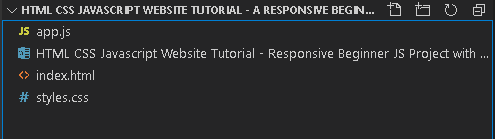
***HTML CSS Javascript Website Tutorial***

***Responsive Beginner JS Project with Smooth Scroll***

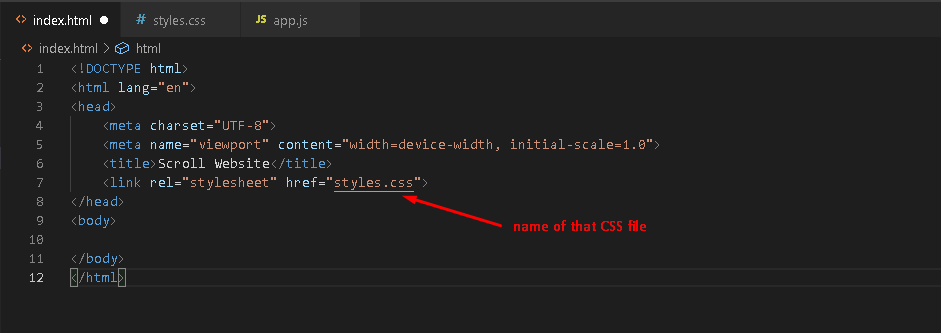
***CREATING NAVIGATION BAR***

Start by creating a file in your folder. Create index.html, styles.css and app.js.



Go to index.html, on short cut press shift + 1 = ! to bring a huge block of html.

Let’s link to style sheets so,

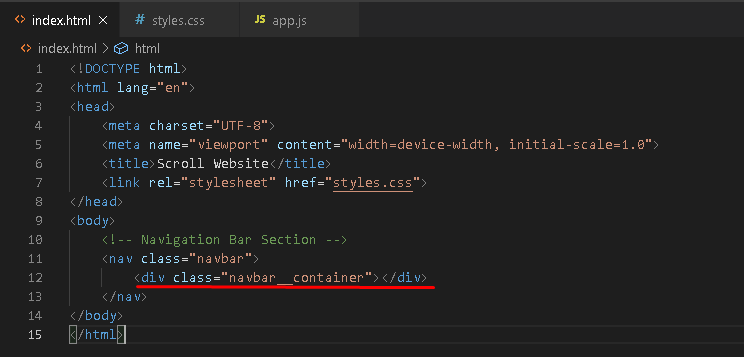


Then let’s go to our body section and go ahead and start creating actual website design. So, first thing first, I’m going to create navigation bar. So,

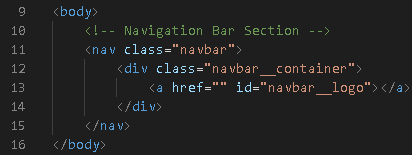


Just go ahead and type nav dot and name of your class, you can see a suggestion. Note that to add a comment just press ctrl + slash. Feel free to name your class.

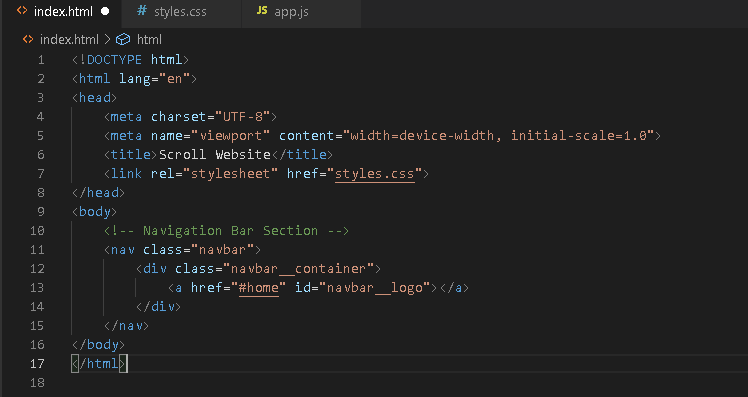
Now, let’s create navbar container, so, type dot navbar double underscore then container,



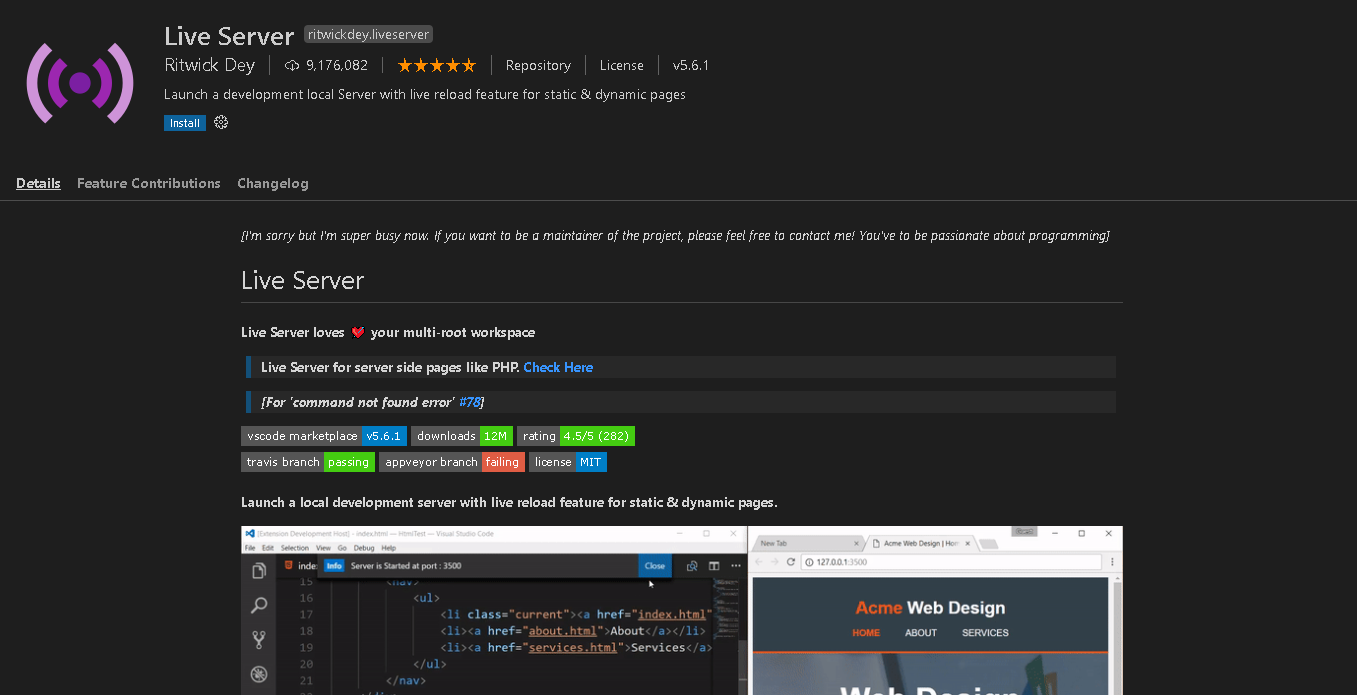
Its automatically create div. Now let’s create <a> text. So, to add <a> text type “a#nameofyourId”.



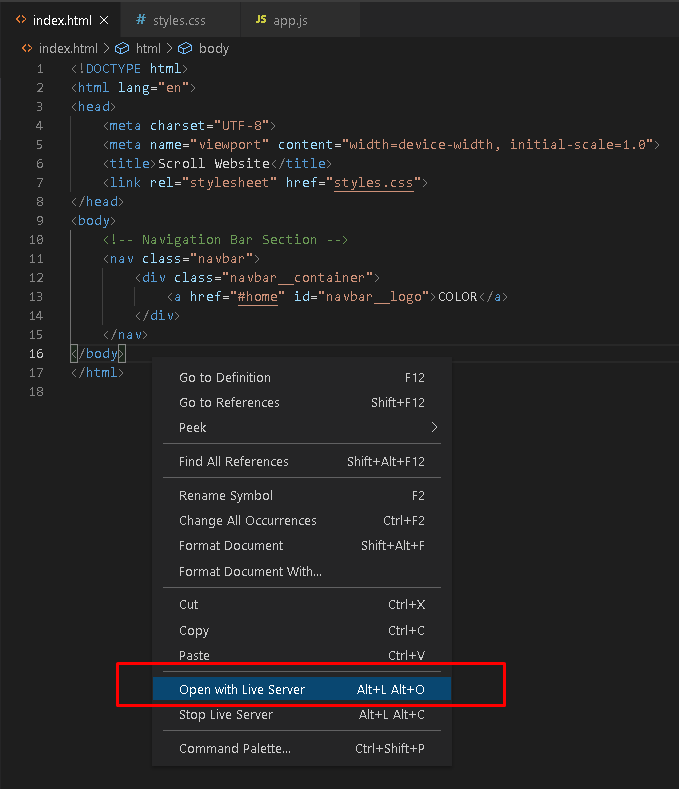
Then for the href, since we’re using html css this is the easiest way to do the little scroll effect.



Then inside this <a> here, let’s say color, by the way, on extension search for live server then install it.

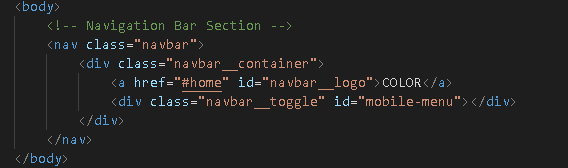


This is what it can do is when I right click

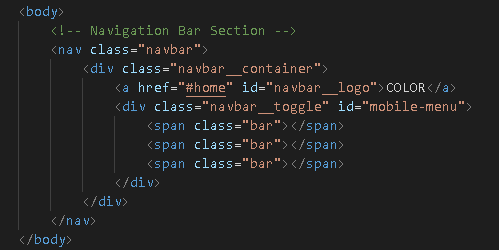


And also, it automatically updates if any codes we’re change. And now underneath this </a> (a closing remark for tags which will be learn on basic html) I will now create the actual navbar.

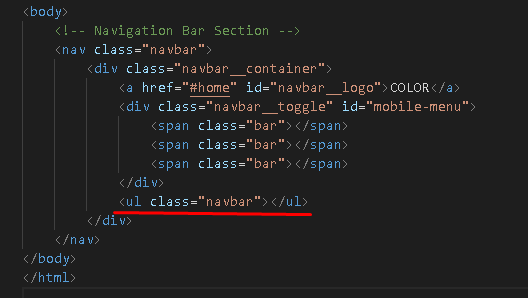
It will be a div so, dot navbar underscore 2 times toggle, it also has an id so add #.



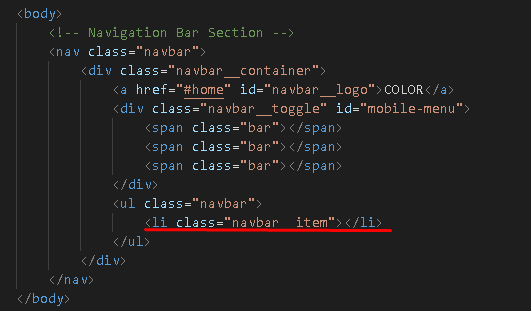
Under this dev here, we add span, so type, “span.bar” leave it empty then add 3 of this so the shortcut is shift + alt + down-arrow.



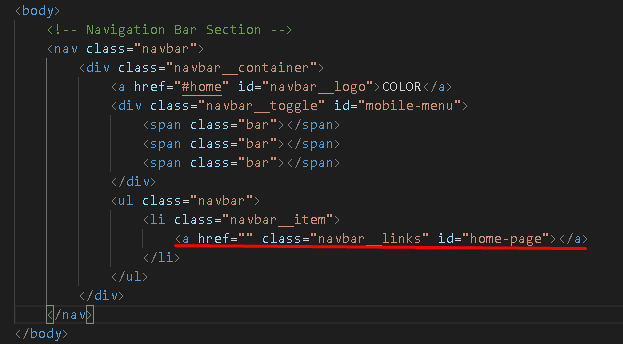
Under this div, now I will create the actual nav menu, using unordered list. So, type “ul dot navbar” always remember that dot is for clash and number sign is for id.



Under this <ul> is will be the place for the list of items so type “li dot navbar double underscore item”



Inside this <li> I will insert <a> with class name of navbar double underscore links and an id of home-page.



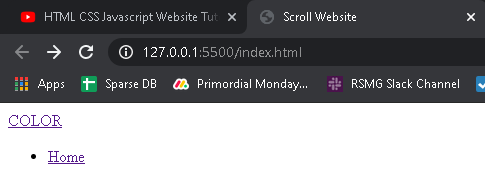
And we put href with #home,



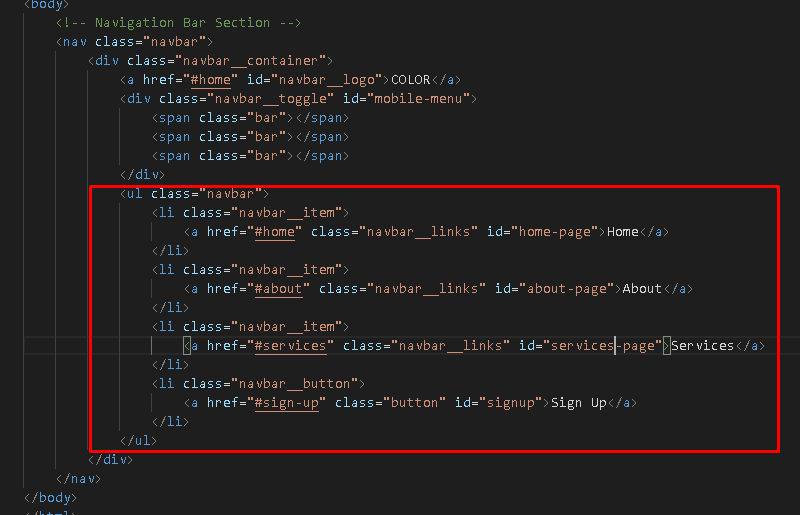
And inside this <a> let’s insert this home,

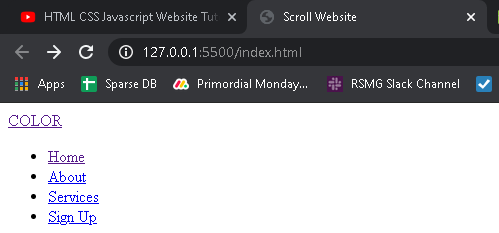


This is the view from the page,

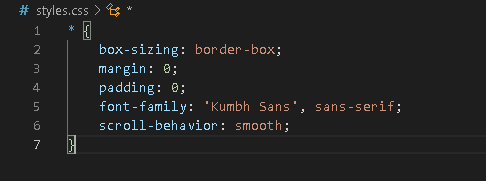


We just need to copy this <li>,

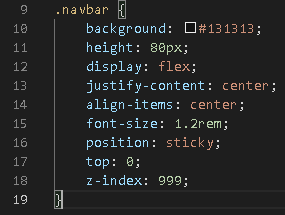


The view from the page,

***Styling our Navigation Bar***

Go to style.css, let’s start by adding this,

You’ve notice that all move to corner because we remove paddings and everything. So, start by dot navbar,



Background is the color and hashtag are the hex number of the color.

Height is very understandable.

Display is the main display of this characters and flex is for flexible in any size of the window.

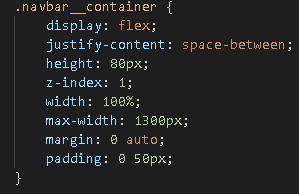
Justify-content just like MS Word, same with align-items.

Font-size is the size of fonts with the unit rem, default value of rem is n x 16.

Position is property specifies the type of positioning method used for an element and sticky is positioned based on the user’s scroll position.

Top value is zero so it’s the very top. Z-index make this thing above everything, think of it like a 3D type where it’s like sticking out then the value is ridiculous, no matter what I do with my site it’s above everything.

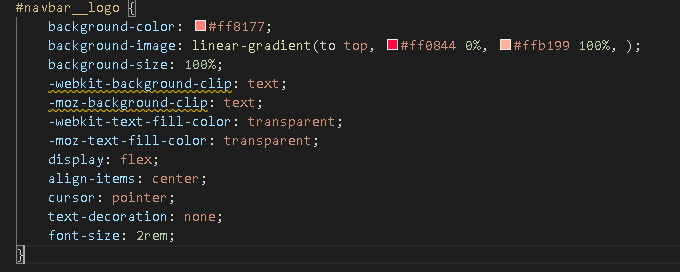
Then the navbar container



Justify-context space-between is going to add some spacing.

For margin sequence is top right bottom left. Top and bottom are just the same so its zero left and right are the same so we just say auto.

Next will be the navbar logo.



-webkit-text-fill-color this (for chrome and safari3) property specifies the fill color of characters of the text.

If this property is not specified, the value of the [color](https://www.w3docs.com/learn-css/color.html) property is used. Alternative for this for firefox browser is moz-text-fill-color.

The background-clip property defines how far the background (color or image) should extend within an element.

The cursor property specifies the mouse cursor to be displayed when pointing over an element.

The text-decoration property specifies the decoration added to text, and is a shorthand property for:

* text-decoration-line (required)
* text-decoration-color
* text-decoration-style

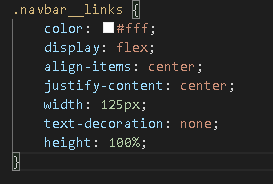
For .navbar\_\_menu



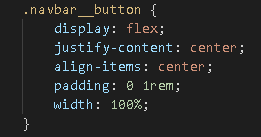
For .navbar\_\_item



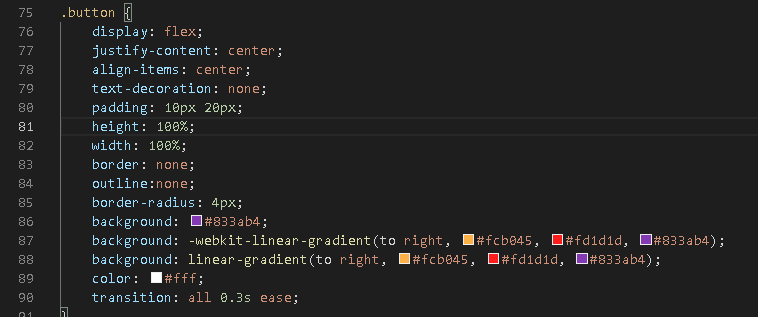
For .navbar\_\_links



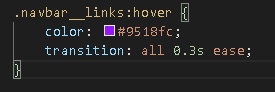
For .navbar\_\_button



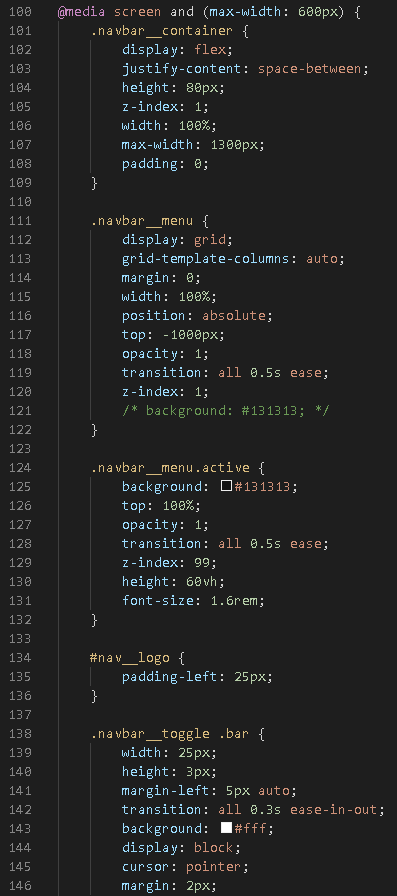
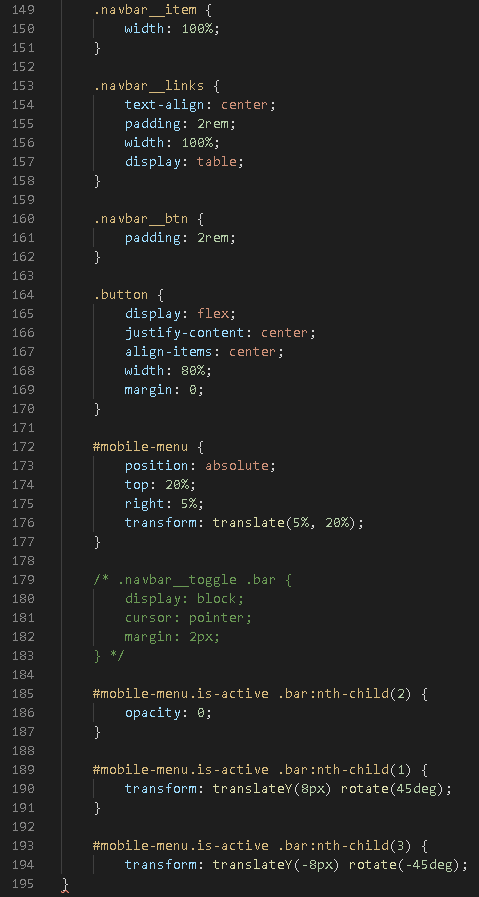
For .button



For .navbar\_\_links:hover



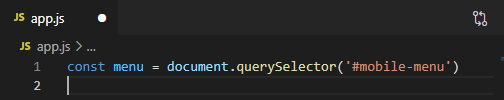
For @media screen and (max-width: 960px), The @media rule is used in media queries to apply different styles for different media types/devices.

Well, you don’t need to memories all of this css property. Take a look on this line of code inside media.

***Toggle Menu with JavaScript***

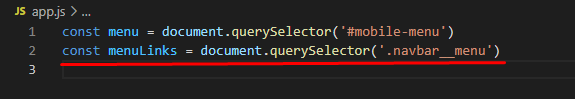
Go to app.js file, let’s target the first thing.

The keyword const does not define a constant value. It defines a constant reference to a value because of this, we cannot change constant primitive values, but we can change the properties of constant objects.

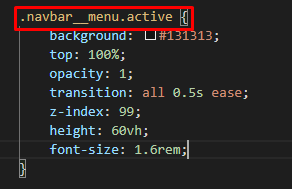


Const (name), you can name it whatever you want. Also I’m using query selector instead of id or document.getclass etc. because I can never always switch it if I have a class in my html, I can switch it easily just by putting a dot instead of a thing and we are targeting a mobile menu which we set by id on our html.

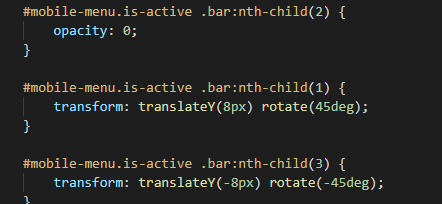
Then I want to target the actual navbars the <ul> let’s say,



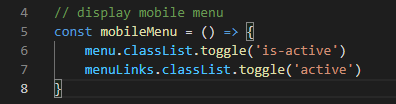
Then we can actually toggle this, so what we do now we have to write a function that adds a class that we have either active or is active which is the ones we have in our css,



This one that we want to trigger for the menu and then also for the actual little animation effect,



So how we going to do that, so I’m going to create an arrow function so in order to do that you simply say,



Const then name of you function (the one you name like menu and menuLinks) so we say menu.classlist.toggle so essentially its like true or false, on or off, think of it like active and not active like a switch and for this one I want to target the class that we put which is called ‘is-active’ the same with menulinks.

Then how do we even toggle this, well you have to actually add an event listener so I’m going to say,



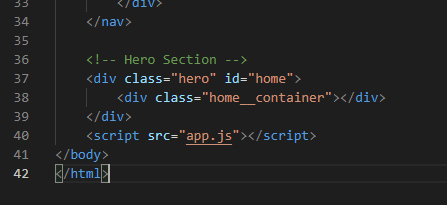
Then we import script tag to link our html and js. (right before the closing </body> tag)



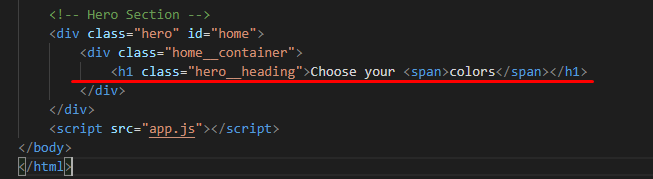
Then you can check your page if that’s work.

Creating Hero Section

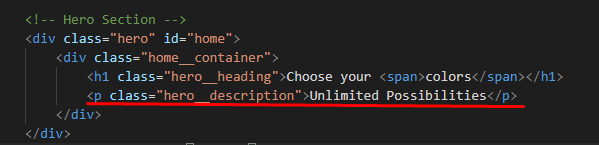
Under </nav> end tag,



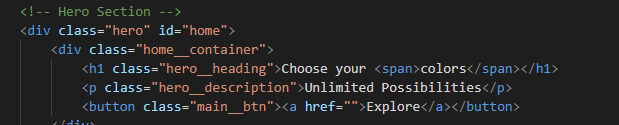
Then let’s put a hero heading, inside <div> class home container,



And then I will insert <p> tag after <h1>

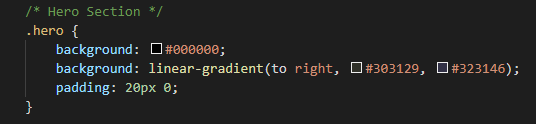


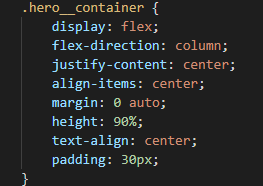
Under the <p> tag we insert <button> with the class of main\_\_btn, then inside this <button> tag I will insert <a> tag,

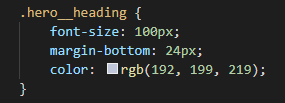


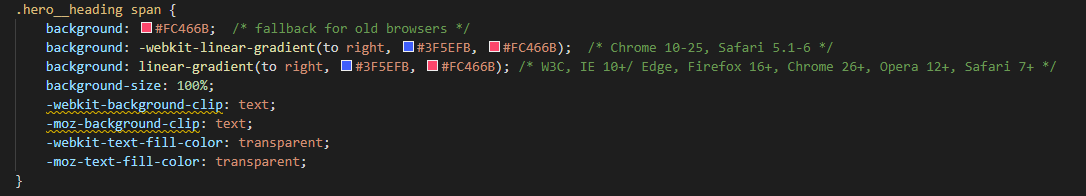
Styling Hero Section

We add a line comment section just to help us not to get lost.

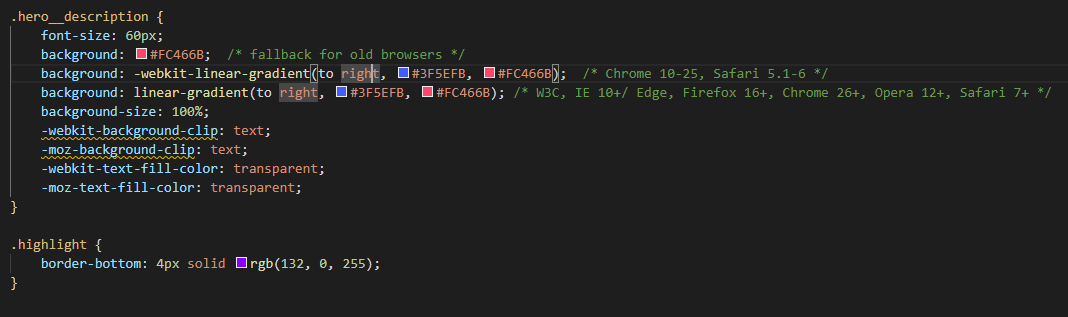




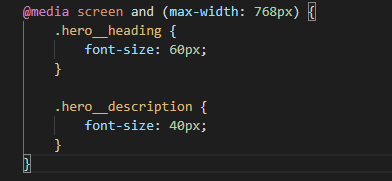




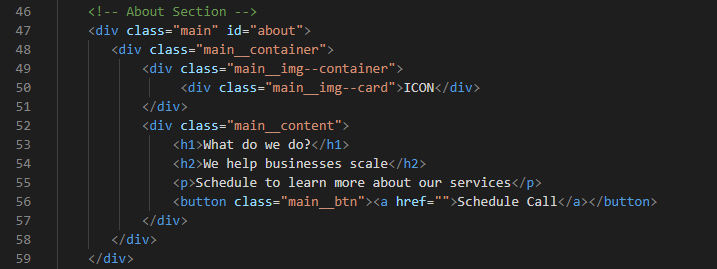
\*Note that these gradients are copied from a gradient website that can be found on google search.



Class highlight is the design when we trigger something in hero section. We’ll design that on JS.

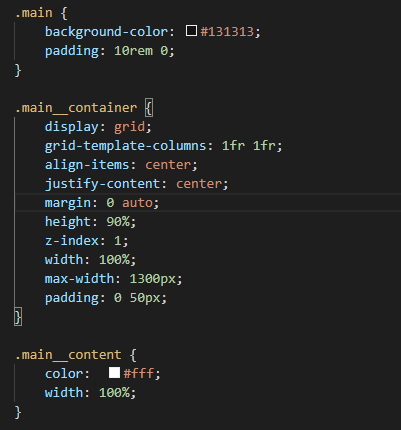


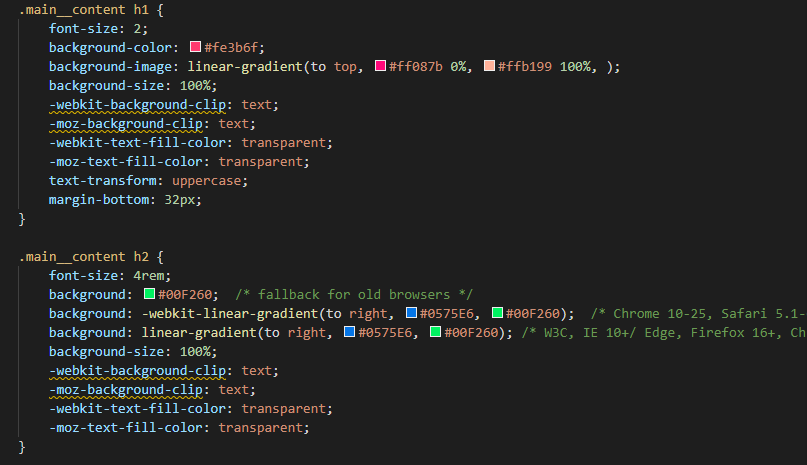
Creating About Section

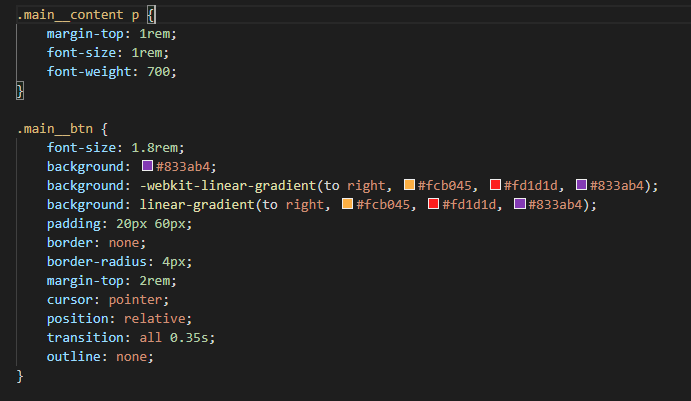


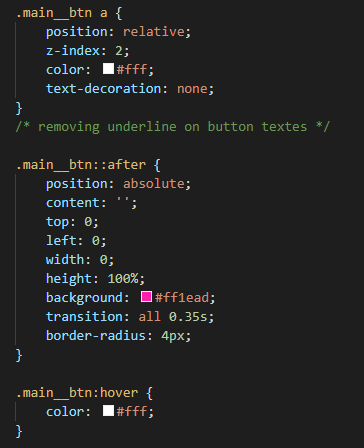
Styling About Section

Creating the Style in About Section, we’ll always habit to put some comments to let you navigate easier.



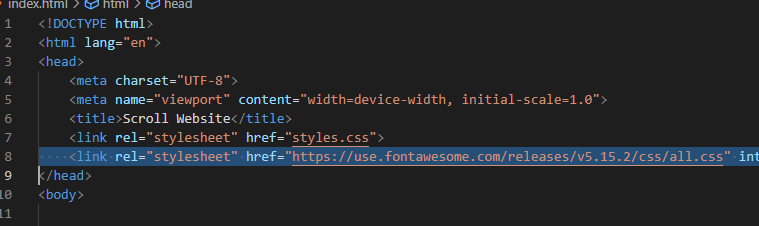




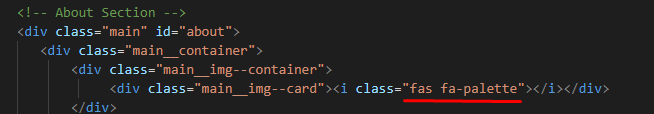




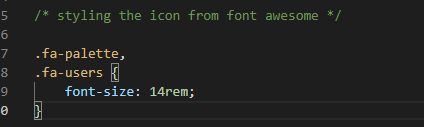
Get the icon on Font Awesome. Now go back to index.html, then under <link> tag, paste the cdn file from font awesome.

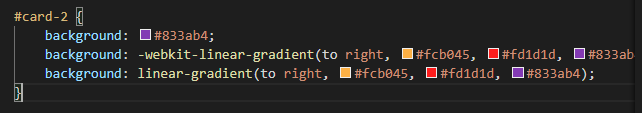


Now, under About Section, on the line which icon belongs, remove the icon string and paste the link from icon that came from font awesome.

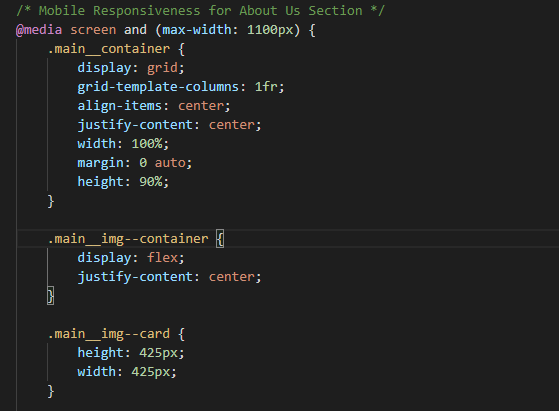


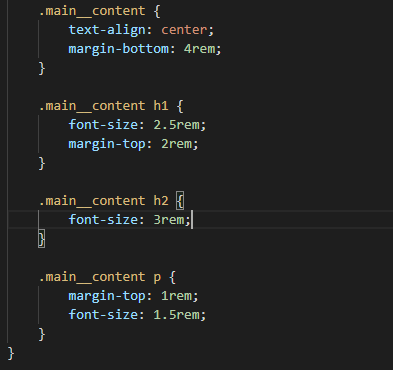
Then back to style.css,

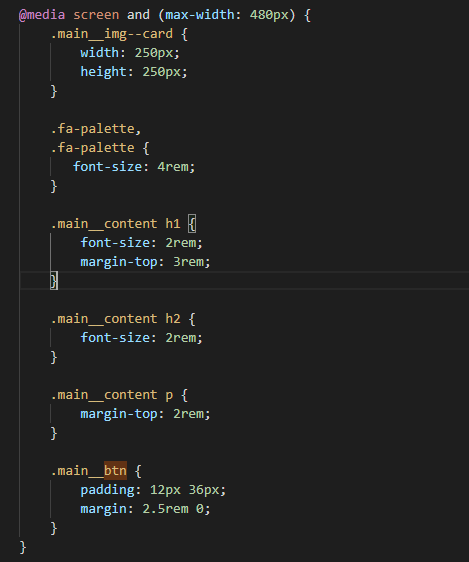




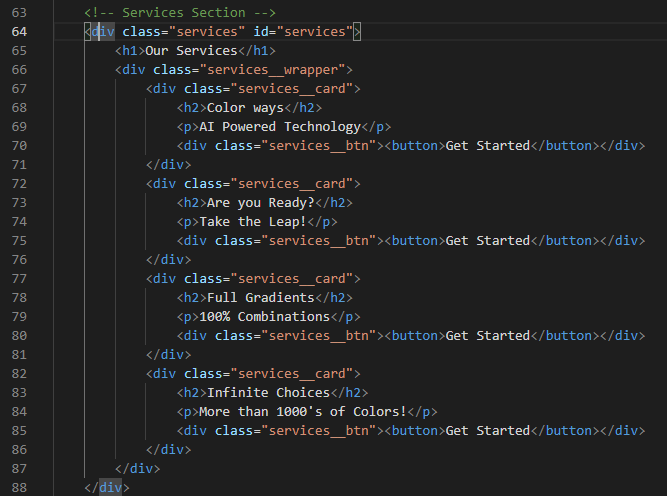
This ID can be used later, it’s not yet on index.html but it will be used later.







Creating Services Section



Styling Services Section