**Bootstrap**

Bootstrap is a front-end framework used to design responsive web pages and web applications. It takes a mobile-first approach to web development. Bootstrap includes pre-built CSS styles and classes, plus some JavaScript functionality. Bootstrap uses a responsive 12 column grid layout and has design templates for:

* buttons
* images
* tables
* forms
* navigation

RESPONSIVE IMAGES

Fortunately, with Bootstrap, all we need to do is add the img-responsive class to your image to make it exactly the width of the phone’s screen.

RESPONSIVE LAYOUT

Bootstrap uses a responsive 12-column grid system, which makes it easy to put elements into rows and specify each element's relative width. Most of Bootstrap's classes can be applied to a div element.

Bootstrap has different column width attributes that it uses depending on how wide the user's screen is. For example, phones have narrow screens, and laptops have wider screens.

Take for example Bootstrap's col-md-\*class. Here, md means medium, and \*is a number specifying how many columns wide the element should be. In this case, the column width of an element on a medium-sized screen, such as a laptop, is being specified.

In the Cat Photo App that we're building, we'll use col-xs-\*, where xs means extra small (like an extra-small mobile phone screen), and \*is the number of columns specifying how many columns wide the element should be.

FONT AWESOME

Font Awesome is a convenient library of icons. These icons are vector graphics, stored in the .svg file format. These icons are treated just like fonts. You can specify their size using pixels, and they will assume the font size of their parent HTML elements.

You can include Font Awesome in any app by adding the following code to the top of your HTML:

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.5.0/css/font-awesome.min.css" integrity="sha384-XdYbMnZ/QjLh6iI4ogqCTaIjrFk87ip+ekIjefZch0Y+PvJ8CDYtEs1ipDmPorQ+" crossorigin="anonymous">

STYLING TEXT INPUTS

All textual <input>, <textarea>, and <select>elements with the class .form-control have a width of 100%.

JQUERY

* Now you know three ways of targeting elements:
  + by type: $("button"),
  + by class: $(".btn"),
  + and by id $("#target1").

In the same way you can add classes to an element with jQuery's addClass()function, you can remove them with jQuery's removeClass()function.

* We can also change the CSS of an HTML element directly with jQuery. jQuery has a function called .css()that allows you to change the CSS of an element.
  + $("#target1").css("color", "red")
* jQuery has a function called .prop()that allows you to adjust the properties of elements.
  + $("#target1").prop("disabled", true)
* jQuery has a function called .html()that lets you add HTML tags and text within an element. Any content previously within the element will be completely replaced with the content you provide using this function.
* Here's how you would rewrite and emphasize the text of our heading:
  + $("h3").html("<em>jQuery Playground</em>");
  + $("#target4").html("<em>#target4</em>")
* jQuery also has a similar function called .text()that only alters text without adding tags. In other words, this function will not evaluate any HTML tags passed to it, but will instead treat it as the text you want to replace the existing content with.
* jQuery has a function called appendTo()that allows you to select HTML elements and append them to another element.
  + $("#target2").appendTo("#right-well")

In addition to moving elements, you can also copy them from one place to another.jQuery has a function called clone() that makes a copy of an element. For example, if we wanted to copy target2 from our left-well to our right-well, we would use:

* $("#target2").clone().appendTo("#right-well");
* Did you notice this involves sticking two jQuery functions together? This is called function chaining and it's a convenient way to get things done with jQuery.

jQuery has a function called parent() that allows you to access the parent of whichever element you've selected.

* $("#left-well").parent().css("background-color", "blue")

jQuery has a function called children() that allows you to access the children of whichever element you've selected.

 jQuery has some other tricks for targeting the right elements. jQuery uses CSS Selectors to target elements. The target:nth-child(n)CSS selector allows you to select all the nth elements with the target class or element type.

* $(".target:nth-child(2)").addClass("animated bounce")

You can also target elements based on their positions using :odd or :even selectors. Note that jQuery is zero-indexed which means the first element in a selection has a position of 0. This can be a little confusing as, counter-intuitively, :odd selects the second element (position 1), fourth element (position 3), and so on.

* $(".target:even").addClass("animated shake")

Tear it all down!

$("body").addClass("animated hinge")

**SASS**

One feature of Sass that's different than CSS is it uses variables. They are declared and set to store data, similar to JavaScript.

In JavaScript, variables are defined using the let and const keywords. In Sass, variables start with a $ followed by the variable name.

Here are a couple examples:

$main-fonts: Arial, sans-serif;  
$headings-color: green;  
  
//To use variables:  
h1 {  
  font-family: $main-fonts;  
  color: $headings-color;  
}