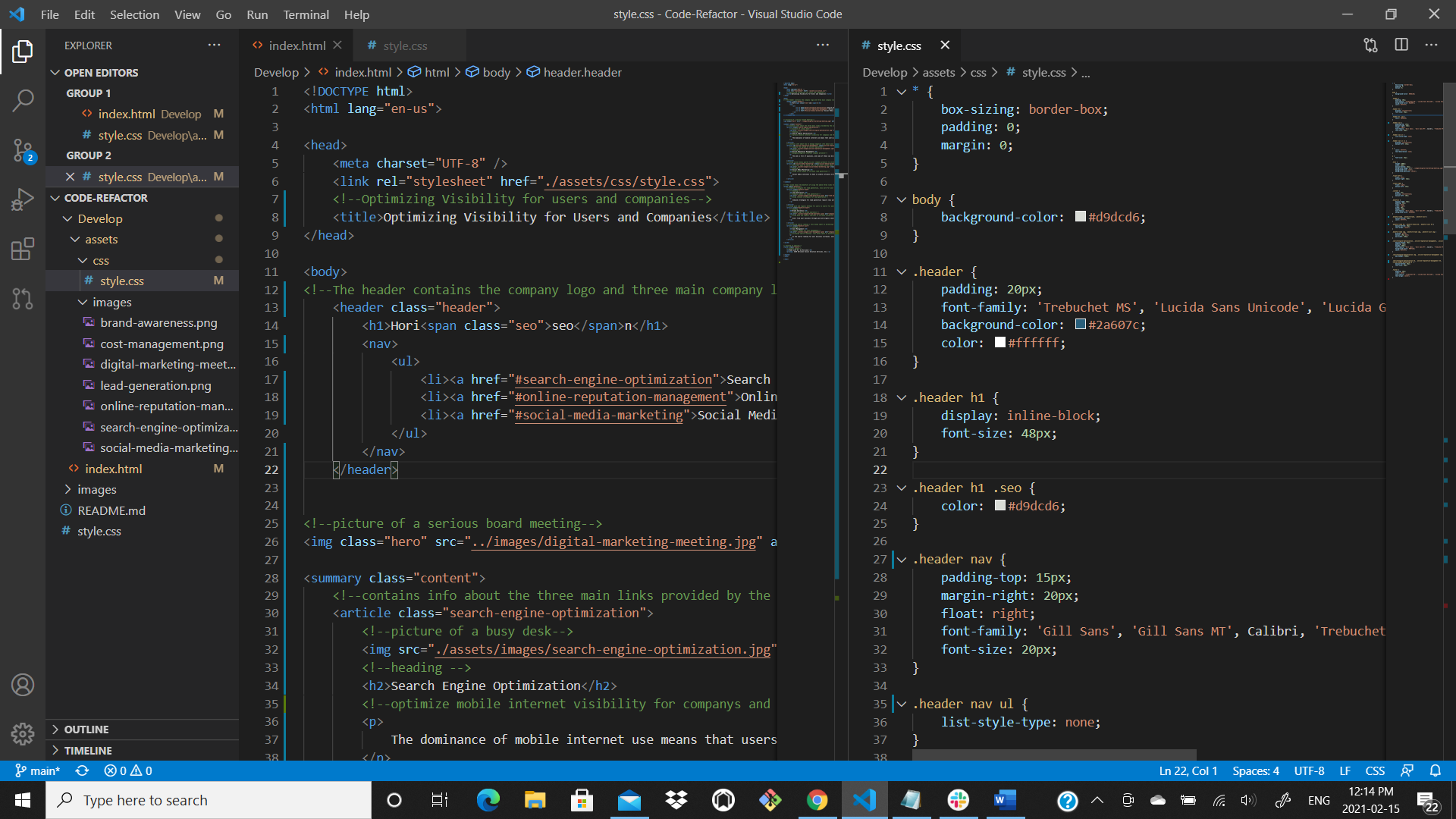
Code-Re-factor

**Repository**

This repository (Code-Refactoring) is an educational homework assignment from CARL-OTT-FSF-PT-02-20121-U-C course. It was designed to help familiarize students with html5 attributes and how to implement them. There is also a CSS component, which allows students to become more acquainted with their interconnectedness.



**ASSIGNMENT**

The assignment was to help a marketing agency refactor existing code so that it meets their specific accessibility standards, which in turn would optimize it for search engines. Their requested accessibility standards included: the source code be updated with semantic html elements; the html elements follow a logical structure independent of styling and positioning; has accessible alt attributes; heading attributes fall in sequential order; the title is concise and descriptive.

**Installation: Code-Refactoring**

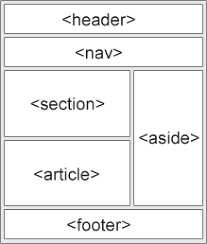
I removed all the <div> attributes from the html doctype and replaced them with semantic html elements. These include but are not limited to: “w3schools list” also using the recommended web page layout.

* <article>
* <aside>
* <details>
* <figcaption>
* <figure>
* <footer>
* <header>
* <main>
* <mark>
* <nav>
* <section>
* <summary>
* <time>

The first replaced <div> was the header section. This allows search engines to easily identify what the section is. The <div> element comes with no descriptors and could contain anything. Once all webpages adopt these elements, search engines will better understand what the page contains. I also replaced the <div> containing the three-header links with a <nav> and cleaned up the un-ordered list to be more visually appealing. I also, had to change the style notes to reflect this change to <nav>, otherwise the CSS would not reflect the required styles. A side note: the title was changed later to “Optimizing Visibility Using Accessibility Standards”.

The second section was replaced with <summary> and not a <section> element as I thought this was the heart of the website and deserved more of a distinction than just <section>. I added an image tag and a src of the image and an alt attribute. This was missing from the html doc and would confuse future developers as to where the picture came from unless they looked in the CSS file. The CSS had set the picture to background and there was no html link other than hero, which appeared to be confusing. I then took the three different <div> sections and placed them under the <summary> and gave them each article elements. This is because each of the three were not directly related but worked together to form the heart of the website.

The third section I decided to label <aside> because they were indirectly related to the summary section of the website. They were the benefits of using the three-above links. Also, in the html layout model below, they are in the aside position.



I did not follow the above html layout exactly as displayed, because calling everything in the section and article section seem to be confusing and was not as visually appealing as I would like it to. I chose this direction for aesthetic reasons. It’s just more clean, clear and concise, just as the assignment requested.

I repeated the same steps for the aside section and cleaned up the code and added developer notes to the entire doc. I then changed to the <h1> to <h6> tags, which came with a new set of problems. The existing code had the same h tags for many of the headings, so they appeared to be the same size on the website. This seemed to appear better than changing them to sequential order. The first section had headings with h2 tags for all three and the second section had h3 tags for the three headings. I added the h tags in sequential order and realized two unintended outcomes.

There were more than 6 headings on the website and the headings were way to small by the time the 6th tag was implemented. In keeping with the sequential order rule, I added the tags up until I reached 5. However, instead of having different heading sizing for the <aside> section, I decided to assign them all h5 tags so the site would appear more visually appealing. The last header was in the footer section and worked out perfectly. As this was all happening, I was debating whether to change some of the header tags so I would have only six on the site. This created a new and unintended problem. The headers all had styles assigned to them and in changing the tags I would have to write new style sheets. Now, this was not a serious problem as it could be easily changed, however it would directly change the flow of the semantic html elements and would be outside the scope of the homework assignment.

All this was done to increase the websites visibility when users use search engines to find similar companies. These changes will meet the accessibility standards which will bring this website higher on the priority list in popular search engines, the end user will see if quicker. Ironically, the company does the same thing that they requested be done. So, the material becomes extremely repetitive. However, this has allowed me to better understand the requirements and to easily attach them what needs to be done.

**Cascading Style sheets**

This section of the homework assignment was much harder to figure out. The style sheet seemed to be working fine and I had made the necessary changes to link the html to the CSS when I changed some of the names. It was not until I stared at the code for awhile and realized a lot of it was repeating. So, I remembered from class that some of the same styles that had the same style attributes could be combined. Once I started doing that (which took some time as left out the.) I began to realize that a lot more of the code was redundant than I initially thought. So, I consolidated all the style elements that had the same values. Then I moved the style headings to be aligned with the flow of the html doc. I also, removed a footer note that had zero value which made no changes to the site.

I wanted to move the paragraph in the footer to heading tag so they would be on the same line and have the same font size. However, the assignment called for the site to look at least 90% the same as the original code. So, I decided to leave the heading and the paragraph the same as they had their own styles in the style notes and did not want to be seen changing the structure of the site.

**What I learnt**

I learned a lot doing this project. I thought it was due on Tuesday February 16th, so I did it all before the class and knew nothing about what I was doing. I had to look up everything for the assignment. I did not even know what semantic html elements where. I extensively used w3schools.com to educate myself as to what they were. I also used various question answer sites to help with questions I may have had. I learnt that any question I may have can be googled and there is an answer 9 times out of 10.

The project itself helped better understand why this type of activity is done. To help companies make their websites more accessible to people by increasing their ranking in search engines. Using proper semantic html tags allows the code to be human and machine readable. Being that most people use their phones to search the web now it allows for better readability across all devices that use the internet. They also increase the ability of people with impairments access the site. This allows people using web accessibility tools such as screen readers the ability to understand what the website is trying to say through <alt’s>. People that are visually impaired or hearing impaired. This is an incentive to reduce the amount of accessibility lawsuits a company may face from discrimination.

It helped me understand the layout of semantic html and how it can benefit future developers with descriptors. It can be hard to guess what a <div>’s original intention was. This easy-to-understand layout allows people to better understand previous developers’ intentions. The developer notes in green on the html doc and the CSS file allow the next user to understand what they are and where they are on the site. All this allows for a consistency in coding that was previously lacking and can be used on phones, tablets, laptops, and desktops.

Update: Figured out the link on the top was missing an <id> and <class>. I first added the id because I noticed it was not the same as the other 2 links. However, that created a new issue, the CSS were not being applied to the search-engine-optimization article, so the website was all over the place. I had to go through a process of elimination to narrow it down to the class. I tried changing the CSS to have it on its own because I consolidated them. That did nothing. Then I tried to check the image and see if there was an issue with CSS for that image. Found there were no issue as all three of the images had been consolidated and none of the styles were missing from the original assignment. Then it clicked, the image was not getting any of the CSS because it was not connected. This took some time cause all the names are the same and I finally noticed the other two have a class and this one did not. Once I added the class, that fixed the issue. Anastasia from class helped with the id and I figured out the rest on my own.

**Contributing**

<https://www.w3schools.com/html/html5_semantic_elements.asp>

<https://www.freecodecamp.org/news/semantic-html5-elements/>

These two websites helped me understand the semantic elements and why they are required in today’s atmosphere of internet usage.