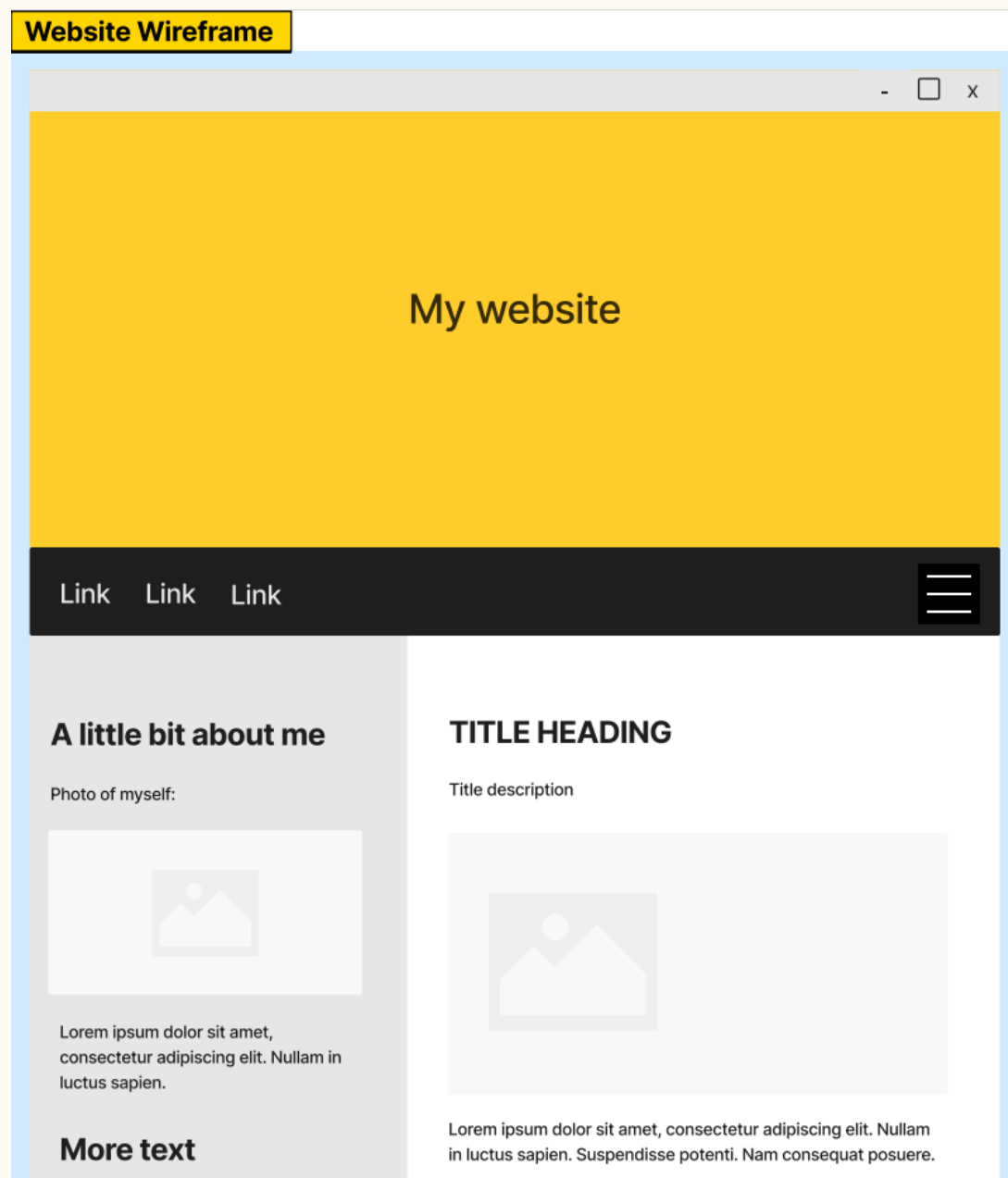


DI BOOTCAMPS WEB DEVELOPMENT EXERCISE SHEET

1. Header - hero image

Hero image is a web development term for a large image usually at the top of the webpage (header) with text and sometimes other elements like buttons.

Client wireframe (objective):

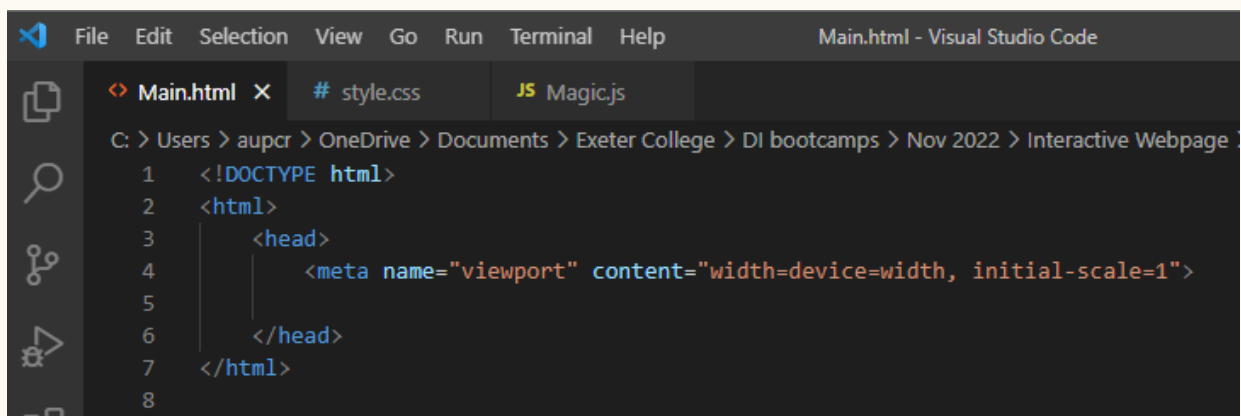


HTML

After setting up the basic HTML to load our page:

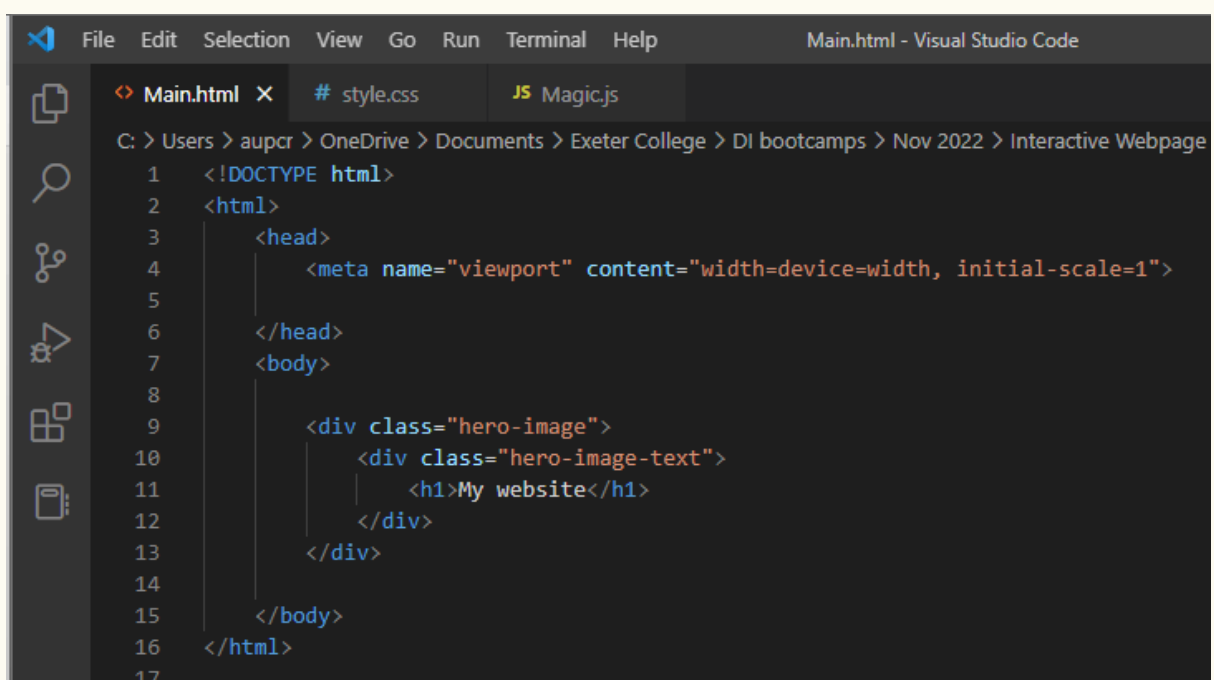
HTML element <>	Description
<!DOCTYPE html>	Tells the browser the file is an HTML 5 format
<html>	Top-level element, all other elements after are descendants.
<head>	Stores meta information and link to CSS file.
<meta name=viewport>	Optimises webpage for various viewport sizes.
<body>	Elements inside are visible to the user.

Our code file should look something like this:



```
File Edit Selection View Go Run Terminal Help Main.html - Visual Studio Code
Main.html x # style.css JS Magic.js
C: > Users > aupcr > OneDrive > Documents > Exeter College > DI bootcamps > Nov 2022 > Interactive Webpage
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta name="viewport" content="width=device-width, initial-scale=1">
5   </head>
6 </html>
```

To create our hero image, we are going to create a div inside another div, a div is a container to hold and position elements on a page using a block like system. We give our divs an identifier (class="name") so that our CSS can know which element in our html to style.



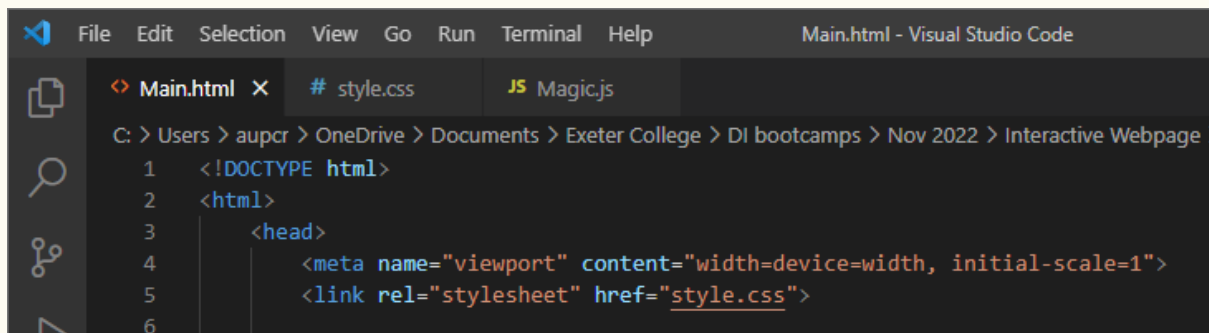
```
File Edit Selection View Go Run Terminal Help Main.html - Visual Studio Code
Main.html x # style.css JS Magic.js
C: > Users > aupcr > OneDrive > Documents > Exeter College > DI bootcamps > Nov 2022 > Interactive Webpage
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta name="viewport" content="width=device-width, initial-scale=1">
5   </head>
6   <body>
7     <div class="hero-image">
8       <div class="hero-image-text">
9         <h1>My website</h1>
10      </div>
11    </div>
12  </body>
13 </html>
```

CSS

Before we start adding our CSS code, we need to make sure our HTML file is linked to it as they exist in separate files. Just after the opening `<head>` tag (line 5 in this example) we add a `<link>` element with `rel` and `href` attributes.

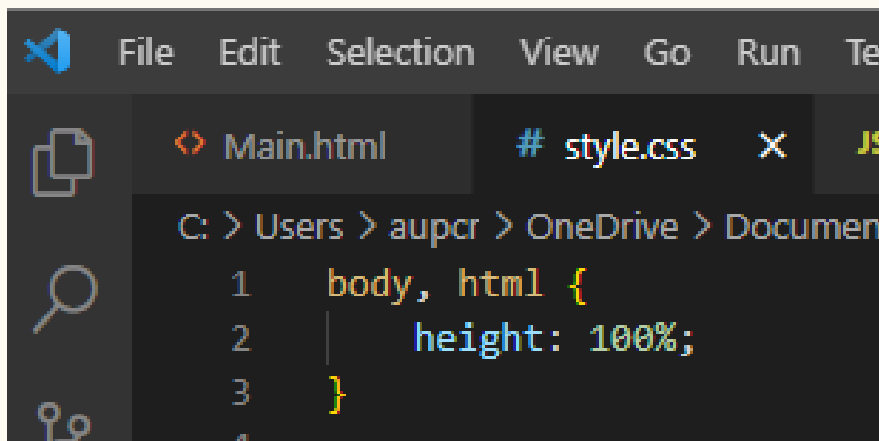
rel	Tells the browser there is a relationship between the files.
href	Https reference (href) includes the exact file to fetch.

The IDE should pick it up as you start typing the name of your CSS file press enter and use the arrow keys on the keyboard when prompted to autofill.



```
File Edit Selection View Go Run Terminal Help Main.html - Visual Studio Code
Main.html x # style.css JS Magic.js
C: > Users > aupcr > OneDrive > Documents > Exeter College > DI bootcamps > Nov 2022 > Interactive Webpage
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta name="viewport" content="width=device=width, initial-scale=1">
5     <link rel="stylesheet" href="style.css">
6
```

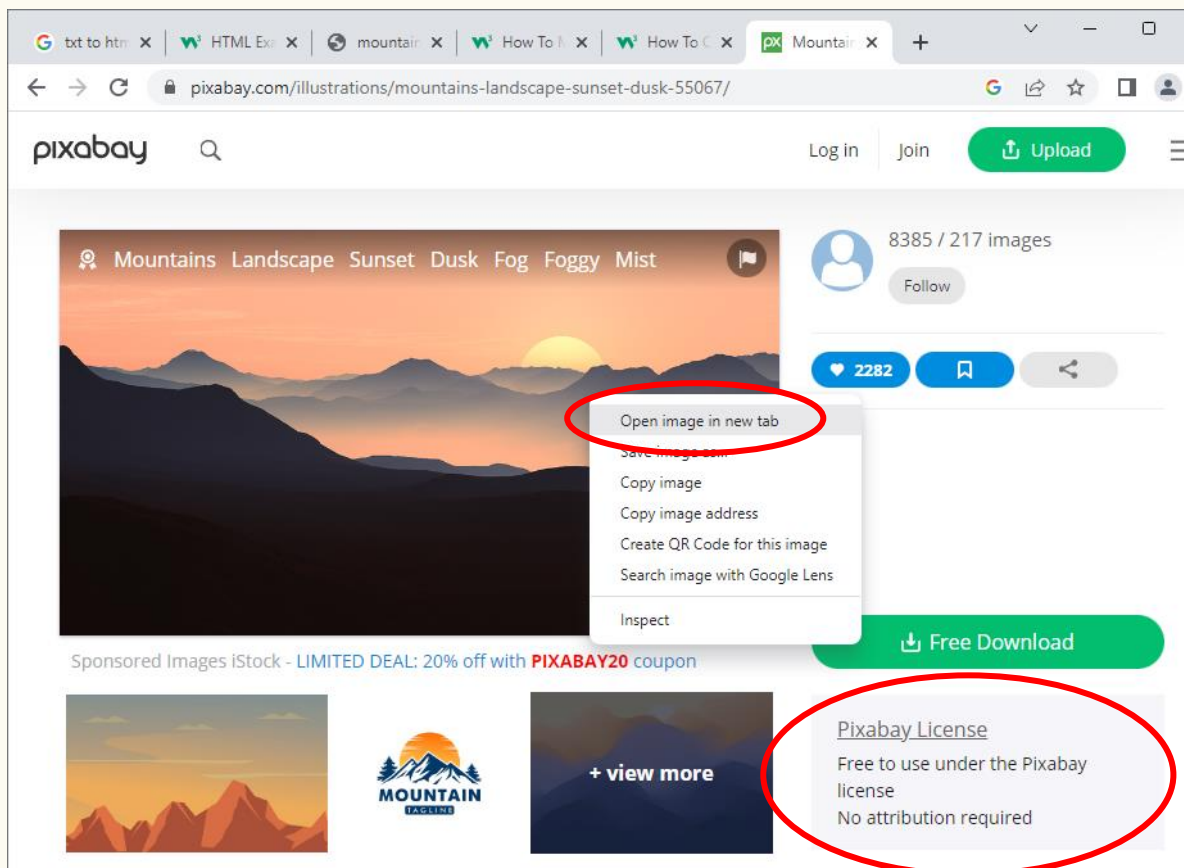
Now, its time to switch tabs into our CSS file. First let's adjust all html body content to fit the page nicely. CSS uses tag id followed by curly brackets to apply styling.



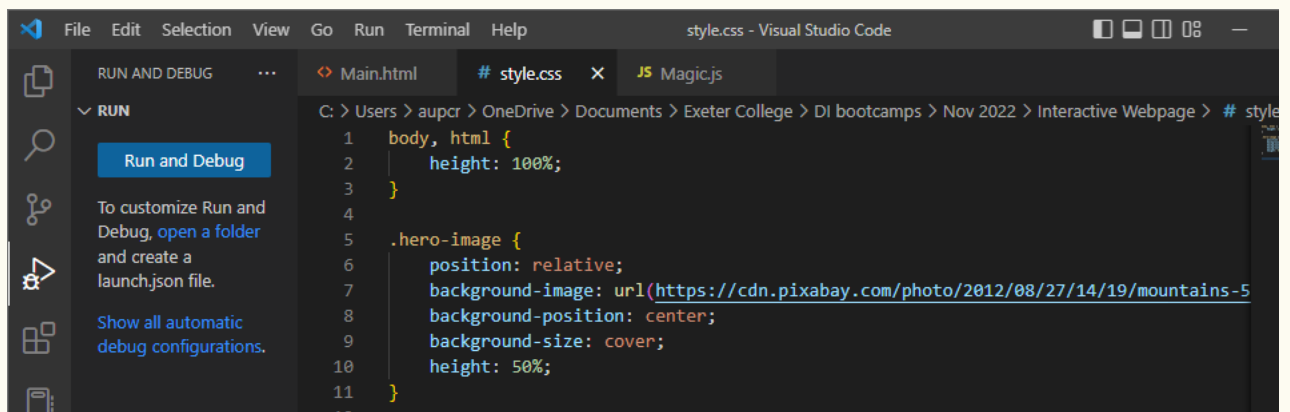
```
File Edit Selection View Go Run Ter
Main.html # style.css x JS
C: > Users > aupcr > OneDrive > Document
1 body, html {
2   height: 100%;
3 }
```

Image task:

Before we grab our divs and create our hero image header we need an image. Go online (Pixabay.com) and find a free-to-use image that we can grab a link from for later. When you find an image you like, you will need to right-click the image and 'open in new tab' then copy the URL to get the true images server destination.



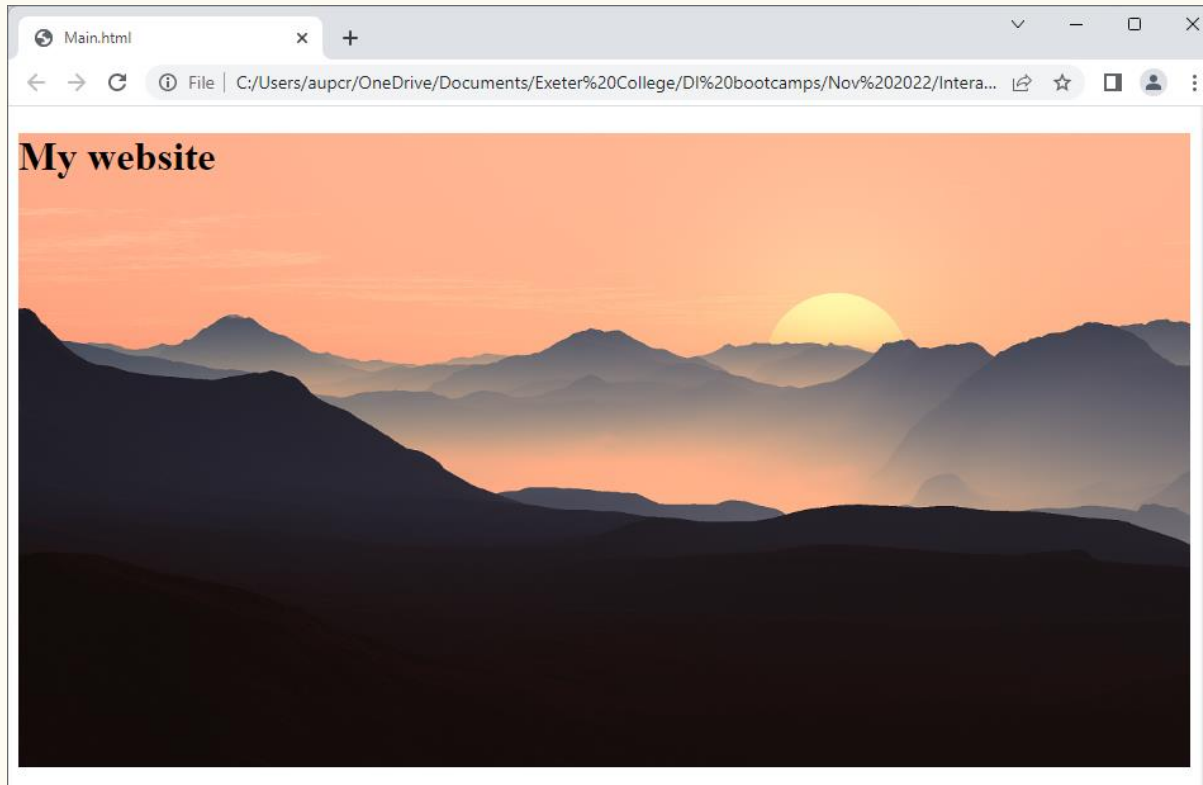
Now that we have our desired image link copied to the clipboard let's go back into VS Code and add the CSS to the Divs. Copy the code below.



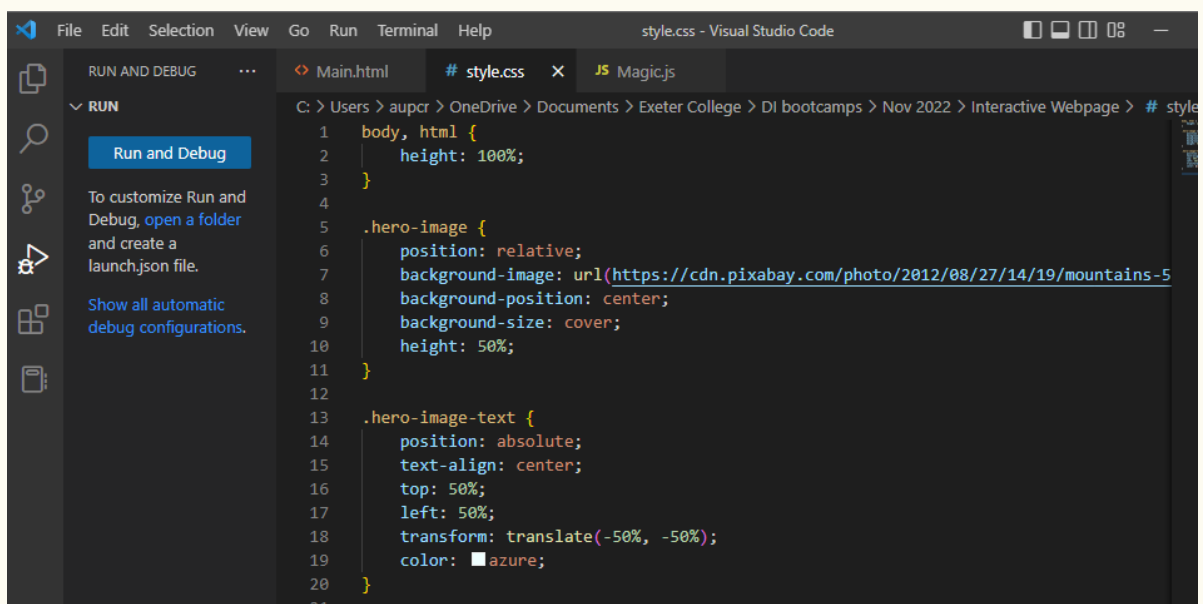
- We grab the div by typing a dot followed by div class name (ID) to start styling this is followed by curly brackets.
- Here we are changing the background image of the div to whichever image we fetch using the URL.
- We specify the position the background image to be centred and to cover the div.
- By specifying height in % we are referring to how much space the div will take on the viewport - **try experimenting with different percentages!**

Viewing our in-progress masterpiece!

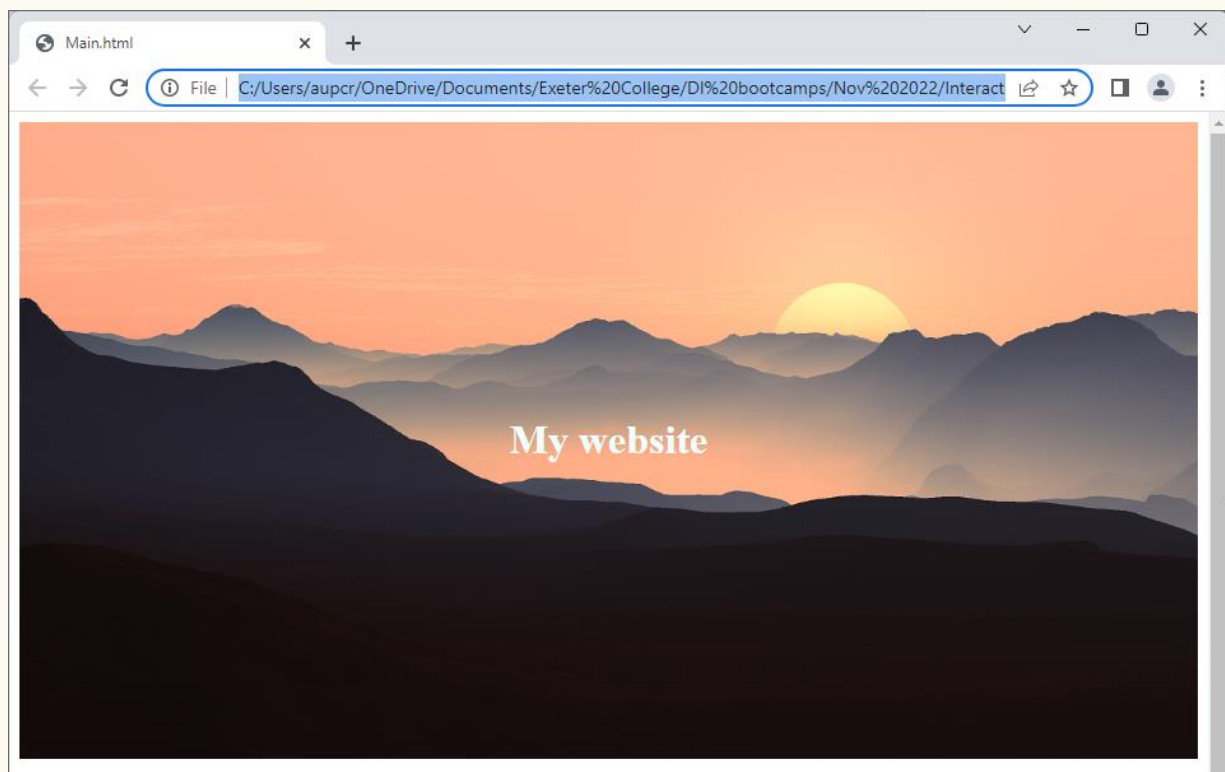
After adding the CSS above make sure both HTML and CSS files are saved (ctrl + s) and navigate back to the HTML tab in VS Code. Hit the run button at the top – followed by run without debugging to launch the browser and see what you’ve done!



It’s looking good but the text isn’t centred. To do this we need to consider positioning of the div containing the heading text. Going back to our CSS we grab the div and place the text in the middle of the image:



And voila!



Hero image text task:

- Change the font.
- Change the text color (HTML uses US spelling)
- Change the text color using a HEX color code - #000000 - <https://htmlcolorcodes.com/>
- Make the text bold.

We have now completed the header section of our webpage. By now we should be warming up to how HTML and CSS work, in the next section we will add the navigation bar which will include some animated buttons.

2. Navigation bar

The navigation bar will essentially be a div that includes buttons, let's start with a single button first. HTML looks like this:

```
15
16     <div class="navigaton-bar">
17         <button class="button home-btn">Home</button>
18     </div>
19
20 </body>
21 </html>
22
```

Key point to note: The difference between using id or class attribute for an html element is that an id tag must only be used once to uniquely identify a single element, a class can include multiple elements. So why have we included two values in the class attribute?

If we click run and we should now see our button. Doesn't quite fit with the style we are going for though so let's change that! In our CSS let's set a default style to all our button elements - using the first class name "button".

```
21
22 .button {
23     background-color: #f3974e;
24     border: none;
25     border-radius: 8px;
26     color: black;
27     padding: 16px 32px;
28     text-align: center;
29     text-decoration: none;
30     display: inline-block;
31     font-size: 16px;
32     margin: 4px 2px;
33     transition-duration: 0.4s;
34     cursor: pointer;
35 }
```

Quick research task:

Take a look at the CSS properties above and any that you haven't come across before search the web and make a note of what they do. By understanding transition-duration we can predict what we are going to do to make our button interactive.

We can animate using CSS. We can overwrite our default button layout by specifying the second class name we gave our button, in this case "btn-home". Feel free to try the code below out, but I would advise experimenting with the CSS properties to better fit your chosen hero image.

```
36
37 .home-btn {
38     background-color: black;
39     color: azure;
40     border: 2px solid #f3974e;
41 }
42
43 .home-btn:hover {
44     background-color: #ebb283;
45     color: azure;
46 }
47
48 .navigaton-bar {
49     background-color: black;
50 }
```

How have we achieved the animation? The html button element supports things called pseudo-elements such as :after, :before and :hover which we can use to further style our buttons.

A great website for finding interactive webpage elements is codepen;
<https://codepen.io/trending>

Button task:

- Using codepen, add the source code to animate and style your buttons (html, css).
- Create two additional buttons. (Projects, about me for example).

Now let's get add some JavaScript to enable our buttons to take us to another webpage (link).

```
<div class="navigaton-bar">
|   <button class="button home-btn" onclick="location.href='https://exe-coll.ac.uk/';">Home</button>
</div>
```

Here we have added an onclick event which listens for when the user clicks the element it is embedded into, in our case the button. "location.href=" is a JavaScript function which is used to set the current webpages URL to that specified, in my case the Exeter College website.

Why not include the URL address to your blog home page? Modify the JavaScript code above to do this.

Burger menu challenge task

Now that we have the navigation bar set up with three buttons it is time to add the final piece - a burger menu. Your final task is to add an animated burger menu to fit the client design. I recommend using some great resources such as:

[How To Create a Menu Icon \(w3schools.com\)](https://www.w3schools.com/html/html_burger_menu.asp)

[How To Create a Collapsible \(w3schools.com\)](https://www.w3schools.com/html/html_collapsible.asp)

There's a really cool HTML and CSS only burger menu you should check out here:

[Fullscreen Menu Enter \(codepen.io\)](https://codepen.io/FullScreenMenu/pen/Enter) .