|  |  |  |  |
| --- | --- | --- | --- |
| Lab 10 Chapter 7 Advanced CSS: Layout  |  | | --- | | What You Will Learn   * How to position and float elements outside of normal page flow * How to construct multi-column layouts * How to make a responsive layout * How to use a CSS Framework to simplify page layout |  |  | | --- | | Approximate Time  The exercises in this lab should take approximately 50 minutes to complete. | |  | |
| Web Application Development  COP3834  Professor Navarro |
| Textbook by Pearson  http://www.funwebdev.com  Date Last Revised: Sep 20, 2021 |

## Creating Tables

|  |
| --- |
| Preparing Directories |
| 1 | Download zip folder.. |
| 2 | Uncompress to your desktop. |

The <table> element in HTML represents information that exists in a two-dimensional grid. Tables can be used to display calendars, financial data, pricing tables, and many other types of data. Just like a real-world table, an HTML table can contain any type of data: not just numbers, but text, images, forms, even other tables.

|  |
| --- |
| Exercise 7. — Document Flow |
| 1 | Open, examine, and test lab07-exercise01.html. |
| 2 | Resize the browser window and experiment by making the width smaller and larger. |
| 3 | Modify the following style inlab07-exercise01.css and test.  h1 span {  color: #8B0000;  font-size: 1.5em;  display: block;  }  This changes the <span> element in the main heading from an inline element to a block-level element. |
| 4 | Add the following style and test.  li {  display: inline;  }  This changes the <li> elements from block-level to inline (they will thus show up on a single line now. |
| 5 | Add the following style and test.  div img {  display: block;  } |

## 

## Positioning

|  |
| --- |
| Exercise 7. — Relative Positioning |
| 1 | Open, examine, and test lab07-exercise02.html. |
| 2 | Modify the following style inlab07-exercise02.css and test.  figure {  background-color: #EDEDDD;  border: 1pt solid #A8A8A8;  padding: 5px;  width: 150px;  top: 150px;  left: 200px;  } |
| 3 | Modify the following style and test.  figure {  background-color: #EDEDDD;  border: 1pt solid #A8A8A8;  padding: 5px;  width: 150px;  top: 150px;  left: 200px;  position: relative;  }  As you can see in Figure 7.1, the original space for the positioned <figure> element is preserved, as is the rest of the document’s flow. As a consequence, the repositioned element now overlaps other content. |

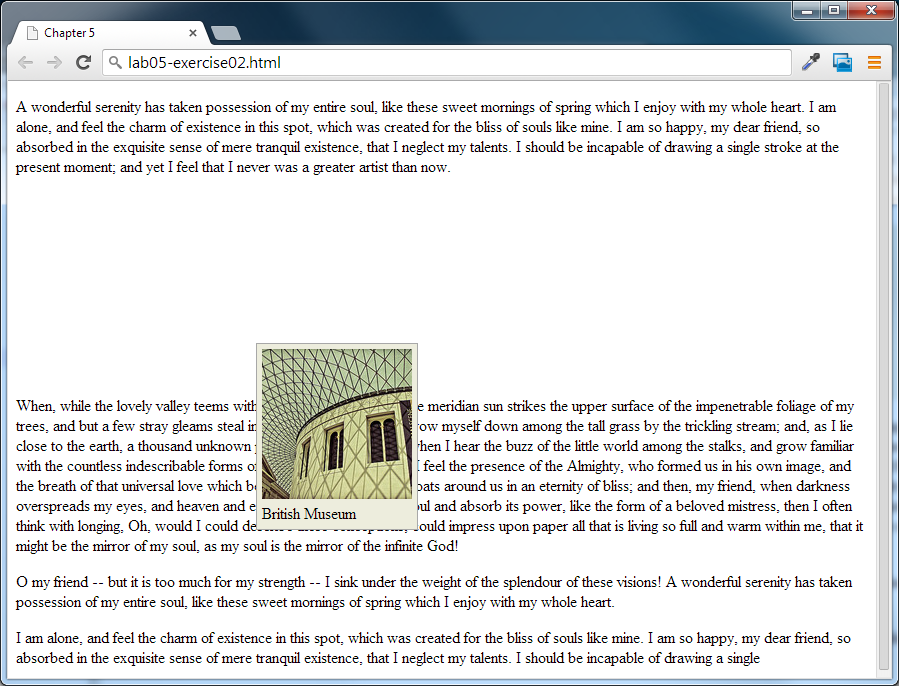


Figure 7.1 – Exercise 7.2 complete



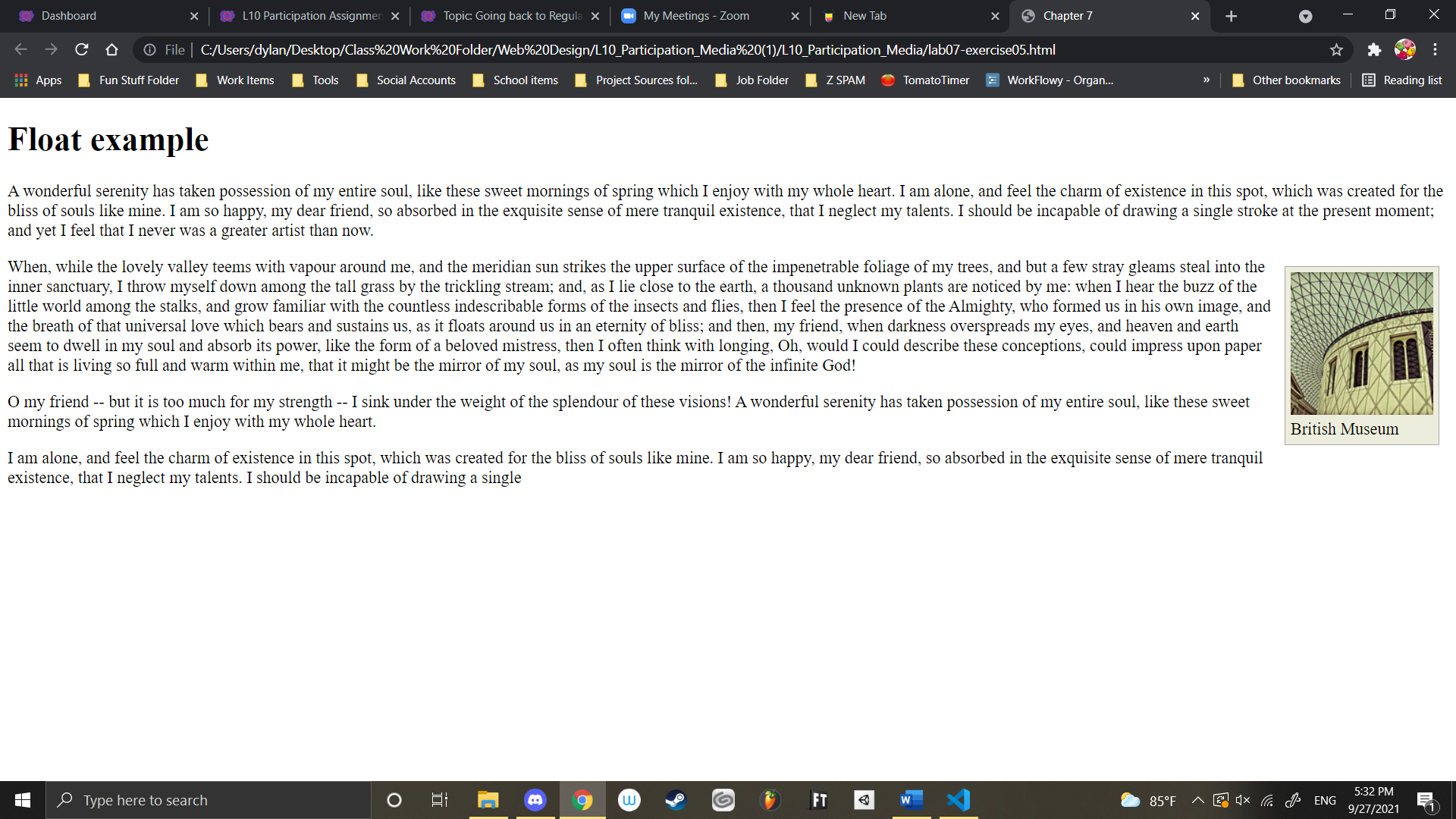
|  |
| --- |
| Exercise 7. — Absolute Positioning |
| 1 | Open, examine, and test lab07-exercise03.html. |
| 2 | Modify the following style inlab07-exercise03.css and test.  Figure {  background-color: #EDEDDD;  border: 1pt solid #A8A8A8;  padding: 5px;  width: 150px;  top: 150px;  left: 200px;  position: absolute;  }  With absolute positioning, space is not left for the moved element, as it is no longer in the normal flow. |
| 3 | Modify the following style and test.  Figcaption {  background-color: #EDEDDD;  padding: 5px;  top: 150px;  left: 200px;  position: absolute;  }  A moved element via absolute position is actually positioned relative to its nearest positioned ancestor container (that is, a block-level element whose position is fixed, relative, or absolute). In this example, the <figcaption> is absolutely positioned; it is moved 150 px down and 200 px to the left of its nearest positioned ancestor, which happens to be its parent (the <figure> element). |
|  |  |
|  |  |
| Exercise 7. — Stacking Using Z-Index |
| 1 | Open, examine, and test lab07-exercise04.html. |
| 2 | Modify the following style and test.  figure {  background-color: #EDEDDD;  border: 1pt solid #A8A8A8;  padding: 5px;  width: 150px;  top: 150px;  left: 200px;  position: absolute;  }  figcaption {  font-size: 1.25em;  background-color: yellow;  padding: 5px;  top: 90px;  left: 140px;  position: absolute;  }  Notice that the caption now overlaps both the underlying paragraph text and the figure image. |
| 3 | Modify the following styles (some CSS omitted here for clarity) and test.  figure {  ...  position: absolute;  z-index: 5;  }  figcaption {  ...  position: absolute;  z-index: 1;  }  Note that this did not move the <figure> on top of the <figcaption> as one might expect. This is due to the nesting of the caption within the figure. |
| 4 | Modify the following styles and test.  figure {  ...  position: absolute;  z-index: 1;  }  figcaption {  ...  position: absolute;  z-index: -1;  }  The <figure> z­index could be any value equal to or above 0 as long as the <figcaption> is below 0. |
| 5 | Modify the following styles and test.  figure {  ...  position: absolute;  z-index: -1;  }  figcaption {  ...  position: absolute;  z-index: 1;  }  If the <figure> z­index is given a value less than 0, then any of its positioned descendants change as well. Thus both the <figure> and <figcaption> move underneath the body text. |

## 

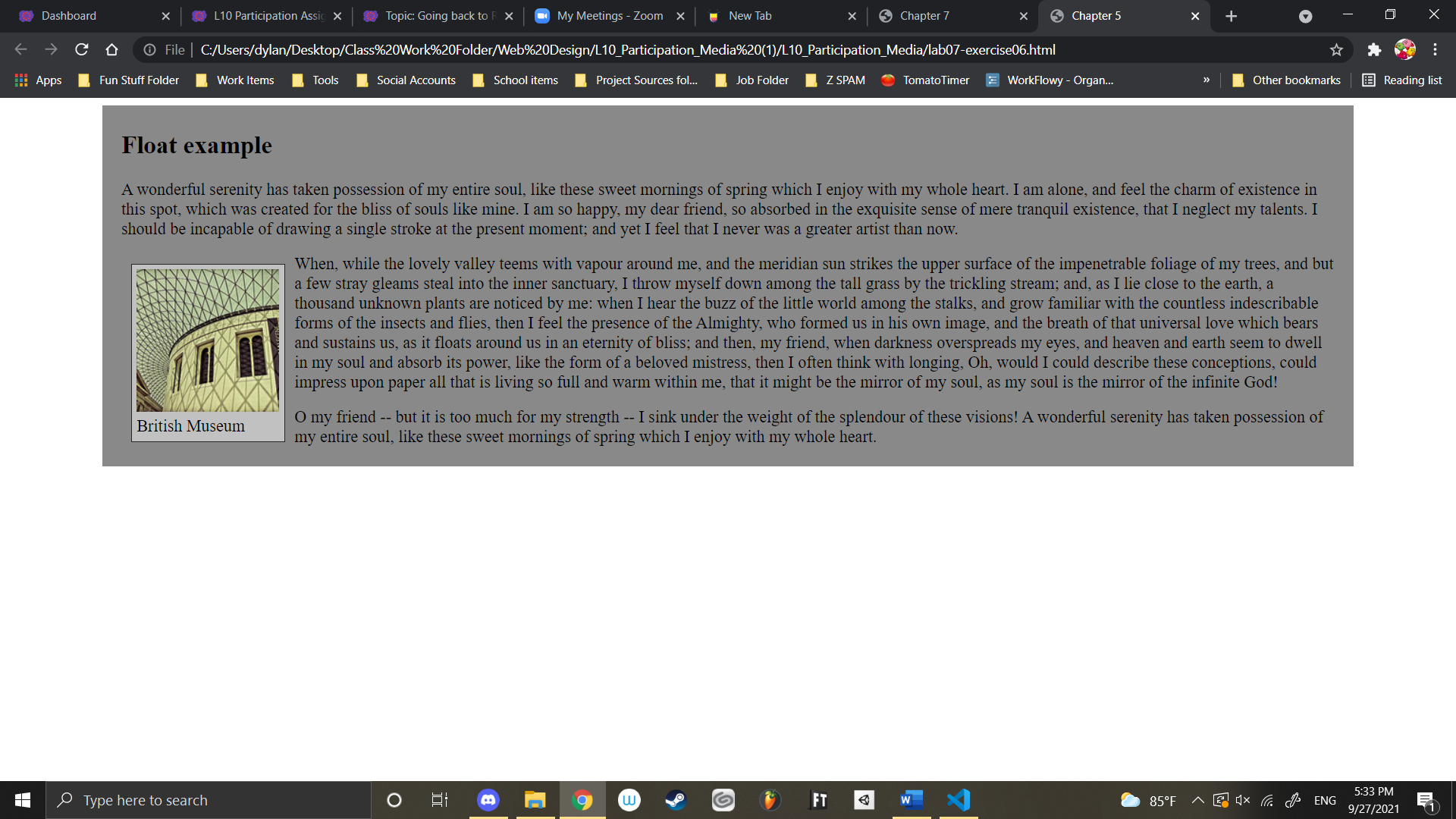
## 

## Floating

|  |
| --- |
| Exercise 7. — Floating Elements |
| 1 | Open, examine, and test lab07-exercise05.html. |
| 2 | Modify the following styles and test.  figure {  border: 1pt solid #A8A8A8;  background-color: #EDEDDD;  padding: 5px;  margin: 10px;  width: 150px;  float: left;  }  Notice that a floated block-level element must also have a width specified. |
| 3 | Modify the following styles and test.  figure {  border: 1pt solid #A8A8A8;  background-color: #EDEDDD;  padding: 5px;  margin: 10px;  width: 150px;  float: right;  } |



|  |
| --- |
| Exercise 7. — Floating in a Container |
| 1 | Open, examine, and test lab07-exercise06.html.  Notice the empty margin space on the left. |
| 2 | Modify the following styles and test.  figure {  border: 1pt solid #262626;  background-color: #c1c1c1;  padding: 5px;  width: 150px;  margin: 10px;  float: left;  }  Notice that a floated item moves to the left or right of its container. In this case the container for the floated <figure> is the <article> element. |



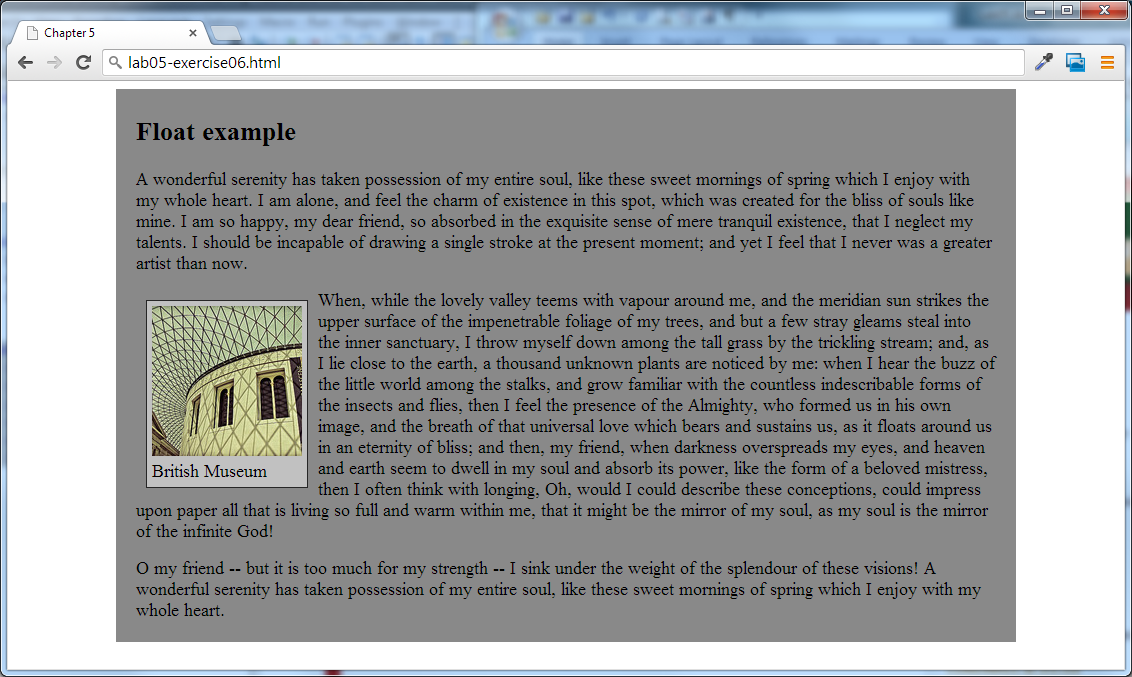
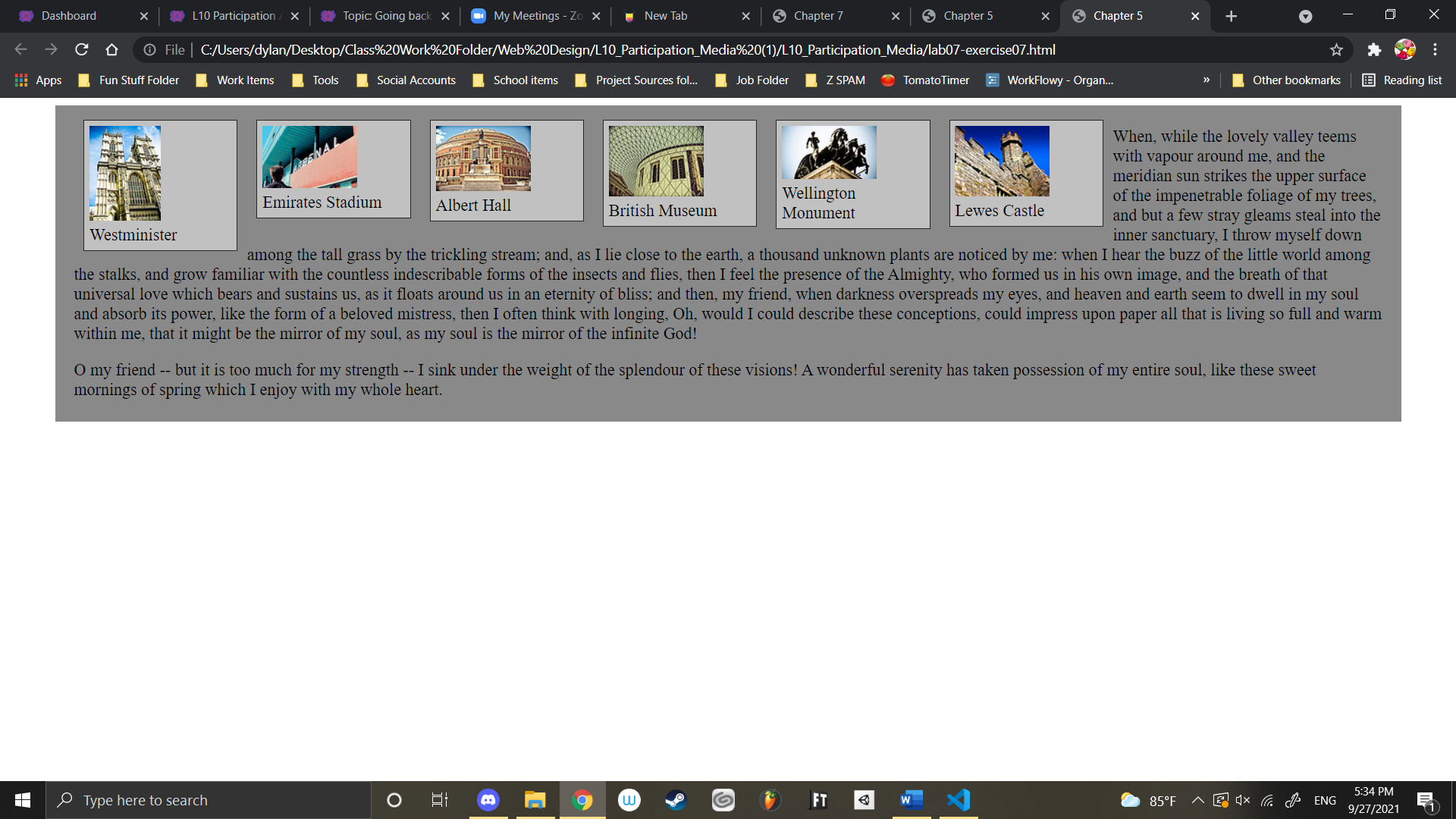


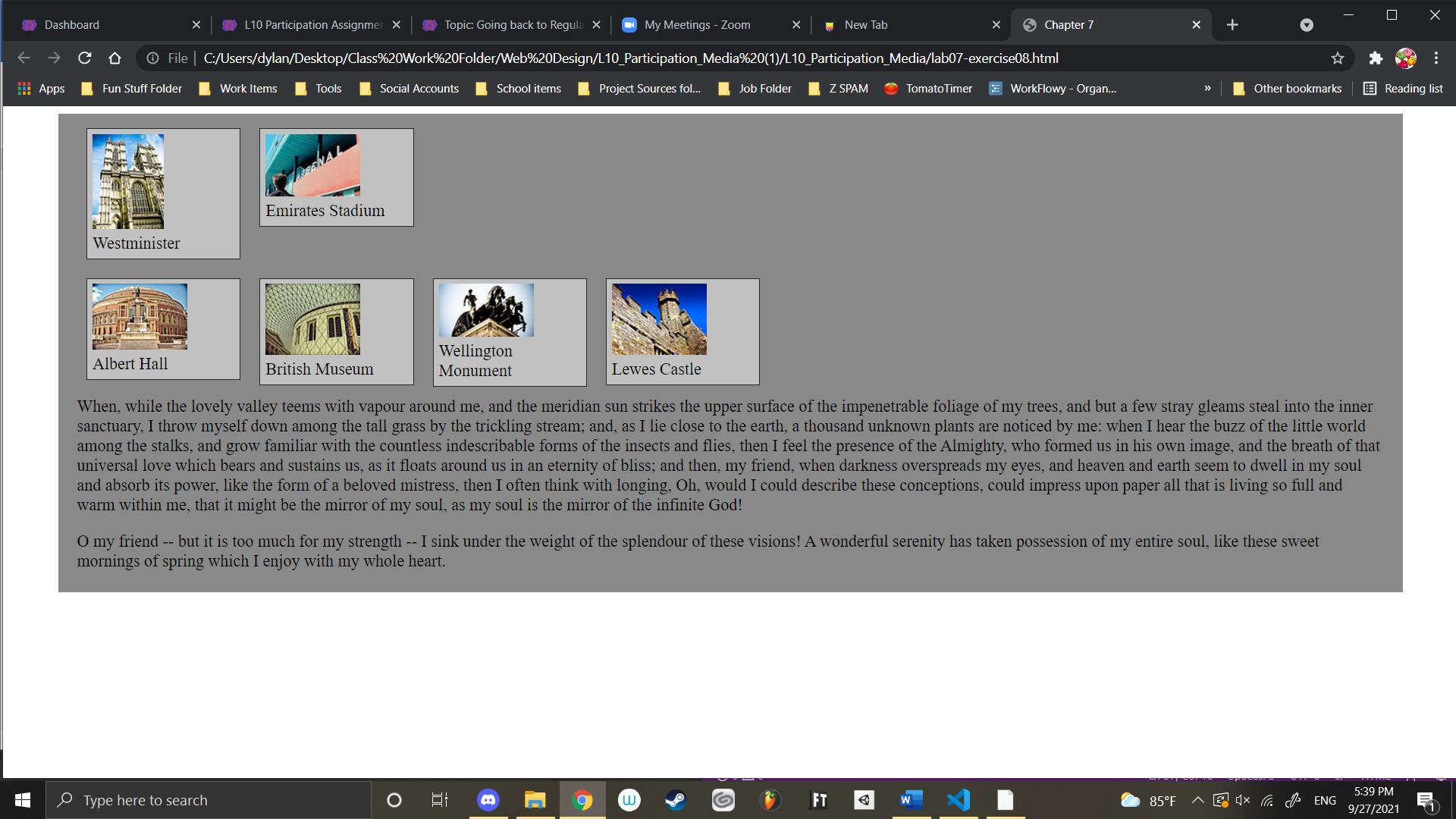
Figure 7.4 – Exercise 7.6 complete

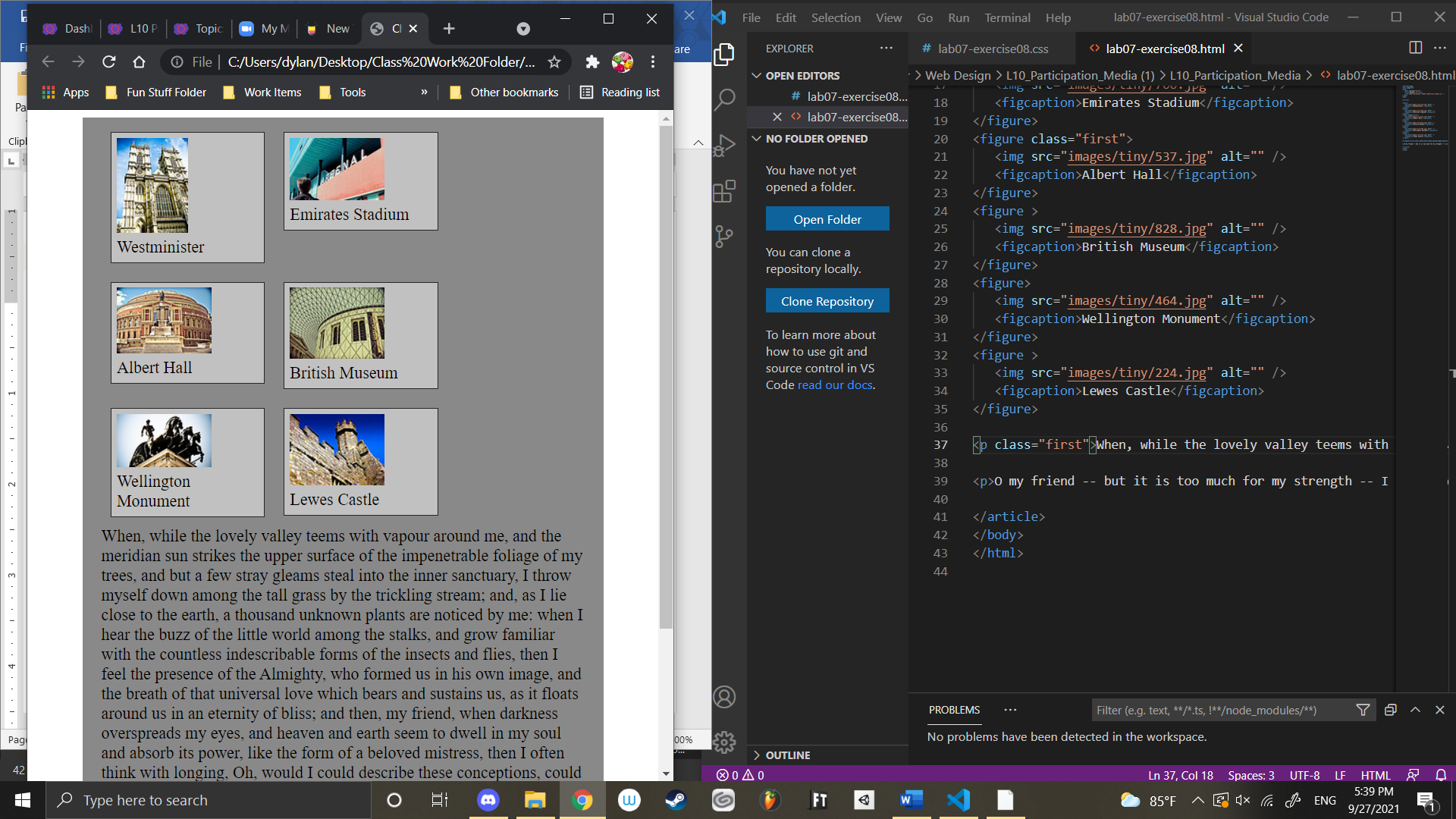
|  |
| --- |
| Exercise 7. — Floating Multiple Items |
| 1 | Open, examine, and test lab07-exercise07.html.  Notice that each <figure> is a block-level element and hence on its own line. |

|  |  |
| --- | --- |
| 2 | Modify the following style and test.  figure {  border: 1pt solid #262626;  background-color: #c1c1c1;  padding: 5px;  width: 150px;  margin: 10px;  float: left;  }  When you float multiple items that are in proximity, each floated item in the container will be nestled up beside the previously floated item. All other content in the containing block (including other floated elements) will flow around all the floated elements. |
| 3 | Experiment by changing the width of the browser window.  This can result in some pretty messy layouts as the browser window increases or decreases in size (that is, as the containing block resizes). The solution is to use the clear property (next exercise). |


|  |
| --- |
| Exercise 7. — Floating and Clearing |
| 1 | Open, examine, and test lab07-exercise08.html. |
| 2 | Add the following style and test.  .first { clear: left; } |
| 3 | Modify the third <figure> element as follows and test:  <figure class="first">  <img src="images/tiny/828.jpg" alt="" />  <figcaption>British Museum</figcaption>  </figure> |
| 4 | Modify the <p> element after the figures as follows and test:  <p class="first">When, while the lovely valley ... |





|  |
| --- |
| Exercise 7. — Using Positioning |
| 1 | Open, examine, and test lab07-exercise09.html.  In this exercise, we are going to position the caption on top of the image. |
| 2 | Modify the following style and test.  figcaption {  width: 100px;  background-color: black;  color: white;  opacity: 0.6;  width: 140px;  height: 20px;  padding: 5px;  position: absolute;  top: 130px;  left: 10px;  }  Since absolute positioning is positioning in relation to the closest positioned ancestor, we will also need to position the <figure>. |
| 3 | Modify the following style and test.  figure {  border: 1pt solid #262626;  background-color: #c1c1c1;  padding: 10px;  width: 200px;  margin: 10px;  position: relative;  }  Notice that we are not moving the figure. Instead by setting its position to relative, we are creating a positioning context for the absolute positioning in the previous step. |
| 4 | Add the following markup and test.  <figure>  <img src="images/828.jpg" alt="" />  <figcaption>British Museum</figcaption>  <img src="images/new-banner.png" alt="" class="overlayed"/>  </figure> |
| 5 | Add the following style and test.  .overlayed {  position: absolute;  top: 10px;  left: 10px;  } |

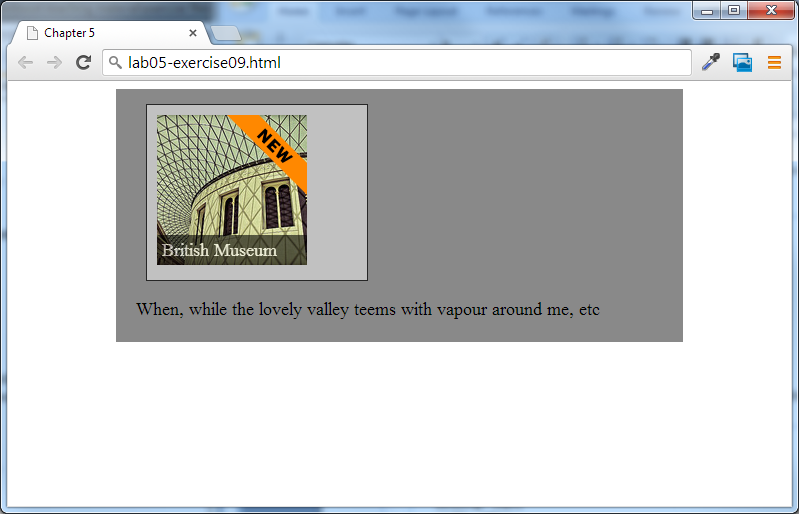


Figure 7.4 – Exercise 7.9 complete

## 

## Layouts

The exercises so far in this lab showed two different ways to move items out of the normal top-down flow, namely, by using positioning and by using floats. They are the raw techniques that you can use to create more complex layouts.

|  |
| --- |
| Exercise 7. — Two-Column Layout |

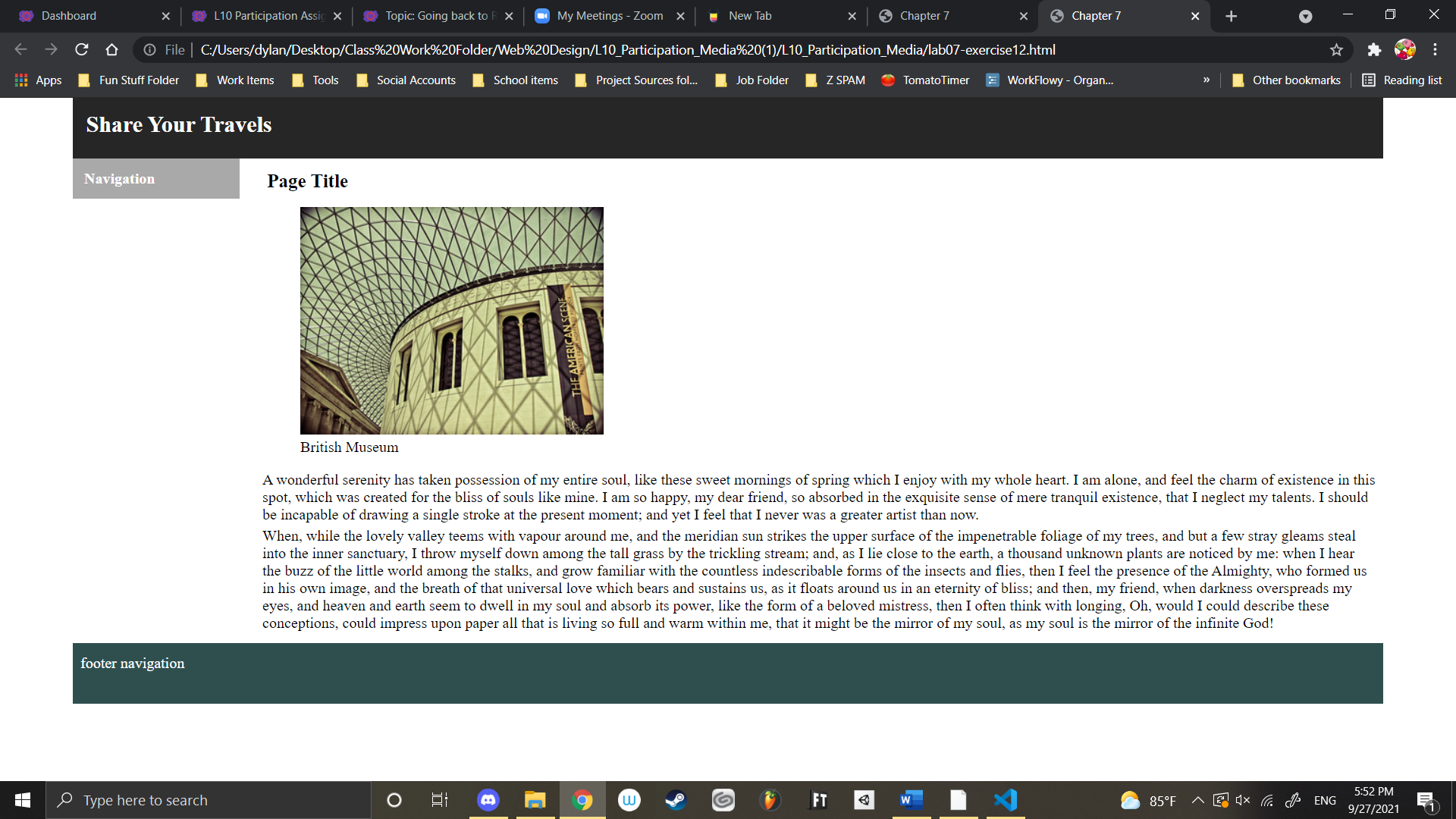
skip

|  |
| --- |
| Exercise 7. — Three-Column Layout |
| 1 | skip | |
| 2 |  | |

## Responsive Design

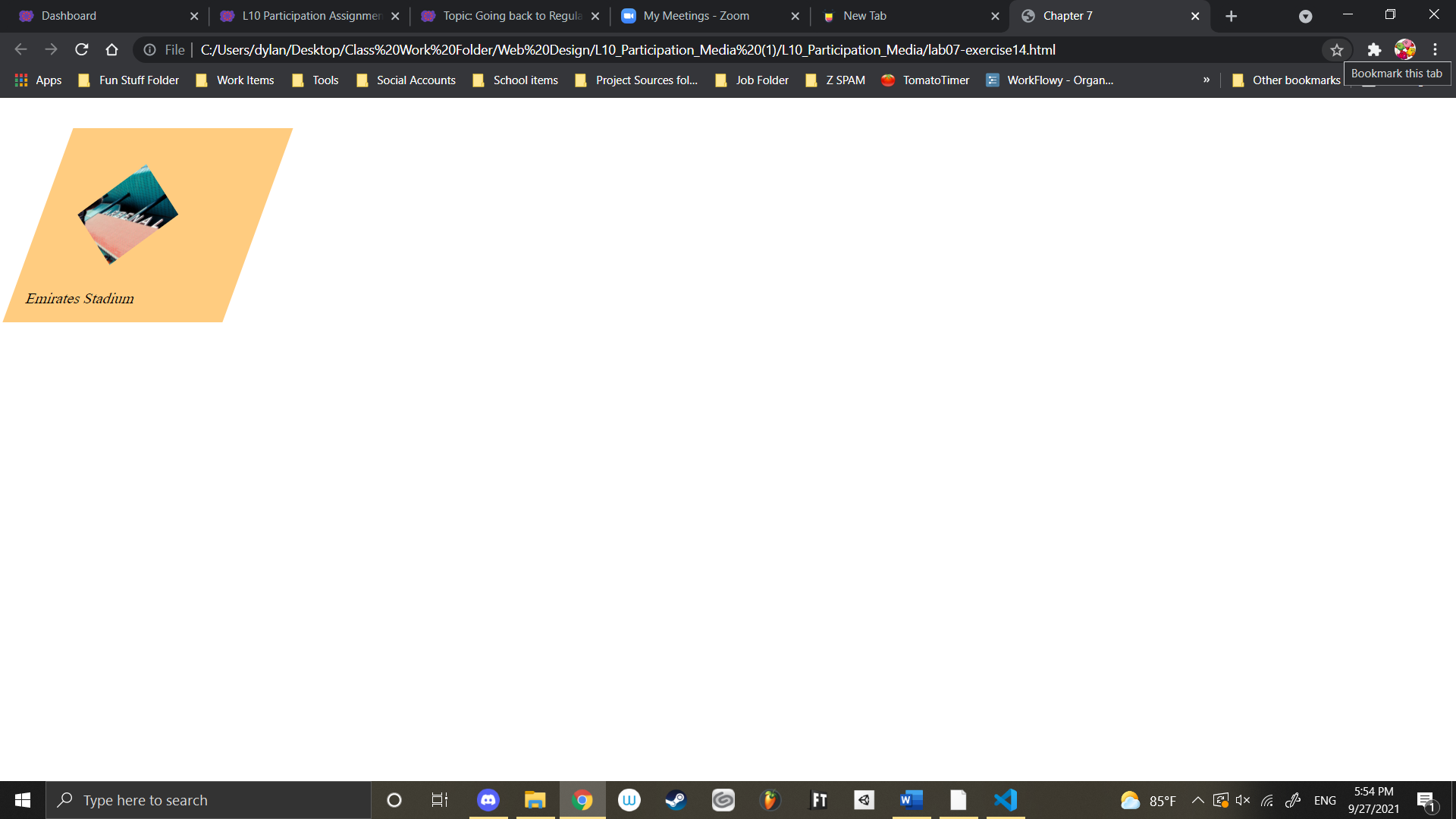
|  |
| --- |
| Exercise 7. — Media Queries |
| 1 | Open, examine, and test lab07-exercise12.html. | |
| 2 | Modify the following markup and test by progressively reducing the width of the browser window .  <link rel="stylesheet" href="lab07-exercise12-desktop.css"  media="screen and (min-width:481px)" />  Once the window shrinks below 481px wide, this stylesheet will no longer be in effect (and hence all CSS styling will disappear). | |
| 3 | Add the following additional media query.  <head lang="en">  <meta charset="utf-8">    <title>Chapter 7</title>  <link rel="stylesheet" href="lab07-exercise12-mobile.css"  media="screen and (max-width:480px)" />  <link rel="stylesheet" href="lab07-exercise12-desktop.css"  media="screen and (min-width:481px)" />  </head>  We are adding an additional style sheet for mobile browser sizes. | |
| 4 | Edit lab07-exercise12-mobile.css (this is the mobile style sheet) as follows and test.  main {  width: 100%;  margin-left: auto;  margin-right: auto;  }  nav {  color: white;  background-color: #A9A9A9;  padding: 0.5em;  width: 100%;  /\* be sure to delete the float as well \*/  }  div#main {  padding: 0.5em;  margin-left: 0;  }  figure img {  max-width: 100%;  }  Notice that when the browser window shrinks to mobile size, the two-column layout is replaced with a single column. This requires removing the floats and the margins. | |

|  |
| --- |
| Exercise 7. — Setting the Viewport |
| 1 | Continue this exercise by using your files from Exercise 12. | |
| 2 | Add the following viewport.  <meta name="viewport" content="width=device-width" /> | |
| 3 | Add the following markup to the <nav> element.  <nav>  <h3>Navigation</h3>  <ul>  <li><a href="#">Australia</a></li>  <li><a href="#">Belgium</a></li>  <li><a href="#">Canada</a></li>  <li><a href="#">France</a></li>  <li><a href="#">United Kingdom</a></li>  <li><a href="#">United States</a></li>  </ul>  <select>  <option>Australia</option>  <option>Belgium</option>  <option>Canada</option>  <option>France</option>  <option>United Kingdom</option>  <option>United States</option>  </select>  </nav>  We are going to use CSS to use the <select> element for mobile sizes, and the <ul> for larger sizes. | |
| 4 | Edit lab07-exercise12-desktop.css (this is the desktop style sheet) as follows by adding the following style.  nav select {  display: none;  } | |
| 5 | Edit lab07-exercise12-mobile.css (this is the mobile style sheet) as follows by adding the following styles and test in browser. Be sure to test using small and larger browser widths.  nav ul { display: none; }  nav select {  display: inline-block;  width: 90%;  } | |

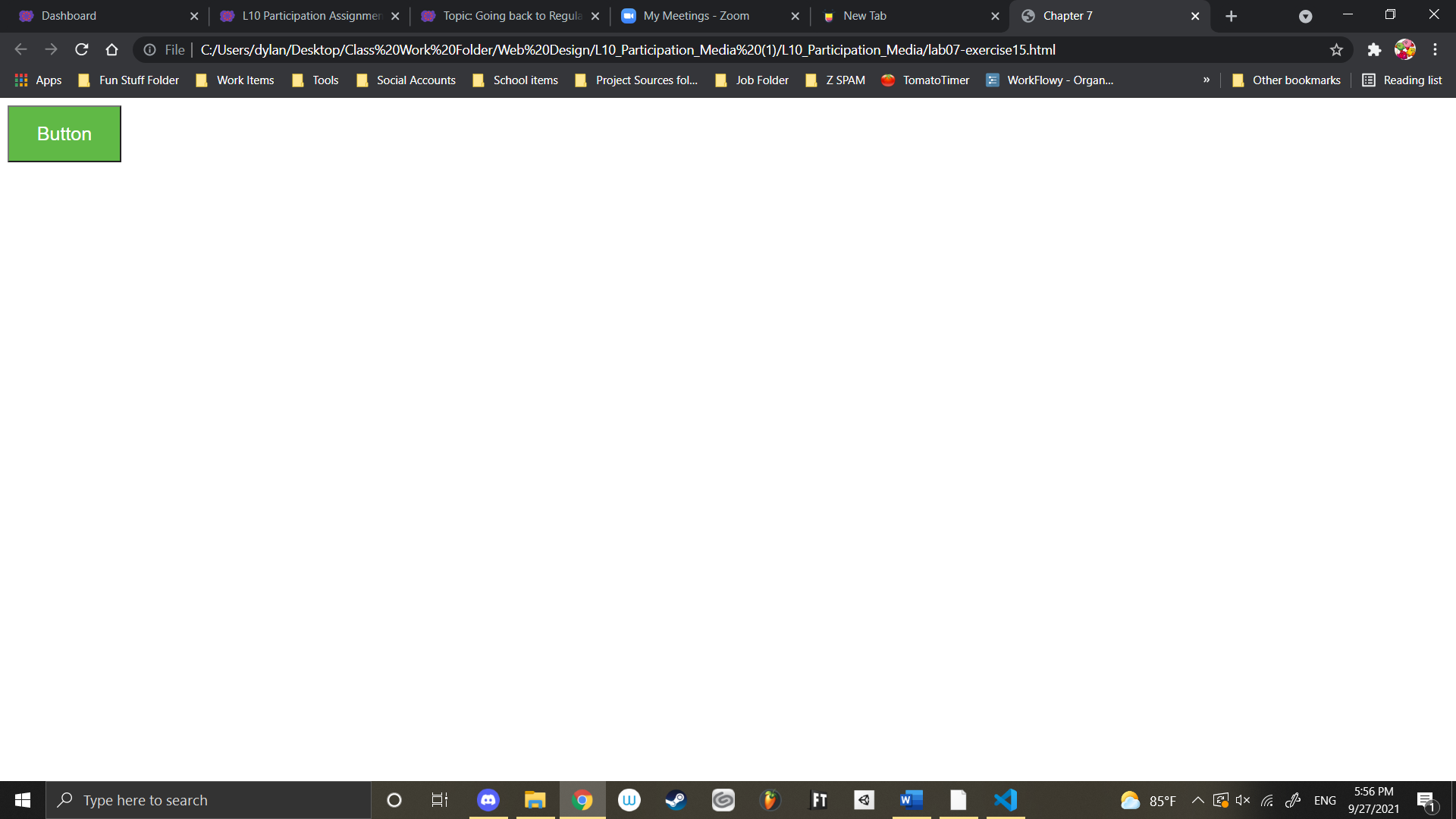


## CSS3 Features

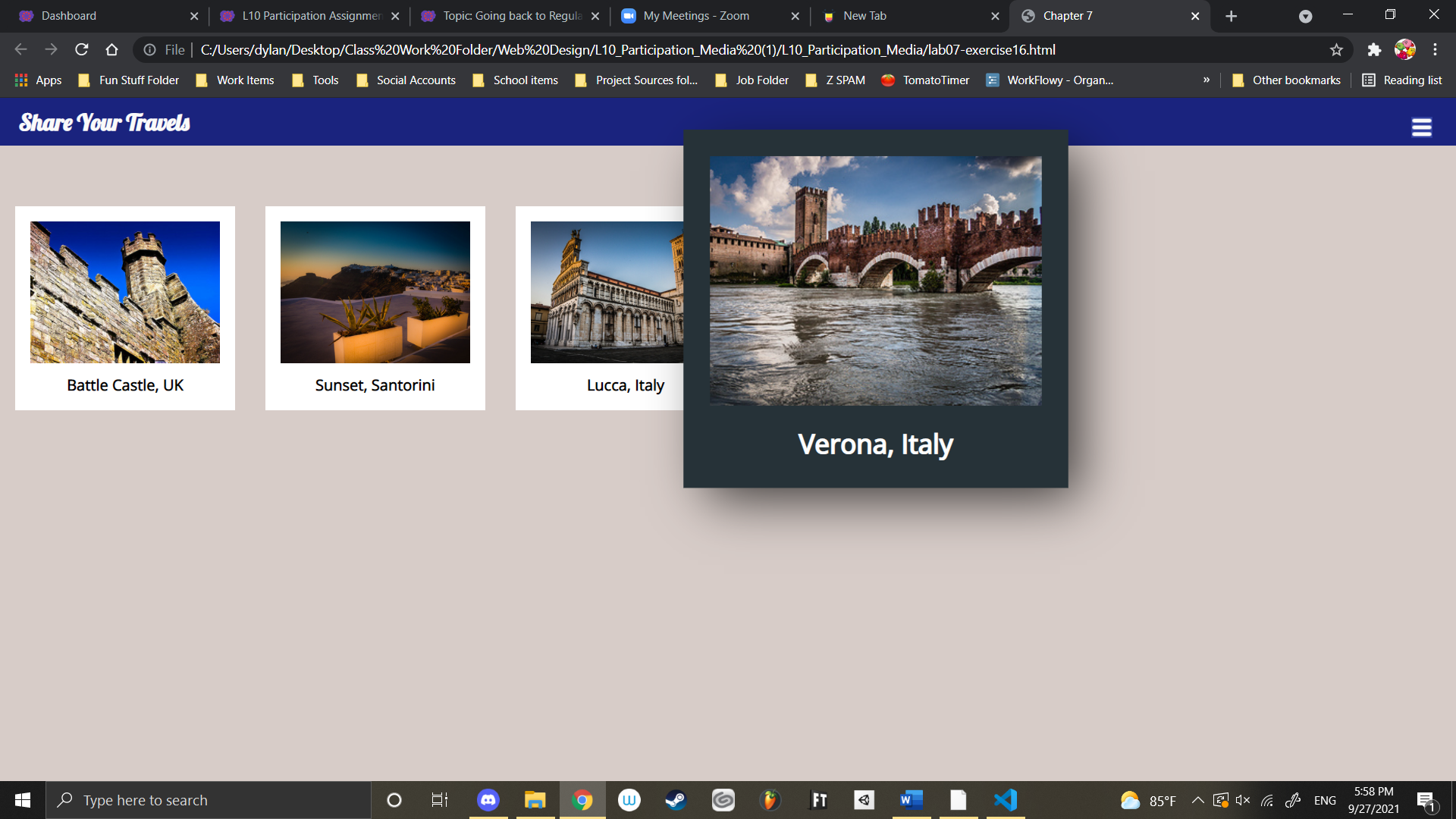
|  |
| --- |
| Exercise 7. — Transforms |
| 1 | Open, examine, and test lab07-exercise14.html. | |
| 2 | Add the following style to the end of lab07-exercise14.css and test.  figure {  transform: rotate(45deg);  }  Notice that the transform affects all the content within the transformed container. | |
| 3 | Modify the style as follows and test.  figure {  transform: skew(-20deg);  } | |
| 4 | Add the following style and test.  figure img {  transform: translatex(100px) translatey(-30px);  }  Notice that you can combine transforms and that the y-axis extends downwards. | |
| 5 | Modify the styles as follows and test.  figure {  transform: rotate(15deg);  }  figure img {  transform: rotate(45deg) scale(0.5);  } | |



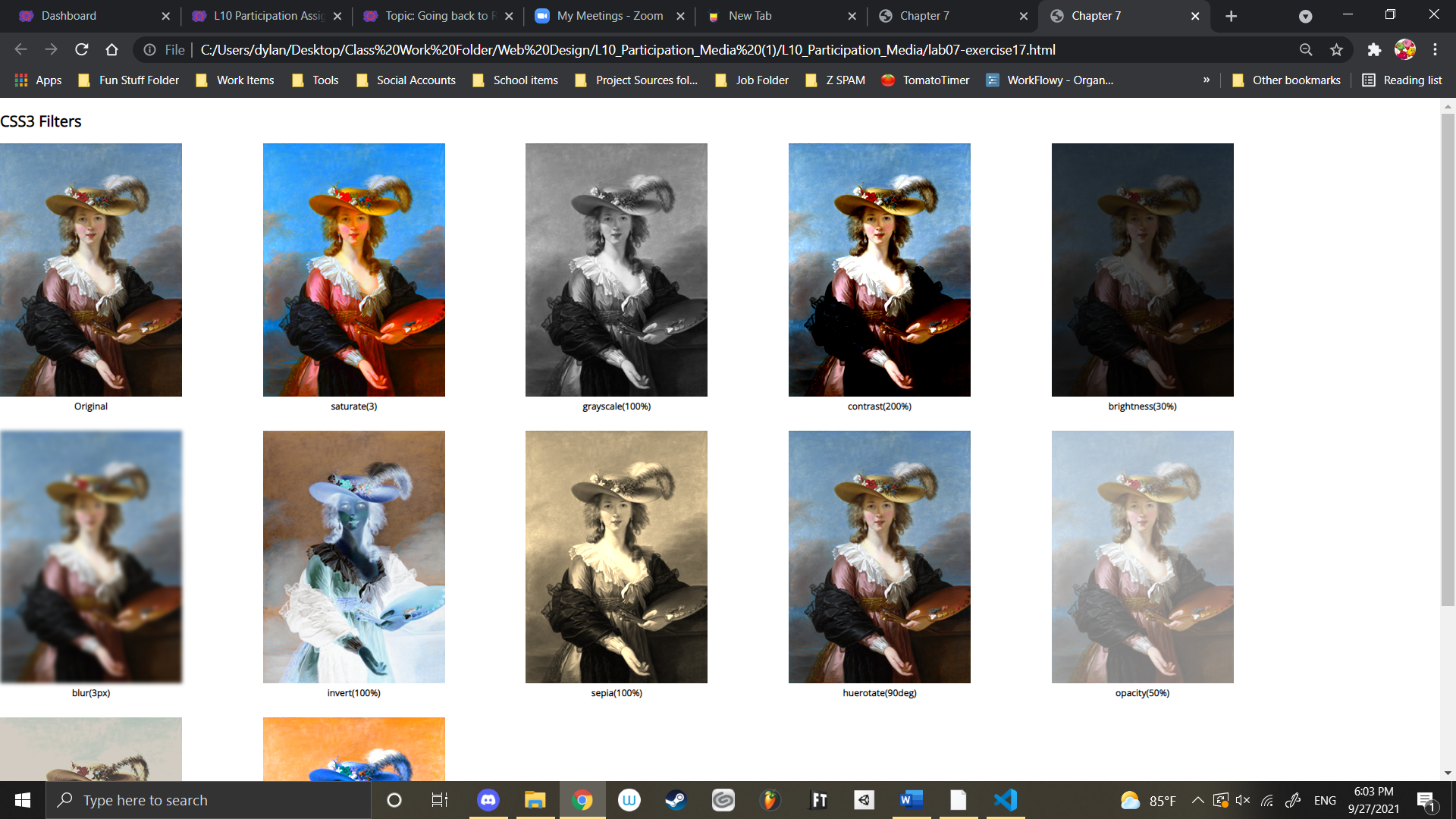
|  |
| --- |
| Exercise 7. — Transitions |
| 1 | Open, examine, and test lab07-exercise15.html.  In this example, we will add a transition to the button. | |
| 2 | Modify the following style in lab07-exercise15.css.  button {  height: 3em;  ...  background-color: #146d37;  transition-property: background-color;  transition-duration: 1s;  transition-timing-function: ease-out;  transition-delay: 0s;  }  This tells the browser to transition the background-color property. For this to work, however, we have to also indicate a different value of the background-color property to transition to or from. | |
| 3 | Add the following style to the end of lab07-exercise15.css and test.  button:hover {  background-color: #60b946;  } | |



|  |
| --- |
| Exercise 7. — More Transitions |
| 1 | Open, examine, and test lab07-exercise16.html.  Notice the different hover state of the images. This is accomplished via four property settings in the figure:hover selector (see next step). | |
| 2 | Modify the following style in lab07-exercise16.css and test.  figure:hover {  background-color: #263238;  color: white;  box-shadow: 10px 10px 32px -4px rgba(0,0,0,0.75);  transform: scale(1.75);  transition: all 1s ease-in 0.25s;  }  This transitions all four of these properties. When you test it you will notice the transition happens on the hover over, but once you move the mouse off the image, it returns immediately to the non-hover property values. Let’s add a transition to the figure as well. | |
| 3 | Modify the following style in lab07-exercise16.css and test.  figure {  margin: 1em;  ...  align-items: center;  justify-content: center;  transition: all 0.6s ease-out 0.25s;  } | |



|  |
| --- |
| Exercise 7. — Filters |
| 1 | Open, examine, and test lab07-exercise17.html.  Notice that each image (other than the first) has a different assigned CSS class. In this exercise we will define these classes to experiment with the different CSS3 filters. | |
| 2 | Add the following style to the end of lab07-exercise17.css and test.  .saturate {  filter: saturate(3);  -webkit-filter: saturate(3);  }  At the time of writing (February 2017), the additional specification with the -webkit- prefix was necessary for Chrome browsers. You should test with and without this extra prefix and see if it works now without it. If so, then you can remove the additional prefix line in the following steps. | |
| 3 | Add the following style and test.  /\* preview the difference \*/  img:hover {  filter:none;  -webkit-filter:none;  }  To help you better see the difference the filter makes, this style definition removes the filter from the filtered image when you hover over it. | |
| 4 | Add the following styles and test.  .grayscale {  filter: grayscale(100%);  -webkit-filter: grayscale(100%);  }  .contrast {  filter: contrast(160%);  -webkit-filter: contrast(160%);  }  .brightness {  filter: brightness(30%);  -webkit-filter: brightness(30%);  }  .blur {  filter: blur(3px);  -webkit-filter: blur(3px);  } | |
| 5 | Add the following styles and test.  .invert {  filter: invert(100%);  -webkit-filter: invert(100%);  }  .sepia {  filter: sepia(100%);  -webkit-filter: sepia(100%);  }  .huerotate {  hue-rotate(90deg);  -webkit-filter: hue-rotate(90deg);  }  .filter-opacity {  filter: opacity(50%);  -webkit-filter: opacity(50%);  } | |
| 6 | Add the following styles and test.  .combo-1 {  filter: brightness(1.5) contrast(3) grayscale(0.6) invert(0.23)  sepia(0.2);  -webkit-filter: brightness(1.5) contrast(3) grayscale(0.6) invert(0.23)  sepia(0.2);  }  .combo-2 {  filter: brightness(1.3) contrast(1.1) hue-rotate(180deg) saturate(2);  -webkit-filter: brightness(1.3) contrast(1.1) hue-rotate(180deg)  saturate(2);  }  This example shows that you can combine multiple filters in once property setting. | |



## 