



[Home](#) [Menu](#) [What's New](#) [Contact](#)

It's not just Coffee It's **Starbucks**

Lorem ipsum dolor sit amet consectetur
adipiscing elit. Laboriosam a iste dolorem
sapiente eius culpa facilis beatae harum nemo.
Nobis fuga voluptates quisquam perspiciatis
molestias magni obcaecati facere, illum
dignissimos.

[Learn more](#)



Webpage Coding Project



Table of Contents

Overview	2
Activities	2
Downloading the Source Code and Previewing the Project	3
Finding Your Source Code	3
Setting up your project	5
Webpage Layout: Title	7
Personalizing your web page Title	7
Beautifying your web page with Styles	8
Webpage Layout: Header	11
Webpage Layout: Content.....	16
Webpage Layout: Bottom Section & Logos.....	21
Animate your webpage.....	21
Sharing your webpage	23
Webpage Layout: Circle	25
Webpage background theme	25
JavaScript: Webpage interactivity.....	27
Change Background Color	28
More Coding Fun	31
Learn More Button	31
Add HTML Links & Pages	31
Add New Styling and Content.....	31
References	32
HTML Definitions.....	33
CSS Unit and Property Definitions (and links)	34



Overview

This activity uses coding languages *HTML*, *JavaScript*, and *CSS* to design and create a stylish, dynamic webpage with numerous potential applications.

You will learn and demonstrate the coding skills to create the Starbucks beverage advertisement shown.

Previewing the project prior to the coding activity session is encouraged. The solution is located on the github repo link: <https://github.com/naiyapatel29/100GirlsOfCode>

Prerequisites: Internet Access, Browser Capabilities (Microsoft Edge, Chrome, etc.)

Activities

- Download Source Code
- Create Webpage layout (5 major sections)
 - Header
 - Content - Text Box and Img Box (image)
 - Bottom-Section
 - Logo-Section
 - Circle
- Create HTML elements within each section
- Create CSS webpage styling
- Create Interactivity with JavaScript



Downloading the Source Code and Previewing the Project

Finding Your Source Code

1. Go to the github repo site by clicking on the below link. See Figure 1.
<https://github.com/naiyapatel29/100GirlsOfCode>

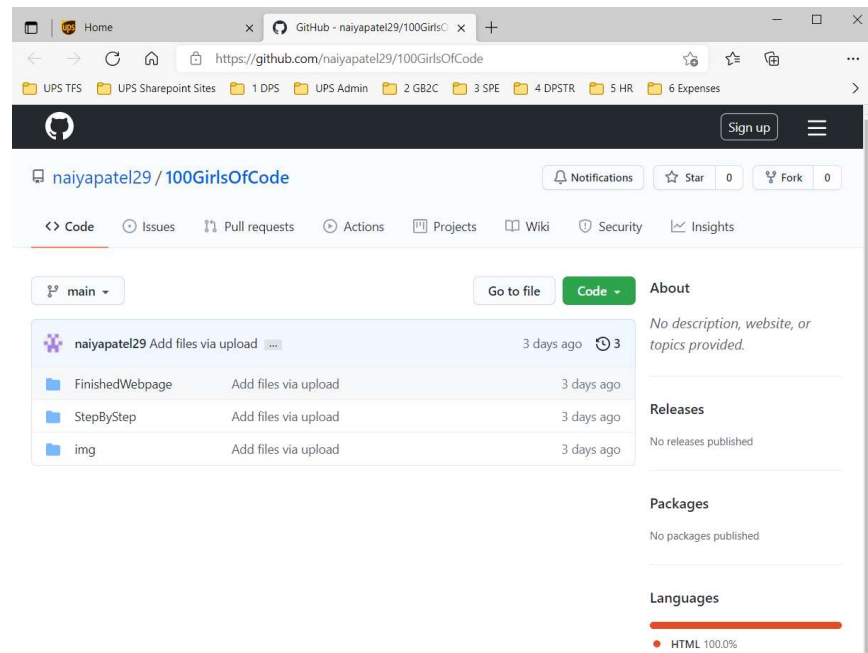


Figure 1

2. Select the dropdown menu from the green Code button and select Download ZIP. Shown in Figure 2.

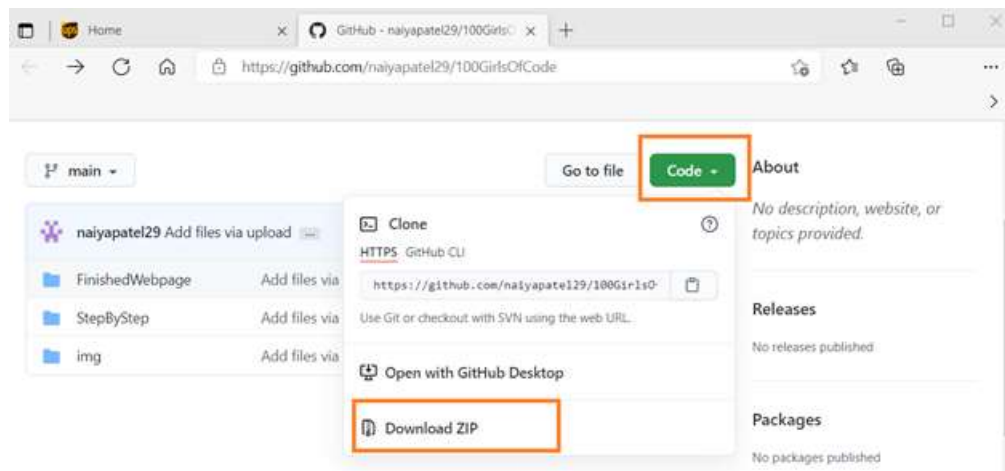




Figure 2

3. Within the *Downloads* folder, find and download file: **100GirlsOfCode-main.zip**
4. Move the **100GirlsOfCode-main.zip** file to the Desktop (for easy access)
5. (Optional depending on Zip application) Right Click on the **100GirlsOfCode-main.zip** to unarchive the file
 - a. Select WinZip
 - b. Select Unzip to here
 - c. A folder named 100GirlsOfCode-main is created on your Desktop
6. Open the Folder and verify three subfolders and its contents
 - a. **Img**: this folder contains the 10 images required for the webpage
 - b. **StepByStep**: this folder contains 9 subfolders; each subfolder contains an index.html file.

The index.html file comprises of the sample solution for how your code should look like after completing each step
 - c. **FinishedWebpage**: this folder contains 11 files, including the completed webpage index.html
7. Preview the webpage by opening the **FinishedWebpage** folder and double clicking the `index.html` file.
 - a. Observe the web page features by hovering over each beverage image and clicking on each one.

Note: This webpage is designed to automatically tailor to different screen sizes by using relative units instead of absolute units.

Setting up your project

Prerequisite: Complete the Project Setup steps + Downloading the Source Code

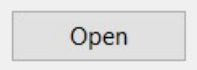
1. Log into your W3Schools Space using <https://spaces.w3schools.com>
2. Open your Space. Click on the "Number of files" number as shown in Figure 3.

Space	Requests	Unique visitors	Data served	Number of files	Storage used
 naiya https://naiya.w3spaces.com	341	9	10.72 MB	14	1.47 MB

Figure 3

3. Add the images required for the project by uploading the **img files** from the source code that we downloaded earlier.

- a. Click on the Upload Files Button 
- b. A File uploader screen is displayed
- c. Navigate to the folder 100GirlsOfCode-main\Img folder on your Desktop
- d. Shift + click on all 10 images, see Figure 4

- e. Click on the "Open" button 

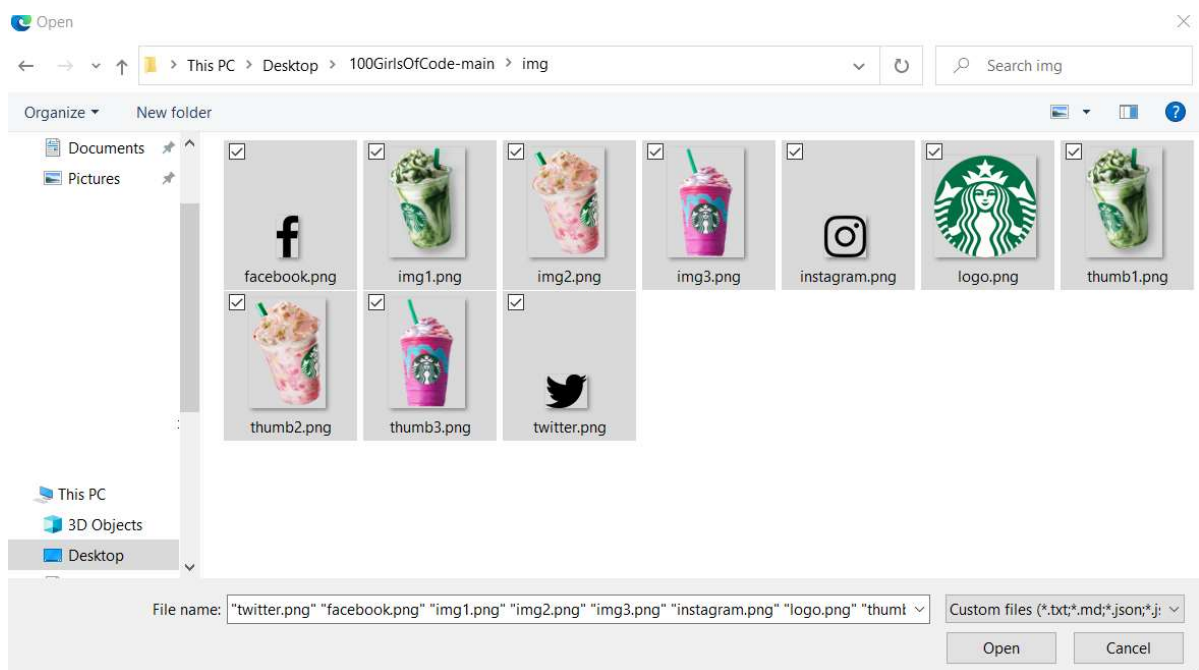



Figure 4

4. Confirm all 10 image files had been successfully uploaded to your Space. The 10 files are displayed in Figure 4 above. The file names are:
 Facebook.png, Img1.png, Img2.png, Img3.png, Instagram.png, logo.png, thumb1.png, thumb2.png, thumb3.png, twitter.png
5. Navigate to the `Index.html` file and click on the Edit button, as shown in Figure 5. This opens the web editor as seen in Figure 6.

During code development, we will always keep this file open, and as we add code, we will periodically save our code by clicking on the Save* button.



Figure 5

6. Deleted the default code generated by Space in between the `<head>` and `<body>` sections as noted in Figure 6. Click the save button .

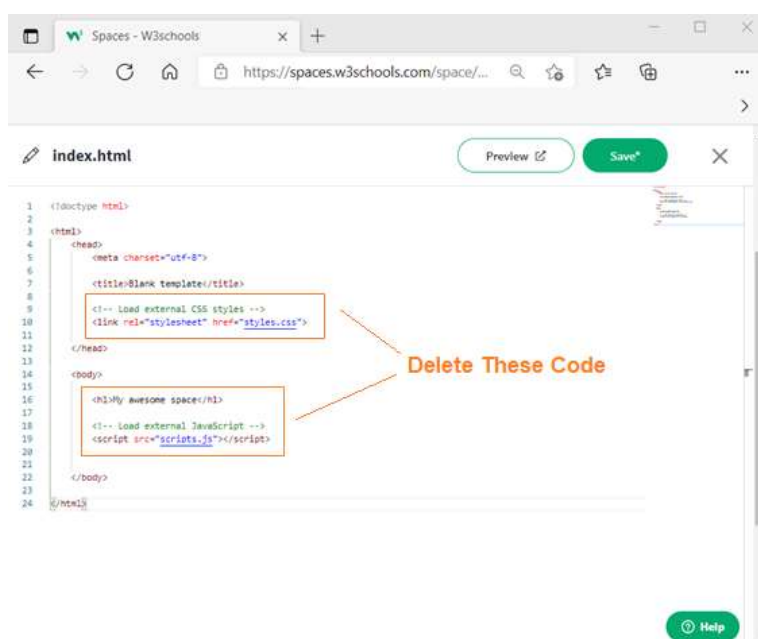


Figure 6



7. The `Index.html` file generated by the web editor should already contain the structure of the html body. If you do not see this structure, copy and paste the below code snippet into your web editor and save.

```
<!DOCTYPE html>
<html>
  <head>
  </head>
  <body>
  </body>
</html>
```

Now that we have our project set up, let's start coding!

Webpage Layout: Title

Personalizing your web page Title

1. Navigate to your `index.html` file [edit view] by clicking on the `index.html` row from your Space as shown in Figure 7.

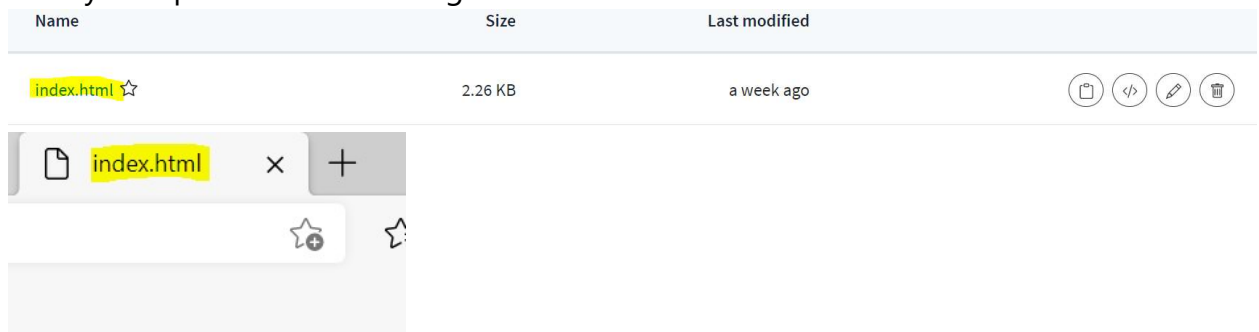


Figure 7

2. Personalize your webpage title by navigating to the `<head>` code block. Change the name of your title to "Starbucks" in between the tags `< title>` `</title>`, as seen in the snippet below:

```
<title>Starbucks</title>
```


3. Save your file.
4. Refresh your browser and preview your page. Verify the name "Starbucks" is the title tag.

Preview the file in your browser. It will display the tab name of your webpage, Starbucks, as shown in the screenshot (Figure 8).

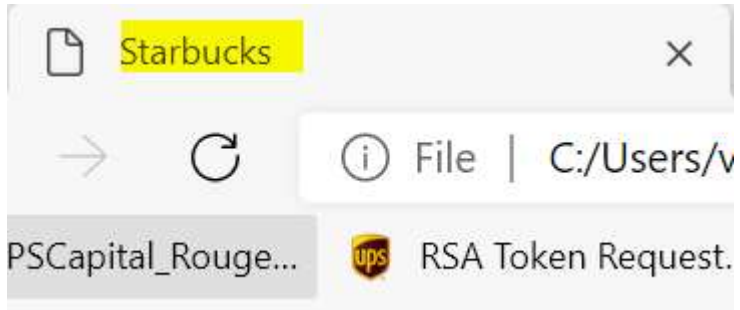


Figure 8

Beautifying your web page with Styles

Styles allows your artistic flair to shine through. Start your design with the basics, such as your favorite font type, font color, italicize/bold/underline, bulleted lists, padding, window sizing, etc. For this project, we will begin with standard inline styles, then move on to advanced styles.

1. Open your `index.html` file
2. Navigate to the closing `</body>` tag
3. Add an opening `<style>` tag and a closing `</style>` tag as shown in Figure 9 below.

Note: All your styling code will be placed between the `<style>` and `</style>` tags.

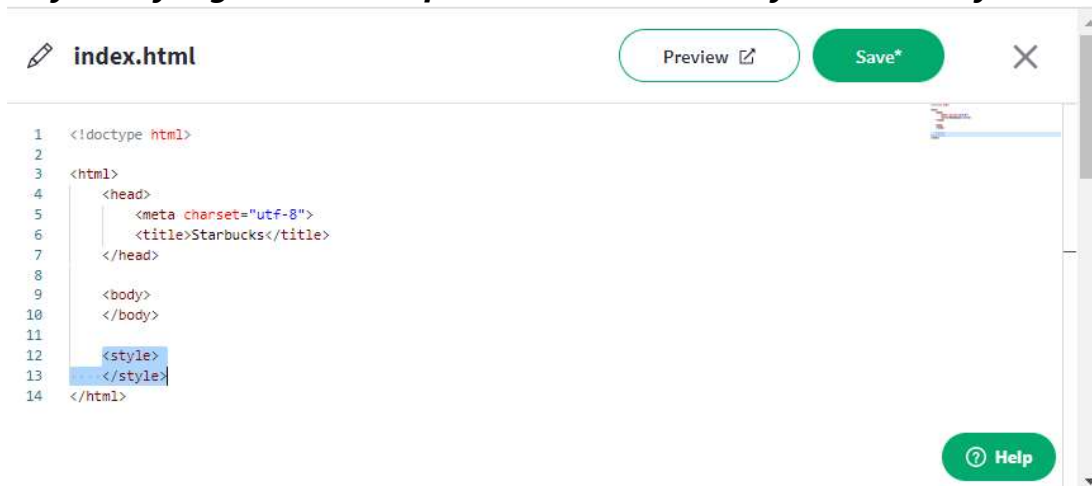




Figure 9

4. We will use **Poppins** font for this project. The font must be imported from a different URL location. Type the following import statement between the `<style>` `</Styles>` tags:

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;900&display=swap');
```

Define the boundaries where our elements will reside within the browser by using Box Sizing. **Box Sizing** specifies the width and height of how an element is calculated such as should the boundaries include padding and borders. This dynamically resizes a web page based on the browser size.

5. Type the following code after your import statement between the `<style>` `</style>` tags, then Save.

```
*{
  margin: 0;
  padding: 0;
  box-sizing: border-box;
  font-family: 'Poppins', sans-serif;
}
```

Add style to the `<body>` tag with the following specifications: a) Place the body elements **relative** to one another (display flex format), b) Specify the % unit for utilizing all the **width** available to fit the webpage content, c) Give **padding** around all element sides with 100 px and, d) Specify **min-height** of 100vh.

6. Type the following code after your Box Size code in between the `<style>` `</style>` tags, then Save.

```
body {
  position: relative;
  width: 100%;
  min-height: 100vh;
  padding: 100px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  background: #FFF;
}
```

Your style section now contains code for importing the Poppin font, defining the Box Size, and setting the body style for your elements. See Figure 10.

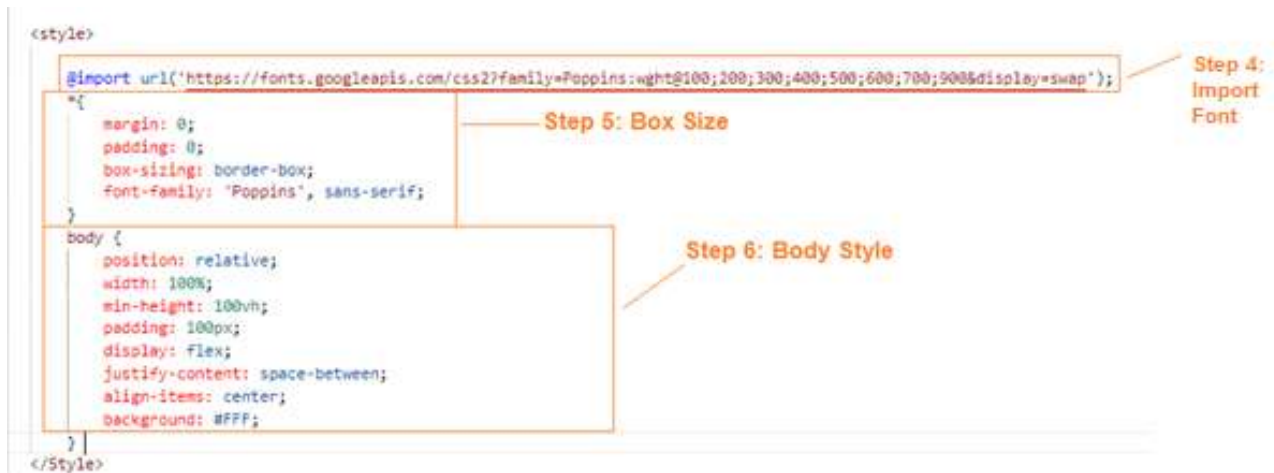


Figure 10

Webpage Layout: Header

A website header is located at the top of each page and serves a very important purpose. It provides a page for your logo and main menu items that makes it easy to navigate your website. This allows for a consistent user experience that all good websites share.

1. Open your `index.html` file, navigate to the `<body>` `</body>` section
2. Add the Header opening and closing tags `<header>` `</header>` into the `<body>` section
3. Inside the `<header>` tag, add our logo using the `` tag and provide the link for image as follows:

```

```

4. Inside the `<header>` section and below the logo, we will add Menu options using a HTML list using the code below:

```
<ul>

    <li><a href="#">Home</a></li>
    <li><a href="#">Menu</a></li>
    <li><a href="#">What's New</a></li>
    <li><a href="#">Contact</a></li>

</ul>
```

Upon completing this step, your code should look like Figure 11.

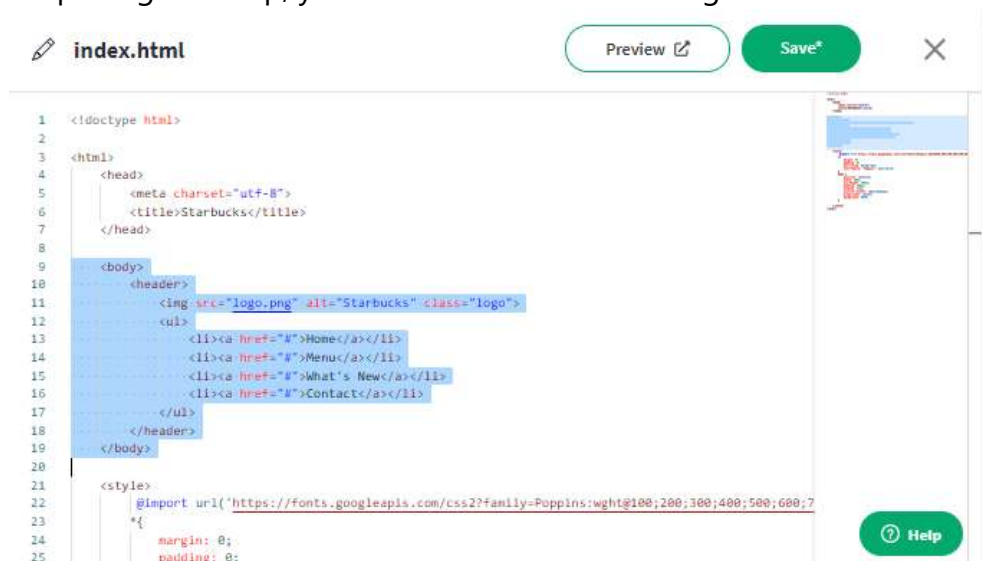


Figure 11

5. Save your file
6. Refresh and preview your webpage. The page should look like Figure 12 at this point:

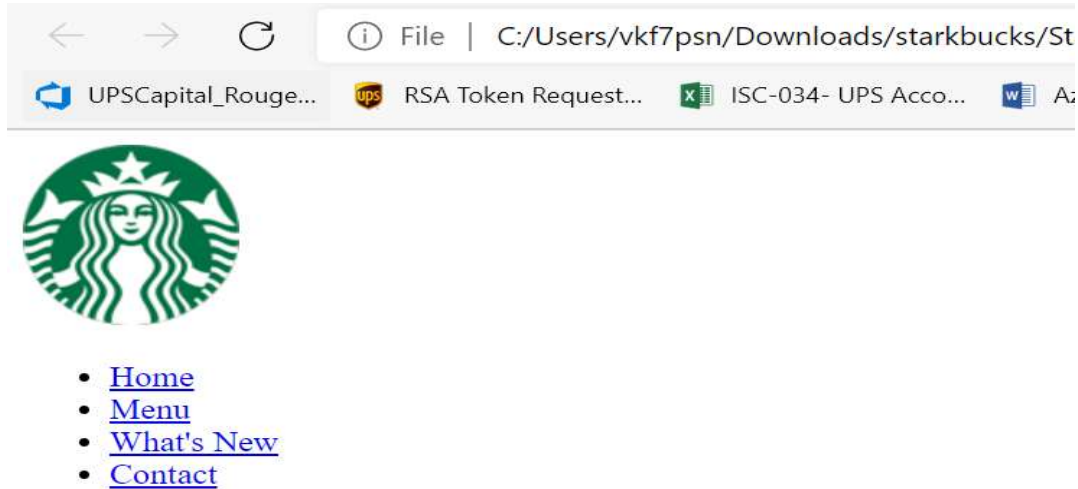


Figure 12

7. Add style to the header via the `<header>` tag where our logo and menu are located. We will use four popular attribute settings by adding the block of code below into our `<style>` section (below the Body Style).
 - a. `position: absolute` = allow the header to be fixed at the top of the webpage
 - b. `display: flex` = automatically aligns the header items on the webpage
 - c. `justify-content: space-between` = aligns the items in the header evenly with spaces
 - d. `align-items: center` = align the items in the center of the webpage

```
header {
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  padding: 20px 100px;
  display: flex;
  justify-content: space-between;
  align-items: center;
}
```



8. Add style to the Starbucks logo to prevent the logo from exceeding a defined width of 80pixels and position it relative to its <header> element using the code block below:

```
header .logo {  
  position: relative;  
  max-width: 80px;  
}
```

9. Horizontally align all menu items by setting the attribute "Display: flex". After adding the below code menu, your webpage will look like Figure 13. Note: Menu item are vertical by default with list element.

```
header ul {  
  position: relative;  
  display: flex;  
}
```



• [HomeMenuWhat's NewContact](#)

Figure 13

10. Black bullets are not part of our design and need to be removed. By default, the element applies the dot into its style. To remove it, set the list-style to none using the code below. The dot should now be removed as shown Figure 14.

```
header ul li {  
  list-style: none;  
}
```



[HomeMenuWhat's NewContact](#)

Figure 14



It's time for applying color and adding space in between the menu items.

11. In the below code, we added color to text margin and set each element to be displayed as a block. The code, `display: inline-block` allows you to set a width and height on the element. See Figure 15.

```
header ul li a {  
  display: inline-block;  
  color: #333;  
  font-weight: 400;  
  margin-left: 40px;  
  text-decoration: none;  
}
```



[Home](#)

[Menu](#)

[What's New](#)

[Contact](#)

Figure 15

12. Your `<style>` code section should look like Figure 16 below. Save your code.

index.html

Preview

Save

✕

```

21 <style>
22 @import url('https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;400;500;600;700;900&display=swap');
23
24 *{
25     margin: 0;
26     padding: 0;
27     box-sizing: border-box;
28     font-family: 'Poppins', sans-serif;
29 }
30 body {
31     position: relative;
32     width: 100%;
33     min-height: 100vh;
34     padding: 100px;
35     display: flex;
36     justify-content: space-between;
37     align-items: center;
38     background: #FFF;
39 }
40 header {
41     position: absolute;
42     top: 0;
43     left: 0;
44     width: 100%;
45     padding: 20px 100px;
46     display: flex;
47     justify-content: space-between;
48     align-items: center;
49 }
50 header .logo {
51     position: relative;
52     max-width: 80px;
53 }
54 header ul {
55     position: relative;
56     display: flex;
57 }
58 header ul li {
59     list-style: none;
60 }
61 header ul li a {
62     display: inline-block;
63     color: #333;
64     font-weight: 400;
65     margin-left: 40px;
66     text-decoration: none;
67 }
68 </style>
</html>

```

?

Help

Figure 16

Webpage Layout: Content

Adding Content to your webpage for your audience's reading pleasure

1. Open your `index.html` file, navigate to the `<body>` `</body>` section
2. Add the division (section) opening and closing tags `<div>` `</div>` into the `<body>` section following the Headers.
3. Inside the `<div>` element, add a class called "content" as seen in Figure 17.

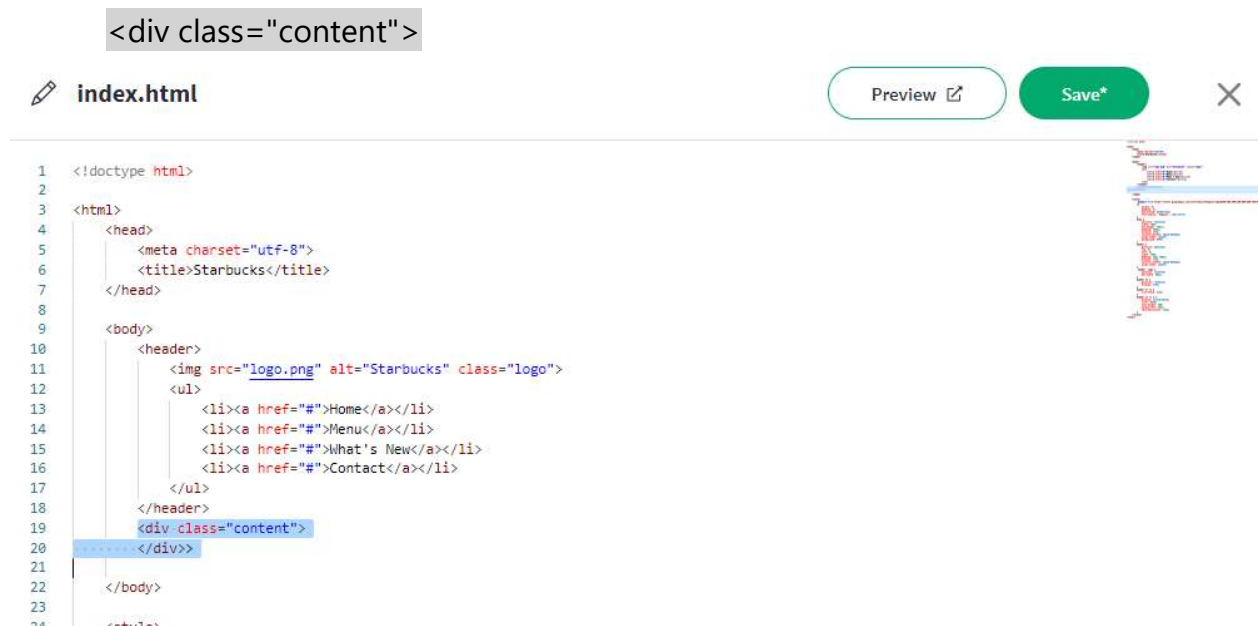


Figure 17

4. Add CSS (Cascading Style Sheet) for the `<div>` content section inside the `<style>` code block. This style will determine how the elements inside the content section will be displayed.

```

.content {
  position: relative;
  width: 100%;
  display: flex;
  justify-content: space-between;
  align-content: center;
}

```

5. Add a nested division inside the `<div class="content">` section to support a textbox for text contents using the following code:

```
<div class="textBox">
</div>
```

```
19 | <div class="content">
20 |   <div class="textBox">
21 |     <h2>It's not just Coffee <br>It's <span>Starbucks</span></h2>
22 |     <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Laboriosam a iste dolorem sapiente eius culpa facilis
23 |     <a href="#">Learn more</a>
24 |   </div>
25 | </div>
```

Figure 18

6. Add all elements with text inside `<div class="textBox">`. The content contains a heading, description, and links.

- a. For heading:

```
<h2>It's not just Coffee <br>It's <span>Starbucks</span> </h2>
```

- b. For text description:

```
<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Laboriosam a
iste dolorem sapiente eius culpa facilis beatae harum nemo. Nobis fuga
voluptates quisquam perspiciatis molestias magni obcaecati facere, illum
dignissimos.</p>
```

- c. For the link:

```
<a href="#">Learn more</a>
```

The finished code block should look like Figure 18 above.

7. Add the image for the large beverage with the code below between the tags `<div class="textBox">` and `</div>`. See Figure 19.

```
<div class="imgBox">
    
</div>
```

```
18 </header>
19 <div class="content">
20   <div class="textBox">
21     <h2>It's not just Coffee <br>It's <span>Starbucks</span></h2>
22     <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Laboriosam a iste dolorem sapiente eius culpa facilis
23     <a href="#">Learn more</a>
24   </div>
25   <div class="imgBox">
26     
27   </div>
28 </div>>
```

Figure 19

8. We made great progress. Let's Save our work!
9. We will now add a CSS between the `<style>` block for each element that we added in the previous steps. Size rem is Relative to font-size of the root element.
 - a. For text :

```
.content .textBox {
  position: relative;
  max-width: 600px;
  margin-top: 10rem;
}
```

- b. For h2 element:

```
.content .textBox h2 {
  color: #333;
  font-size: 4rem;
  line-height: 1.4em;
  font-weight: 500;
}
```



c. For span element:

em means relative to the font-size of the element (2em means 2 times the size of the current font)

```
.content .textBox h2 span {  
  color: #017143;  
  font-size: 1.2em;  
  font-weight: 900;  
}
```

d. For link:

"border-radius:" gives the curved edges to the link.


```
.content .textBox a {  
  display: inline-block;  
  margin-top: 20px;  
  padding: 8px 20px;  
  background: #017143;  
  color: #fff;  
  border-radius: 40px;  
  font-weight: 500;  
  letter-spacing: 1px;  
  text-decoration: none;  
}
```

e. For Image:

"flex-end" will display the cup on the left side of the screen.

```
.content .imgBox {  
  display: flex;  
  justify-content: flex-end;  
  padding-right: 50px;  
  margin-top: 50px;  
}  
  
.content .imgBox img {  
  max-width: 340px;  
}
```

10. Your code now contains the new CSS styles between the tags `<style>` `</style>` shown in Figure 20. Save the file.


index.html

Preview
Save
X

```

71     header ul li a {
72         display: inline-block;
73         color: #333;
74         font-weight: 400;
75         margin-left: 40px;
76         text-decoration: none;
77     }
78     .content {
79         position: relative;
80         width: 100%;
81         display: flex;
82         justify-content: space-between;
83         align-content: center;
84     }
85     .content .textBox {
86         position: relative;
87         max-width: 600px;
88         margin-top: 10rem;
89     }
90     .content .textBox h2 {
91         color: #333;
92         font-size: 4rem;
93         line-height: 1.4em;
94         font-weight: 500;
95     }
96     .content .textBox h2 span {
97         color: #017143;
98         font-size: 1.2em;
99         font-weight: 900;
100    }
101    .content .textBox a {
102        display: inline-block;
103        margin-top: 20px;
104        padding: 8px 20px;
105        background: #017143;
106        color: #fff;
107        border-radius: 40px;
108        font-weight: 500;
109        letter-spacing: 1px;
110        text-decoration: none;
111    }
112    .content .imgBox {
113        display: flex;
114        justify-content: flex-end;
115        padding-right: 50px;
116        margin-top: 50px;
117    }
118    .content .imgBox img {
119        max-width: 340px;
120    }
121
122    </style>
123    </html>

```

Figure 20

11. Refresh and Preview your webpage as seen in Figure 21.

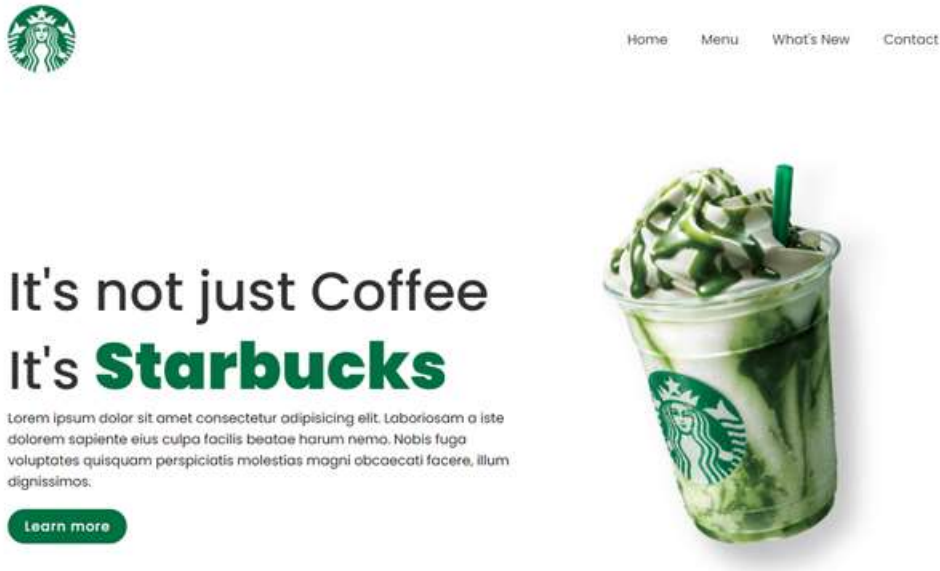


Figure 21

Webpage Layout: Bottom Section & Logos

Animate your webpage

Mouse Over effects can provide an animated feel to bring your website to life and improve the experience of your site visitors.

1. Open your `index.html` file, navigate to the `<body>` `</body>` section
2. Now we will add the bottom part of the webpage containing the three varieties of delicious Starbucks treats that Naiya enjoys. Add the following code in between the `<body>` `</body>` tags after the textbox class to create the division for the bottom-section.

```
<ul class="bottom-section">
  <li> </li>
  <li> </li>
  <li> </li>
</ul>
```

3. Add styles for the entire section with class bottom-section.



Note: The property `transform: translateX(-50%);` means -50% along the x-axis "move me leftwards by half my computed width"

```
.bottom-section{  
  position: absolute;  
  left: 50%;  
  bottom: 20px;  
  transform: translateX(-50%);  
  display: flex;  
  transition: 0.5s;  
}
```

4. Remove the list style and add a margin to make the image look nice.

```
.bottom-section li {  
  list-style: none;  
  display: inline-block;  
  margin: 0 20px;  
}
```

5. Add animation to your treat cups on mouse hover with the following transform style:

```
.bottom-section li:hover {  
  transform: translateY(-15px);  
}
```

6. Limit the image size to a maximum of 60px with style using the below code:

```
.bottom-section li img {  
  max-width: 60px;  
}
```

7. Save your file.

8. Refresh and Preview your webpage as seen in Figure 22. Hover your mouse over the treat cups to see some action!

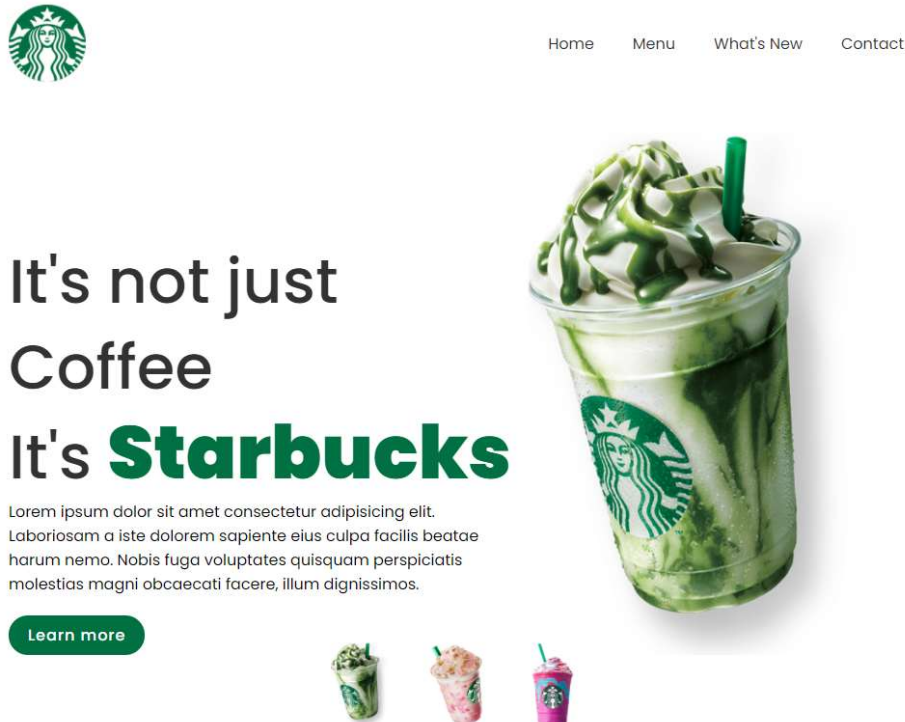


Figure 22

Sharing your webpage

Webpages are intended to be shared. You can easily accomplish this by sharing your webpage through social media platforms such as Facebook, Twitter, and Instagram.

Tip: Logos should always appear on the center, right most side of the webpage.

1. Open your `index.html` file, navigate to the `<body>` `</body>` section
2. Add the following code to load images between the `<body>` `</body>` tags, after the bottom-section, for your logo-section class.

```
<ul class="logosection">
  <li><a href="#"></a></li>
  <li><a href="#"></a></li>
  <li><a href="#"></a></li>
</ul>
```




3. Position your logos between the tags `<style>` `</style>` with the following attributes:
 - a. `absolute position` = provides a fixed position on the webpage for the logos.
 - b. `flex-direction column` = aligns the logos vertically instead of horizontally.
Note: By default, it is set to horizontal with the value "row".
 - c. `translateY(-50%)` = -50% along the y-axis to "move me horizontally by half my computed width". Note: we coded the x-axis earlier with `translateX(-50%)`.

```
. logo-section {  
  position: absolute;  
  top: 50%;  
  right: 30px;  
  transform: translateY(-50%);  
  display: flex;  
  justify-content: center;  
  align-content: center;  
  flex-direction: column;  
}
```

4. Remove the default placement of the bulleted list by setting the list-style to none.

```
. logo-section li {  
  list-style: none;  
}
```

5. Save your code.
6. Preview the webpage. It should now look like Figure 23.

It's not just Coffee
It's **Starbucks**

Lorem ipsum dolor sit amet consectetur adipisicing elit. Laboriosam a iste dolorem sapiente eius culpa facilis beatae harum nemo. Nobis fuga voluptates quisquam perspiciatis molestias magni obcaecati facere, illum



Figure 23



Webpage Layout: Circle

Webpage background theme

The background holds the theme of the website and the possibilities are limitless when designing it. There is a fine line between a beautiful background and a busy background that won't look so good and takes away from the content. The trick is to strike the right balance.

1. Open your `index.html` file, navigate to the `<style> </style>` section
2. Add a theme to your background with a splash of green by adding the below code to the `<style>`. We will change the logo color to white to contrast the green circle with the following settings:
 - a. `filter: invert(1)` = applied to reverse the color of the logos, since our logos are black, it will turn it white.
 - b. `Transform: scale` = scales the logo images down to 0.6 units to achieve our desired size

```
. logo-section li a {  
    display: inline-block;  
    margin: 5px 0;  
    transform: scale(0.6);  
    filter: invert(1);  
}
```

3. Navigate to the top of your source code file, to the inside of the `<body>` tag above the header tag. Add a division for the Circle class.

Tip: Adding the code in this order places the Circle in the background verses the foreground. The elements afterwards are placed on top.

```
<div class="circle"> </div>
```

4. Navigate to the `<style> </style>` section and apply style to the Circle division `<div>` and add color to it with the following CSS attributes:
 - a. `clip-path : circle(40% at 98% 79%)` = clips an element to a basic shape. In our project, the half circle at the bottom left of the screen uses the function to set 40% of circle to display to the height and width specified by percentage on the screen.

- b. dimensions = consists of width, height. These attributes could be set instead of percentage.

```
.circle {
  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background: #017143;
  clip-path: circle(40% at 98% 79%);
}
```

7. Save your file and Preview your webpage. See Figure 24. We are almost done!



Figure 24

JavaScript: Webpage interactivity

JavaScript is a scripting programming language that adds interactivity to your website. This happens when, for example, a button is pressed, and a response occurs. In advanced JavaScript programming, websites can support games, dynamic styling, complex animations, and many more capabilities. For our project, we will begin with the basics.

1. Open your `index.html` file, navigate to the bottom of the file, below the closing `</style>` tag
2. Add `<script>` tag using the code below. See Figure 25

```
<script type="text/javascript">
</script>
```



Figure 25



Change Background Color

We can change the circle's background color when a treat cup image located in the bottom-section is clicked. We will accomplish this by adding a JavaScript function to our code.

Below is a breakdown of the structure of a function code block:

- A JavaScript function is created using the "function" keyword. The name of our function is

`changeCircleColor`

- Functions can accept parameters enclosed in between parenthesis. The JavaScript function, below, takes the color as it's input parameter.

`changeCircleColor(color)`

- Open curly brace { begins the body of the function's code
- Close curly brace } ends the body of the function's code
- Inside a function, variables can be declared and initialized with a value
 - In our example, the variable circle is declared as a const and initialized with `document.querySelector('.circle')`
 - `const circle = document.querySelector('.circle');`
 - Object variables can contain the object's class attributes such as a circle's background color. The below line of code fetches the circle div class and applies the style background color to it.

`circle.style.background = color;`

1. Add the below Javascript Code:

```
function changeCircleColor(color) {  
    const circle = document.querySelector('.circle');  
    circle.style.background = color;  
}
```



Clicking on the treat cup image from the bottom-section should change the large beverage image in front of the circle to reflect our selection. We will do this by adding a second JavaScript function inside the `<script>` tag. This new function will receive an image path as the input parameter to set as the value for the function call `"document.querySelector('.starbucks')."`. This assignment will fetch the div with class `starbucks` and replace the image source path with the one provided by the input parameter.

2. Add the below Javascript Code:

```
function imgSlider(imgPath){  
    document.querySelector('.starbucks').src = imgPath;  
}
```

Now that we have two functions created, to active and use it, the functions will need to be called.

3. Begin by navigating to the class bottom-section of your code (as seen below) and delete it.

```
<ul class=" bottom-section">  
  <li></li>  
  <li></li>  
  <li></li>  
</ul>
```

3. Replace the above lines with the following:

```
<ul class="bottom-section">  
  <li></li>  
  <li></li>  
  <li></li>  
</ul>
```

The new code will add an onclick event which will point to the two new functions we created in the previous steps. When we mouse click the treat cups, the functions will display the expected beverage and background color that corresponds with the user's selection.

4. Save your code.

At your leisure, you may verify your code against the full solution downloaded at the beginning of this project in the folder "FinishedWebPage" directory. The source code of the webpage can be viewed by right clicking the `index.html` file and open it with Notepad.

5. Preview your webpage. It should now look like Figure 26.

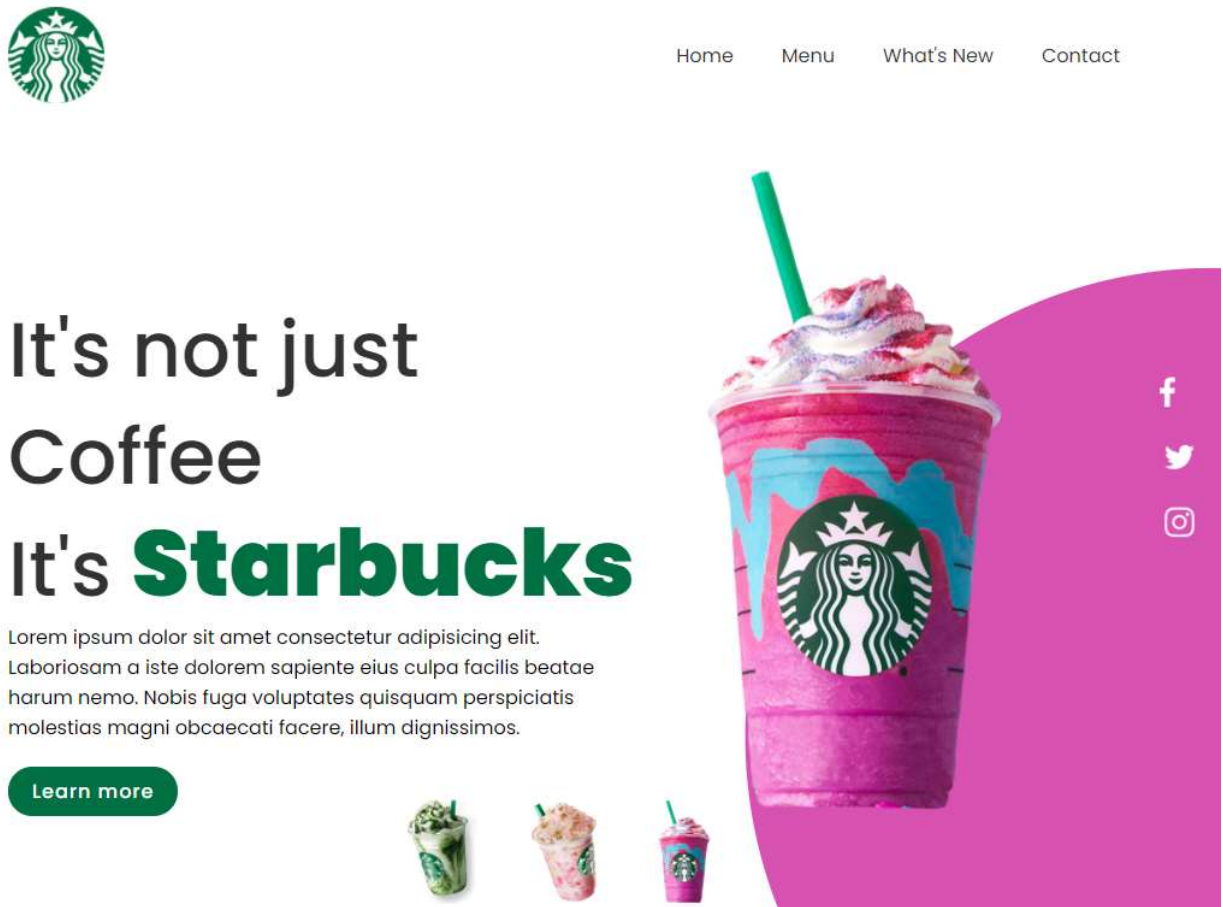


Figure 26

Congratulations! You have successfully completed your code project!!



More Coding Fun

Great job completing the primary coding activity. Read more to learn how to implement additional functionality to your webpage like making all buttons and menus functional.

Learn More Button

To make it functional, add a div section below the learn more button and add content into that div. Click the below link for a tutorial on toggling the hide/show functionality of this button:

https://www.w3schools.com/howto/howto_js_toggle_hide_show.asp

Add HTML Links & Pages

To add functionality to the Menu, What's New and Contact options, use the below tutorial on HTML links:

https://www.w3schools.com/html/html_links.asp

Once you confirm links are working, adding content in each page can be accomplished by completing the following steps:

- Reference this document's [Header](#) section. Copy paste the header section and style for header section in newly created html pages.
- Use the following link for content inspiration on image/text alignment.
<https://www.hostpapa.com/knowledgebase/align-float-images-website/>

Add New Styling and Content

Use your creativity to:

- Create a list of and describe different products. Leverage the same styling from the main coding activity or create your own.
- Provide contact information for the contact page
- Add dynamic content with JavaScript
- Use the following tutorial to add HTML geo location (which identifies the user's current location): https://www.w3schools.com/html/html5_geolocation.asp



References

Explore each element below by selecting the appropriate link

Element	Link
Box Sizing	https://www.w3schools.com/css/css3_box-sizing.asp
CSS Units	https://www.w3schools.com/css/css_units.asp
Flex Layout	https://www.w3schools.com/css/css3_flexbox.asp
Inline block	https://www.w3schools.com/css/css_inline-block.asp
Class Attribute	https://www.w3schools.com/html/html_classes.asp
Transforms	https://www.w3schools.com/css/css3_2dtransforms.asp
TranslateX	https://developer.mozilla.org/en-US/docs/Web/CSS/transform-function/translateX()
Filter Invert	https://www.w3schools.com/cssref/css3_pr_filter.asp
Clip Path	https://www.w3schools.com/cssref/css3_pr_clip-path.asp
Javascript function	https://www.w3schools.com/js/js_functions.asp
OnClick event	https://www.w3schools.com/jsref/event onclick.asp
Document.querySelector	https://www.w3schools.com/jsref/met_document_queryselector.asp



HTML Definitions

Element	Definition
<!DOCTYPE html>	declaration defines that this document is an HTML5 document
<html>	element is the root element of an HTML page
<head>	element contains meta information about the HTML page
<title>	element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab) https://www.w3schools.com/tags/tag_title.asp
<body>	element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
<h1>	element defines a large heading
<p>	element defines a paragraph
<div>	defines a division or a section in an HTML document https://www.w3schools.com/tags/tag_div.ASP
<a>	defines a hyperlink https://www.w3schools.com/html/html_links.asp
	allow web developers to group a set of related items in lists https://www.w3schools.com/html/html_lists.asp
	used to embed an image in a web page https://www.w3schools.com/html/html_images.asp



CSS Unit and Property Definitions (and links)

Unit/Property	Definition
cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	Relative to the parent element
align-content	Modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines
align-items	Vertically aligns the flex items when the items do not use all available space on the cross-axis
display	Specifies the type of box used for an HTML element
flex-direction	Specifies the direction of the flexible items inside a flex container
flex-flow	A shorthand property for flex-direction and flex-wrap
flex-wrap	Specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line
justify-content	Horizontally aligns the flex items when the items do not use all available space on the main-axis