Assignment 3 Write Up

The process of adding CSS to our HTML templates with two different approaches was enlightening. Our group is composed of mostly web-programming beginners, so while we had some general idea about the differences between using bootstrap and vanilla css, this assignment exposed us to the true tradeoffs of these approaches.

The Foodex team decided to get their hands dirty by styling the HTML templates from the previous assignment with vanilla CSS before using Bootstrap. Before starting the assignment, the team was aware of the purpose of Bootstrap and its renowned use across the web for making a website "responsive." We immediately knew that we were going to have to tackle the problem of making our web pages responsive using pure CSS. This was an issue for the team since no one really had a solid understanding of how to make a website responsive without Bootstrap. So the team was forced to search across the web for solutions. One way we made our website responsive was through the use of flex boxes. The flex box proved to be a very simple and effective method for laying out our recipe profiles on the Foodex main page and Personal page. We also made use of media queries mainly for controlling the font-sizes, paddings, and margins across the different media platforms. Besides having to learn how to make our website responsive, we invested time into making the Foodex website visually appealing. We had to create many CSS rules from scratch in order to personalize our input, buttons, footers, and nav tags. When it came to using Bootstrap, we didn't need to style those tags as heavily because the framework already did most of the work. Although we had to invest a lot of time styling our website using vanilla CSS from the ground up, we definitely had a lot more freedom in the overall design of our website which made it feel a lot more closer to our vision of the Foodex website. Another advantage of using vanilla CSS was the fact that the HTML pages that used our vanilla CSS were definitely more semantic than the HTML pages that used Bootstrap. Being semantic meant that our structuring was clear and intentful. However, it also meant we had to write more frustrating CSS code. In addition to being semantic, the lines of codes were significantly less than the Bootstrap version. This is obviously a clear plus since the overall byte count of our HTML files remained low.

When it came to styling our templates with Bootstrap, we did our best to acknowledge Professor Powell's caveats - rather than downloading the entire Bootstrap library, we obtained a custom build containing only what (we think) we need. However, while we tried to be minimal in what we chose to include, we still wound up with a couple huge css files that obviously contains more than we need. Additionally, we found that, while Bootstrap provides a huge amount of custom styles, we still had to add our own css for our application specific needs. This may be due to a lack of familiarity with what Bootstrap has to offer (not knowing what to look for if we've never used it).

However, it wasn’t hard to pick up on what Bootstrap had to offer. The grid system is very intuitive and makes it extremely easy to create pages that behave well in all screen sizes. Tasks such as making the navbar move from the left side to the top of the screen were suddenly made very simple by employing both the grid system and Bootstrap’s custom CSS classes.

Below are some statistics on how we found the different approaches affect the experience of the user (load time, byte count) and developer (hours of work, lines of code). Having implemented the same project with both methods, it is clear to us that they both have various situations that call for their application.

Version	**Hours of work**	**Lines of Code**
Vanilla	14	394
Bootstrap	9	5960

Vanilla (Mobile 3G)

Web Page	Load Time (in seconds)	Byte Count (in kilobytes)
Login	2.789	33
Main	6.661	746
Personal	3.605	226
Recipe View	2.962	91
Add/Edit	1.965	66
Sign Up	1.98	19

Bootstrap (Mobile 3G)

Web Page	Load Time (in seconds)	Byte Count
Login	2.723	61
Main	3.723	232
Personal	3.652	236
Recipe View	3.268	85
Add/Edit	2.314	45
Sign Up	2.666	52

While the majority of our group was unfamiliar with Bootstrap, its learning curve proved to be relatively easy to climb. Its intuitive nature, wide breadth of offered features and aesthetically pleasing presentation made it easy to whip up a sexy, responsive web page quickly. The downside is potential lack of performance due to large CSS files. However, this can be mitigated with a custom build if the developer is familiar enough with their requirements and Bootstrap.

Vanilla CSS also has pros and cons. One of the pros was how quickly we could jump into writing CSS code for precisely what we wanted to do. This was different from the Bootstrap approach because we were spending time learning the library to avoid reinventing its features. A con about vanilla CSS is the actually implementation of all the code. This was a bit more difficult because we didn’t have the luxury of calling premade classes from bootstrap, so as beginners, we were slow to start.

Choosing one approach over the other will no doubt affect the speed of development - it is up to the developer to make sure performance is affected as little as possible. Bootstrap can be a convenient choice for prototyping or other situations where performance is less important. Either a custom build of Bootstrap or vanilla CSS are good alternatives when performance (in page load time and byte count) is more important.