Styling Web Pages with CSS

1 Introduction

Web pages are plain text documents formatted with HTML. Different HTML elements -- or tags -- format plain text to change their appearance. For instance, heading tags format their text to be much larger than their surrounding plain text. Paragraph tags add vertical spacing. List tags enumerate or bullet the line items they surround. Table tags can format data into rows and columns. Other than basic appearance and layout, HTML does not offer much else to make a Web page more appealing or distinguished from other Web pages.

To make Web pages stand out, we must use ***CSS*** (***Cascading Style Sheets***), a computer language that can describe the appearance of an existing HTML document. CSS can refer to different parts of a document and configure various appearance attributes such as foreground and background color, font, alignment, spacing, borders, paddings, etc. This assignment will give us a chance to learn about how to use CSS to style Web pages. CSS is a powerful language allowing complex Web page styling and layout. Various patterns and best practices have evolved over time in the industry that have become popular. Some of these have been collected into commercial and open source libraries such as:

- Bootstrap
- Foundation
- Tailwind
- Material Design
- Bulma

All these libraries define a set of ***CSS rules*** that you can be readily applied to achieve a professional look and feel, powerful layouts, and ***responsive designs***. Using a library consists of becoming familiar with the CSS rules and applying them appropriately to your HTML to achieve a particular purpose. For this course we are going to be using ***Bootstrap*** throughout our assignments. Feel free to explore and use other libraries for your ***final project***.

Assignments in this course contain three main sections: ***Labs***, ***Kanbas***, and ***Graduates***. The ***Labs*** section in this assignment will give you an opportunity to practice the concepts described in this assignment, i.e., ***CSS*** and ***Bootstrap***. Once you've had a chance to practice, in the ***Kanbas*** section you'll be asked to apply what you've learned to build a Website called ***Kanbas***, inspired on a popular learning management system. Create a new branch called ***a2*** and do all your work there. When done, add, commit and push the branch to GitHub. Deploy the new branch to ***Netlify*** and confirm it's available in a new URL based on the branch name ***a2***.

2 Learning objectives

- Styling Web content with ***Cascading Style Sheets*** (***CSS***)
- Layout and responsive design with ***Bootstrap***
- Laying out Webpages with ***Bootstrap***
- Use ***Fontawesome*** icons

3 Labs

This section presents several CSS, Bootstrap, and Fontawesome exercises to practice and learn how to style HTML documents. Use the same project you created last assignment. After you work through the exercises you will apply the skills to create ***Kanbas*** on your own. Using ***IntelliJ*** or ***VS Code***, open the project you created in the previous assignment (***kanbas-react-web-app***), and do all your work under the ***public*** directory of your project. Under the ***public/labs*** directory,

create a new directory called ***a2*** and create ***index.html*** under ***public/labs/a2***. For this assignment, do all your work in ***public/labs/a2/index.html***. To make it easy on the TAs, add a link to this new ***index.html*** file in ***public/index.html***.

3.1 Styling Webpages with CSS (Cascading Style Sheets)

3.1.1 Styling a document's look and feel

3.1.1.1 Styling elements with the style attribute

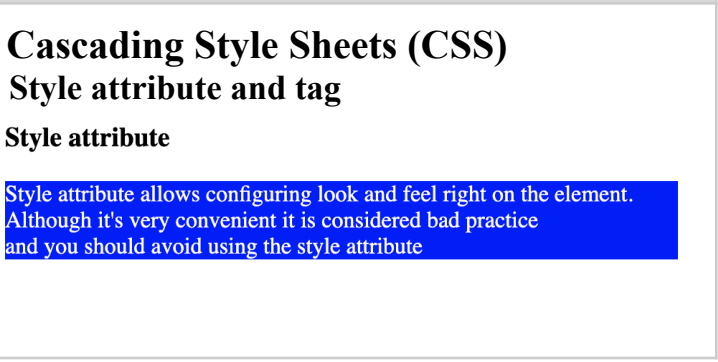
An HTML element's ***style*** attribute can configure the look and feel of the element by changing the values of its style properties as shown below.

```
<element style="property1: value1; property2: value2">
  element body
</element>
```

Examples of properties ***property1*** and ***property2*** are foreground color, background color, font size, etc. The value of the ***style*** attribute is a string formatted with a language called ***CSS*** or ***Cascading Style Sheets*** and is the focus of this assignment. To practice using the ***style*** attribute, copy and paste the example below into ***public/labs/a2/index.html***

/public/labs/a2/index.html

```
<html>
  <head></head>
  <body>
    <h1>Cascading Style Sheets (CSS)</h1>
    <h2>Style attribute and tag</h2>
    <h3>Style attribute</h3>
    <p style="background-color: blue; color: white">
  Style attribute allows configuring look and feel right on
the element. Although it's very convenient it is considered
bad practice and you should avoid using the style attribute
    </p>
  </body>
</html>
```



In the exercise above we styled the paragraph element with its ***style*** attribute. We changed the color of its background by setting the ***background-color*** property to ***blue*** and also changing the foreground color to white by setting the ***color*** property to ***white***. There are 100s of style attributes of which we'll cover the most relevant.

3.1.1.2 Styling documents with the style tag

A slightly better way to configure the look and feel of a Web page is to declare ***CSS rules*** in the ***<style>*** tag in the ***<head>*** and then refer to content by their ***selector***, e.g., name, ID, or class. It's better because you can control the style of a whole page from a central place instead of dealing with individual elements.

```
<style>
   selector1 {
       property1: value1;
       property2: value2;
   }
</style>
```

To practice using the style tag, copy and paste the highlighted style tag below at the end of the head tag as shown.

/public/labs/a2/index.html

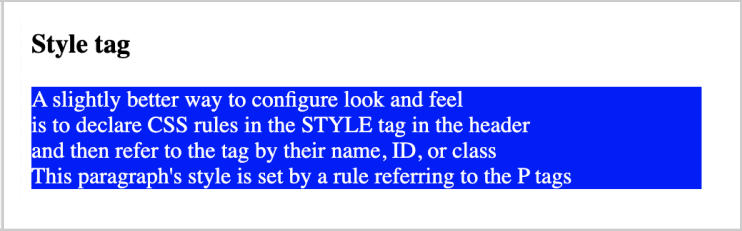
```
<head>
   <style>                                    <!-- new style tag
       p {                                         selector "p" refers to all paragraphs
           background-color: blue;                 sets all paragraph background color to blue
           color: white;                           sets all paragraph foreground color to white
       }                                      -->
   </style>
</head>
```

Then copy the following paragraph at the end of the ***public/labs/a2/index.html*** document. The paragraph should render with a

blue background and white text as shown below.

index.html

```
<h3>Style tag</h3>
<p>
A slightly better way to configure look and feel
is to declare CSS rules in the STYLE tag in the
header and then refer to the tag by their name,
ID, or class This paragraph's style is set by a
rule referring to the P tags
</p>
```



3.1.1.3 Linking CSS styling documents with the link tag

Although the style tag is slightly better than the style attributes, styling an entire website will entail editing many style tags in their HTML documents. Instead of changing styles inside many HTML documents, it is a ***best practice*** to do all styling configuration in separate CSS files and then link the files from the HTML document with the ***link*** tag. To practice using the ***link*** tag create a brand new file called ***public/labs/a2/index.css*** in the same directory of the ***public/labs/a2/index.html*** document, and copy the following content.

index.css

```
p {
   background-color: blue;            /* This is the same content currently in the style tag. */
   color: white;                      /* We'll use this file from now on for this assignment  */
}
```

Then, as shown below, comment out the ***style*** tag in ***index.html*** highlighted in red, since we won't be using it anymore -- ever. Instead, use the ***link*** tag to load the ***index.css*** file created earlier, as highlighted in green.

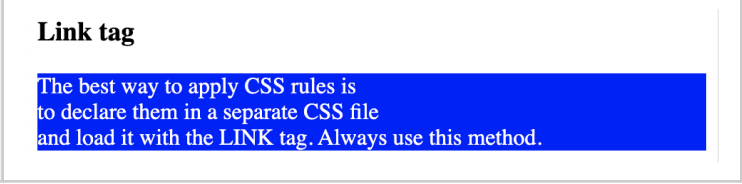
/public/labs/a2/index.html

```
<head>
   <!--style>                                  <!--
       p {                                          Comment out, or remove, the style tag.
           background-color: blue;                  We moved this content to the index.css file
           color: white;                       -->
       }
   </style-->
   <link href="index.css"                      <!-- Instead load the index.css
         rel="stylesheet"/>                         file created earlier -->
</head>
```

Finally copy the following content to the end of ***index.html*** which should render a blue and white paragraph as shown.

/public/labs/a2/index.html

```
<h3>Link tag</h3>
<p>
   The best way to apply CSS rules is to declare
   them in a separate CSS file and load it with
   the LINK tag. Always use this method.
</p>
```



3.1.2 Selecting content to style with selectors

3.1.2.1 Selecting content with *ID* selectors

The CSS rules in previous exercises styled all paragraphs at once by using the name of the tag ***p*** and then specifying the style property values. Instead of changing the look and feel of all the elements of the same name, e.g., ***p***, we can refer to a specific element by their ID, an attribute specifying a unique identifier. To practice using ID selectors, in ***index.css***, comment out the

highlighted paragraph CSS rule as shown and add the two CSS rules referring to paragraphs with IDs ***id-selector-1*** and ***id-selector-2***.

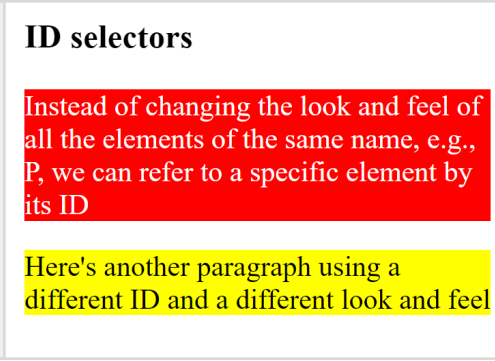
index.css

```
/*p {
 background-color: blue;
 color: white;}*/
p#id-selector-1 {
 background-color: red;
 color: white;
}
p#id-selector-2 {
 background-color: yellow;
 color: black;
}
```

index.html

```
<h3>ID selectors</h3>
<p id="id-selector-1">
 Instead of changing the look and
 feel of all the elements of the
 same name, e.g., P, we can refer
 to a specific element by its ID
</p>
<p id="id-selector-2">
 Here's another paragraph using a
 different ID and a different look
 and feel
</p>
```

Browser



3.1.2.2 Selecting content with ***class*** selectors

Instead of using IDs to refer to specific elements, you can use an element's ***class*** attribute instead, or a combination of both. Class selectors can be used just like ID selectors if you keep them unique, but can also be used to apply the same style to several elements, even if they are different types of elements. We will be using class selectors exclusively from now on. To practice using class selectors, copy the CSS rule below into ***index.css***, and the HTML at the end of ***index.html***.

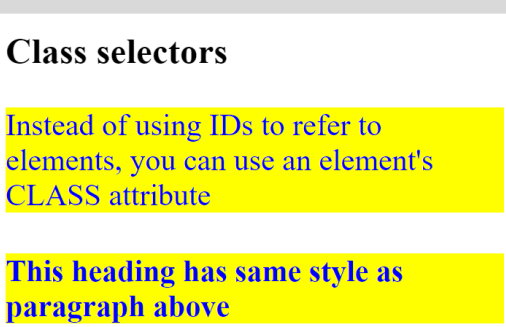
index.css

```
.wd-class-selector {
 background-color: yellow;
 color: blue;
}
```

index.html

```
<h3>Class selectors</h3>
<p class="wd-class-selector">
 Instead of using IDs to refer
 to elements, you can use an
 element's CLASS attribute</p>
<h4 class="wd-class-selector">
 This heading has same style as
 paragraph above</h4>
```

Browser



The example above declares a selector that declares a style that transforms the background and foreground color. We can then use the selector to apply the transformation to several elements. The above example applies the style to two elements, the paragraph and the heading.

3.1.2.3 Selecting content based on the document structure

Selectors can be combined to refer to elements in particular places in the document. A set of selectors separated by a space can refer to elements in a hierarchy. For instance: ***.selector1 .selector2 { … }*** refers to an element whose class is ***.selector2*** and is inside some ***descendant*** of another element whose class is ***.selector1***. If we use ">" instead to separate the classes, then we can establish a direct parent/child relationship. To practice selecting elements using a set of selectors, copy the following content in ***index.html***

/public/labs/a2/index.html

```
<div class="wd-selector-1">
 <h3>Document structure selectors</h3>
 <div class="wd-selector-2">
  Selectors can be combined to refer elements in
  particular places in the document
  <p class="wd-selector-3">
   This paragraph's red background is referenced as
   <br/>.selector-2 .selector3<br/>
   meaning the descendant of some ancestor.
   <br/>
```

```
<!-- this is parent element with selector
  .wd-selector-1
  .wd-selector-2 is a direct child of
  .wd-selector-1
  .wd-selector-3 is a descendant of
  .wd-selector-1 and a direct child of
  .wd-selector-2
```

```
   <span class="wd-selector-4">Whereas this span is a
     direct child of its parent</span>                    this is a descendant of .selector-1 and
   <br/>                                                  .wd-selector-2 and a direct child of
   You can combine these relationships to create          .wd-selector-3
   specific styles depending on the document             -->
   structure
  </p>
 </div>
</div>
```

Let's now style elements .wd-selector-3 and .wd-selector-4. Copy the CSS below into ***index.css***.

/public/labs/a2/index.css

```
.wd-selector-1 .wd-selector-3 {          /* refers to .wd-selector-3 as a descendant of .wd-selector-1   */
   background-color: red;
   color: white;
}
.wd-selector-2 > .wd-selector-3 >
.wd-selector-4 {                         /* refers to .wd-selector-4 as a direct child of .wd-selector-3 */
   background-color: yellow;             /* which is a direct child of .wd-selector-2                    */
   color: blue;
}
```

3.1.3 Styling color

3.1.3.1 Styling the foreground color

Foreground colors can be configured using the CSS ***color*** property as follows

```
.some-css-selector {                        /* selects some DOM element    */
   color: blue;                             /* sets color property to blue */
}
```

Colors can be defined as follows

- As strings, e.g., white, red, blue, etc
- As hexadecimals, e.g., #ABCDEF
- As RGB, e.g., rgb(12, 34, 56)

Here are a couple examples:

`.the-sun { color: rgb(255,255,0); }`	`.the-sky { color: blue; }`	`.ketchup { color: #FF0000; }`

To practice working with foreground colors, copy the CSS rules below into ***index.css*** to declare several useful color classes. Copy the HTML code below into ***index.html***. Confirm it renders as shown.

index.css

```
.wd-fg-color-black { color: black; }
.wd-fg-color-white { color: white; }
.wd-fg-color-blue { color: #7070ff; }
.wd-fg-color-red { color: #ff7070; }
.wd-fg-color-green { color: green; }
```

index.html

```
<h2>Colors</h2>
<h3 class="wd-fg-color-blue">Foreground color</h3>
<p class="wd-fg-color-red">
  The text in this paragraph is red but
  <span class="wd-fg-color-green">this text is green</span>
</p>
```

Foreground color

The text in this paragraph is red but this text is green

3.1.3.2 Styling the background color

To practice working with background colors, copy the CSS rules shown below into ***index.css*** and the HTML code into ***index.html***. Confirm the browser renders as shown.

index.css

```
.wd-bg-color-yellow {
   background-color: #ffff07;
}
.wd-bg-color-blue {
   background-color: #7070ff;
}
.wd-bg-color-red {
   background-color: #ff7070;
}
.wd-bg-color-green {
   background-color: green;
}
```

index.html

```
<h3 class="wd-bg-color-blue wd-fg-color-white">Background color</h3>
<p class="wd-bg-color-red wd-fg-color-black">
   This background of this paragraph is red but
   <span class="wd-bg-color-green wd-fg-color-white"> the background
  of this text is green and the foreground white</span>
</p>
```

Background color

This background of this paragraph is red but the background of this text is green and the foreground white

3.1.4 Styling the box model

3.1.4.1 Styling borders

Use CSS border properties to configure the look and feel of the border around content. Here's a sample of the properties that can be configured.

```
.some-selector {
  border-width: 10px;
  border-style: solid|dotted|dashed|double;
  border-color: red | blue ...;
}
```

```
/* configure border with several properties*/
/* border's width. Can also provide per border*/
/* the style of the border*/
/* the color of the border     */
```

To practice styling borders, copy the CSS code below into ***index.css*** and the HTML code into ***index.html*** and confirm the browser renders as shown.

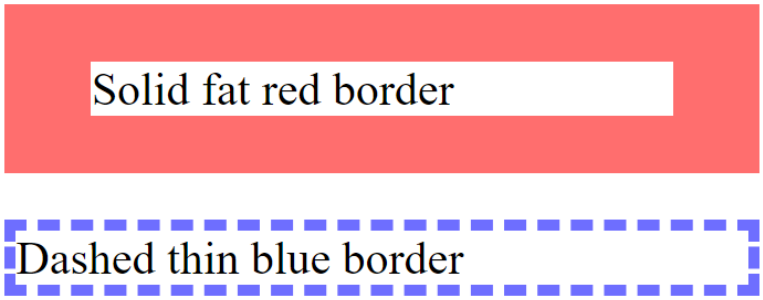
/public/labs/a2/index.css

```
.wd-border-fat {
   border-width: 20px 30px 20px 30px; }
.wd-border-thin {
   border-width: 4px; }
.wd-border-solid {
   border-style: solid; }
.wd-border-dashed {
   border-style: dashed; }
.wd-border-yellow {
   border-color: #ffff07; }
.wd-border-red {
   border-color: #ff7070; }
.wd-border-blue {
```

```
<h2>Borders</h2>
<p class="wd-border-fat wd-border-red wd-border-solid">
   Solid fat red border</p>
<p class="wd-border-thin wd-border-blue wd-border-dashed">
   Dashed thin blue border</p>
```

Borders

```
  border-color: #7070ff; }
```



3.1.4.2 Staying margins and paddings

You can also configure the spacing between elements. To practice with paddings, copy the CSS code below into ***index.css*** and the HTML into ***index.html***. Confirm browser renders as shown.

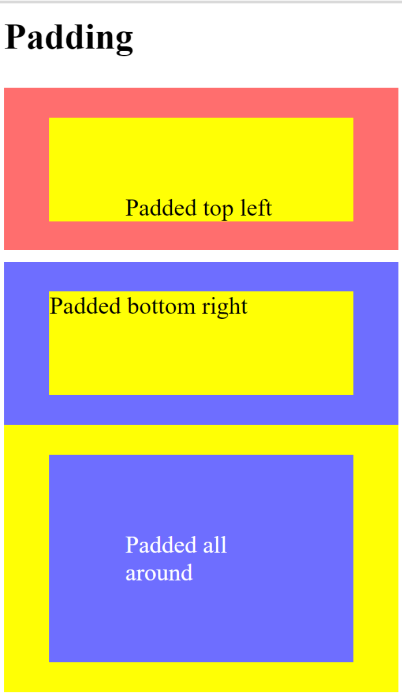
index.css

```
.wd-padded-top-left {
  padding-top: 50px;
  padding-left: 50px;
}


.wd-padded-bottom-right {
  padding-bottom: 50px;
  padding-right: 50px;
}

.wd-padding-fat {
  padding: 50px;
}
```

index.html

```
<h2>Padding</h2>
<div class="wd-padded-top-left wd-border-fat
            wd-border-red wd-border-solid
            wd-bg-color-yellow">
   Padded top left
</div>

<div class="wd-padded-bottom-right wd-border-fat
            wd-border-blue wd-border-solid
            wd-bg-color-yellow">
   Padded bottom right
</div>
<div class="wd-padding-fat wd-border-fat
            wd-border-yellow wd-border-solid
            wd-bg-color-blue wd-fg-color-white">
   Padded all around
</div>
```

Browser



To practice with margins, copy the CSS code below into ***index.css*** and the HTML into ***index.html***. Confirm browser renders as shown.

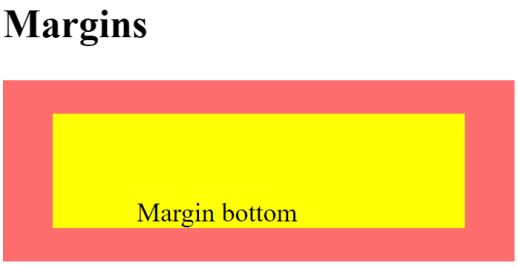
index.css

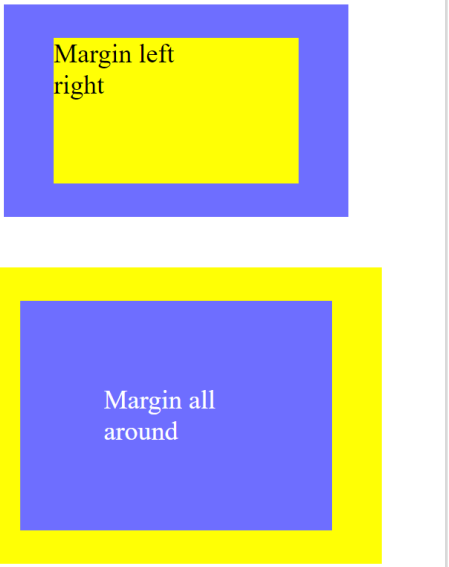
```
.wd-margin-bottom {
  margin-bottom: 50px;
}

.wd-margin-right-left {
  margin-left: 50px;
  margin-right: 50px;
}

.wd-margin-all-around {
  margin: 30px;
}
```

index.html

```
<h2>Margins</h2>
<div class="wd-margin-bottom
            wd-padded-top-left
            wd-border-fat border-red
            wd-border-solid
            wd-bg-color-yellow">
   Margin bottom</div>
<div class="wd-margin-right-left
            wd-padded-bottom-right
            wd-border-fat border-blue
            wd-border-solid
            wd-bg-color-yellow">
   Margin left right
</div>
```

Browser

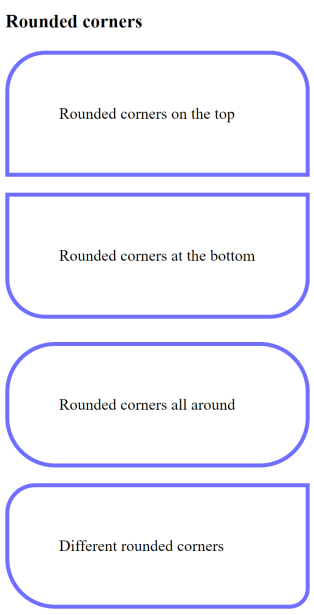


	<div class="wd-margin-all-around wd-padding-fat border-fat wd-border-yellow wd-border-solid wd-bg-color-blue wd-fg-color-white"> Margin all around 	

3.1.4.3 Staying corners

You can configure the corners of element borders to be rounded. Either all of them at once or specific corners. You can do this by configuring a border's radius. To practice rounding some corners, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

index.css	*index.html*	*Browser*

.wd-rounded-corners-top {
border-top-left-radius: 40px;
border-top-right-radius: 40px;
}
 	
.wd-rounded-corners-bottom {
border-bottom-left-radius: 40px;
border-bottom-right-radius: 40px;
}
 	
.wd-rounded-corners-all-around {
border-radius: 50px;
}
 	
.wd-rounded-corners-inline {
border-radius: 30px 0px 20px 50px;
}
	Rounded corners
	
	Rounded corners on the top
	
	Rounded corners at the bottom
	
	Rounded corners all around
	
	Different rounded corners
	
		

3.1.5 Styling dimensions

You can configure an element's dimensions with ***width*** and ***height*** properties. To practice setting element's dimensions, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

index.css	*index.html*	*Browser*

`.wd-dimension-portrait {`
`  width: 75px;`
`  height: 100px;`
`}`
 	
`.wd-dimension-landscape {`
`  width: 100px;`
`  height: 75px;`
`}`
 	
`.wd-dimension-square {`
`  width: 75px;`
`  height: 75px;`
`}`
	`<h2>Dimension</h2>`
	`<div>`
	`  <div class="wd-dimension-portrait wd-bg-color-yellow">`
	`  Portrait`
	`  </div>`
	`  <div class="wd-dimension-landscape wd-bg-color-blue`
	`  wd-fg-color-white">`
	`  Landscape`
	`  </div>`
	`  <div class="wd-dimension-square wd-bg-color-red">`
	`  Square`
	`  </div>`
	`</div>`
		

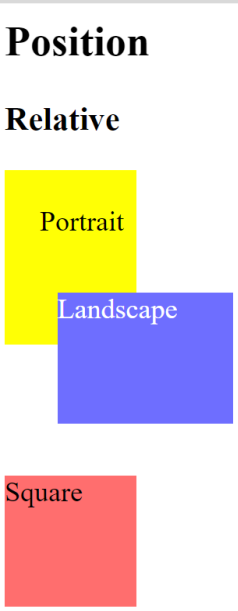
3.1.6 Styling positions

You can configure an element's position with the ***position*** property. The property has many possible values, but we'll explore ***relative***, ***absolute***, and ***static***.

3.1.6.1 Styling relative position

Setting ***position*** property to ***relative*** allows moving the element relative to its original position. To practice setting element's relative position, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

index.css	*index.html*	*Browser*

`.wd-pos-relative-nudge-up-right {`
`  position: relative;`
`  bottom: 30px;`
`  left: 30px;`
`}`
`.wd-pos-relative-nudge-down-right {`
`  position: relative;`
`  top: 20px;`
`  left: 20px;`
`}`
`.wd-pos-relative {`
`  position: relative;`
`}`
	`<h2>Position</h2>`
	`<h3>Relative</h3>`
	`<div>`
	`  <div class="wd-bg-color-yellow`
	`  wd-dimension-portrait">`
	`  <div class="wd-pos-relative-nudge-down-right">`
	`  Portrait</div>`
	`  </div>`
	`  <div class="wd-pos-relative-nudge-up-right`
	`  wd-bg-color-blue wd-fg-color-white`
	`  wd-dimension-landscape">`
	`  Landscape</div>`
	`  <div class="wd-bg-color-red wd-dimension-square">`
	`  Square</div>`
	`</div>`
		

3.1.6.2 Styling absolute position

Setting ***position*** property to ***absolute*** allows moving the element relative to the position of its parent. To practice setting element's absolute position, copy the CSS and HTML below into ***index.css*** and ***index.html***. Notice several ***
*** elements were added at the end of the example to make room for the next exercise.

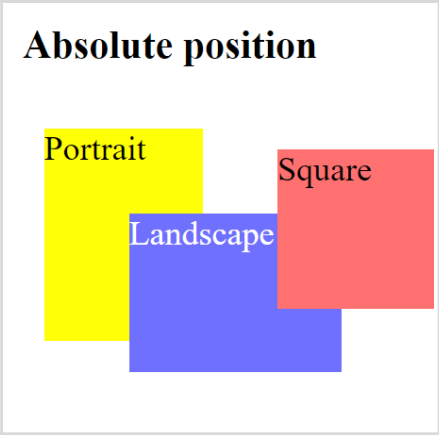
index.css

```
.wd-pos-absolute-10-10 {
   position: absolute;
   top: 10px;
   left: 10px;
}
.wd-pos-absolute-50-50 {
   position: absolute;
   top: 50px;
   left: 50px;
}
.wd-pos-absolute-120-20 {
   position: absolute;
   top: 20px;
   left: 120px;      }
```

index.html

```
<h3>Absolute position</h3>
<div class="wd-pos-relative">
   <div class="wd-pos-absolute-10-10
    wd-bg-color-yellow wd-dimension-portrait">
      Portrait
   </div>
   <div class="wd-pos-absolute-50-50
        wd-bg-color-blue wd-fg-color-white
        wd-dimension-landscape">
       Landscape   </div>
   <div class="wd-pos-absolute-120-20
        wd-bg-color-red wd-dimension-square">
       Square      </div>
</div><br/><br/><br/><br/><br/><br/><br/>
```

Browser



3.1.6.3 Styling fixed position

Setting ***position*** property to ***fixed*** allows setting the element relative to the window. That means that if you scroll the content of the page, the element will not scroll with it. To practice setting element's fixed position, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown. Your display may be different depending on the actual size of the screen and scrolling.

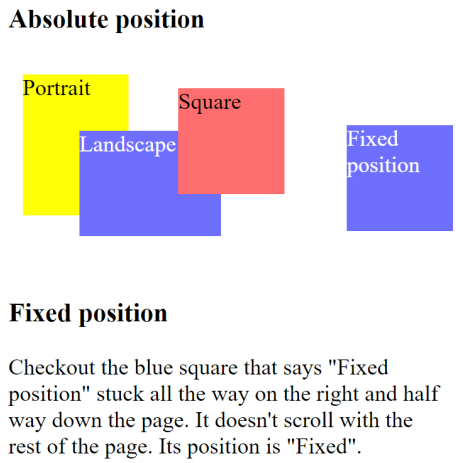
index.css

```
.wd-pos-fixed {
  position: fixed;
  right: 0px;
  bottom: 50%;
}
```

index.html

```
<h3>Fixed position</h3>

  Checkout the blue square that says
  "Fixed position" stuck all the way
  on the right and half way down the
  page. It doesn't scroll with the
  rest of the page. Its position is
  "Fixed".

<div class="wd-pos-fixed
            wd-dimension-square
            wd-bg-color-blue
            wd-fg-color-white">
  Fixed position
</div>
```

Browser



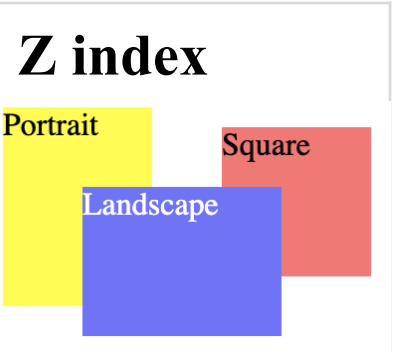
3.1.6.4 Styling z-index (1pt)

When the browser renders content declared in HTML documents, it calculates positions and dimensions so every element has a dedicated rectangle on the window. Typically elements don't fall on top of each other. When you start moving elements with ***position***, then overlapping elements are possible. By default elements are rendered in the order declared in HTML documents. Elements declared later render on top of elements declared earlier. The ***z-index*** CSS property overrides this behavior. Default value of ***z-index*** is ***auto***, which corresponds to 0. Increasing z-index can make elements render later, or on top of, others. To practice setting an element's ***z-index***, copy the CSS and HTML below into ***index.css*** and ***index.html***

index.css *index.html* *Browser*

```css
.wd-zindex-bring-to-front {
   z-index: 10;
}
```

```html
<h2>Z index</h2>
<div class="wd-pos-relative">
   <div class="wd-pos-absolute-10-10
        wd-bg-color-yellow
        wd-dimension-portrait">
        Portrait</div>
   <div class="wd-zindex-bring-to-front
        wd-pos-absolute-50-50
        wd-bg-color-blue wd-fg-color-white
        wd-dimension-landscape">
        Landscape</div>
   <div class="wd-pos-absolute-120-20
        wd-bg-color-red wd-dimension-square">
        Square</div>
</div><br/><br/><br/><br/><br/><br/><br/>
```



3.1.6.5 Floating Content

HTML does not support laying out content horizontally. The CSS float property allows fixing that. To practice laying out content horizontally, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

index.css

```css
.wd-float-left {
   float: left;
}

.wd-float-right {
   float: right;
   height: 100px;
}

.wd-float-done {
   clear: both;
}
```

index.html

```html
<h2>Float</h2>
<div>
   <div class="wd-float-left wd-dimension-portrait wd-bg-color-yellow">
        Yellow</div>
   <div class="wd-float-left wd-dimension-portrait wd-bg-color-blue wd-fg-color-white">
        Blue</div>
   <div class="wd-float-left wd-dimension-portrait wd-bg-color-red">
        Red</div>
   <img class="wd-float-right"
src="https://www.staradvertiser.com/wp-content/uploads/2021/08/web1_Starship-gap2.jpg"/>
   <div class="wd-float-done"></div>
</div>
```

3.1.6.6 Laying out content in a grid (4pts)

Using float we can implement a grid layout made up of rows and columns. To practice laying out content in a grid, copy the CSS and HTML below into ***index.css*** and ***index.html*** and confirm the browser renders as shown.

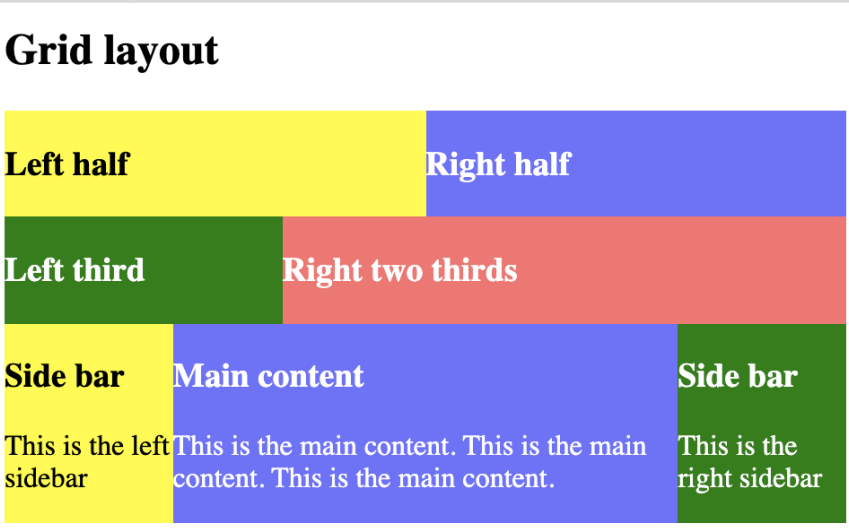
index.css ***index.html***

```css
.wd-grid-row {
   clear: both;
}

.wd-grid-col-half-page {
   width: 50%;
   float: left;
}
```

```html
<h2>Grid layout</h2>
<div class="wd-grid-row">
  <div class="wd-grid-col-half-page wd-bg-color-yellow">
    <h3>Left half</h3>
  </div>
  <div class="wd-grid-col-half-page wd-bg-color-blue
              wd-fg-color-white">
    <h3>Right half</h3>
  </div>
</div>
```

```css
.wd-grid-col-third-page {
   width: 33%;
   float: left;
}

.wd-grid-col-two-thirds-page {
   width: 67%;
   float: left;
}

.wd-grid-col-left-sidebar {
   width: 20%;
   float: left;
}

.wd-grid-col-main-content {
   width: 60%;
   float: left;
}

.wd-grid-col-right-sidebar {
   width: 20%;
   float: left;
}
```

```html
<div class="wd-grid-row">
  <div class="wd-grid-col-third-page wd-bg-color-green
              wd-fg-color-white">
    <h3>Left third</h3>
  </div>
  <div class="wd-grid-col-two-thirds-page wd-bg-color-red
              wd-fg-color-white">
    <h3>Right two thirds</h3>
  </div>
</div>
<div class="wd-grid-row">
  <div class="wd-grid-col-left-sidebar wd-bg-color-yellow">
    <h3>Side bar</h3>
    <p>This is the left sidebar</p>
  </div>
  <div class="wd-grid-col-main-content wd-bg-color-blue
              wd-fg-color-white">
    <h3>Main content</h3>
    <p>This is the main content. This is the main content.
       This is the main content. </p>
  </div>
  <div class="wd-grid-col-right-sidebar wd-bg-color-green
              wd-fg-color-white">
    <h3>Side bar</h3>
    <p>This is the right sidebar</p>
  </div>
</div>
```



3.1.6.7 Flex

Flexbox Layout (***Flexible Box***, or just ***Flex***) provides a simpler way to layout and distribute content in an HTML document. It all starts with creating a container element that configures the layout and behavior of its child elements. To illustrate some of the features of flex, let's create create a container with display configured to flex as shown below.

index.html	*index.css*	*Browser*

`<h2>Flex</h2>`
`<div class="wd-flex-row-container">`
`  <div class="wd-bg-color-yellow">Column 1</div>`
`  <div class="wd-bg-color-blue">Column 2</div>`
`  <div class="wd-bg-color-red">Column 3</div>`
`</div>`
	`.wd-flex-row-container {`
	`  display: flex;`
	`  flex-direction: row;`
	`}`
		Flex
		Column 1Column 2Column 3

Note how DIV elements inside the container render horizontally as a row of element instead of stacking the elements vertically. Flex simplifies laying out content horizontally. Flex child elements can also be configured to grow and expand to fill into empty spaces. The styling below illustrates how the last column can expand into the empty space to its right.

index.html	*index.css*	*Browser*

`<h2>Flex</h2>`
`<div class="wd-flex-row-container">`
`  <div class="wd-bg-color-yellow">Column 1</div>`
`  <div class="wd-bg-color-blue">Column 2</div>`
`  <div class="wd-bg-color-red wd-flex-grow-1">`
`  Column 3`
`  </div>`
`</div>`
	`.wd-flex-row-container {`
	`  display: flex;`
	`  flex-direction: row;`
	`}`
	`.wd-flex-grow-1 {`
	`  flex-grow: 1;`
	`}`
		Flex
		Column 1Column 2Column 3

The rest of the flex child elements can be configured independently to have specific widths to fit whatever content is needed as shown below.

index.html	*index.css*	*Browser*

`<h2>Flex</h2>`
`<div class="wd-flex-row-container">`
`  <div class="wd-bg-color-yellow wd-width-75px">`
`  Column 1`
`  </div>`
`  <div class="wd-bg-color-blue">Column 2</div>`
`  <div class="wd-bg-color-red wd-flex-grow-1">`
`  Column 3`
`  </div>`
`</div>`
	`.wd-flex-row-container {`
	`  display: flex;`
	`  flex-direction: row;`
	`}`
	`.wd-flex-grow-1 {`
	`  flex-grow: 1;`
	`}`
	`.wd-width-75px {`
	`  width: 75px;`
	`}`
		Flex
		Column 1 Column 2Column 3

3.2 Styling Webpages with the Bootstrap CSS Library

This section presents exercises of how to use the ***Bootstrap*** CSS library to style and layout Web pages. Keep working under the ***public*** directory of your project in the same HTML document ***public/labs/a2/index.html***. To make it easy on the TAs, make sure there's a link to this new ***index.html*** file in ***public/index.html***.

3.2.1 Installing bootstrap

Bootstrap can be installed with npm from the root of you project as follows.

```
$ npm install bootstrap                    # install bootstrap from the root of your project
```

The ***bootstrap*** CSS library will be downloaded from ***npm*** and installed in you ***node_modules*** directory located at the root of your project. Once installed, copy the ***bootstrap.min.css*** file located in ***node_modules/bootstrap/dist/css/bootstrap.min.css*** into a new folder ***public/libs/bootstrap***. You can then import the ***bootstrap*** library from your ***HTML*** documents using the following ***link*** tag

```
<link href="/libs/bootstrap/bootstrap.min.css" rel="stylesheet"/>
```

3.2.2 Laying out content with containers and grids

3.2.2.1 Containers

Bootstrap containers establish the root of your HTML document providing a basis of default styles such as the overall margins, paddings, and font of your document. There are two main classes that control container elements: ***.container*** and ***.container-fluid***. The ***.container*** class centers the content with margins on either side and defines several responsive design thresholds. The ***.container-fluid*** class just defines a constant thin margin all around the document. To practice with containers, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown. The heading is not flush with the left hand side and the font is not the default browser font.

index.html	*Browser*

`<body>`
`<div class="container">`
`  <h1>Bootstrap</h1>`
`  <p>Welcome to Bootstrap!</p>`
`  <!-- main content goes here -->`
`</div>`
`</body>`
	Bootstrap
	
	Welcome to Bootstrap!

3.2.2.2 Grid system (2pts)

It's easy to break a page vertical in HTML with the ***p*** and ***div*** tags. It's a little harder to layout content horizontally. Some resort to HTML tables to layout content horizontally using table ***rows*** and ***columns***, but this is generally considered a bad practice. HTML tables should be used for displaying tabular content only, not laying out HTML content. Nevertheless, laying out content like a table is convenient, so to achieve the same functionality as tables, but without tables, we can use CSS instead. Bootstrap provides classes such as ***.row*** and ***.col*** to layout content in a ***grid***. To practice with ***Bootstrap grids***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html	*Comments / Browser*

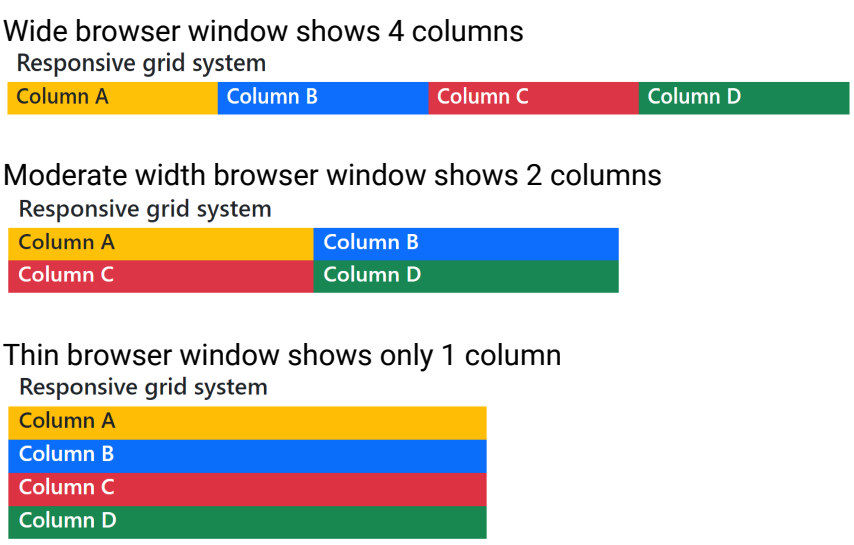
`<h2><a`
`href="https://getbootstrap.com/docs/5.1/layout/grid/">`
`  Grid system</a></h2>`
`<div class="row">`
`  <div class="col bg-danger text-white">`
`  <h3>Left half</h3></div>`
`  <div class="col bg-primary text-white">`
`  <h3>Right half</h3></div></div>`
`<div class="row">`
`  <div class="col-4 bg-warning">`
`  <h3>One thirds</h3></div>`
`  <div class="col-8 bg-success text-white">`
`  <h3>Two thirds</h3></div></div>`
`<div class="row">`
`  <div class="col-2 bg-dark text-white">`
`  <h3>Sidebar</h3></div>`
`  <div class="col-8 bg-secondary text-white">`
`  <h3>Main content</h3></div>`
`  <div class="col-2 bg-info">`
`  <h3>Sidebar</h3></div></div>`
	`<!-- a row containing`
	`  two columns applying class col with default`
	`  width evenly spread over all columns`
	
	`  another row with`
	`  two columns applying classes col-4 and col-8`
	`  where 4 + 8 = 12, the total number of columns`
	`  so 4/12 is ⅓ and 8/12 is ⅔ of the screen`
	`-->`
	Grid system
	Left half Right half
	One thirds Two thirds
	Sideba Main content Sidebar

3.2.2.3 Responsive Grids

Bootstrap grids can adapt to the size of the screen, that is, they can be responsive. We can achieve this by applying more than one ***.col*** class which only applies for a given window size. To practice with ***Bootstrap responsive*** grids, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown. Resize the browser and confirm that the grid shows 4 columns, then 2 and then just 1.

index.html | ***Comments / Browser***

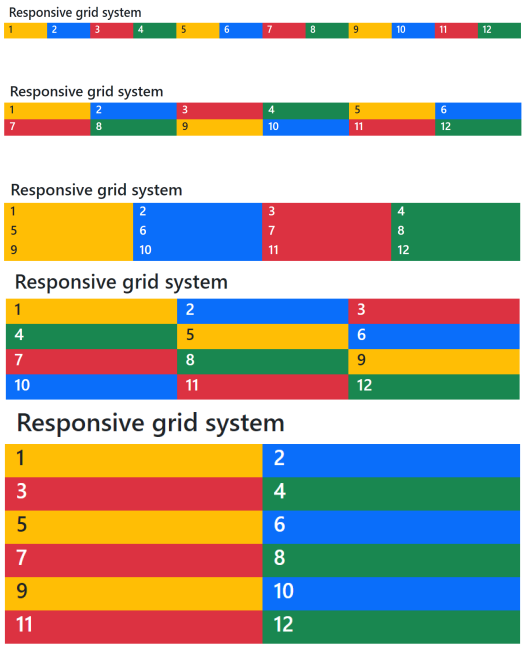
```
<h2>Responsive grid system</h2>
<div class="row">
 <div class="col-12 col-md-6 col-xl-3
bg-warning">
  <h3>Column A</h3>
 </div>
 <div class="col-12 col-md-6 col-xl-3
bg-primary text-white">
  <h3>Column B</h3>
 </div>
 <div class="col-12 col-md-6 col-xl-3
bg-danger text-white">
  <h3>Column C</h3>
 </div>
 <div class="col-12 col-md-6 col-xl-3
bg-success text-white">
  <h3>Column D</h3>
 </div>
</div>
```

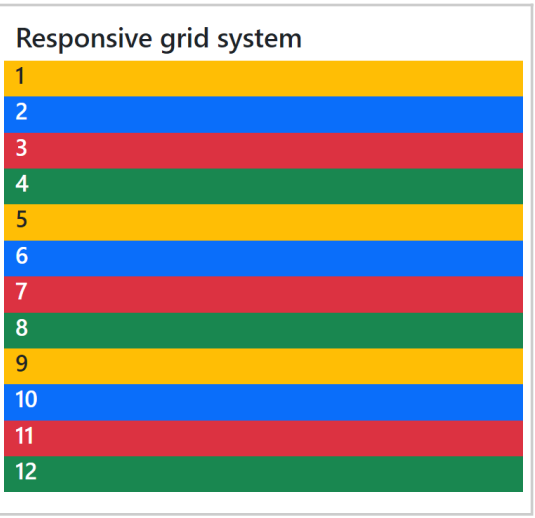


Let's try a more dramatic example by adding more content spread over more columns and rows. Copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm that the column layout changes as you resize the browser window.

index.html | ***Browser***

```
<h2>Responsive grid system</h2>
<div class="row">
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-warning">
   <h4>1</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-primary text-white">
   <h4>2</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-danger text-white">
   <h4>3</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-success text-white">
   <h4>4</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-warning">
   <h4>5</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-primary text-white">
   <h4>6</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-danger text-white">
   <h4>7</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-success text-white">
   <h4>8</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-warning">
   <h4>9</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-primary text-white">
   <h4>10</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-danger text-white">
   <h4>11</h4></div>
 <div class="col-12 col-sm-6 col-md-4 col-lg-3 col-xl-2 col-xxl-1
bg-success text-white">
   <h4>12</h4></div>
</div>
```





3.2.2.4 Hiding and showing responsive content (2pts)

As users shrink or widen the browser window, there may be more or less space to show some content. Bootstrap can configure content to show or hide depending on the width of the screen. As described earlier in Responsive grids, Bootstrap breaks up the screen into 5 sizes: ***extra small***, ***small***, ***medium***, ***large***, ***extra large***, and ***extra extra large***. Use Bootstrap display classes as shown below that illustrates hiding and showing content as you resize the screen. Resize the window and confirm that the content only shows at certain widths.

index.html

```
<h2>Hiding and showing responsive content</h2>
<div class="d-block d-sm-none fa-2x">XS</div>
<div class="d-none d-sm-block d-md-none fa-2x">S</div>
<div class="d-none d-md-block d-lg-none fa-2x">M</div>
<div class="d-none d-lg-block d-xl-none fa-2x">L</div>
<div class="d-none d-xl-block d-xxl-none fa-2x">XL</div>
<div class="d-none d-xxl-block fa-2x">XXL</div>
```

Browser displays	@ window width
XS	<576px
S	≥576px
M	≥768px
L	≥992px
XL	≥1200px
XXL	≥1400px

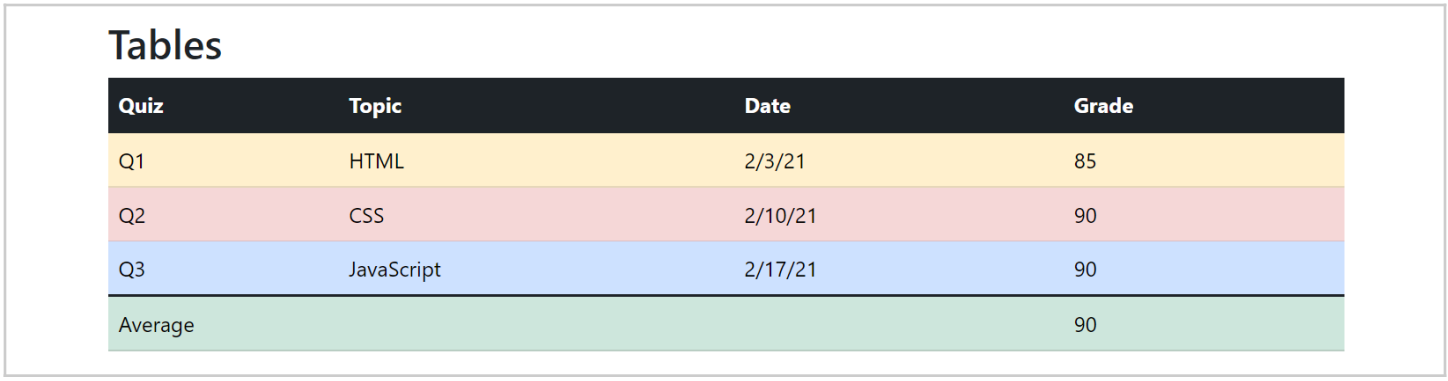
3.2.3 Tables and lists

3.2.3.1 Styling tables

Bootstrap provides several classes that enhance the look and feel of common HTML widgets such as tables, lists, and form elements. Let's start with tables. To practice with styling ***HTML tables***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h2>Tables</h2>
<table class="table">
   <thead>
   <tr class="table-dark"><th>Quiz</th><th>Topic</th><th>Date</th><th>Grade</th></tr>
   </thead>
   <tbody>
   <tr class="table-warning"><td>Q1</td><td>HTML</td><td>2/3/21</td><td>85</td></tr>
   <tr class="table-danger"><td>Q2</td><td>CSS</td><td>2/10/21</td><td>90</td></tr>
   <tr class="table-primary"><td>Q3</td><td>JavaScript</td><td>2/17/21</td><td>90</td></tr>
   </tbody>
   <tfoot>
   <tr class="table-success"><td colspan="3">Average</td><td>90</td></tr>
   </tfoot>
</table>
```



3.2.3.2 Making tables responsive (2pts)

In general it is a good practice to minimize the number of scrollbars shown at any one time in a browser screen. In browsers large amounts of content extends vertically beyond the height of the window, and then scrollbars allow you to access that extra content. Sometimes it is necessary to use additional scrollbars when appropriate such as tables that might be too wide to fit horizontally. Bootstrap provides tables that can show scrollbars when they don't fit in their parent window. To practice with ***Bootstrap responsive tables***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown. Resize the window and confirm that the table shows a horizontal scroll bar when the screen is too small to fit the table comfortably.

index.html

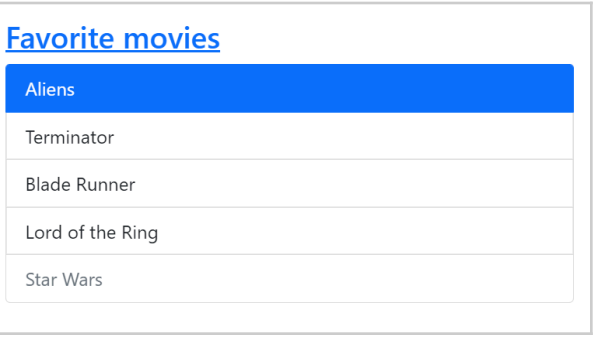
```
<h2>
   <a href="https://getbootstrap.com/docs/5.1/content/tables/#responsive-tables">
       Responsive tables</a></h2>
<div class="table-responsive">
  <table class="table">
    <thead>
      <tr><th>Very</th><th>long</th><th>set</th><th>of</th><th>columns</th>
          <th>Very</th><th>long</th><th>set</th><th>of</th><th>columns</th>
          <th>Very</th><th>long</th><th>set</th><th>of</th><th>columns</th>
      </tr>
    </thead>
    <tbody>
      <tr>
         <td>Very</td><td>long</td><td>set</td><td>of</td><td>columns</td>
         <td>Very</td><td>long</td><td>set</td><td>of</td><td>columns</td>
         <td>Very</td><td>long</td><td>set</td><td>of</td><td>columns</td>
      </tr>
    </tbody>
  </table>
</div>
```

3.2.3.3 Styling Lists

Another set of Bootstrap classes can make simple HTML lists look more presentable. The ***.list-group*** and ***.list-group-item*** classes can be applied to ***ul*** and ***li*** tags correspondingly to make list stand out. The ***.active*** class can be applied to an ***li*** tag to highlight it. To practice with ***Bootstrap lists***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html ***Browser***

```
<h2>
   <a
href="https://getbootstrap.com/docs/5.1/components/list-group/">
       Favorite movies</a>
</h2>
<ul class="list-group">
   <li class="list-group-item active">Aliens</li>
   <li class="list-group-item">Terminator</li>
   <li class="list-group-item">Blade Runner</li>
   <li class="list-group-item">Lord of the Ring</li>
   <li class="list-group-item disabled">Star Wars</li>
</ul>
```

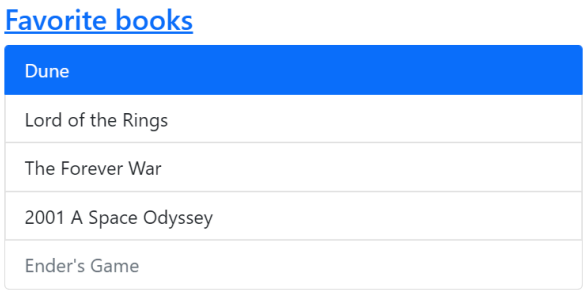


3.2.3.4 Styling a List of Hyperlinks

The same ***.list-group*** and ***.list-group-item*** Bootstrap classes can be applied to ***div*** and ***a*** tags to implement a list of anchor links. To practice with ***Bootstrap hyperlink lists***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown and that the links work.

index.html | ***Browser***

```
<h3>
   <a
href="https://getbootstrap.com/docs/5.1/components/list-group/#links
-and-buttons">
       Favorite books</a>
</h3>
<div class="list-group">
  <a href="https://en.wikipedia.org/wiki/Dune_(novel)"
class="list-group-item list-group-item-action active">
       Dune</a>
   <a href="https://en.wikipedia.org/wiki/The_Lord_of_the_Rings"
class="list-group-item list-group-item-action">
       Lord of the Rings</a>
   <a href="https://en.wikipedia.org/wiki/The_Forever_War"
class="list-group-item list-group-item-action">
       The Forever War</a>
   <a
href="https://en.wikipedia.org/wiki/2001:_A_Space_Odyssey_(novel)"
class="list-group-item list-group-item-action">
       2001 A Space Odyssey</a>
   <a href="https://en.wikipedia.org/wiki/Ender%27s_Game"
class="list-group-item list-group-item-action disabled"
tabindex="-1" aria-disabled="true">Ender's Game</a>
</div>
```



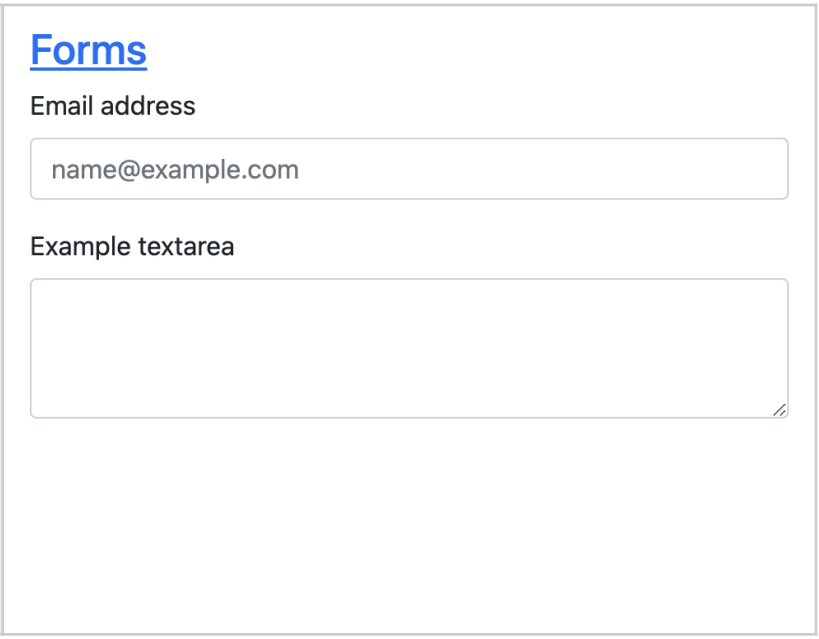
3.2.4 Styling Forms

3.2.4.1 Basic Form Styling

Bootstrap has tons of classes to style form elements especially to make them friendly for mobile Web applications. To practice with ***Bootstrap form classes***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html | ***Browser***

```
<h2>
  <a
href="https://getbootstrap.com/docs/5.1/forms/form-
control/">
    Forms
  </a>
</h2>
<div class="mb-3">
  <label for="input1"
         class="form-label">
    Email address</label>
  <input type="email"
         class="form-control"
         id="input1"
         placeholder="name@example.com">
</div>
<div class="mb-3">
  <label for="textarea1"
         class="form-label">
    Example textarea</label>
  <textarea class="form-control"
            id="textarea1"
            rows="3"></textarea>
</div>
```



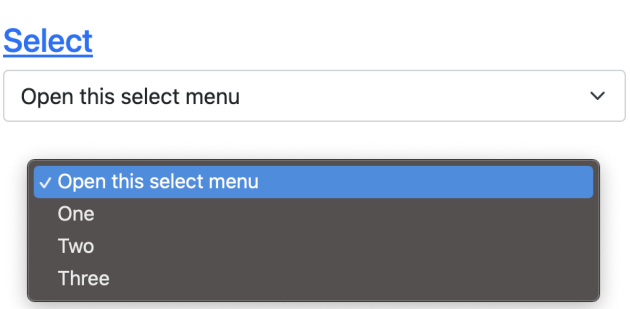
3.2.4.2 Styling Dropdowns

Dropdowns can also be styled professionally. To practice with ***Bootstrap dropdown styling***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h3>
   <a href="https://getbootstrap.com/docs/5.1/forms/select/">
       Dropdowns</a>
</h3>
<select class="form-select">
   <option selected>Open this select menu</option>
   <option value="1">One</option>
   <option value="2">Two</option>
   <option value="3">Three</option>
</select>
```

Browser



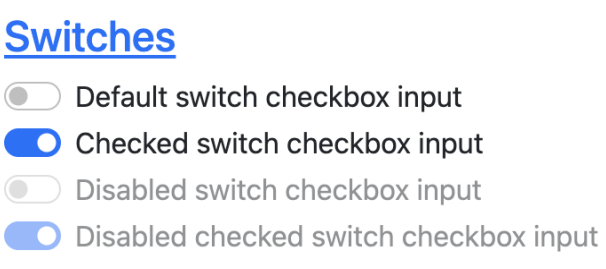
3.2.4.3 Styling Switches

Checkboxes can be styled to look like switches with Bootstrap classes ***.form-check*** and ***.form-switch***. To practice with ***Bootstrap form switches***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h3>
 <a
href="https://getbootstrap.com/docs/5.1/forms/checks-radios/#swi
tches">Switches</a>
</h3>
<div class="form-check form-switch">
 <input class="form-check-input"
        type="checkbox"
        id="switch1">
 <label class="form-check-label" for="switch1">
     Default switch checkbox input</label>
</div>
<div class="form-check form-switch">
 <input class="form-check-input"
        type="checkbox"
```

Browser



```
        id="switch2"
        checked>
 <label class="form-check-label"
        for="switch2">
     Checked switch checkbox input</label>
</div>
<div class="form-check form-switch">
 <input class="form-check-input"
        type="checkbox"
        id="switch3"
        disabled>
 <label class="form-check-label"
        for="switch3">
     Disabled switch checkbox input</label>
</div>
<div class="form-check form-switch">
 <input class="form-check-input"
        type="checkbox"
        id="switch4"
        checked disabled>
 <label class="form-check-label"
        for="switch4">
     Disabled checked switch checkbox input</label>
</div>
```

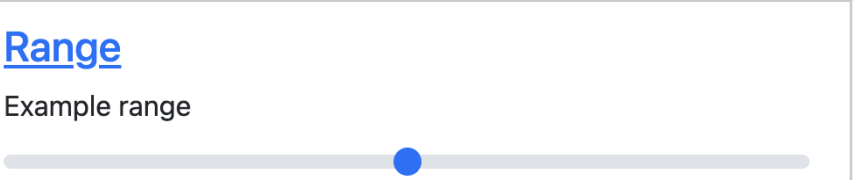
3.2.4.4 Styling Range and Sliders

Range input fields render as sliders. To practice with ***Bootstrap sliders***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h3>
 <a
href="https://getbootstrap.com/docs/5.1/forms
/range/#steps">
    Range
 </a>
</h3>
<label for="range1"
      class="form-label">
 Example range</label>
<input type="range"
      class="form-range"
      min="0"
      max="5"
      step="0.5"
      id="range1"/>
```

Browser



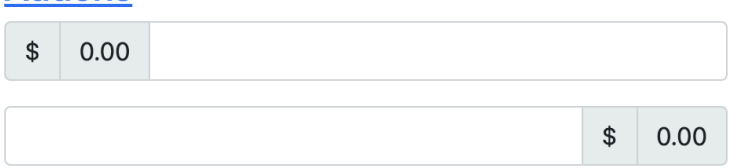
3.2.4.5 Styling Addons

Addons decorate input fields to give some context on the type and formate of the information to type in the input field. To practice with ***Bootstrap addons***, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h3><a
href="https://getbootstrap.com/docs/5.1/forms/input-
group/">Addons</a></h3>
<div class="input-group mb-3">
  <span class="input-group-text">$</span>
  <span class="input-group-text">0.00</span>
  <input type="text" class="form-control">
</div>
<div class="input-group">
  <input type="text" class="form-control">
  <span class="input-group-text">$</span>
  <span class="input-group-text">
```

Browser



```
    0.00</span>
</div>
```

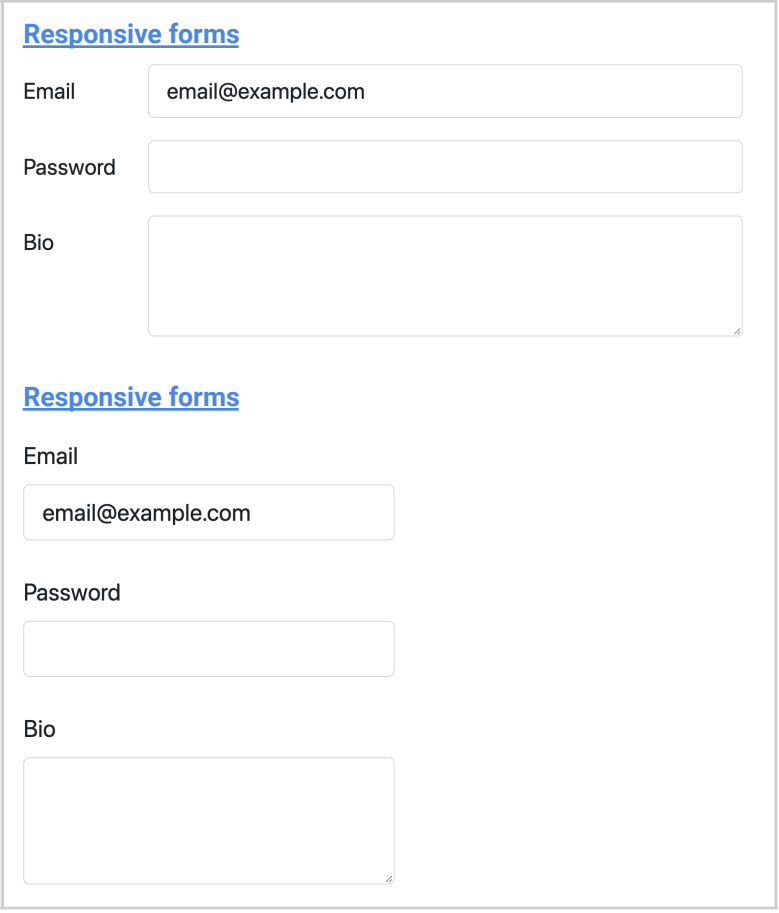
3.2.4.6 Making Forms Responsive

Forms can be configured to display either horizontally or vertically depending on the size of the containing element. To practice with Bootstrap responsive forms, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown. Resize the window to show how the form changes layout as the window resizes.

index.html

```
<h3><a
href="https://getbootstrap.com/docs/5.1/forms/layout/#
horizontal-form">Responsive forms</a></h3>
<div class="mb-3 row">
  <label for="email1"
         class="col-sm-2 col-form-label">
    Email</label>
  <div class="col-sm-10">
    <input type="text"
           class="form-control"
           id="email1"
           value="email@example.com">
  </div>
</div>
<div class="mb-3 row">
  <label for="password1"
         class="col-sm-2 col-form-label">
    Password</label>
  <div class="col-sm-10">
    <input type="password"
           class="form-control"
           id="password1">
  </div>
</div>
<div class="mb-3 row">
  <label for="textarea1"
         class="col-sm-2 col-form-label">
    Bio</label>
  <div class="col-sm-10">
  <textarea class="form-control"
            id="textarea1"
            rows="3"></textarea>
  </div>
</div>
```

Browser

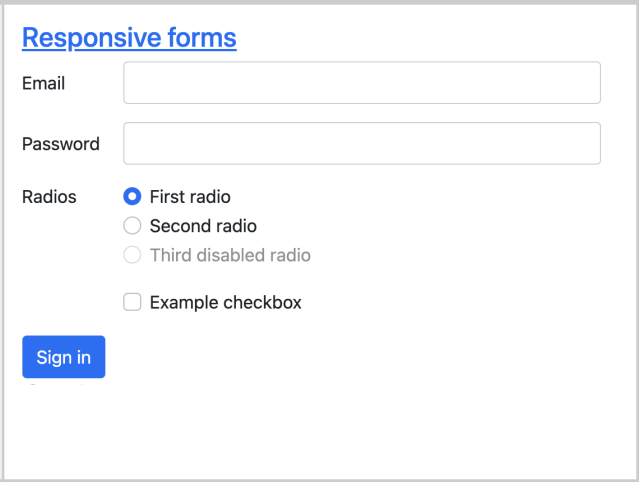


Here's another example. Copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown. Resize the window to show how the form changes layout as the window resizes.

index.html

```
<h3><a
href="https://getbootstrap.com/docs/5.1/forms/layout/#horizontal-
form">Responsive forms</a>
</h3>
<form>
 <div class="row mb-3">
   <label for="r1" class="col-sm-2 col-form-label">
     Email</label>
   <div class="col-sm-10">
     <input type="email" class="form-control"
            id="r1"/>
   </div>
 </div>
 <div class="row mb-3">
   <label for="r2" class="col-sm-2 col-form-label">
       Password</label>
   <div class="col-sm-10">
     <input type="password" class="form-control"
```

Browser



```
          id="r2"/>
  </div>
 </div>
 <fieldset class="row mb-3">
  <legend class="col-form-label col-sm-2 pt-0">
    Radios</legend>
  <div class="col-sm-10">
    <div class="form-check">
      <input class="form-check-input" type="radio"
             name="gridRadios" id="r3"
             value="option1" checked/>
      <label class="form-check-label" for="r3">
        First radio</label>
    </div>
    <div class="form-check">
      <input class="form-check-input" type="radio"
             name="gridRadios" id="r4"
             value="option2"/>
        <label class="form-check-label" for="r4">
          Second radio</label>
        </div>
        <div class="form-check disabled">
          <input class="form-check-input"
                 type="radio"
                 name="gridRadios" id="r5"
                 value="option3" disabled/>
          <label class="form-check-label" for="r5">
            Third disabled radio</label>
        </div>
    </div>
 </fieldset>
 <div class="row mb-3">
  <div class="col-sm-10 offset-sm-2">
    <div class="form-check">
      <input class="form-check-input"
             type="checkbox" id="r6"/>
        <label class="form-check-label" for="r6">
          Example checkbox</label>
    </div>
  </div>
 </div>
 <button type="submit" class="btn btn-primary">Sign in</button>
</form>
```

Responsive forms

Email

Password

Radios

- First radio
- Second radio
- Third disabled radio

Example checkbox

Sign in

3.2.4.7 Overriding Bootstrap Styles

Bootstrap makes it easy to apply professional styles, but sometimes we will need override some of the styles declared by Bootstrap. Complete the exercise described in section ***6.3 Overriding Bootstrap Styles*** in the ***Full Stack Developer*** book.

3.2.5 Navigating with tabs, pills and cards

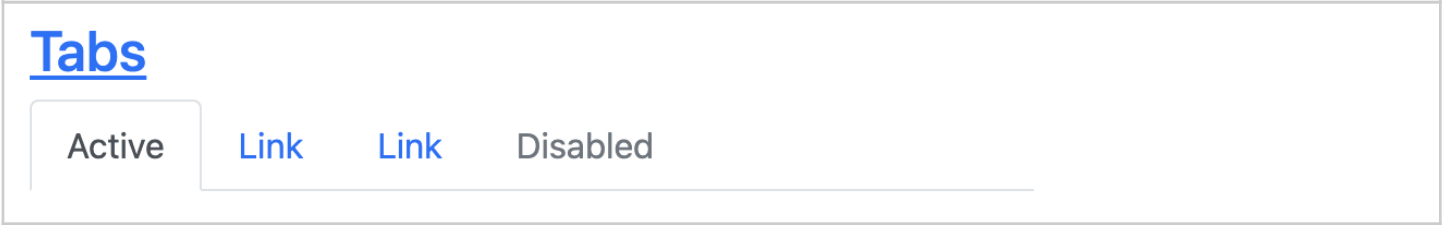
3.2.5.1 Navigating with Tabs

Bootstrap provides several common navigation widgets such as tabs, menus, and pills. Let's take a look at tabs first. To practice with Bootstrap tabs, copy the HTML code below to the end of ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h2><a href="https://getbootstrap.com/docs/5.1/components/navs-tabs/">Tabs</a></h2>
<ul class="nav nav-tabs">
   <li class="nav-item">
       <a class="nav-link active" href="#">Active</a>
   </li>
   <li class="nav-item">
       <a class="nav-link" href="#">Link</a>
   </li>
   <li class="nav-item">
```

```
      <a class="nav-link" href="#">Link</a>
   </li>
   <li class="nav-item">
      <a class="nav-link disabled" href="#" tabindex="-1">Disabled</a>
   </li>
</ul>
```



3.2.5.2 Navigating with Pills

Pills are another navigation widget listing several links. Complete the exercise described in section ***6.2 Implementing Navigation Pills*** in the ***Full Stack Developer*** book.

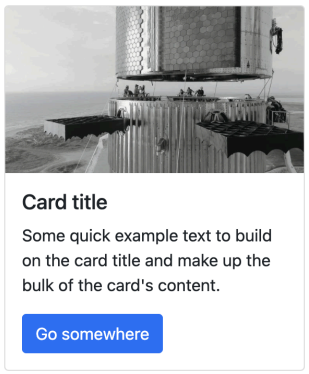
3.2.5.3 Navigating with cards

Cards combine images, titles, paragraphs and buttons into a reusable component that can quickly summarize a topic. To practice with ***Bootstrap cards***, copy the HTML code into ***index.html***, and save. Refresh the browser and confirm it looks similar to image shown. Use an image of your own, save it in ***public/images***, and reference it from the ***img*** tag.

index.html

```
<h2>
   <a href="https://getbootstrap.com/docs/5.1/components/card/">
      Cards
   </a>
</h2>
<div class="card" style="width: 18rem;">
   <img src="../../images/starship.jpg"
        class="card-img-top" alt="...">
   <div class="card-body">
      <h5 class="card-title">Card title</h5>
      <p class="card-text">
         Some quick example text to build on the card
         title and make up the bulk of the card's content.</p>
      <a href="#" class="btn btn-primary">Go somewhere</a>
   </div>
</div>
```

Browser



3.2.6 Bootstrap positions, margins and paddings

Instead of using CSS to style positions, margins and paddings, use Bootstrap utility classes as shown below. ***m*** stands for ***margin***, ***p*** stands for ***padding***, ***t*** for ***top***, ***b*** for ***bottom***, ***s*** for ***start***, and ***e*** for ***end***. Integers are sizes.

Positions	Margins	Paddings
position-absolute, position-relative, position-fixed, position-sticky top, bottom, start, end, top-0, top-50, top-100, bottom-0, bottom-50, bottom-100, start-0, start-50, start-100, end-0, end-50, end-100,	mt-0 mb-0 ms-0 me-0 mt-1 mb-1 ms-1 me-1 mt-2 mb-2 ms-2 me-2 ... ... mt-5 mb-5 ms-5 me-5 mt-auto ms-auto mb-auto me-auto	pt-0 pb-0 ps-0 me-0 pt-1 pb-1 ps-1 me-1 pt-2 pb-2 ps-2 me-2 ... . pt-5 pb-5 ps-5 me-5 pt-auto ps-auto pb-auto pe-auto

3.3 Decorating Documents with Fontawesome Icons

Fontawesome can be installed from the root of your project with ***npm*** as follows.

```
$ npm install font-awesome                # install font-awesome from the root of your project
```

The ***fontawesome*** CSS library will be downloaded from ***npm*** and installed in you ***node_modules*** directory located at the root of your project. Once installed, copy the content of the ***node_modules/font-awesome*** folder into a new ***public/libs/font-awesome*** folder. You can then import the ***fontawesome*** library from your ***HTML*** documents using the following ***link*** tag

```
<link href="/libs/font-awesome/css/font-awesome.min.css" rel="stylesheet"/>
```

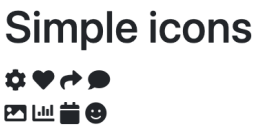
3.3.1 Using simple icons

To practice with Fontawesome icons, copy the HTML and CSS code below to the end of ***index.html*** and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h2>Simple icons</h2>
<i class="fa fa-cog"></i> <i class="fa fa-heart"></i>
<i class="fa fa-share"></i> <i class="fa fa-comment"></i><br/>
<i class="fa fa-image"></i> <i class="fa fa-chart-bar"></i>
<i class="fa fa-calendar"></i> <i class="fa fa-smile"></i>
```

Browser



3.3.2 Resizing icons

To practice with Fontawesome icons, copy the HTML and CSS code below to the end of ***index.html*** and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h2>Resizing icons</h2>
<i class="fa fa-cog fa-2x"></i><i class="fa fa-heart fa-2x"></i><i class="fa fa-share fa-3x"></i>
<i class="fa fa-comment fa-3x"></i><br/><i class="fa fa-image fa-3x"></i><i class="fa fa-chart-bar
fa-4x"></i><i class="fa fa-calendar" style="font-size: 5em"></i><i class="fa fa-smile"
style="font-size: 3.5em"></i>
```

Browser

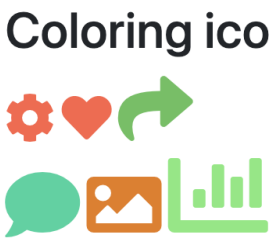
3.3.3 Coloring Icons

To practice with Fontawesome icons, copy the HTML and CSS code below to the end of ***index.html*** and save. Refresh the browser and confirm it looks similar to image shown.

index.html

```
<h2>Coloring icons</h2>
<i class="fa fa-cog fa-2x" style="color: tomato"></i>
<i class="fa fa-heart fa-2x" style="color: tomato"></i>
<i class="fa fa-share fa-3x" style="color: #5ec15b"></i><br/>
<i class="fa fa-comment fa-3x" style="color: #19d49e"></i>
<i class="fa fa-image fa-3x" style="color: rgb(234,123,12)"></i>
<i class="fa fa-chart-bar fa-4x" style="color: rgb(123,234,123)"></i>
```

Browser



3.3.4 Brand Icons

To practice with Fontawesome icons, copy the HTML and CSS code below to the end of ***index.html*** and save. Refresh the browser and confirm it looks similar to image shown.

index.html	*Browser*
`<h2>Brand icons</h2>` `<i class="fab fa-twitter fa-2x"></i> ` `<i class="fab fa-facebook fa-2x"></i> ` `<i class="fab fa-apple fa-2x"></i> `	Brand icons 

3.3.5 Stacking icons

To practice with stacking Fontawesome icons, copy the HTML and CSS code below to the end of ***index.html*** and save. Refresh the browser and confirm it looks similar to image shown.

index.html	*Browser*
`<h2>Stacking icons</h2>` `<span class="fa-stack fa-2x">` `<i class="fas fa-square fa-stack-2x"></i>` `<i class="fab fa-twitter fa-stack-1x fa-inverse"></i>` `</span> ` `<span class="fa-stack fa-2x">` `<i class="fas fa-circle fa-stack-2x"></i>` `<i class="fas fa-flag fa-stack-1x fa-inverse"></i>` `</span> ` `<span class="fa-stack fa-2x">` `<i class="fas fa-camera fa-stack-1x"></i>` `<i class="fas fa-ban fa-stack-2x" style="color:Tomato"></i>` `</span> `	Stacking icons 

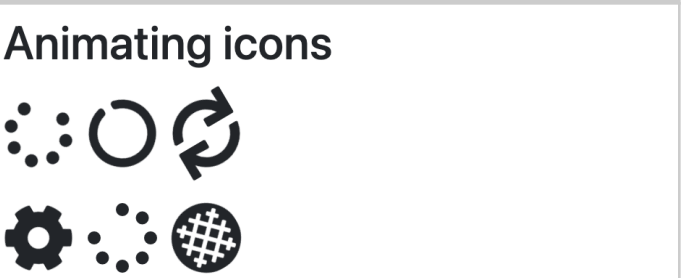
3.3.6 Rotating icons (2pts)

To practice with rotating Fontawesome icons, copy the HTML and CSS code below to the end of ***index.html*** and save. Refresh the browser and confirm it looks similar to image shown.

index.html	*Browser*
`<h2>Rotating icons</h2>` `<div class="fa-4x">` `<i class="fas fa-snowboarding"></i>` `<i class="fas fa-snowboarding fa-rotate-90"></i>` `<i class="fas fa-snowboarding fa-rotate-180"></i> ` `<i class="fas fa-snowboarding fa-rotate-270"></i>` `<i class="fas fa-snowboarding fa-flip-horizontal"></i>` `<i class="fas fa-snowboarding fa-flip-vertical"></i> ` `<i class="fas fa-snowboarding fa-flip-both"></i> ` `</div>`	Rotating icons 

3.3.7 Animating icons

To practice animating Fontawesome icons, copy the HTML and CSS code below to the end of ***index.html*** and save. Refresh the browser and confirm it looks similar to image shown.

index.html	*Browser*
`<h2>Animating icons</h2>` `<div class="fa-3x">` `  <i class="fas fa-spinner fa-spin"></i>` `  <i class="fas fa-circle-notch fa-spin"></i>` `  <i class="fas fa-sync fa-spin"></i> ` `  <i class="fas fa-cog fa-spin"></i>` `  <i class="fas fa-spinner fa-pulse"></i>` `  <i class="fas fa-stroopwafel fa-spin"></i>` `</div>`	

4 Styling Kanbas with CSS

In this section we're going to revisit the ***Kanbas*** screens we worked on in the previous assignment where we left the documents in their natural, unstyled, default look and feel. We're going to use ***CSS*** and ***Bootstrap*** to layout and color the screens so they look more like the screenshots provided. Students should make an effort to style their HTML code to make the screens look as close as possible to their intended look and feel, but it is not required that the resulting screens are pixel perfect matches. Instead we will provide guidelines and requirements you should adhere to.

4.1 Styling the Kanbas Navigation Sidebar

The ***Kanbas*** Web application has several screens implementing different features. Figures 4.1a and 4.1b bellow illustrate the ***Kanbas Navigation*** sidebar with the ***Dashboard*** and ***Courses*** links selected, respectively. When the ***Dashboard*** is selected, a grid of courses is displayed that allow navigating to different courses. When the ***Courses*** link is selected, the submenu ***Course Navigation*** side bar provides further navigation to course related features. In this section we'll style the ***Kanbas Navigation*** as shown below and configure it to navigate to the ***Kanbas Dashboard*** and ***Courses Home*** screens.

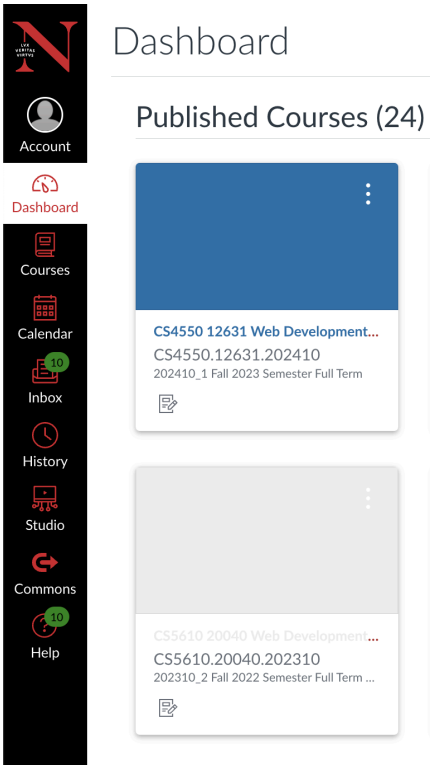


Figure 4.1a - Dashboard

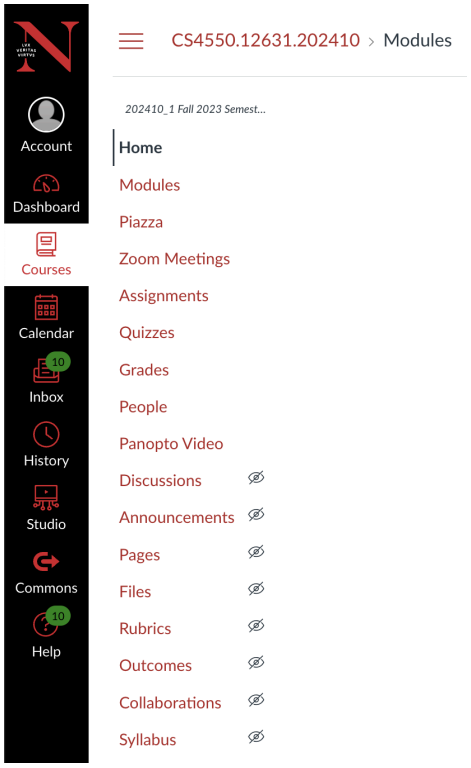


Figure 4.1b - Courses

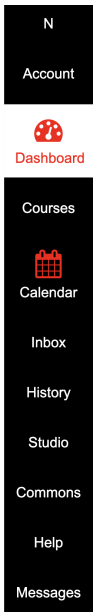
In a previous assignment we implemented the ***Kanbas Navigation*** sidebar in ***/public/Kanbas/Navigation/index.html*** as rendered here on the right. Let's style this with CSS so that it looks more like the sidebar shown in Figures 4.1a and 4.1b. We'll add ***classes*** to the ***index.html*** HTML file and implement the styling in a new ***index.css*** files. The following code illustrates how the ***Kanbas Navigation*** sidebar can be styled. The example code below demonstrates how to use the ***fontawesome*** library to render the ***Dashboard*** and ***Calendar*** as icons. Explore other icons for the rest of the links. The icons don't have to match the ones used in Figures 4.1a and 4.1b, but should reflect the meaning and intention of the link's text and target screen. Also the example code below demonstrates splitting styling into two CSS files: ***styles.css*** and ***index.css***. The ***styles.css*** is implemented at the root of the ***Kanbas*** folder and is meant to implement styles that are common across multiple screens and is intended to be linked from those screens that share a common set of styles. These might include the document's overall paddings and margins, background color, and various common foreground colors and fonts. On the other hand, the ***index.css*** file is in the same folder as the ***index.html*** file and is intended to implement styles that are specific to the ***index.html*** file only.

- Account
- Dashboard
- Courses
- Calendar
- Inbox
- History
- Studio
- Commons
- Help

These will include styling the black background color of the ***Kanbas Navigation*** sidebar, the white background of the highlighted link, the red foreground color of the links, etc. We added the class ***wd-kanbas-navigation*** to the ***ul*** element to style the overall background color, and width of the sidebar. The ***wd-active*** class sets the white background color of the selected link. This class will have to be added to different links when the sidebar renders in the respective screen. The ***href*** attribute of hyperlinks that don't link to any existing HTML documents were set to the ***hash*** (#) value meaning that we don't yet know where they should link to.

/public/Kanbas/Navigation/index.html

```
<html>
  <head>
    <link rel="stylesheet" href="/Kanbas/styles.css" />
    <link rel="stylesheet" href="/Kanbas/Navigation/index.css" />
    <link rel="stylesheet" href="/libs/font-awesome/css/font-awesome.min.css" />
  </head>
  <body>
    <ul class="wd-kanbas-navigation">
      <li><a href="http://northeastern.edu">N</a></li>
      <li><a href="/Kanbas/Account/Profile/screen.html">Account</a></li>
      <li class="wd-active">
        <a href="/Kanbas/Dashboard/screen.html">
          <i class="fa fa-tachometer"></i> Dashboard</a>
      </li>
      <li><a href="/Kanbas/Courses/Home/screen.html">Courses</a> </li>
      <li><a href="#"><i class="fa fa-calendar"></i> Calendar</a></li>
      <li><a href="#">Inbox</a></li>
      <li><a href="#">History</a></li>
      <li><a href="#">Studio</a></li>
      <li><a href="#">Commons</a></li>
      <li><a href="#">Help</a></li>
    </ul>
  </body>
</html>
```



The ***index.css*** file shown below illustrates how CSS can be used to style the ***Kanbas Navigation*** sidebar implemented above. The ***wd-kanbas-navigation*** class styles the black background color, the overall width, and margin. The ***.wd-kanbas-navigation li*** CSS rule styles each of the line items in the unordered list including padding, margin, and font. The ***.wd-kanbas-navigation li.wd-active*** CSS rule styles selected link. The ***styles.css*** file shown below implements common styles applicable to all the screens across the Website including things such as overall margin, padding, and fonts. Navigate to the sidebar and confirm it renders as shown above.

/public/Kanbas/Navigation/index.css

```
.wd-kanbas-navigation {
  background-color: black;
  list-style: none;
  padding: 0px;  margin: 0px;
  width: 80px;
}
.wd-kanbas-navigation li {
  padding: 10px;     margin-bottom: 10px;
  font-size: 0.8em;  text-align: center;
}
.wd-kanbas-navigation li.wd-active {
  background-color: white;
}
.wd-kanbas-navigation li.wd-active a {
  color: red;
}
.wd-kanbas-navigation a {
  color: white;
  text-decoration: none;
}
.wd-kanbas-navigation a i {
  color: red;
  font-size: 2em;
  margin-bottom: 5px;
}
```

/public/Kanbas/styles.css

```
body {
  margin: 0;
  padding: 0;
}
body * {
  font-family: Arial, Helvetica, sans-serif;
}
```

Your implementation does not need to be pixel perfect, but here are some rough requirements you should consider when implementing the ***Kanbas Navigation***. Use your own access to ***Canvas*** to help guide your design

- The ***Kanbas Dashboard*** link must be the default screen when navigating to ***Kanbas***
- Icons must be ***red***, except the ***Account*** icon which must be ***white***
- Icons don't have to match exactly, but must be similar. Use an equivalent fontawesome icon
- The font size and style must be similar
- The whole sidebar must have a black background
- Selected (active) links must have a white background with red text
- Non selected links must have a black background with white text
- The width of the sidebar must be about 80 pixels wide, +/- 5 pixels.
- The spacing between the text, icons, and links must be 5 pixels, +/- 2 pixels.
- The space between the bottom of the text and the top of the icon below must be about 10 pixels, +/- 5 pixels.
- The icons and text must be centered in the sidebar

Once you are satisfied with the styling of the ***Kanbas Navigation*** sidebar, replace all occurrences of the ***Kanbas Navigation*** in all the screens implemented so far, e.g., ***Home/screen.html***, ***Assignments/screen.html***, ***Assignments/Editor/screen.html***, and ***Grades/screen.html***. You'll need to also include the ***link*** tags that reference the ***styles.css*** and ***index.css*** that style the ***Kanbas Navigation***. Visit all the screens and make sure that the ***Kanbas Navigation*** styling is as shown in Figures 4.1a and b. Use this new styled ***Kanbas Navigation*** implementation going forward when implementing new screens.

4.2 Implementing and Styling the Kanbas Dashboard Screen

The ***Kanbas Dashboard*** is the default screen when you first login to ***Kanbas*** and it lists courses as illustrated below in figures 4.2a through 4.2e. The ***Kanbas Dashboard*** gives access to courses a user is enrolled in as a student and courses a faculty is teaching. Clicking on a course navigates to that specific course. Since we don't yet have JavaScript, for this assignment we'll just implement a single course and all courses in the ***Kanbas Dashboard*** will navigate to the same course. Later assignments will implement multiple courses. The courses are rendered in a responsive grid pattern that adapts (responds) to the width of the screen. When the screen is wide, each row of the grid displays several courses, e.g., 4 courses. As the screen becomes narrower, the courses that don't fit the narrower width wrap and to the next row as shown in figure 4.2b and c. When the screen is too narrow, the ***Kanbas Navigation*** is hidden and courses are displayed in a single column as shown in figure 4.2e.

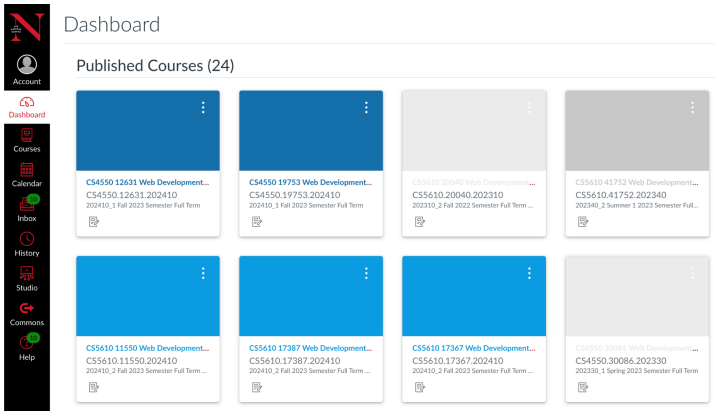


Figure 4.2a

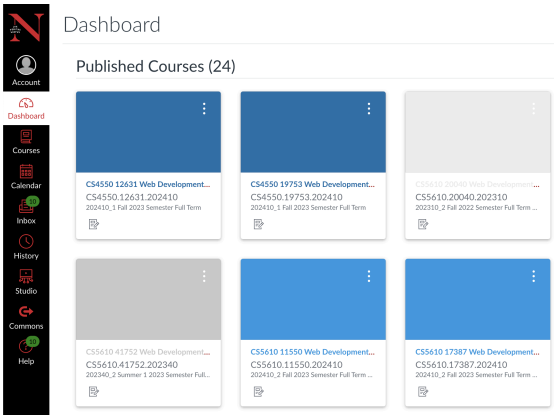


Figure 4.2b

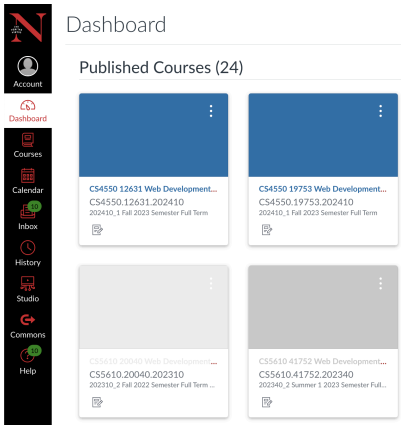


Fig. 4.2c

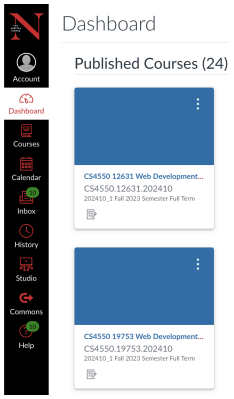


Fig. 4.2d

Fig. 4.2e

In a new ***index.html*** file under ***/public/Kanbas/Dashboard***, create the ***Kanbas Dashboard*** displaying at least 7 courses. Use the ***index.html*** file shown below as an example. Feel free to come up with your own courses. Download images for each course and save them to ***/public/images***. The code below uses an image of the React.js logo.

/public/Kanbas/Dashboard/index.html

```
<html>
  <head>
    <link href="/libs/bootstrap/bootstrap.min.css" rel="stylesheet" />
  </head>
  <body>
    <div class="p-4">
      <h1>Dashboard</h1> <hr />                                    <!-- Dashboard Title -->
      <h2>Published Courses (12)</h2> <hr />                       <!-- Published Courses -->
      <div class="row">
        <div class="row row-cols-1 row-cols-md-5 g-4">             <!-- Single Row -->
          <div class="col" style="width: 300px">
            <div class="card">                                     <!-- Course 1 -->
              <img src="/images/reactjs.jpg" class="card-img-top"
                   style="max-height: 150px"/>
              <div class="card-body">
                <a class="card-title" href="/Kanbas/Courses/Home/screen.html"
                   style="text-decoration: none; color: navy; font-weight: bold">
                   CS1234 React JS</a>
                <p class="card-text">Full Stack software developer</p>
                <a html="#" class="btn btn-primary"> Go </a>
              </div>
            </div>
          </div>
          <div class="col" style="width: 300px"> ... </div>        <!-- Course 2 -->
          <div class="col" style="width: 300px"> ... </div>        <!-- Course 3 -->
        </div>                                                     <!-- Add at least 7 courses
      </div>                                                           in total -->
    </div>
  </body>
</html>
```

Once you are satisfied with the ***Kanbas Dashboard***, combine it with the ***Kanbas Navigation*** to create the whole ***Kanbas Dashboard Screen*** so that the ***Kanbas Navigation*** renders as a left sidebar, and the ***Kanbas Dashboard*** renders as the main content on the right hand side. Use a single row ***table*** to layout the ***Kanbas Navigation*** in one column, and the ***Kanbas Dashboard*** in the second column, as suggested in the code below.

/public/Kanbas/Dashboard/screen.html

```
<html>
  <head>
    <link rel="stylesheet" href="/libs/bootstrap/bootstrap.min.css" />
    <link rel="stylesheet" href="/libs/font-awesome/css/font-awesome.css" />
    <link rel="stylesheet" href="/Kanbas/styles.css" />
    <link rel="stylesheet" href="/Kanbas/Navigation/index.css" />
  </head>
  <body>
```

//

```
    <table width="100%">
      <tbody>
        <tr>
          <td valign="top" style="width: 80px">
            <!-- copy the Kanbas Navigation in here. Only what's inside the body tag -->
            <!-- here are the first few lines as an example -->
            <ul class="wd-kanbas-navigation">
              <li><a href="http://northeastern.edu">N</a></li>
              <li><a href="/Kanbas/Account/Profile/screen.html">Account</a></li>
              <!-- copy the rest of the links here -->
            </ul>
          </td>
          <td>
            <!-- copy the Dashboard in here. Only what's inside the body tag -->
            <!-- here are the first few lines as an example -->
            <div class="p-4">
              <h1>Dashboard</h1>
              <!-- copy the rest in here -->
            </div>
          </td>
        </tr>
      </tbody>
    </table>
  </body>
</html>
```

Your implementation does not need to pixel perfect, but here are some rough requirements you should consider when implementing the ***Kanbas Dashboard***. Use your own access to ***Canvas*** to help guide your design

- Modify the ***Kanbas*** link in the root ***index.html*** file so that it points to ***/Kanbas/Dashboard/screen.html***.
- The ***Dashboard*** link must be rendered as selected in the ***Kanbas Navigation*** sidebar with red text, red icon, and white background
- A ***Dashboard*** title must appear at the top of the screen as shown
- Clicking the ***Dashboard*** link must display the ***Kanbas Dashboard*** screen
- Clicking any course in the ***Kanbas Dashboard*** screen must display the ***Home*** screen implemented in previous assignments
- A ***horizontal rule*** (`<hr/>`) must appear below the title as shown
- A ***Published Courses*** sub title must appear as shown
- At least 7 courses must be rendered under the ***Published Courses*** as shown
- Courses must render in a grid of rows and columns as shown
- Use ***Bootstrap*** classes such as ***d-flex***, ***flex-row*** and ***flex-wrap*** to implement the grid of rows and columns. Alternative classes can be used as long as they have a similar functionality
- ***Bootstrap card*** classes must be used to render each course as a card with an image or solid color at the top and the course's title, term, year and section linked to the course ***Home*** screen (described later) as shown
- The widths of the courses must be between 250 and 270 pixels, and can not change as the width of the window changes
- There must be white spacing between courses above and below of between 30 and 40 pixels
- There must be white spacing between the right most edge of the ***Kanbas Navigation*** sidebar and the left most edge of the left most course
- When the window is at its widest, rows must show at least 4 courses per row
- When the window shrinks and 4 courses don't fit, then at most 3 courses per row must display and the remaining courses must wrap to the next row
- When the window shrinks and 3 courses don't fit, then at most 2 courses per row must display and the remaining courses must wrap to the next row

To make the ***Kanbas Dashboard*** the default screen, modify the root ***index.html*** file so that clicking ***Kanbas*** navigates to the ***Kanbas Dashboard*** screen instead of the ***Courses Home*** screen as shown below.

public/index.html

```
<!DOCTYPE html>
<html lang="en">
  <head> ...
  </head>
  <body>
    <h1>Welcome to Web Development!</h1>
    <ul>
      <li><a href="/labs/a1/index.html">Assignment 1</a></li>
      <li><a href="/labs/a2/index.html">Assignment 2</a></li>
      <li><a href="/Kanbas/Courses/Home/screen.html">Kanbas</a></li>
      <li><a href="/Kanbas/Dashboard/screen.html">Kanbas</a></li>
    </ul>
  </body>
</html>
```

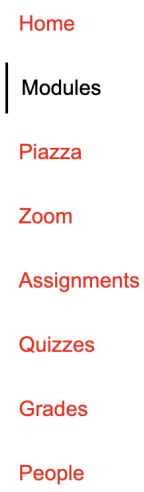
```
//
```

4.3 Styling the Course Navigation Sidebar

Clicking a course in the ***Kanbas Dashboard*** navigates to that course's ***Home*** screen. For now we'll implement the ***Kanbas Dashboard*** to navigate to the same course. When we introduce JavaScript later in the course we'll revisit this so that we navigate to the corresponding course. In an earlier assignment we implemented the ***Course Navigation*** sidebar in ***/public/Kanbas/Courses/Navigation/index.html***. Let's style the ***Course Navigation*** sidebar to look more like in figure 4.1b. The code below illustrates how we can apply styles to the ***Courses Navigation*** sidebar so that it renders closer to the target look and feel shown below on the right. The ***index.css*** referenced in the HTML document below is implemented in the same folder as the ***index.html*** HTML document and is documented later in this document. The ***styles.css*** file is implemented at the root of the ***Kanbas*** folder and is the same we used in the ***Kanbas Navigation*** sidebar and is documented earlier in this document. Feel free to use and modify the code below as necessary.

/public/Kanbas/Courses/Navigation/index.html

```
<html>
  <head>
    <link rel="stylesheet" href="/Kanbas/styles.css" />
    <link rel="stylesheet" href="/Kanbas/Courses/Navigation/index.css" />
  </head>
  <body>
    <ul class="wd-navigation">
      <li>
        <a href="/Kanbas/Courses/Home/screen.html">Home</a></li>
      <li class="wd-active">
        <a href="/Kanbas/Courses/Modules/screen.html">Modules</a>
      </li>
      <li><a href="http://piazza.com">Piazza</a></li>
      <li><a href="#">Zoom</a></li>
      <li>
        <a href="/Kanbas/Courses/Assignments/screen.html">Assignments</a>
      </li>
      <li><a href="#">Quizzes</a></li>
      <li><a href="/Kanbas/Courses/Grades/screen.html">Grades</a></li>
      <li><a href="#">People</a></li>
    </ul>
  </body>
</html>
```



The ***index.css*** file shown below is implemented in the same directory of the ***Courses Navigation*** sidebar since it is meant to style only that HTML document. It styles the white background color, the width, margin, padding, and black border of the highlighted link.

/public/Kanbas/Courses/Navigation/index.css

```
.wd-navigation {
  background-color: white;
  list-style: none;
  padding: 0px;
  width: 125px;
  margin: 0px;
  margin-left: 20px;
```

```
//
```

```
}
.wd-navigation li {
  padding-left: 10px;
  margin-bottom: 10px;
  margin-top: 10px;
  padding-top: 10px;
  padding-bottom: 10px;
}
.wd-navigation li.wd-active {
  border-left: 2px solid black;
}
.wd-navigation li.wd-active a {
  color: black;
}
.wd-navigation a {
  color: red;
  text-decoration: none;
}
```

Once you are satisfied with the styling of the ***Courses Navigation*** sidebar, replace all occurrences of the ***Courses Navigation*** in all the screens implemented so far, e.g., ***Home/screen.html***, ***Assignments/screen.html***, ***Assignments/Editor/screen.html***, and ***Grades/screen.html***. You'll need to also include the ***link*** tags that reference the ***styles.css*** and ***index.css*** that style the ***Courses Navigation***. Visit all the screens and make sure that the ***Courses Navigation*** styling is as shown in Figure 4.1b. Use this new styled ***Courses Navigation*** implementation going forward when implementing new screens. In each screen, style the ***Courses Navigation*** sidebar so that the corresponding link is highlighted, e.g., in ***Home/screen.html***, the sidebar's ***Home*** link should be highlighted, in the ***Grades/screen.html***, the sidebar's ***Grades*** link should be highlighted, and so on and so forth.

4.4 Styling the Course Home Screen

Clicking on a course in the ***Kanbas Dashboard*** displays the course's ***Home*** screen as shown below. The ***Course Navigation*** sidebar provides various links to navigate to different screens related to the course such as ***Home***, ***Modules***, ***Assignments***, and ***Grades***. The ***Course Home*** screen is displayed by default and its link is rendered as selected when you first navigate to a course from the ***Dashboard***. In a wide screen the ***Course Home*** screen renders 4 columns as shown in Figure 4.4a. The first column is the ***Kanbas Navigation*** sidebar, the second column is the ***Course Navigation*** sidebar, and third column contains the ***Course Home*** showing a list of modules starting at ***Week 0***. The last column are various buttons and links. If the window narrows below some threshold, the last column is hidden and the third column takes the remaining width as shown in Figure 4.4b. If the window narrows even further below some other threshold, the first and second column are hidden, and the third column takes the remaining width as shown in Figure 4.4c. If hidden, the ***Kanbas Navigation*** is still accessible by clicking the ***sandwich*** icon at the top left corner and displays the ***Kanbas Navigation*** as shown in Figure 4.4d. Similarly, if hidden, the ***Course Navigation*** is still accessible by clicking the ***chevron*** icon at the top right corner and displays the ***Course Navigation*** as shown in Figure 4.4e. Both, the ***Kanbas Navigation*** and ***Course Navigation*** can be dismissed by clicking the ***X*** icon at their top right corner. When hidden, the ***Kanbas*** and ***Course Navigation*** should still work allowing navigation as when they are not hidden. Implement the ***Course Home*** screen as described below.

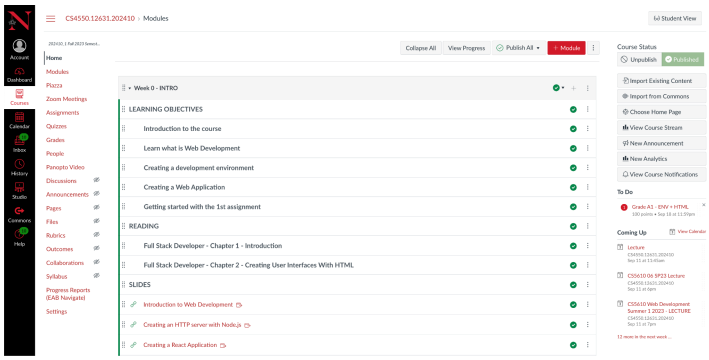


Figure. 4.4a

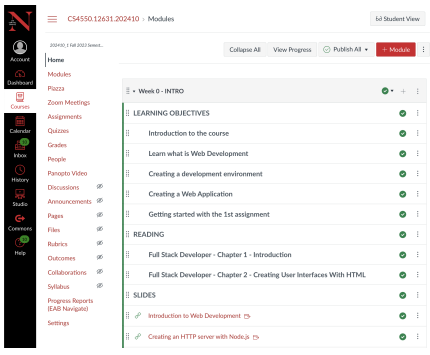


Figure. 4.4b

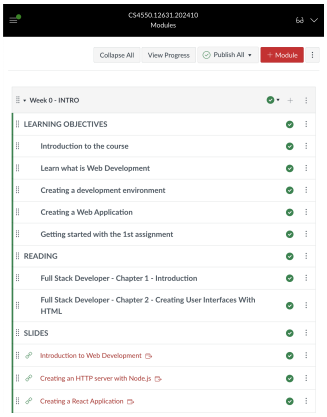


Figure. 4.4c

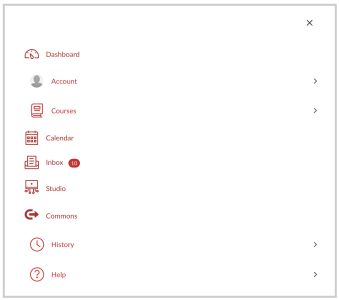


Figure. 4.4d

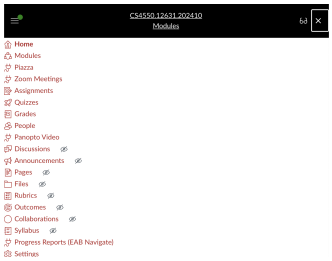


Figure. 4.4e

The following code illustrates how you might implement and style modules Week 1 and Week 2. Feel free to use it as a guide. You'll still need to add the buttons at the top: ***Collapse All***, ***View Progress***, etc.

/public/Kanbas/Courses/Modules/index.html

```
<html>
  <head>
    <link rel="stylesheet" href="/libs/bootstrap/bootstrap.min.css" />
    <link rel="stylesheet" href="/libs/font-awesome/css/font-awesome.css" />
    <link rel="stylesheet" href="/Kanbas/Courses/Modules/index.css" />
  </head>
  <body>
    <!-- Implement buttons Collapse All, View Progress, etc. here -->
    <ul class="list-group wd-modules">
      <li class="list-group-item">
        <div>
          <i class="fa fa-ellipsis-v"></i> Week 1
          <span class="float-end">
            <i class="fa fa-check-circle text-success"></i>
            <i class="fa fa-plus ms-2"></i>
            <i class="fa fa-ellipsis-v ms-2"></i>
          </span>
        </div>
        <ul class="list-group">
          <li class="list-group-item">
            <i class="fa fa-ellipsis-v"></i> Learning Objectives
            <span class="float-end">
              <i class="fa fa-check-circle text-success"></i>
              <i class="fa fa-ellipsis-v ms-2"></i>
            </span>
          </li>
        </ul>
      </li>
      <li class="list-group-item">
        <div>
          <i class="fa fa-ellipsis-v"></i> Week 2
          <span class="float-end">
            <i class="fa fa-check-circle text-success"></i>
            <i class="fa fa-plus ms-2"></i>
            <i class="fa fa-ellipsis-v ms-2"></i>
          </span>
        </div>
      </li>
    </ul>
  </body>
</html>
```

index.css

```
.wd-modules,
.wd-modules * {
  border-radius: 0%;
  border: none;
  margin: 0px;
  padding: 0px;
}

.wd-modules {
  margin: 5px;
}

.wd-modules > li {
  background-color: lightgray;
  border: 1px solid gray !important;
  border-left: none !important;
}

.wd-modules > li > div {
  padding: 6px;
  border-left: 1px solid gray !important;
}

.wd-modules > li > ul > li {
  padding: 5px;
  border-top: 1px solid gray !important;
  border-left: 2px solid green;
}

.wd-modules > li {
  margin-bottom: 20px;
}
```

Your implementation does not need to pixel perfect. Use your own access to ***Canvas*** to help guide your design. Once you are satisfied with the ***Course Home***, combine it with ***Kanbas Navigation***, and ***Course Navigation*** to implement the ***Course Home*** screen.

4.4.1 Implementing responsive design

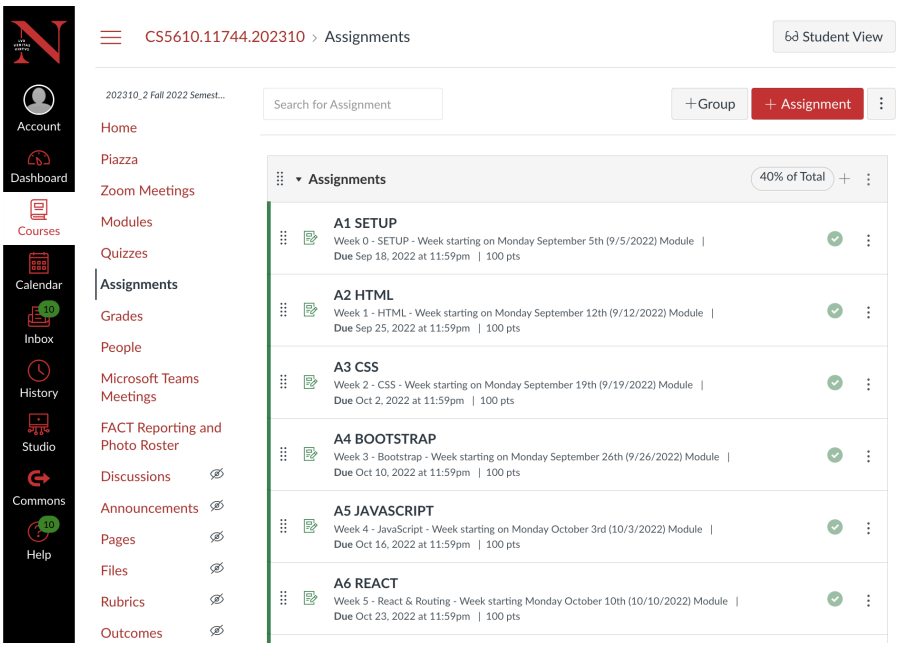
Previous assignments used ***table*** tags to layout screens horizontally. Reimplement the ***Course Home*** screen to use DIVs and CSS flex instead of tables as shown below. Use bootstrap's display classes to implement the responsive design as described in Figures 4.4a through 4.4e. Below is an example of how you could implement the ***Course Home*** screen.

/public/Kanbas/Courses/Home/screen.html

```
<html>
  <head>
    <link rel="stylesheet" href="/libs/bootstrap/bootstrap.min.css" />
    <link rel="stylesheet" href="/libs/font-awesome/css/font-awesome.min.css" />
    <link rel="stylesheet" href="/Kanbas/styles.css" />
    <link rel="stylesheet" href="/Kanbas/Navigation/index.css" />
    <link rel="stylesheet" href="/Kanbas/Courses/Navigation/index.css" />
    <link rel="stylesheet" href="/Kanbas/Courses/Home/index.css" />
  </head>
  <body>
    <!-- Implement links to Kanbas and Course Navigation -->
    <div class="d-flex">
      <div class="d-none d-md-block">
        <ul class="wd-kanbas-navigation">
          <!-- Copy Kanbas Navigation Here -->
        </ul>
      </div>
      <div class="d-none d-md-block">
        <ul class="wd-navigation">
          <!-- Copy Course Navigation Here -->
        </ul>
      </div>
      <div class="flex-fill">
        <ul class="list-group wd-modules">
          <!-- Copy Course Home here -->
        </ul>
      </div>
      <div class="flex-grow-0 me-2 d-none d-lg-block" style="width: 250px">
        <h3>Course Status</h3>
          <!-- Implement Course Status here -->
      </div>
    </div>
  </body>
</html>
```

4.5 Styling the Assignments Screen (grads only)

Clicking ***Assignment*** in the ***Course Navigation*** sidebar displays the ***Assignment*** screen as shown below. Use ***Bootstrap***'s ***float-end*** classes to send the ***Group***, ***Assignment*** and ***Edit*** (⋮) buttons to the right of the screen. Use the ***form-control*** and ***w-25*** to style the ***Search for Assignment*** input field. Use the ***list-group*** and ***list-group-item*** classes to style the list of assignments. Override the ***Bootstrap***'s ***list-group*** and ***list-group-item*** classes to remove the rounded corners. Use ***Fontawesome*** icons for the ***plus***, ***checkmark*** and ***ellipsis*** icons in the buttons. Use the ***rounded*** classes to style the border around ***40% of Total***. Use the margin and padding classes to add white spaces around and between the content as shown below.



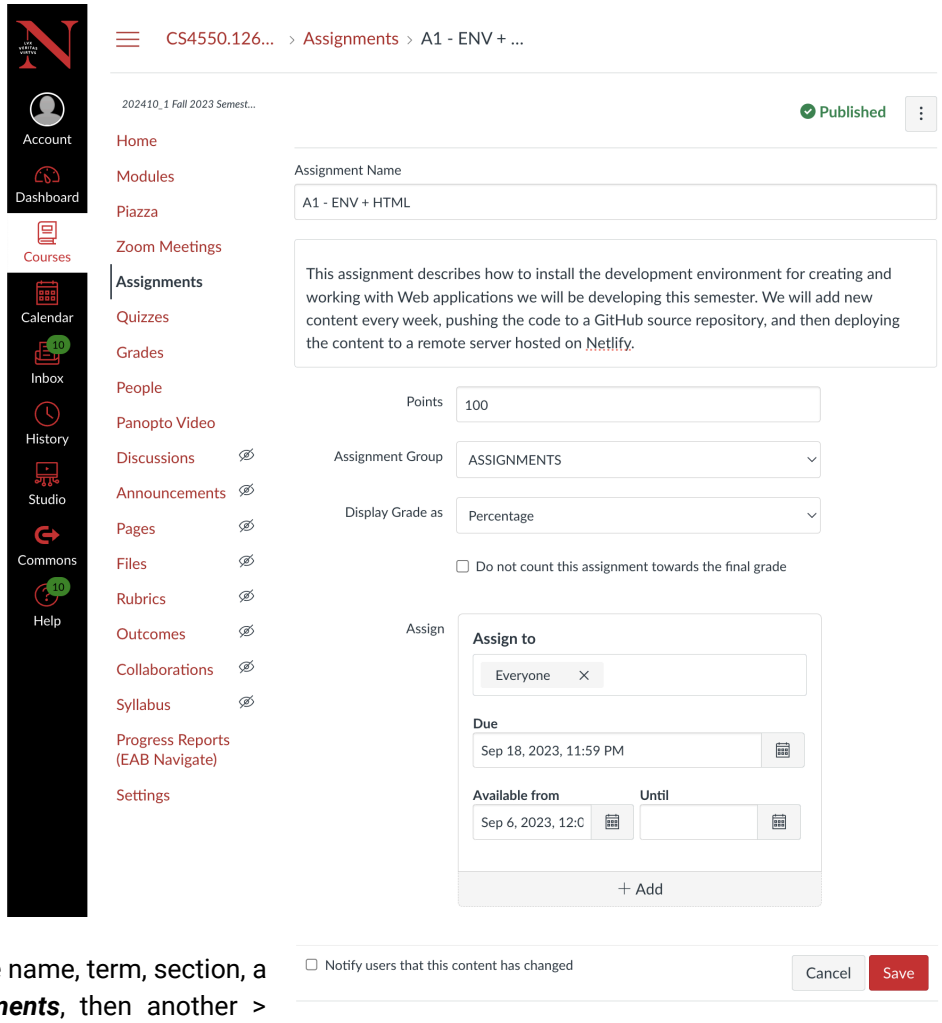
4.5.1 Assignments Screen Requirements

Make an effort to style the ***Assignment*** screen as shown above, but it doesn't need to be pixel perfect. Here are some requirements you need to follow

- The ***Assignments*** link must be rendered as selected in the ***Course Navigation*** sidebar with black text and black border on the left
- Use ***Bootstrap***'s ***breadcrumb*** class to render the breadcrumb at the top of the screen. The breadcrumb must display the course name, term, section, a ">" character, then followed by ***Assignments*** to indicate we are in the ***Assignments*** screen
- Buttons must be floated to the right as shown
- The ***Search for Assignment*** text field must render as shown including its ***placeholder***
- The icons on the screen must be rendered with similar ***Fontawesome*** icons
- White spaces around and between content must be rendered with ***Bootstrap***'s margin and padding classes
- The border to the left of the line items must be rendered green as shown
- The titles of the assignments, ***A1 SETUP***, ***A2 HTML***, etc, must be rendered as shown
- The subtext under the titles such as the due dates, start times, and points, can be abridged and simplified

4.6 Styling Edit Assignment Screen (grads only)

Clicking on the title of any assignment displays the ***Edit Assignment*** screen as shown below. For now this screen displays the same content regardless which assignment you click. In later assignments the content will be different depending which assignment you click. The screen provides a form for faculty to edit the assignment including the ***Assignment Name***, ***Description***, ***Points***, and ***Due Date***. Use ***Bootstrap*** classes such as ***form-control***, ***float-end***, ***btn***, ***btn-success***, ***row***, and ***col***, to style the ***Edit Assignment*** screen as close to the screen shots shown below.



4.6.1 Edit Assignment Screen Requirements

Make an effort to style the screen as close as possible to the wireframes provided, but it is not required to be pixel perfect. Here are some guidelines and requirements to consider.

- The ***Assignments*** link must be rendered as selected in the ***Course Navigation*** sidebar with black text and black border on the left
- The ***Breadcrumb*** must display the course name, term, section, a > character, then followed by ***Assignments***, then another > character followed by the name of the assignment, e.g., ***A1 ENV***
- All ***input***, ***textarea***, and ***select*** form elements must be styled with ***Bootstrap form-control*** class
- The input fields ***Points***, ***Assignment Group***, ***Display Grade as***, ***Submission Type***, and ***Assign***, must appear on the top left of their related fields

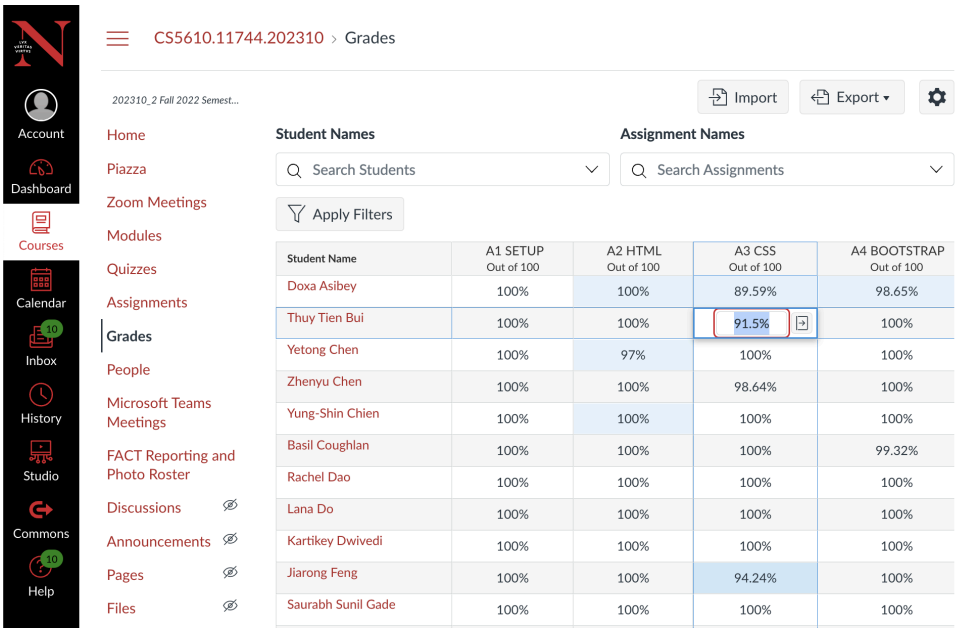
- ***Bootstrap*** grid classes such as ***row*** and ***col***, must be used to implement the rows and columns that separate the form labels and their related fields and sections
- The calendar icons next to the ***Due***, ***Available from***, and ***Until*** fields are not required
- The date format of the ***Due***, ***Available from***, and ***Until*** fields is not required. Dates can render as MM/DD/YYYY
- White spaces around and between content must be implemented with ***Bootstrap*** margin and padding classes

4.7 Grades Screen (grads only)

Clicking ***Grades*** on the ***Kanbas Navigation*** sidebar, displays the ***Grades*** screen as shown below. This screen displays all the grades for all students for all assignments. Use Bootstrap's ***table*** and ***table-striped*** classes to style the tables with alternating row background colors. Wrap the table in a DIV with the ***table-responsive*** class so that you can scroll the table horizontally if it does not fit in the window. Use Bootstrap's ***form-control*** to style the ***Student Names*** and ***Assignment Names*** search elements and the ***row*** and ***col*** classes to render them side by side as shown in the screen shots below. Use Bootstrap's ***btn*** and other button related classes to style the various buttons. Choose icons at your discretion to render the various icons on the screen. Ignore the blue background that appears on some of the cells.

4.7.1 Grades Screen Requirements

Make an effort to render your screen as close as possible as shown in the wireframe, but it does not need to be pixel perfect. Here are some requirements you must achieve.



- The ***Grades*** link must be rendered as selected in the ***Course Navigation*** sidebar with black text and black border on the left
- The ***Breadcrumb*** must display the course name, term, section, a > character, then followed by ***Grades*** to indicate we are in the ***Grades*** screen
- The ***Import*** button must render as shown with a similar "import" ***Fontawesome*** icon
- The ***Export*** dropdown must render as shown with a similar "export" ***Fontawesome*** icon. Don't worry about additional options for now
- The ***Edit*** button must render with a similar ***Fontawesome*** icon
- At least one value in the table must be an input field with some default value as shown
- All white spaces around and between the content must be implemented with ***Bootstrap***'s margin and padding classes

5 Delivery

1. In the same React.js application created in earlier assignments, ***kanbas-react-web-app***, complete all the exercises described in this document
2. In a branch called ***a2***, add, commit and push the source code of the React.js application ***kanbas-react-web-app*** to the same remote source repository in ***GitHub.com*** created in an earlier assignment. Here's an example of how to add, commit and push your code

```
$ git checkout -b a2
$ git add .
```

```
$ git commit -am "a2 CSS and Bootstrap"
$ git push
```

3. Deploy the ***a2*** branch to the same ***Netlify*** created in an earlier assignment. Configure Netlify to deploy all branches to separate URLs. From your Netlify's dashboard go to ***Site settings > Build & deploy > Branches > Branch deploys*** and select ***All***. Now each time you commit to a branch, the application will be available at a URL that contains the name of the branch
4. As a deliverable in ***Canvas***, submit the following URLs
    a. The source repository in GitHub.com. It should look something like **`https://github.com/jannunzi/kanbas-react-web-app.git`**
    b. The React.js application running on Netlify. It should look something like **`https://a2-loving-torvalds-effde8.netlify.app/`**
    Note that the URL contains the name of the branch