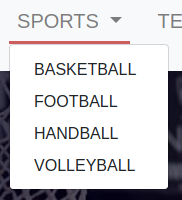
# **Home Page**

Overall, it was easier to achieve responsiveness on the web page using Bootstrap because the css is essentially done for you. The biggest issues I encountered usually occurred when i would add css properties to bootstrap components. For instance, I was adding ‘d-flex’ to carousel items so they take up 100% of remaining space in ‘carousel-inner’ - this would essentially break the carousel. This was tricky because some of my css would end up overriding the bootstrap’s default css properties. I was able to get around this issue by inspecting the bootstrap components’ default css and making sure my code didn't interfere too much. In my particular example, it was much easier to add ‘height:100%’ to the carousel items, this way the display property of carousel items was unaffected.

**Sports page**

Wanted the dropdown menu of sports to show while hovering over it with your mouse. That was easy to apply

.dropdown:hover>.dropdown-menu {

display: block;

}

Then there was a problem, if you clicked sports and moved your mouse cursor away, the dropdown menu won't disappear again.

This was fixed with:

.dropdown > .dropdown-toggle:active {

/\*Without this, clicking will make it sticky\*/

pointer-events: none;

}

Another problem with the hovering. The dropdown menu by default had a top margin wich would make it disappear if you moved the cursor down slowly. That was fixed by removing the default margin

.dropdown-menu{

margin-top: 0px;

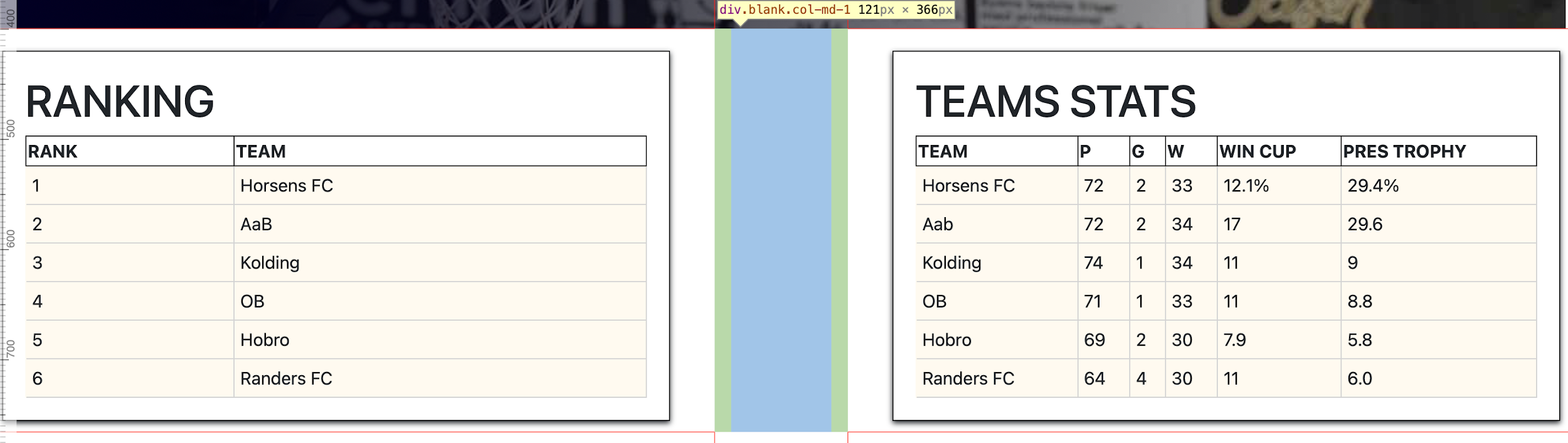
}

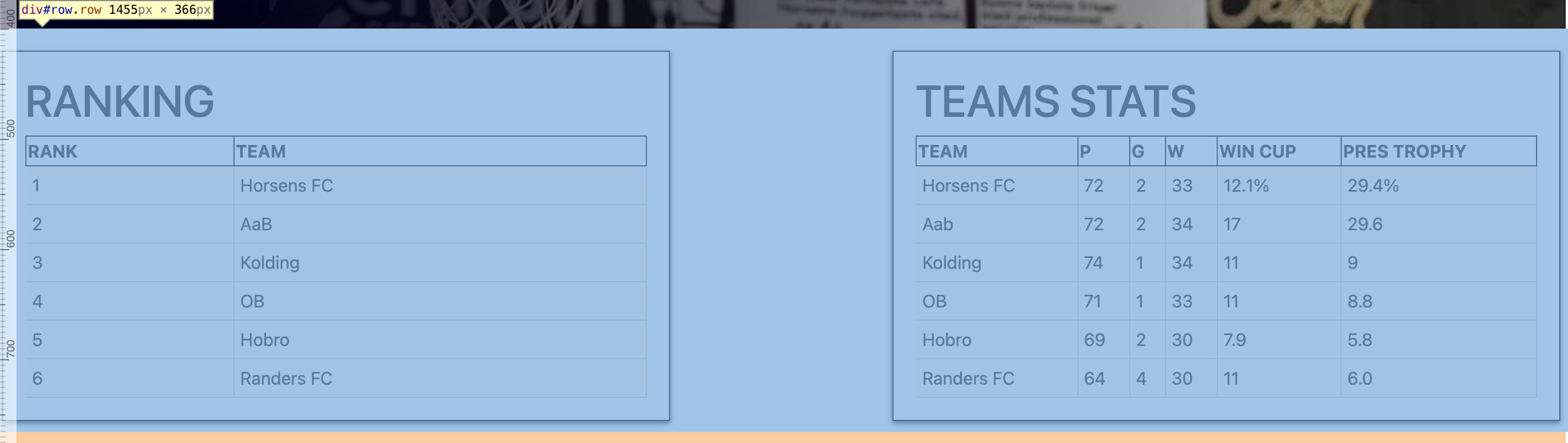
# 

# 

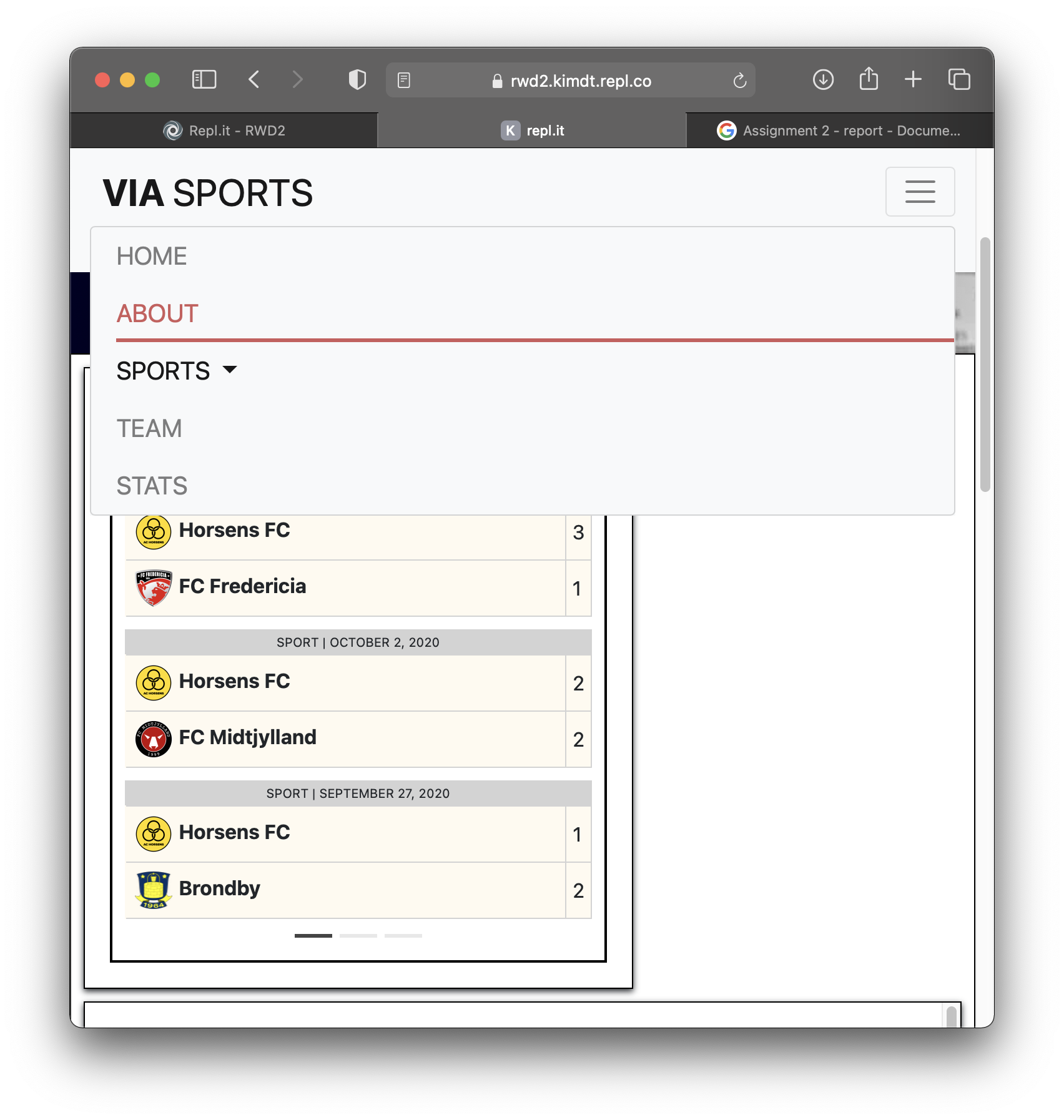
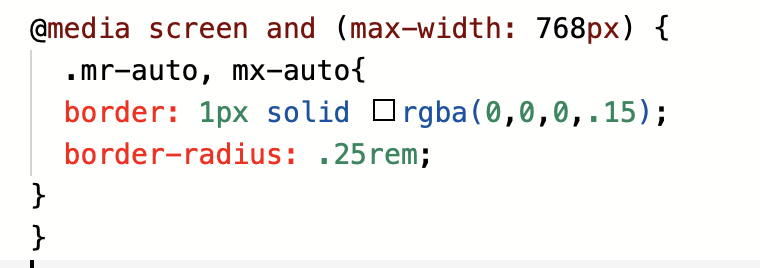
# **Stats Page**

One of the issues that occurred when developing this part of the website was the space between the two tables. At first, I have tried to make use of div’s offsets which were not properly laying out the elements. By this I mean that the code I was trying to implement was overwritten by bootstrap properties. Therefore, I have used a blank div in between the two tables of size 1 just so they are equally distributed. Moreover, the divs are also distanced from each side by using the code “justify-content: space-around” so that the page has a symmetrical layout.



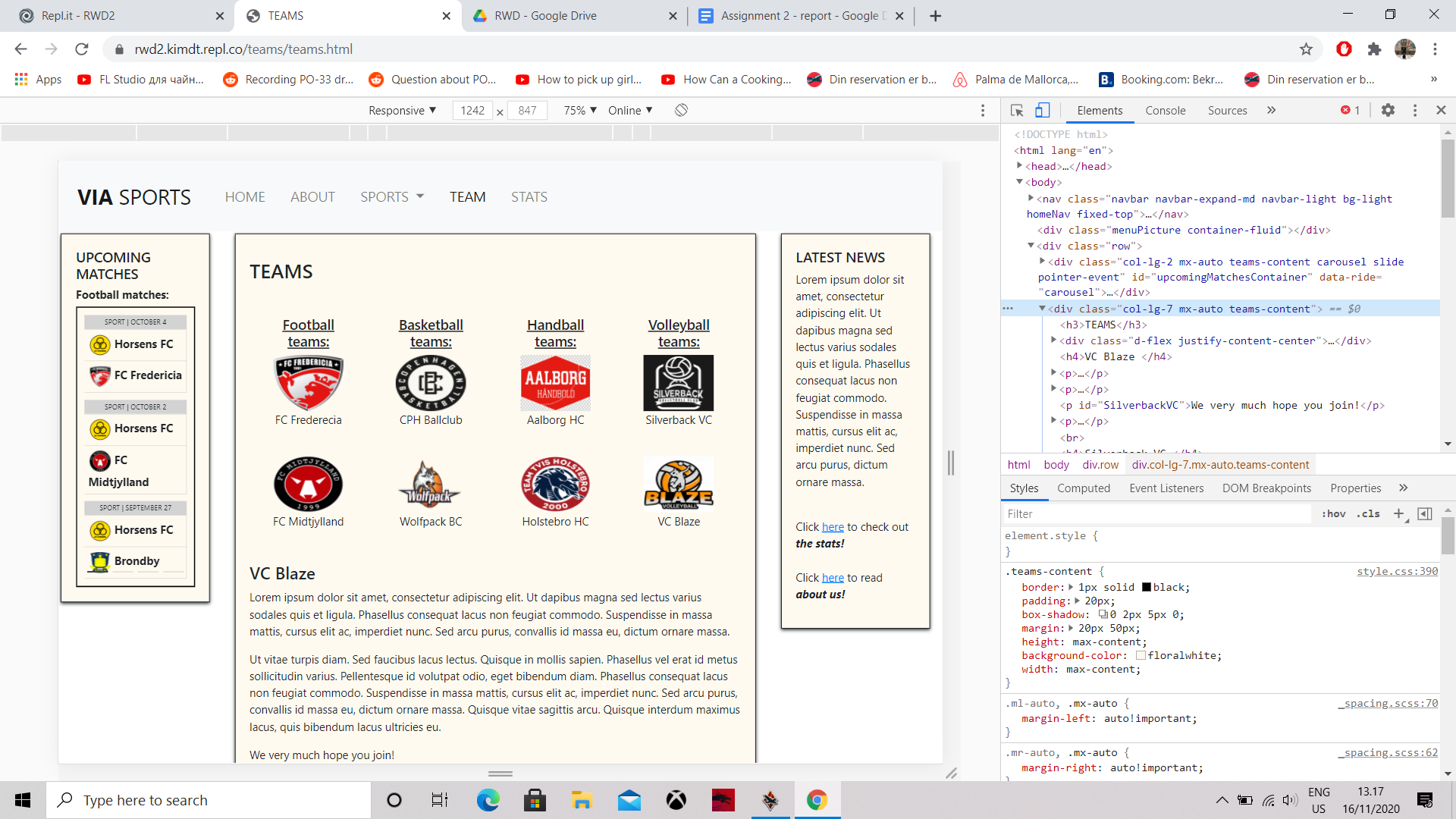


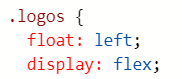
# **Navigation Bar**

The navbar was a subjective aspect about this site in particular. The reason for this is the fact that we had some issues with the visibility of it. To be more specific, whenever we were scrolling through the website on a screen smaller than a laptop, the tabs were implemented inside a menu which could be accessed by clicking the top right button. The tabs of the menu were not visible compared to the rest of the website, so we decided to add a border. We have tried a variety of border styles and we came to the conclusion that the menu should be surrounded by a border rather than having each tab covered by these. Also, the borders were added to the buttons through media queries just when the menu collapsed, but not when the navbar is completely visible.

# **Teams page**

There were a lot of problems fixing the logos to center and stretch within the box, when resizing the screen. To fix the problem I’ve put all divs for each sport in one separate div and created a .logo class to manipulate some css styling. With that however, I didn’t achieve the desired effect (when resizing the screen the logos would overlap the “Latest news” box) and after some research and trial i’ve found d-flex justify-content-center to work perfectly to fix the error.







Because we chose to have a fixed Navbar this time, the links from the team logos went further down on the page than expected. To work around this issue, the id’s in the html needed to be set a little closer to the top, so the links are actually sending the reader to a right spot in the text.