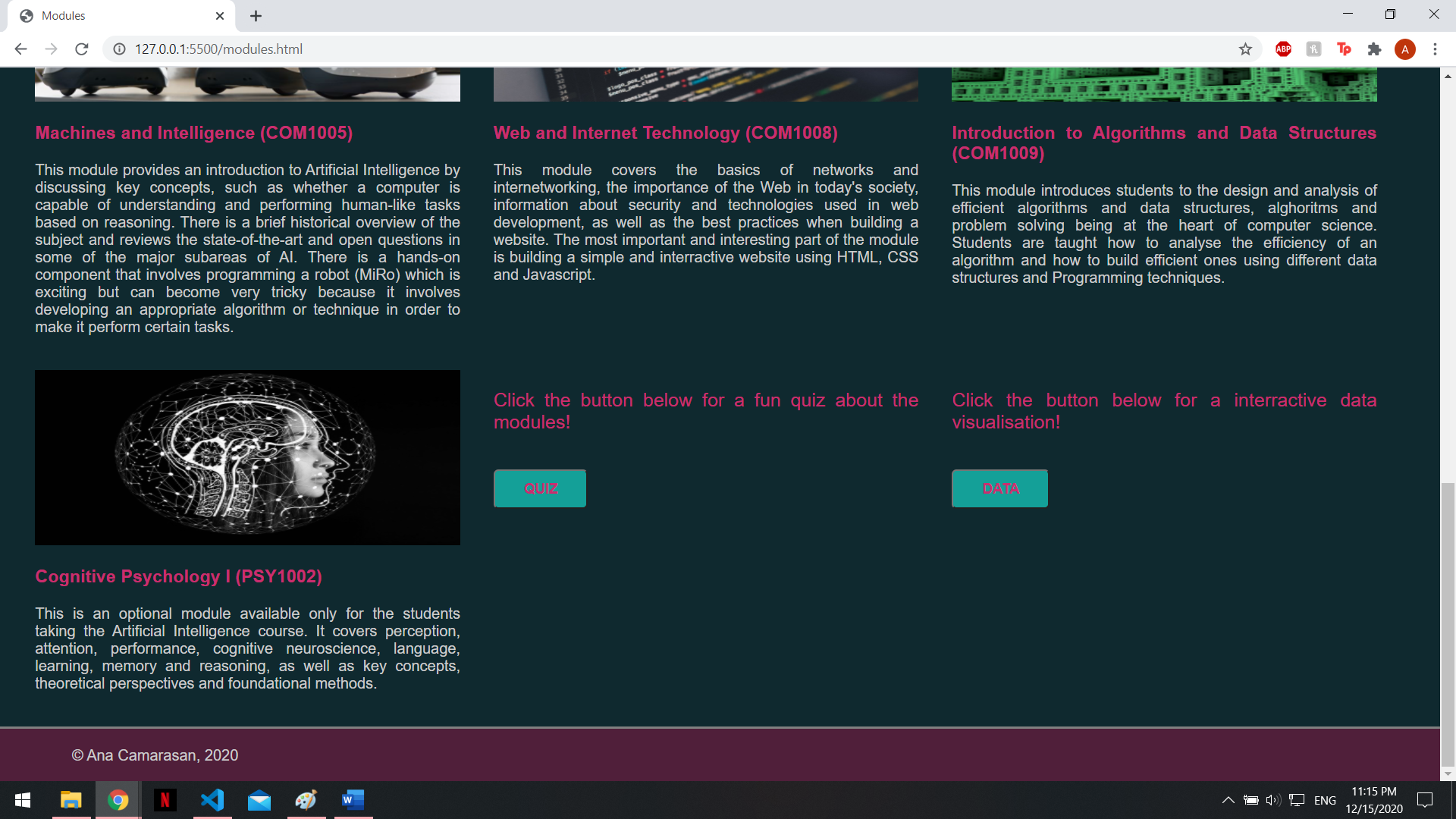
I did not use many debugging tools, only a HTML and CSS validator (<https://validator.w3.org/> for HTML and <https://jigsaw.w3.org/css-validator/> for CSS). I used the terminal in VS Code for syntax errors and the console in the web browser to debug the JavaScript functions.

* **Menu**

I organized the menu in a very simple and predictive way, as I stated in the design document, the website becoming user-friendly. The home page can be accessed from every page by clicking the logo on the top left corner. The modules, contact and accessibility pages can be accessed from the menu displayed on all the pages. The quiz and data visualization can be accessed only from the modules page by clicking on the buttons displayed at the bottom of the page.

* **JavaScript**

I used JavaScript on two of the pages: the quiz page and the data visualization page. On the quiz page, I revealed the correct answers by changing their color to green when the user clicks on the “Check your answers” button. The most of JavaScript was used on the data visualization page for drawing the three types of visualization on the canvas and for switching between them and their colors with buttons.



**Figure 3. Quiz and Data buttons for accessing the pages from the Modules page**

**Optimization**

The site is not very complex, and does not require many simultaneous processes, things that result in a very quick page loading. The original images have large resolutions and appear resized on the website. Despite their large sizes, the pages do not show problems while loading, so I do not see what optimizations can be done on my website considering its reduced size and complexity as well.

**Security**

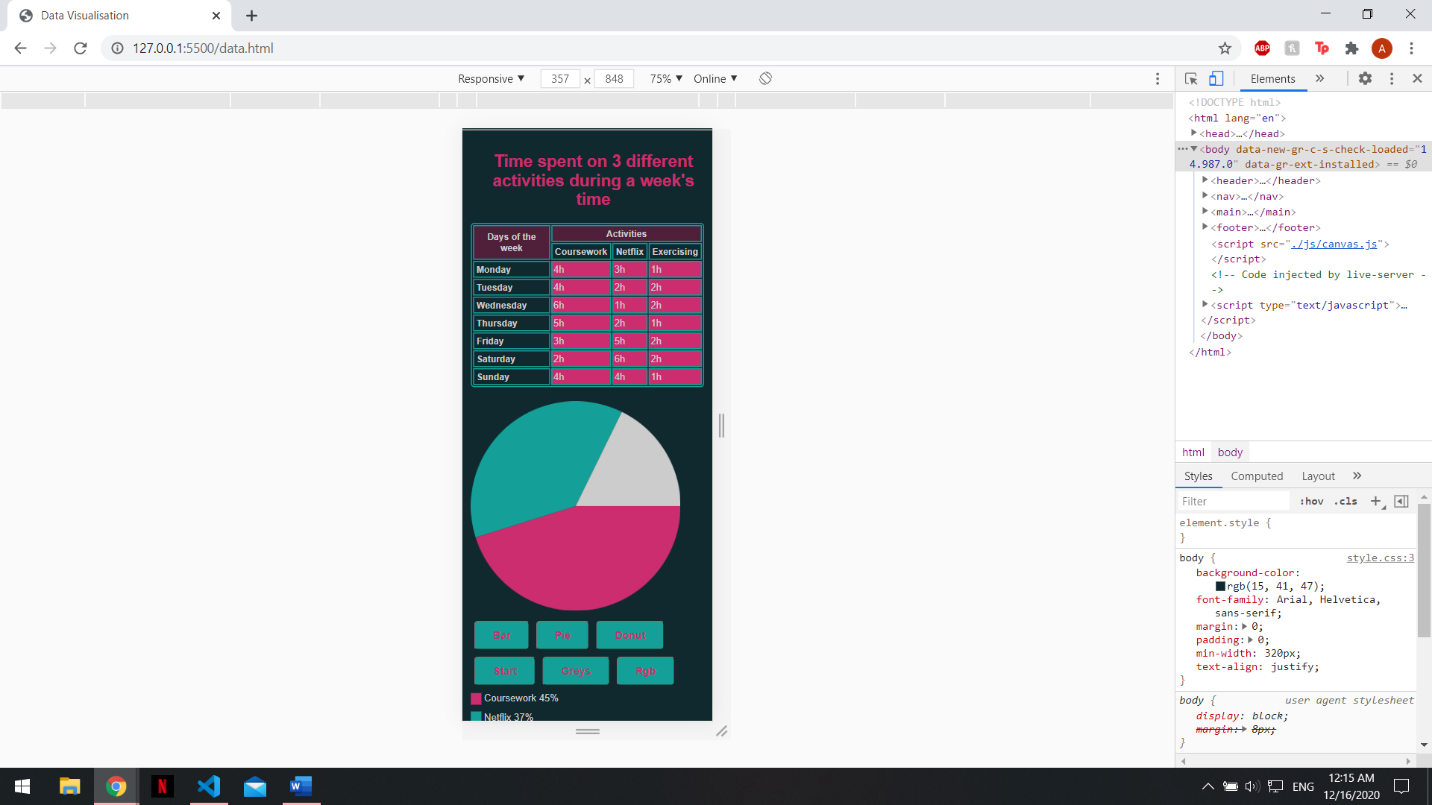
The contact page asks the users to introduce their email address in order to send me comments or suggestions about my website. I used HTML 5 semantics in the form, as suggested in the lectures. The website does not have a database, so the user inputs are not stored anywhere, not even displayed on the website, their data being protected this way. Only I can get their email if they want to send me a message. As my website is locally stored and developed, I do not need to make use of http or https.

**Debugging**

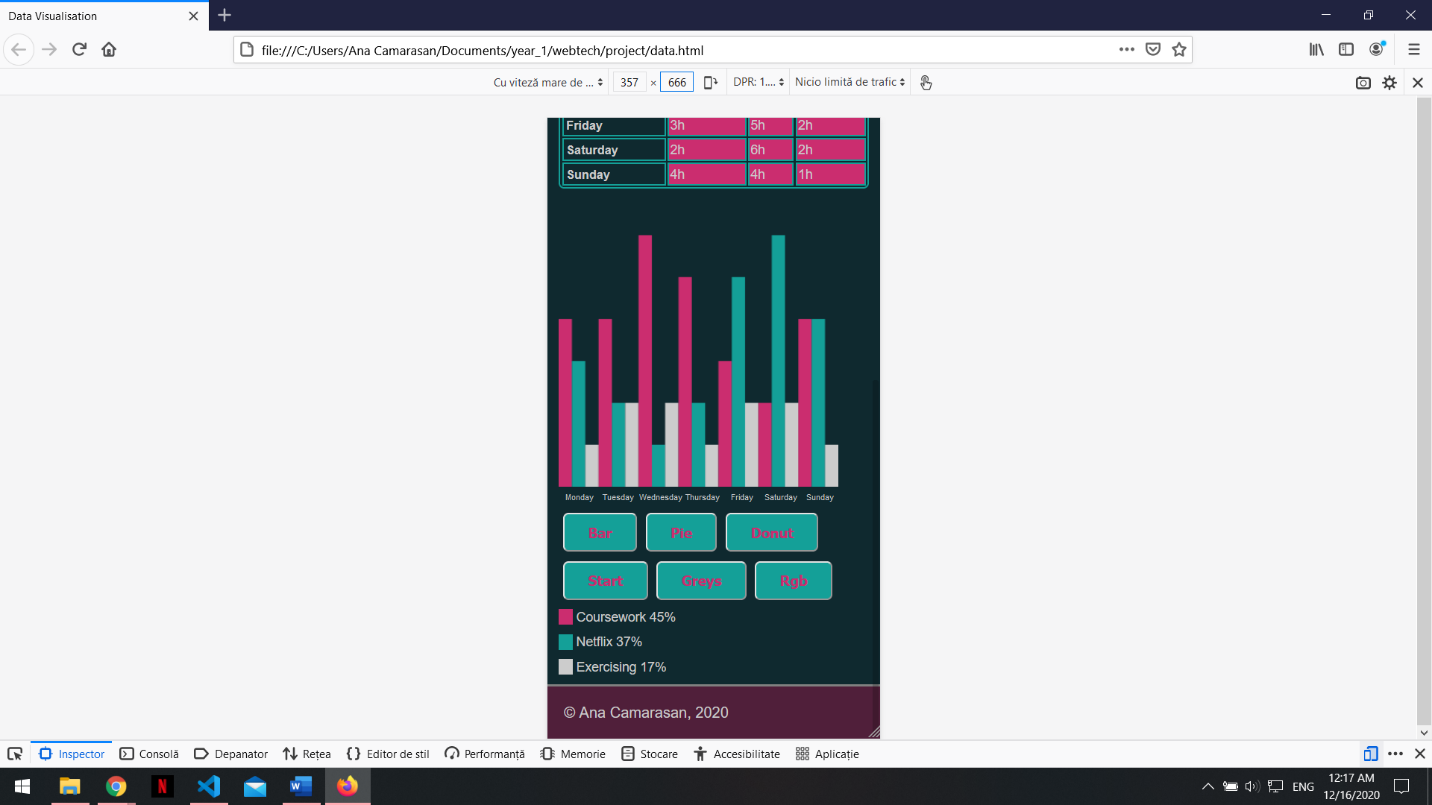
I used the validators mentioned above to test my code. I was surprised to see that I almost had no errors. The ones I had were very easy to fix and the code ran smoothly. I was very careful while writing every piece of code and I think this is the reason why I did not have to deal with many errors. I followed the guidelines in the lectures and I made use of the knowledge I gained from the practical work.

**Testing**

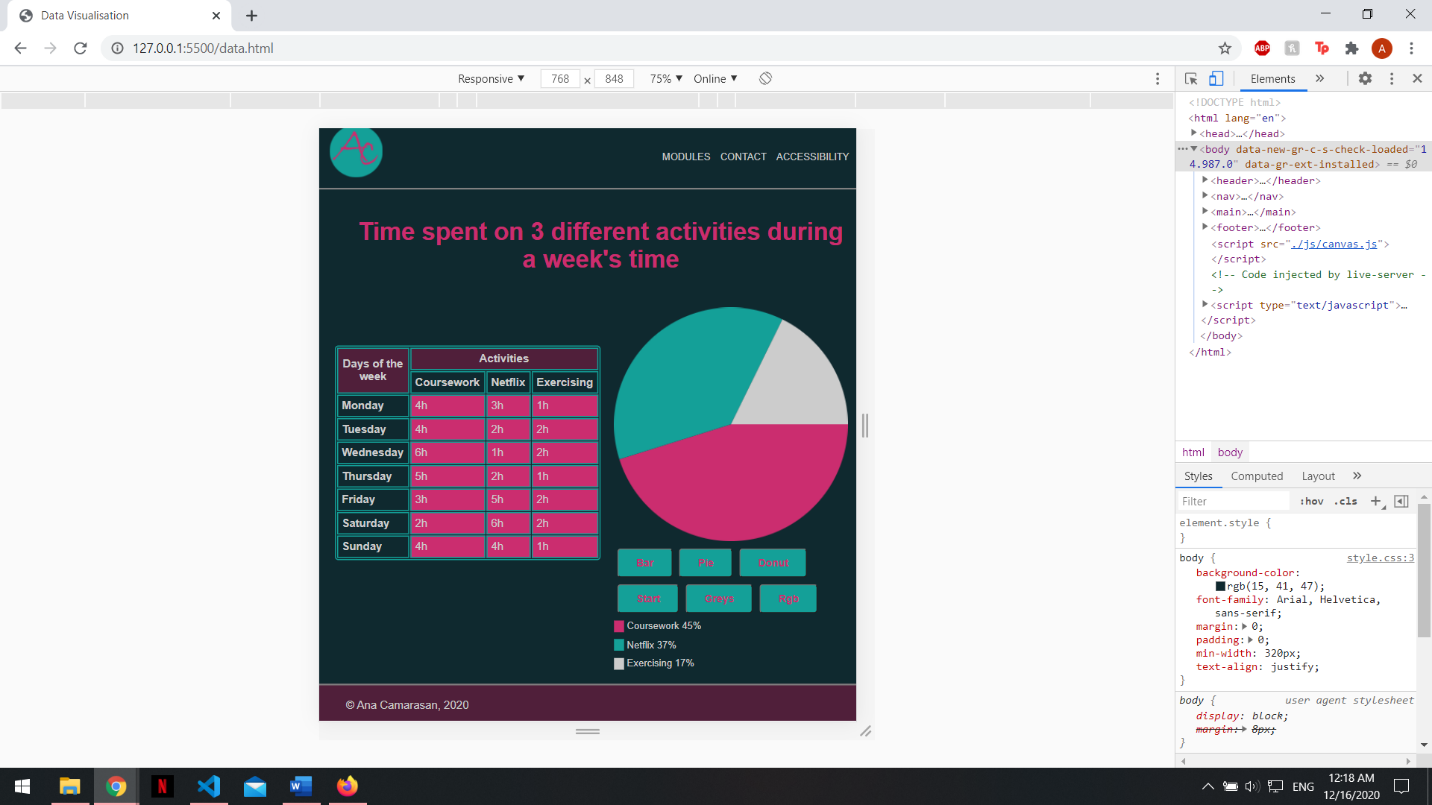
I did the testing during the development. I tested every page I created on different device sizes using the Chrome’s Device Mode and Responsive Design Mode available in Firefox. I made very much use of the Chrome’s Device Mode as I set my breakpoints and tweakpoints using the device sizes suggested there. The inspect element feature on Chrome shows whether there is contrast on every element one hovers over. I made sure all the content has contrast and meets the points stated in the accessibility statement.



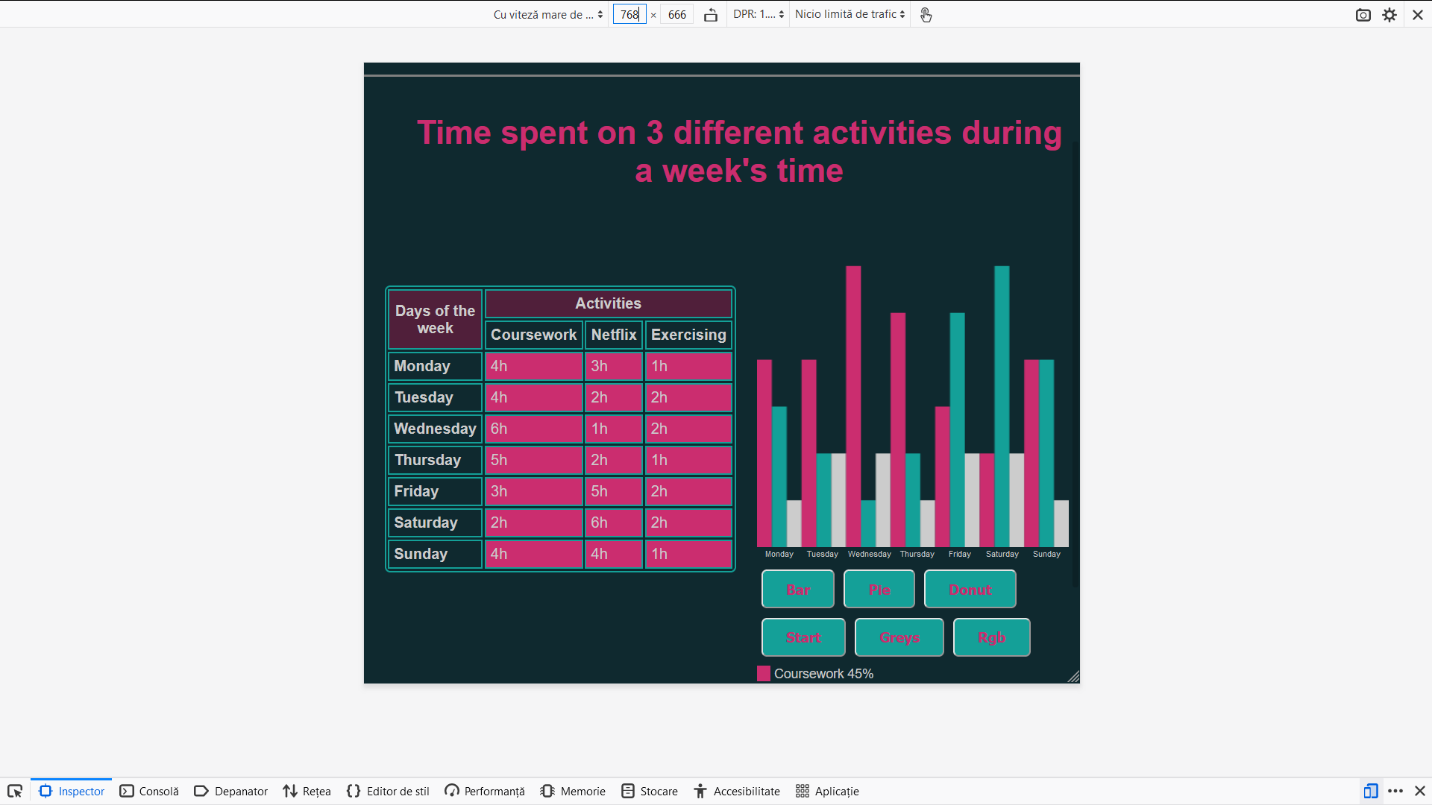
**Figure 4. Data visualization on Chrome’s Device Model, mobile version**



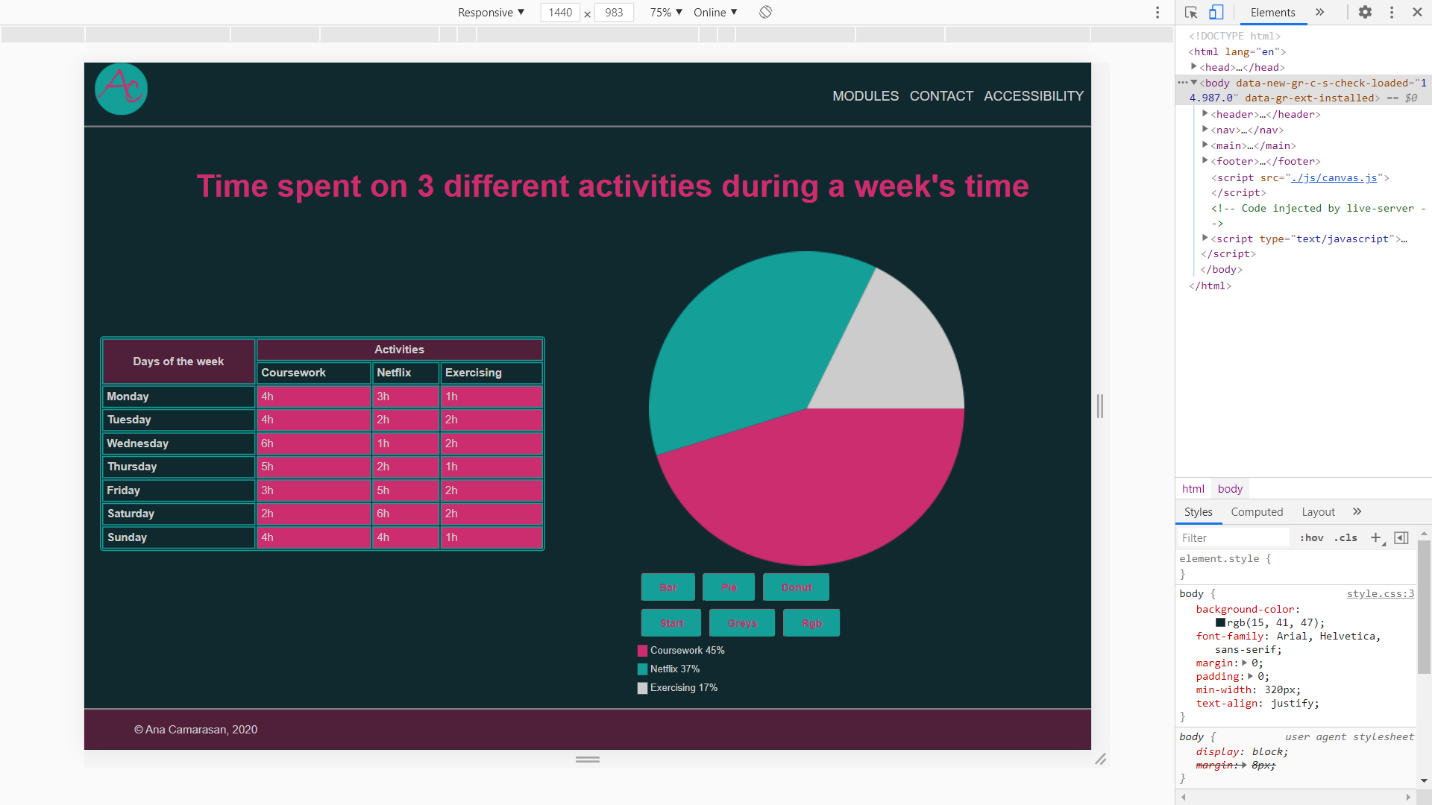
**Figure 5. Data visualization on Firefox Responsive Design Mode, mobile version**



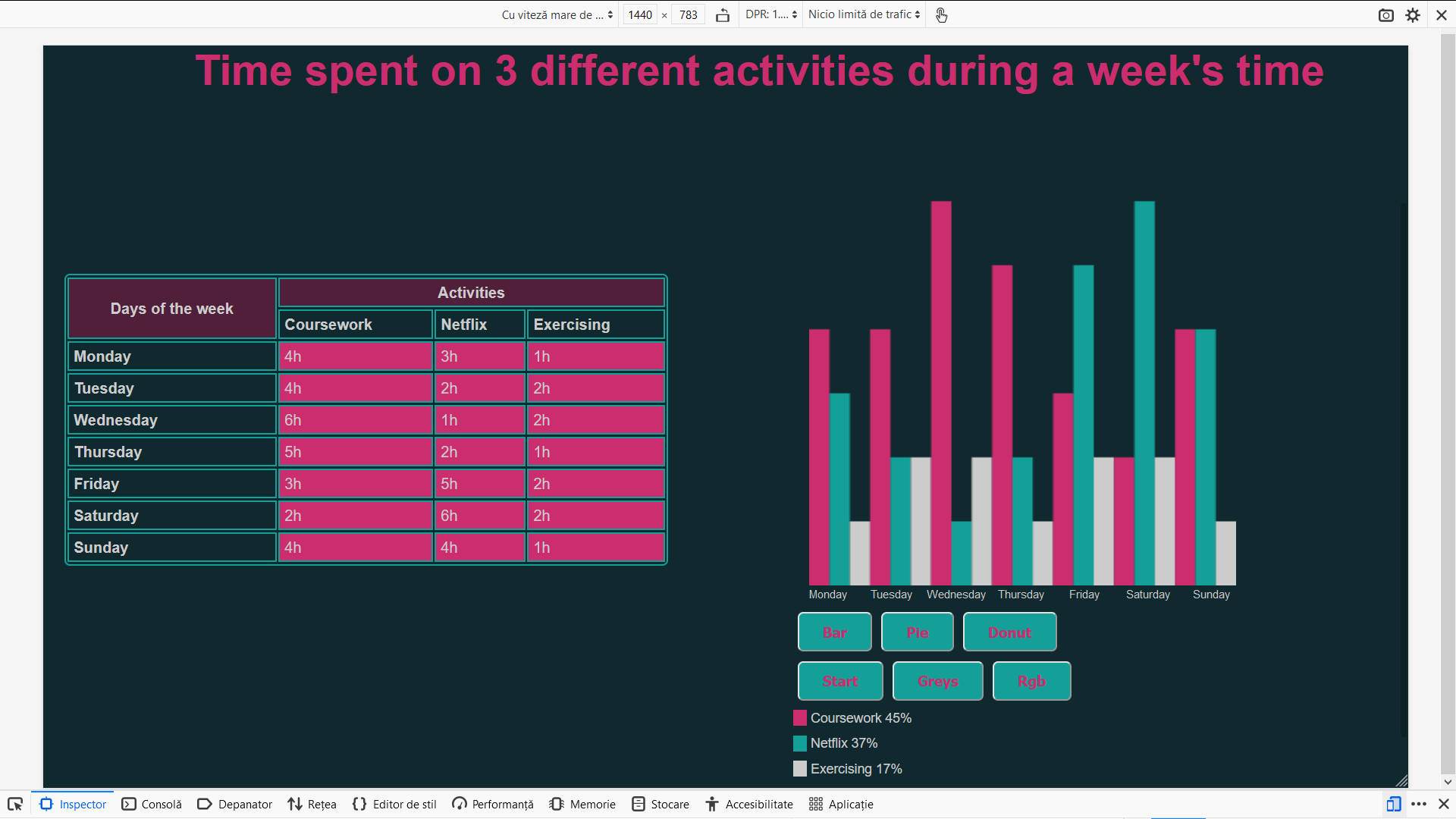
**Figure 6. Data visualization on Chrome’s Device Model, tablet version**



**Figure 7. Data visualization on Firefox Responsive Design Mode, tablet version**



**Figure 8. Data visualization on Chrome’s Device Model, desktop version**



**Figure 9. Data visualization on Firefox Responsive Design Mode, desktop version**