

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 (50%)

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>
```

Cascading Style Sheets (CSS) Style attribute and tag

Style attribute

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

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>
    p {
      background-color: blue;
      color: white;
    }
  </style>
</head>
```

!-- new style tag
selector "p" refers to all paragraphs
sets all paragraph background color to blue
sets all paragraph foreground color to white

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	Style tag
<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>	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

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 {
      background-color: blue;
      color: white;
    }
  </style>
  <link href="index.css"
        rel="stylesheet"/>
</head>
```

!--
Comment out, or remove, the style tag.
We moved this content to the index.css file

!-- Instead Load the index.css
file created earlier -->

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

```
<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>
```

Link tag

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.

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	index.html	Browser
<pre>/*p { background-color: blue; color: white;}*/ <p background-color:="" black;="" color:="" id="id-selector-1" p#id-selector-2="" pre="" red;="" white;="" yellow;="" {="" }="" }<=""></p></pre>	<pre><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></pre>	<p>ID selectors</p> <p>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>Here's another paragraph using a different ID and a different look and feel</p>

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	index.html	Browser
<pre>.wd-class-selector { background-color: yellow; color: blue; }</pre>	<pre><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></pre>	<p>Class selectors</p> <p>Instead of using IDs to refer to elements, you can use an element's CLASS attribute</p> <p>This heading has same style as paragraph above</p>

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

!-- 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

this is a descendant of .selector-1 and
.wd-selector-2 and a direct child of
.wd-selector-3
-->

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	index.html
.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; }	<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 this text is green </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	index.html
.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; }	<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 the background of this text is green and the foreground white </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 {  
    border-color: #7070ff; }
```

```
<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

Solid fat red border

Dashed thin blue border

3.1.4.2 Styling 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
```

Padding

Padded top left

Padded bottom right

Padded all around

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

<i>index.css</i>	<i>index.html</i>	<i>Browser</i>
<pre>.wd-margin-bottom { margin-bottom: 50px; } .wd-margin-right-left { margin-left: 50px; margin-right: 50px; } .wd-margin-all-around { margin: 30px; }</pre>	<pre><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> <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 </div></pre>	<p>Margins</p>

3.1.4.3 Styling 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.

<i>index.css</i>	<i>index.html</i>	<i>Browser</i>
<pre>.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; }</pre>	<pre><h3>Rounded corners</h3> <p class="wd-rounded-corners-top wd-border-thin wd-border-blue wd-border-solid wd-padding-fat"> Rounded corners on the top </p> <p class="wd-rounded-corners-bottom wd-border-thin wd-border-blue wd-border-solid wd-padding-fat"> Rounded corners at the bottom </p> <p class="wd-rounded-corners-all-around wd-border-thin wd-border-blue wd-border-solid wd-padding-fat"> Rounded corners all around </p> <p class="wd-rounded-corners-inline wd-border-thin wd-border-blue wd-border-solid wd-padding-fat"> Different rounded corners </p></pre>	<p>Rounded corners</p>

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.

<i>index.css</i>	<i>index.html</i>	<i>Browser</i>
<pre>.wd-dimension-portrait { width: 75px; height: 100px; } .wd-dimension-landscape { width: 100px; height: 75px; } .wd-dimension-square { width: 75px; height: 75px; }</pre>	<pre><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></pre>	<p>Dimension</p> <p>Portrait</p> <p>Landscape</p> <p>Square</p>

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.

<i>index.css</i>	<i>index.html</i>	<i>Browser</i>
------------------	-------------------	----------------

```

.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;
}

```

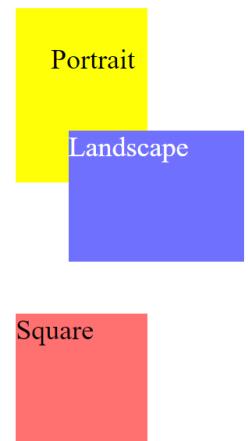
```

<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>

```

Position

Relative



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.

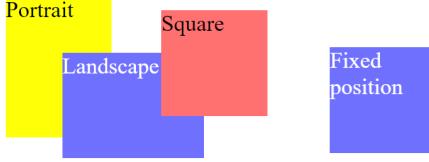
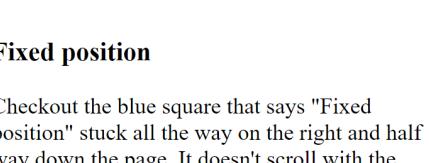
<i>index.css</i>	<i>index.html</i>	<i>Browser</i>
<pre> .wd-pos-absolute-10-10 { position: absolute; top: 10px; left: 10px; } .wd-pos-absolute-50-50 { position: absolute; top: 50px; left: 50px; } .wd-pos-absolute-120-20 { position: absolute; top: 20px; left: 120px; } </pre>	<pre> <h3>Absolute position</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>

 </pre>	<p>Absolute position</p> <p>The diagram illustrates the effect of absolute positioning. It shows three distinct elements: a yellow box labeled 'Portrait' positioned at the top-left, a blue box labeled 'Landscape' positioned to the right of the 'Portrait' box, and a red box labeled 'Square' positioned below the 'Landscape' box. All elements are rendered in their intended colors and positions relative to the surrounding text.</p>

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.

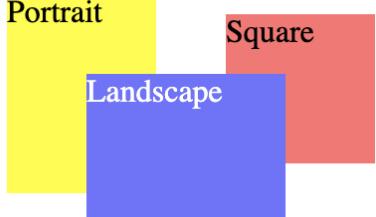
<i>index.css</i>	<i>index.html</i>	<i>Browser</i>
------------------	-------------------	----------------

<pre>.wd-pos-fixed { position: fixed; right: 0px; bottom: 50%; }</pre>	<p><h3>Fixed position</h3></p> <p>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".</p> <pre><div class="wd-pos-fixed wd-dimension-square wd-bg-color-blue wd-fg-color-white"> Fixed position </div></pre>	<p>Absolute position</p>  <p>Fixed position</p>  <p>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".</p>
--	---	--

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
<pre>.wd-zindex-bring-to-front { z-index: 10; }</pre>	<pre><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>

</pre>	<p>Z index</p> 

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	index.html
-----------	------------

```

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

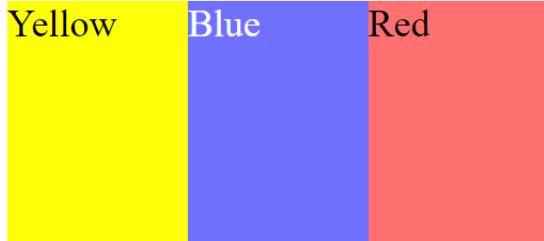
.wd-float-right {
  float: right;
  height: 100px;
}

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

<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>
  
  <div class="wd-float-done"></div>
</div>

```

Float



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

```

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

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

```

index.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>

```

```

.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;
}

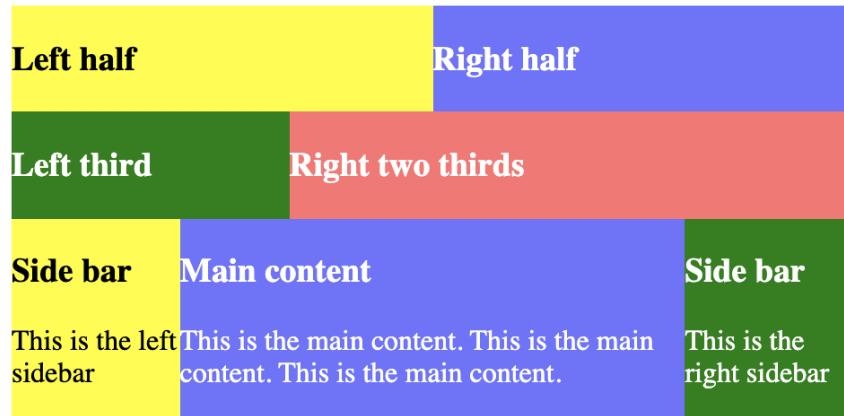
```

```

<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>

```

Grid layout



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 a container with display configured to flex as shown below.

<i>index.html</i>	<i>index.css</i>	<i>Browser</i>
<pre> <h2>Flex</h2> <div class="wd-flex-row-container"> <div class="bg-color-yellow">Column 1</div> <div class="bg-color-blue">Column 2</div> <div class="bg-color-red">Column 3</div> </div> </pre>	<pre> .wd-flex-row-container { display: flex; flex-direction: row; } </pre>	<p>Flex</p> <p>Column 1 Column 2 Column 3</p>

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.

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

<i>index.html</i>	<i>index.css</i>	<i>Browser</i>
<pre><h2>Flex</h2> <div class="wd-flex-row-container"> <div class="bg-color-yellow wd-width-75px"> Column 1 </div> <div class="bg-color-blue">Column 2</div> <div class="bg-color-red wd-flex-grow-1"> Column 3 </div> </div></pre>	<pre>.wd-flex-row-container { display: flex; flex-direction: row; } .wd-flex-grow-1 { flex-grow: 1; } .wd-width-75px { width: 75px; }</pre>	Flex  Column 1 Column 2 Column 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 in a couple of ways. You can [download the latest version from their download Website](#), unzip the zip file, and copy the content under their **dist** folder into your project. A common place to put libraries you download is under a **vendors** directory. Below is an example of having copied the content of the **dist** folder into a directory under **public/vendors/bootstrap**. This alternative is best for software developers where they might have local copies of various versions of libraries they use during the development effort. Local copies of libraries simplify the development process. Your actual location may differ so fix as needed.

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

Alternatively you can reference a CSS file hosted from any number of **CDNs (Content Delivery Network)**, remote servers dedicated to hosting common files and resources used by a community. This alternative is best for production environments when the Website is live. Your URL may differ so fix as needed.

```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet"/>
```

Feel free to use either of the alternatives described above.

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.

<code>index.html</code>	<i>Browser</i>
<pre><body> <div class="container"> <h1>Bootstrap</h1> <p>Welcome to Bootstrap!</p> <!-- main content goes here --> </div> </body></pre>	<h1>Bootstrap</h1> <p>Welcome to Bootstrap!</p>

3.2.2.2 Grid system (2pts)

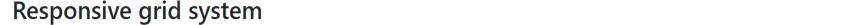
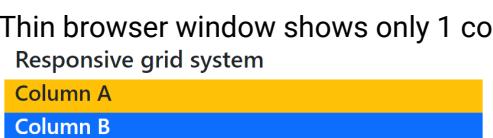
It's easy to break a page vertical in HTML with the `p` and `div` tags. It's a little harder to layout content horizontally. Some resort to HTML tables to layout content horizontally using table `rows` and `columns`, but this is generally considered a bad practice. HTML tables should be used for displaying 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.

<code>index.html</code>	<i>Browser</i>							
<pre><h2> Grid system</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></pre>	<p><i>!-- a row containing two columns applying class col with default width evenly spread over all columns</i></p> <p><i>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 1/3 and 8/12 is 2/3 of the screen</i></p> <p><i>--></i></p> <p><u>Grid system</u></p> <table border="1" style="width: 100%;"><tr><td style="background-color: #e74c3c; color: white; padding: 5px;">Left half</td><td style="background-color: #28a745; color: white; padding: 5px;">Right half</td></tr><tr><td style="background-color: #ffd700; color: black; padding: 5px;">One thirds</td><td style="background-color: #28a745; color: white; padding: 5px;">Two thirds</td></tr><tr><td style="background-color: #6c757d; color: white; padding: 5px;">Sidebar</td><td style="background-color: #6c757d; color: white; padding: 5px;">Main content</td><td style="background-color: #007bff; color: white; padding: 5px;">Sidebar</td></tr></table>	Left half	Right half	One thirds	Two thirds	Sidebar	Main content	Sidebar
Left half	Right half							
One thirds	Two thirds							
Sidebar	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	Browser
<pre data-bbox="226 335 631 485"><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></pre>	<p data-bbox="660 272 1183 297">Wide browser window shows 4 columns</p> <p data-bbox="660 297 951 323">Responsive grid system</p> 
	<p data-bbox="660 399 1313 422">Moderate width browser window shows 2 columns</p> <p data-bbox="660 422 951 447">Responsive grid system</p> 
	<p data-bbox="660 549 1227 574">Thin browser window shows only 1 column</p> <p data-bbox="660 574 951 599">Responsive grid system</p> 

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

```
<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">
```

Browser

Responsive grid system																							
1	2	3	4	5	6	7	8	9	10	11	12												
1		2	3	4	5	6	7		8	9	10	11	12										
1			2			3			4			5			6								
1				2				3				4				5							
1		2		3		4		5		6		7		8		9		10		11		12	

```
<h4>12</h4></div>
</div>
```

Responsive grid system



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

XS
S
M
L
XL
XXL

@ window width
<576px
≥576px
≥768px
≥992px
≥1200px
≥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>
```

Tables

Quiz	Topic	Date	Grade
Q1	HTML	2/3/21	85
Q2	CSS	2/10/21	90
Q3	JavaScript	2/17/21	90
Average			90

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>
```

Responsive tables

Very	long	set	of	columns	Very	long	set	of	columns	Very	long	set	of	columns
Very	long	set	of	columns	Very	long	set	of	columns	Very	long	set	of	columns

Responsive tables

mns	Very	long	set	of	columns	Very	lor
nns	Very	long	set	of	columns	Very	lor

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>

```

Favorite movies

- | |
|------------------|
| Aliens |
| Terminator |
| Blade Runner |
| Lord of the Ring |
| Star Wars |

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

```

<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>

```

Browser

Favorite books

- | |
|----------------------|
| Dune |
| Lord of the Rings |
| The Forever War |
| 2001 A Space Odyssey |
| Ender's Game |

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>


<label for="input1" class="form-label">
    Email address</label>
  <input type="email" class="form-control" id="input1" placeholder="name@example.com">



<label for="textarea1" class="form-label">
    Example textarea</label>
  <textarea class="form-control" id="textarea1" rows="3"></textarea>


```

Forms

Email address

Example textarea

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

Select

Open this select menu

✓ Open this select menu

One

Two

Three

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/#switches">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"

```

Browser

Switches

Default switch checkbox input

Checked switch checkbox input

Disabled switch checkbox input

Disabled checked switch checkbox input

```

        type="checkbox"
        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.

<i>index.html</i>	<i>Browser</i>
<pre> <h3> Range </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"/> </pre>	<p><u>Range</u></p> <p>Example range</p>

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.

<i>index.html</i>	<i>Browser</i>
<pre> <h3>Addons</h3> <div class="input-group mb-3"> \$ 0.00 <input type="text" class="form-control"> </div> <div class="input-group"> <input type="text" class="form-control"> </pre>	<p><u>Addons</u></p>

```

<span class="input-group-text">$</span>
<span class="input-group-text">
  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	Browser
<pre> <h3>Responsive forms</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> </pre>	<p>Responsive forms</p> <p>Email <input type="text" value="email@example.com"/></p> <p>Password <input type="text"/></p> <p>Bio <input type="text"/></p> <p>Responsive forms</p> <p>Email <input type="text" value="email@example.com"/></p> <p>Password <input type="text"/></p> <p>Bio <input type="text"/></p>

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	Browser

```

<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" 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

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>
```

Tabs

Active Link Link Disabled

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 below to the end of **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

Browser

```

<h2>
  <a href="https://getbootstrap.com/docs/5.1/components/card/">
    Cards
  </a>
</h2>
<div class="card" style="width: 18rem;">
  
  <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>

```

Cards



Card title

Some quick example text to build on the card title and make up the bulk of the card's content.

[Go somewhere](#)

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	mt-0	mb-0	ms-0	me-0	pt-0	pb-0	ps-0	me-0
top, bottom, start, end, top, bottom, start, end,	mt-1	mb-1	ms-1	me-1	pt-1	pb-1	ps-1	me-1
top-0, top-50, top-100, bottom-0, bottom-50, bottom-100,	mt-2	mb-2	ms-2	me-2	pt-2	pb-2	ps-2	me-2
start-0, start-50, start-100, end-0, end-50, end-100,
	mt-5	mb-5	ms-5	me-5	pt-5	pb-5	ps-5	me-5
	mt-auto		ms-auto		pt-auto		ps-auto	
	mb-auto		me-auto		pb-auto		pe-auto	

3.3 Decorating Documents with Fontawesome Icons

Fontawesome can be installed in a couple of ways. You can [download the latest version from their download Website](#), unzip the zip file, and copy the unzipped folder to **public/vendors/fontawesome** in your project. This alternative is best for software developers where they might have local copies of various versions of libraries they use during the development effort. Local copies of libraries simplify the development process. Add the following link tag to your HTML document.

```
<link href=".../vendors/fontawesome/css/all.css" rel="stylesheet"/>
```

Alternatively you can reference a CSS file hosted from any number of CDNs. CDN stands for Content Delivery Network and are remote servers dedicated to hosting common files and resources used by a community. This alternative is best for production environments when the Website is live.

```
<link href=
  "https://cdn.jsdelivr.net/npm/@fortawesome/fontawesome-free@6.2.1/css/fontawesome.min.css"
  rel="stylesheet"/>
```

Feel free to use either of the alternatives described above.

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	Browser
<pre><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>
 <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></pre>	<h1>Simple icons</h1> 

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	Browser
<pre><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>
 <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></pre>	<h1>Resizing icons</h1> 

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	Browser
<pre><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>
 <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></pre>	<h1>Coloring icons</h1> 

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
<pre><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>
</pre>	<p>Brand icons</p> 

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
<pre><h2>Stacking icons</h2> <i class="fas fa-square fa-stack-2x"></i> <i class="fab fa-twitter fa-stack-1x fa-inverse"></i>
 <i class="fas fa-circle fa-stack-2x"></i> <i class="fas fa-flag fa-stack-1x fa-inverse"></i>
 <i class="fas fa-camera fa-stack-1x"></i> <i class="fas fa-ban fa-stack-2x" style="color:Tomato"></i>
</pre>	<p>Stacking icons</p> 

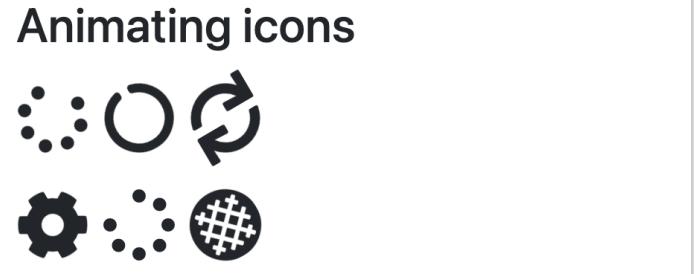
3.3.6 Rotating icons (2pts)

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
<pre><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></pre>	<p>Rotating icons</p> 

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
<pre><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></pre>	<p>Animating icons</p> 

4 Kanbas (50%)

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 screen shots 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 Kanbas Navigation Sidebars

The **Kanbas** Web application has several screens implementing different features. Figures 2.1, 2.2, and 2.3 illustrates the **Kanbas Navigation** sidebar with the **Account**, **Dashboard** and **Courses** links selected, respectively. When the **Account** link is selected, the submenu **Account Navigation** sidebar provides further navigation to the **Profile** and other account related features. When the **Courses** link is selected, the submenu **Course Navigation** side bar provides further navigation to course related features.

Implement the **Kanbas Navigation**, **Account Navigation**, and **Course Navigation** sidebars as illustrated below. Your implementation should try to look as close as possible to the screen shots in Figures 2.1 through 2.3, but they don't need to be pixel perfect. Below is a list of requirements your implementation must adhere to

4.1.1 Kanbas Navigation Sidebar Requirements

- Icons must be red, except the **Account** icon
- The **Account** icon 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
- Selected (active) links must have a white background with red text
- Non selected links must have a black background with white text
- The whole sidebar must have a black background
- The width of the sidebar must be between 75 and 85 pixels wide
- The spacing between (margin) text, icons, and links must be between 5 and 10 pixels
- The spacing around (padding) the links must be between 5 and 10 pixels

- The icons and text must be centered in the sidebar
- Clicking the **Account** link must display the **Account Navigation** sidebar and **Profile** screen described later
- Clicking the **Dashboard** link must display the **Dashboard** screen described later

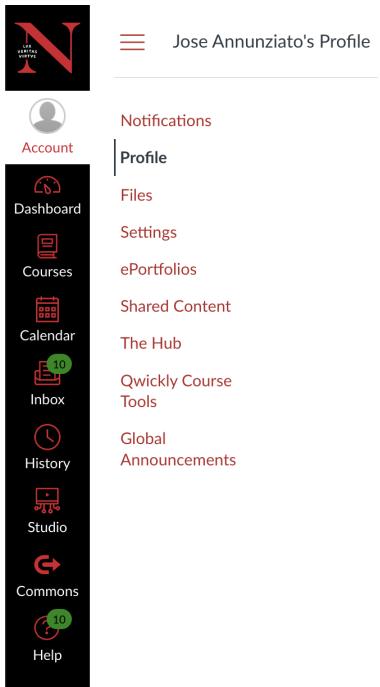


Fig 2.1 - Account

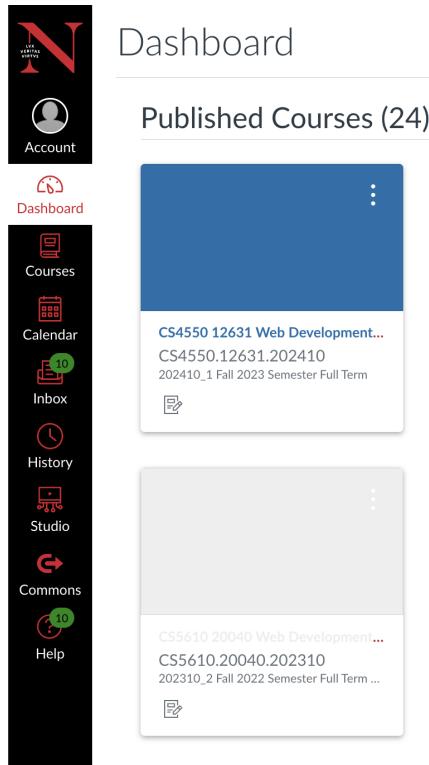


Fig 2.2 - Dashboard

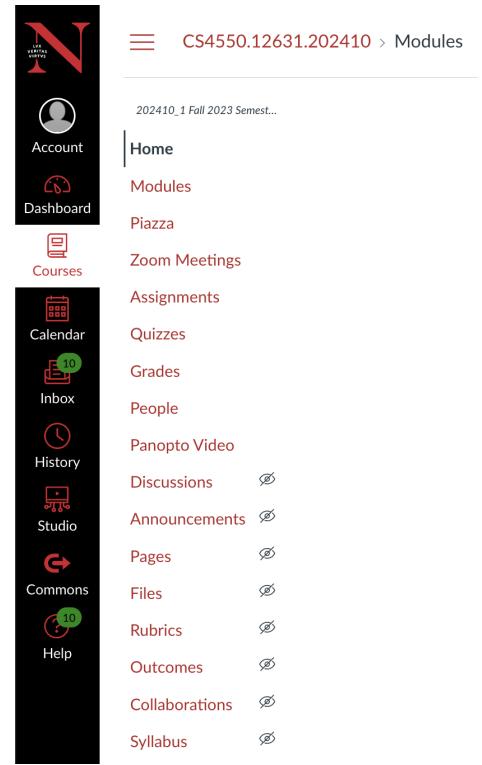


Fig. 2.3 - Courses

4.2 Account Navigation and Profile Screen

Clicking **Account** in the **Kanbas Navigation** sidebar, displays the **Account Navigation** sidebar and **Profile** screen as illustrated below in Figure 2.4. The **Profile** screen displays a user's account information. Clicking the **Edit Profile** button replaces the **Profile** screen with the **Edit Profile** screen where users could change their account information. Clicking **Cancel Editing** displays the **Profile** screen again. Implement the **Account Navigation** sidebar, **Profile** screen, and **Edit Profile** screen as shown below. The resulting implementation must be as close as possible to the screen shots in Figures 2.4 and 2.5, but they don't need to be pixel perfect. Below is a list of the requirements that must be met.

4.2.1 Account Navigation Sidebar Requirements

- Clicking **Account** in the **Kanbas Navigation** sidebar must display the **Account Navigation** sidebar and **Profile** screen
- Selected links (e.g., **Profile**) must have black text and a black border to the left to indicate they are selected
- Non selected links must have red text
- All the links must not be underlined
- The link text must be left justified
- The width of sidebar must be between 190 and 200 pixels
- There must be between 20 and 25 pixels between the right most edge of the **Kanbas Navigation** sidebar and the left most edge of a selected link in the **Account Navigation** sidebar
- There must be at least 9 links starting from **Notifications** through **Global Announcements** in the order shown

- Links with long text that do not fit in the width of the sidebar, must wrap
- The space between the links must be between 15 and 20 pixels
- In a selected link, the space between the left black border and the link's text, must be between 8 and 10 pixels

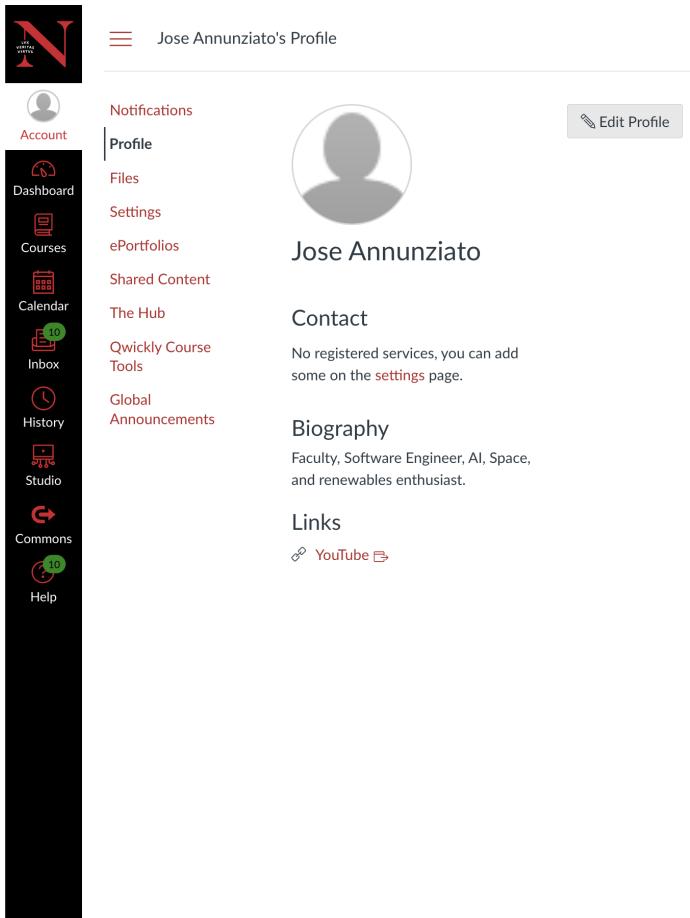


Figure 2.4 - Profile screen

The screenshot shows the 'Edit Profile' screen for 'Jose Annunziato'. It includes the same sidebar as Figure 2.4. The main content area has a 'Cancel Editing' button at the top right. It contains fields for 'Name:' (with 'Jose Annunziato' entered), 'Pronouns:' (set to 'None'), and 'Title:' (empty). Below these is a 'Contact' section and a 'Biography' section with the same text as the original profile. A 'Links' section is present with a table showing one entry: 'YouTube' with the URL 'https://www.youtube.com'. At the bottom are 'Cancel' and 'Save Profile' buttons.

Figure 2.5 - Edit Profile screen

4.2.2 Profile Screen Requirements

- A large account icon must render at top left of the screen using [Fontawesome's user icon](#)
- The full name of the account user must render below the icon as shown
- The **Contact** section must render as shown
- The **Biography** section must render as shown
- The **Links** section must render at least one hyperlink to the YouTube channel associated with this course, as shown
- Clicking the **Edit Profile** button displays the **Edit Profile** screen
- The **Edit Profile** and **Cancel Editing** buttons have a pencil icon to the left
- The buttons and their background colors must be implemented with [Bootstrap's btn classes](#). In particular the **Save Profile** has a red background as shown

4.2.3 Edit Profile Screen Requirements

- The fields **Name**, **Pronouns**, and **Title** must render labels and input fields as shown
- The **Name** and **Title** must be text input fields implemented with [Bootstrap's form-control class](#)
- The Contact section must render as shown

- The **Biography** section must render a textarea implemented with **Bootstrap's [form-control](#)** class with default value as shown and render as a block element
- The **Links** section must display two columns with headings **Title** and **URL** as shown
- The **Links** section must render at least one link with input form elements for the Title and the URL, rendering under the corresponding column
- The two columns and rows in the **Links** section must be implemented with **Bootstrap's [row and col](#)** classes
- Clicking **Cancel Editing**, **Cancel**, or **Save Profile** must display the **Profile** screen
- The pencil icon in the **Edit Profile** and **Cancel Editing** buttons must be implemented with **Fontawesome's [pencil](#)** icon
- The **Bootstrap's [float-end](#)** class must be used to render the **Edit Profile** and **Cancel Editing** to the right
- The buttons and their background colors must be implemented with **Bootstrap's [btn classes](#)**. In particular the **Save Profile** has a red background as shown

4.3 Dashboard Screen

Clicking **Dashboard** on the **Kanbas Navigation** sidebar, displays the **Dashboard** screen as shown below. This screen displays all the courses a user is related to either as a student or faculty. Clicking on a course displays the details of that course, described later in this assignment. Implement the **Dashboard** screen as described in this section making an effort to make it look as close as possible as the screenshots.

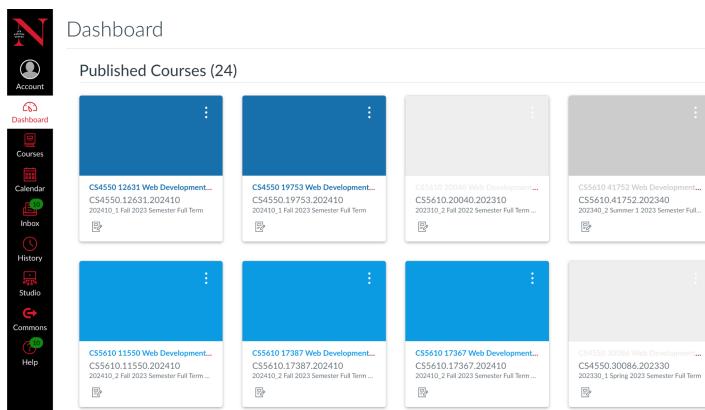


Figure 2.6

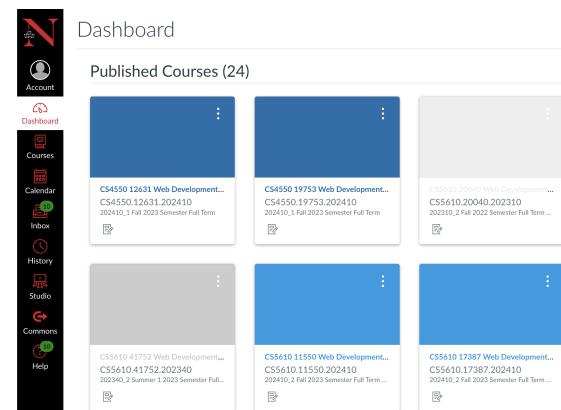


Figure 2.7

4.3.1 Dashboard Screen Requirements

Your implementation does not need to pixel perfect, but here is a list of general requirements you must adhere to.

- 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
- 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
- Bootstrap** classes **`d-flex`**, **`flex-row`**, and **`flex-wrap`** must be used to implement the grid of rows and columns
- 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 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 most 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

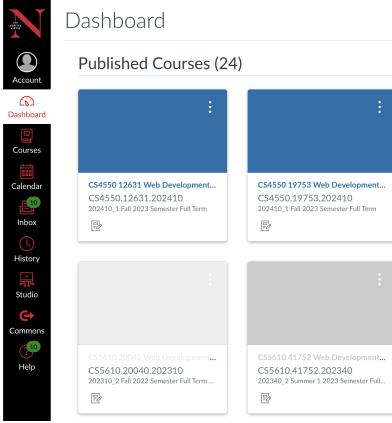


Fig. 2.8

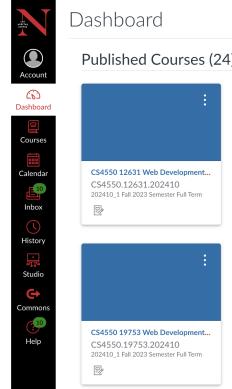


Fig. 2.9



Fig. 2.10

4.4 Course Navigation and Home Screen

Clicking on a course in the **Dashboard**, displays the course's details as shown below. The **Course Navigation** sidebar provides various links to navigate to different screens related to the course including **Home**, **Modules**, **Assignments**, and **Grades**. The **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 **Home** screen renders 4 columns as shown in Figure 2.11. The first column is the **Kanbas Navigation** sidebar, the second column is the **Course Navigation** sidebar, and third column contains the course's modules starting at **Week 0**, and 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 2.12. 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 2.13. 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 2.14. 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 2.15. Both, the **Kanbas Navigation** and **Course Navigation** can be dismissed by clicking the **X** icon at the top right. Implement the **Home** screen as described below.

4.4.1 Home Screen Requirements

- The **Home** 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 **Home** to indicate we are in the **Home** screen. Not **Modules** as shown in the screen shot
- Modules must be implemented with **Bootstrap's** **list-group** and **list-group-item** classes
- The module's titles must be highlighted using **Bootstrap's** **list-group-item-secondary** class
- White spaces must be added using **Bootstrap's** **margin** and/or **padding** classes, such as the white space before **Introduction to the course**

- Buttons, checkmarks and other elements shown on the right side of the screen must be implemented using **Bootstrap's *float-end*** class
- Button background colors must be implemented with **Bootstrap's *btn*, *btn-secondary*, and *btn-danger***
- Icons such as ***vertical ellipsis***, ***plus***, and ***check*** must be implemented with **Fontawesome**
- The **Collapse All**, **View Progress**, and **Module** buttons must be implemented as shown
- The **Publish All** dropdown must be implemented with **Bootstrap's dropdown classes**
- All padding and margins must be implemented as shown in the screen shot
- Hyperlinks must be color red and not underlined
- The checkmark icons must be green
- At least two modules must be implemented. Use the sample content provided
- At least two title line items must be implemented in each module
- At least one title line item must be indented as shown
- At least two hyperlinks must be implemented in each module
- You can ignore chevrons next to the module title, and the checkmark to its right

This screenshot shows the course navigation sidebar for Week 0 - INTRO. The sidebar includes sections for Home, Modules, Pizza, Zoom Meetings, Assignments, Quizzes, Grades, People, Panopto Video, Discussions, Announcements, Pages, Files, Rubrics, Outcomes, Collaborations, and Sylabus. Each section has a list of items, some with checkmark icons.

Fig. 2.11

This screenshot shows a similar course navigation sidebar for Week 0 - INTRO, but with more detailed content and icons. It includes sections for Home, Modules, Pizza, Zoom Meetings, Assignments, Quizzes, Grades, People, Panopto Video, Discussions, Announcements, Pages, Files, Rubrics, Outcomes, Collaborations, and Sylabus. The interface appears more modern and integrated.

Fig. 2.12

This screenshot shows a simplified course navigation sidebar for Week 0 - INTRO. It includes sections for Home, Modules, Pizza, Zoom Meetings, Assignments, Quizzes, Grades, People, Panopto Video, Discussions, Announcements, Pages, Files, Rubrics, Outcomes, Collaborations, and Sylabus. The layout is more compact and focused on the main modules.

Fig. 2.13

This screenshot shows a course navigation sidebar for Week 0 - INTRO with a different design. It includes sections for Home, Modules, Pizza, Zoom Meetings, Assignments, Quizzes, Grades, People, Panopto Video, Discussions, Announcements, Pages, Files, Rubrics, Outcomes, Collaborations, and Sylabus. The interface uses rounded corners and icons for better readability.

Fig. 2.14

This screenshot shows a very clean and minimalist course navigation sidebar for Week 0 - INTRO. It includes sections for Home, Modules, Pizza, Zoom Meetings, Assignments, Quizzes, Grades, People, Panopto Video, Discussions, Announcements, Pages, Files, Rubrics, Outcomes, Collaborations, and Sylabus. The design is focused on a few key modules and links.

Fig. 2.15

4.5 Assignments Screen

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

The screenshot shows the 'Assignments' screen for course CS5610.11744.202310. The sidebar on the left has 'Assignments' selected. The main content area displays a table of assignments with green borders on the left side of each row. The first assignment is 'A1 SETUP'.

Assignment	Description	Due Date	Points
A1 SETUP	Week 0 - SETUP - Week starting on Monday September 5th (9/5/2022) Module Due Sep 18, 2022 at 11:59pm 100 pts		100 pts
A2 HTML	Week 1 - HTML - Week starting on Monday September 12th (9/12/2022) Module Due Sep 25, 2022 at 11:59pm 100 pts		100 pts
A3 CSS	Week 2 - CSS - Week starting on Monday September 19th (9/19/2022) Module Due Oct 2, 2022 at 11:59pm 100 pts		100 pts
A4 BOOTSTRAP	Week 3 - Bootstrap - Week starting on Monday September 26th (9/26/2022) Module Due Oct 10, 2022 at 11:59pm 100 pts		100 pts
A5 JAVASCRIPT	Week 4 - JavaScript - Week starting on Monday October 3rd (10/3/2022) Module Due Oct 16, 2022 at 11:59pm 100 pts		100 pts
A6 REACT	Week 5 - React & Routing - Week starting Monday October 10th (10/10/2022) Module Due Oct 23, 2022 at 11:59pm 100 pts		100 pts

4.6 Edit Assignment Screen

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.

The screenshot shows the 'Edit Assignment' screen for assignment 'A1 - ENV + ...'. The sidebar on the left has 'Assignments' selected. The main content area includes fields for 'Assignment Name' (A1 - ENV + HTML), 'Description' (a text box with a long explanatory text), 'Points' (100), 'Assignment Group' (ASSIGNMENTS), 'Display Grade as' (Percentage), and a checkbox for 'Do not count this assignment towards the final grade'. There are also sections for 'Submission Type' (Online), 'Online Entry Options' (Text Entry, Website URL, Media Recordings), and checkboxes for 'Student Annotation' and 'File Uploads'.

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

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.

The wireframe shows a form titled "Assign". It has fields for "Assign to" (set to "Everyone"), "Due" (set to "Sep 18, 2023, 11:59 PM"), "Available from" (set to "Sep 6, 2023, 12:00 AM") and "Until" (empty). A "Notify users that this content has changed" checkbox is checked. At the bottom are "Cancel" and "Save" buttons.

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

The screenshot shows the Grades screen for CS5610.11744.202310. The sidebar includes links for Account, Dashboard, Courses, Calendar, Inbox, History, Studio, Commons, and Help. The main area shows a breadcrumb path: CS5610.11744.202310 > Grades. It has tabs for Home, Student Names, and Assignment Names. Search bars for "Search Students" and "Search Assignments" are present, along with an "Apply Filters" button. A table lists student names and their assignment scores:

Student Name	A1 SETUP Out of 100	A2 HTML Out of 100	A3 CSS Out of 100	A4 BOOTSTRAP Out of 100
Doxa Asibey	100%	100%	89.59%	98.65%
Thuy Tien Bui	100%	100%	91.5%	100%
Yetong Chen	100%	97%	100%	100%
Zhenyu Chen	100%	100%	98.64%	100%
Yung-Shin Chien	100%	100%	100%	100%
Basil Coughlan	100%	100%	100%	99.32%
Rachel Dao	100%	100%	100%	100%
Lana Do	100%	100%	100%	100%
Kartikey Dwivedi	100%	100%	100%	100%
Jiarong Feng	100%	100%	94.24%	100%
Saurabh Sunil Gade	100%	100%	100%	100%

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 Graduates

Revisit the Course Details Tab and Course Navigation Tab screens you implemented in the earlier assignment. Apply CSS and Bootstrap classes so that they look similar to the screen shots provided

6 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