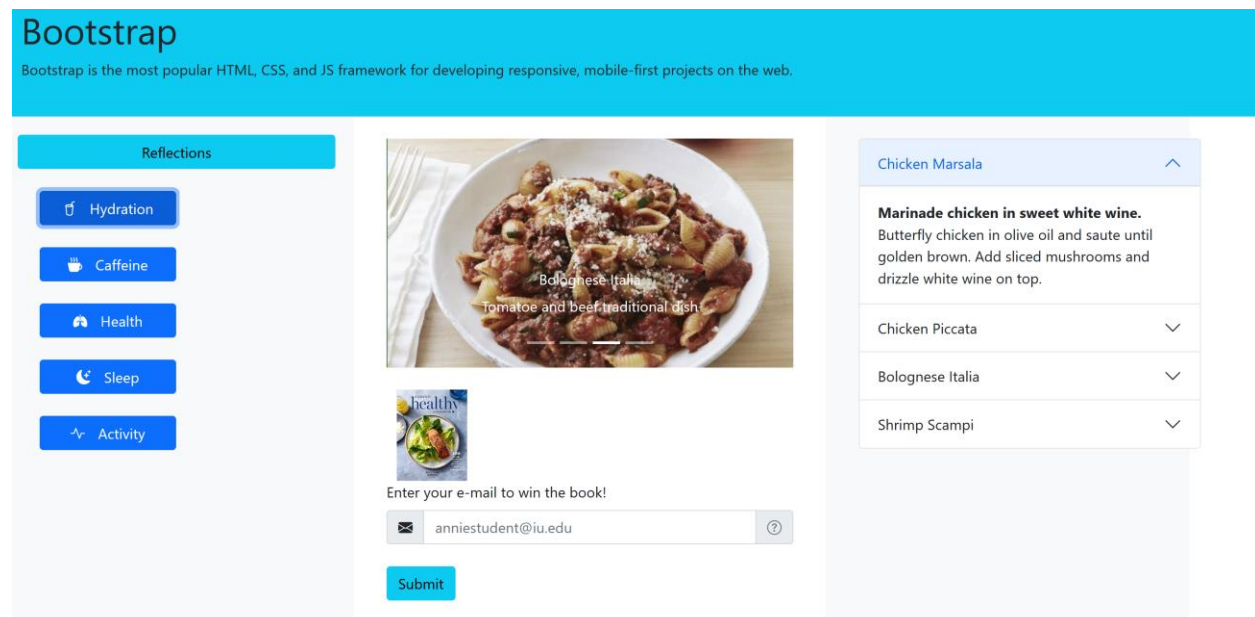


Introduction

This week, you will learn how to integrate sophisticated UI features, specifically a

- Bootstrap carousel (with indicators and handles)
- Bootstrap Tooltips
- Bootstrap Popovers
- Bootstrap Accordions
- Bootstrap Modals

The topic is food and health!



Before you start:

- Developing code is like both following and creating a recipe. There is foundational tasking to follow (some things have to be included for the code to work), after that you can use the concepts to create and deliver a more individualized solution.
- Bootstrap changes, and has changed very significantly. This is always something you have to watch for when using a 3rd party platform. See here: <https://getbootstrap.com/docs/5.0/migration/#jumbotron>
- In this lab, you will see many references to aria. A-r-i-a stands for accessible rich internet applications. Aria content is added to improve accessibility for impaired users, and is an important part of access and useability. We will discuss this more over the next weeks.
- Note any occurrences of the aria-xxx classes. These are specifically for assistive technology. We will be discussing ARIA, the references here are from the Bootstrap documentation, and considered part of recommended use.
- You will see many references to data-bs-toggle. Here the -bs- refers to to bootstrap, and the toggle refers to a change of state, which an action (like hover or click) causes.

- You will see many references to data-bs-target. This wires an action (like hover or click) to the display of another content block that may have been hidden.
- You will see references to a state of collapse(d). This is a special state recognized by the browser and by Bootstrap. It means that the associated element is not taking up any space.

1 – Bootstrap Buttons

Let's start by adding a set of buttons, already covered in a prior class.

This time, let's add a Bootstrap icon to enhance the visual impact.

```
<button class="btn btn-md btn-primary d-block m-4">
  <span> <i class="bi bi-cup-straw"></i></span>
  <span style="margin-left: 10px">Hydration</span>
</button>
```

Here, with span tags, we use both an icon and wording. d-block displayed as block, and m-4 adds 4 units of margin

Now add 3 more of these buttons, to a button container,

```
<div class="button-container">
  <h4>This week's thoughts ...</h4>...
```


Use

bi bi-cup-hot-fill	Caffeine
bi bi-lungs-fill	Health
bi bi-activity	Activity
bi bi-moon-stars-fill	Sleep

Add a little styling to make the buttons of uniform width.

```
.button-container .btn {
  width: 150px;
}
```

This week's thoughts ...

 Hydration

 Caffeine

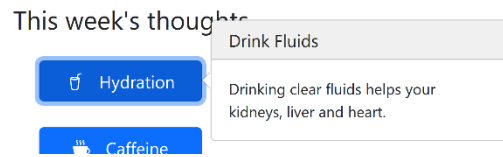
 Health

 Sleep

 Activity

2 – Bootstrap Popovers

Popovers are features that enable a popup box to display when the user clicks on an element. They are supported by popper.js (which we have been including as part of our base set of includes for Bootstrap).



Popovers are one of the set of 3 features (popover, modal, tooltip) in this lab, that conserve space on the page by allowing the user to elect the display of some more information. Popovers are perfect for enhancing the button set that we created.

Let's start by wiring up the popover for the first button.

First requirement is some JavaScript. Put this in a separate file, enabler.js, and make sure to include the defer directive when including the file.

```
<script src="enabler.js" defer></script>
```

You can get the JavaScript directly from the Bootstrap site, it looks complicated but it is only selecting the popover components from the page, and creating a element that corresponds to the popover enabled element.

```
var popoverTriggerList = [].slice.call(
  document.querySelectorAll('[data-bs-toggle="popover"]')
);
var popoverList = popoverTriggerList.map(function (popoverTriggerEl) {
  return new bootstrap.Popover(popoverTriggerEl);
});
```

Ok, next step is to extend the button behaviour. Add attributes,

<code>data-bs-container="body"</code>	puts the popover in the body of the document
<code>data-bs-toggle="popover"</code>	establishes popover behaviour
<code>data-bs-placement="top"</code>	directs where the popover appears: top, right, bottom, left
<code>data-bs-content="Right popover"</code>	the text content of the body of the popover
<code>title="Drink Fluids"</code>	the text content of the title of the popover

Let's modify that first button,

```
<button
  class="btn btn-md btn-primary d-block m-4"
  data-bs-container="body"
  data-bs-toggle="popover"
  data-bs-placement="right"
```

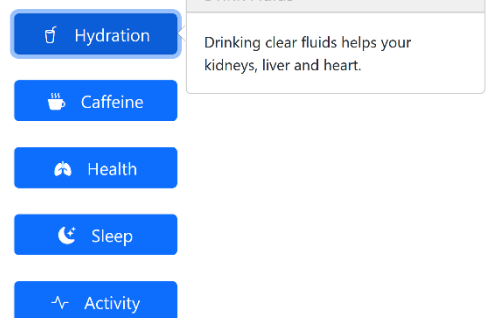
```

        title="Drink Fluids"
        data-bs-content="Drinking clear fluids helps your kidneys, liver and
heart."
    >
    <span> <i class="bi bi-cup-straw"></i></span>
    <span style="margin-left: 10px">Hydration</span>
</button>

```

OK, so now when you click the button, you should see,

This week's thoughts



You have to click again on the button to close the

popover – this is the popover behaviour.

Now add the content and change the way the other buttons react.

	Title	Text
Caffeine	Careful	Caffeine raises your heart rate and causes headaches and dehydration
Health	Wellness	Feeling good is all about how you treat your body
Sleep	Rest	A good night's sleep is essential for good health
Activity	Exercise	Burn calories and add muscle mass with activity

Now all 5 buttons should be enabled as popovers.

Let's add the set of buttons to a button group, and put that in the first column of a Bootstrap container.

3 - Bootstrap Carousel

A Bootstrap carousel is a special control that cycles through content, like a carousel. It is a very commonly used Bootstrap control.



Let's add the carousel to the next column over, you have already put the buttons in the first column. Here is an overview of what you should have, although the column/row arrangement is up to you. Of course, you can start with the col/row spec here, and then change it later.

```

<div class="container">
  <div class="row row-cols-sm-2 row-cols-md-3 row-cols-lg-4 row-cols-xl-5">
    <!-- side buttons -->
    <div class="col button-container">

```

```

</div>
<!-- end side buttons-->
<div class="col carousel-area">
    .....
</div>
</div>

```




So, we will be adding the carousel markup to the carousel column.

Let's add an empty carousel first,

<pre> <div id="course_carousel" class="carousel carousel-light" data-bs-ride="carousel" data-bs-interval="2000"> ... </div> </pre>	<p>The id of the carousel is important because we need that to associate the carousel items with the carousel. The data-bs-ride attribute animates the carousel once the page is loaded. This is typical, because otherwise you have to manually sequence through the items. The class of carousel-light applies a light text, there is also carousel-dark.</p> <p>The interval (data-bs-interval) is in milliseconds, so here the interval is 2 seconds between automatic progressions.</p>
--	--

Now let's add the inner part of the carousel, this is where the carousel items are stored.	<pre> <div id="course_carousel" > <div class="carousel-inner"> </div> </div> </pre>
--	--

Typically, each carousel item is composed of at least an image, and sometimes some caption text. Carousels are created for images, so using other content (although it is possible to include in the carousel) is not recommended. These controls were created to add visual impact. Here are images you can use for the other items.

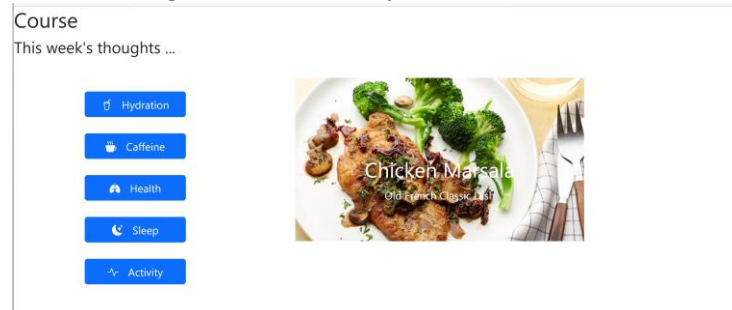
		
Bolognese Pasta	Chicken Piccata	Shrimp Scampi
Hearty Italian comfort food	Low calorie flavourful Italian dinner	Universal healthy seafood dish

Let's add an item to the carousel, inner part. Note, for the lab, we are using an assets folder to hold the images, so we need to access the images from here (or wherever you have the images).

```
<div class="carousel-item active">
  
  <div class="carousel-caption">
    <h2>Chicken Marsala</h2>
    <p>Old French Classic Dish</p>
  </div>
</div>
```

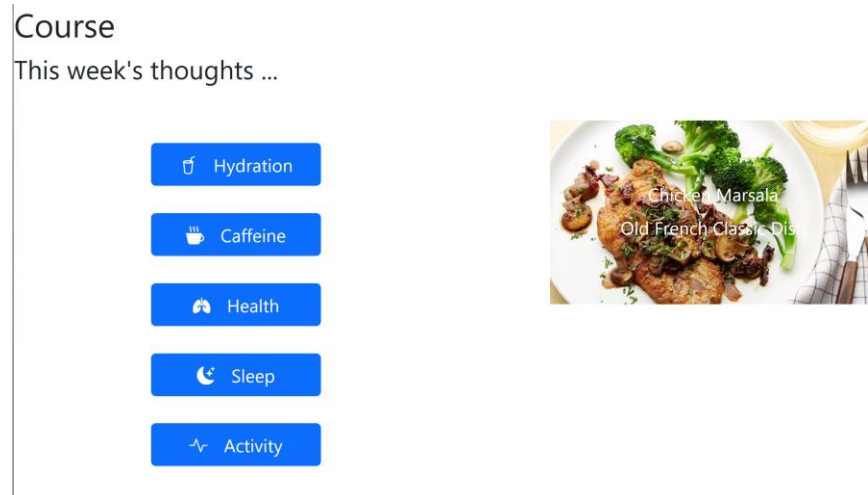
Nothing unusual here, just the image and some caption text. This is the first carousel item, we want it to appear, so we include the **active** class. Don't include *active* in the subsequent carousel items.

Before adding the other items, you should see:



Now add the other 3 items.

You should see a progression to the next image after 2 seconds.



Now let's add the handles for previous and next, so that there is also manual movement, and some indicators to reflect the position of the current image in the sequence. These will appear at the sides of

the image in the carousel, previous button on the left, next button the right. By default, they are very subtle. We have carousel-light, so they will be light.



The previous (<) and next(>) handles are placed after the carousel-inner div, and can be buttons. They can also be any of the other Bootstrap representations for button elements.

Here is how the previous button should be styled

```
<button
  class="carousel-control-prev"
  type="button"
  data-bs-target="#course_carousel"
  data-bs-slide="prev">
  <span
    class="carousel-control-prev-icon"
    aria-hidden="true">
  </span>
  <span
    class="visually-hidden">
    Previous
  </span>
</button>
```

It is important to set the data-bs-target attribute to the id of the carousel, course-carousel.

The arrow symbol is given by the carousel-control-prev-icon class.

The second span element is specifically for assistive technology, and is hidden.

Here is how the elements should be structured:

```
<div class="carousel-inner">
  <div class="carousel-item active">
  </div>
  ...
  <div class="carousel-item">
  </div>
</div>
<!-- buttons to control navigation are placed outside carousel inner -->
<button
  class="carousel-control-prev"
  type="button"
  data-bs-target="#course_carousel"
  data-bs-slide="prev"
>
  ...
</button>
<button
  class="carousel-control-next"
  type="button"
  data-bs-target="#course_carousel"
  data-bs-slide="next"
>
  <span
    class="carousel-control-next-icon"
    aria-hidden="true"
  ></span>
  <span class="visually-hidden">Next</span>
</button>
```

Now you should see the previous and next handles. These can be clicked to progress to the next or previous carousel item – this is what the attribute value for ***data-bs-slide*** impacts.

Note, the span element with class of visually-hidden, is specifically for screen readers. If you find a screen reader, the content is voiced as ‘next(or previous) slide accessible from controller’.

Indicators are used to give feedback about the sequence in the carousel. They are subtle also, by default.

Indicators should be placed just inside the carousel div, and before the carousel-inner.


```

<div
  id="course_carousel"
  class="carousel carousel-light"
  data-bs-ride="carousel"
  data-bs-interval="2000"
>
  <div class="carousel-indicators"> ...
  </div>
  <div class="carousel-inner">
    <div class="carousel-item active">

```

Start by adding the carousel-indicators container,

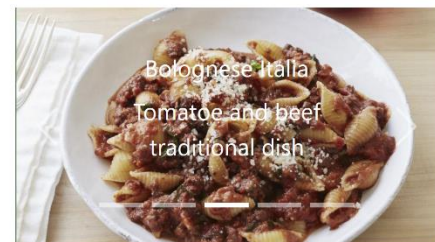
```
<div class="carousel-indicators"></div>
```

Now, for each carousel-item, add an indicator. It is important to include an indicator for each carousel-item (make sure the number of carousel items is the same as the number of indicators), and use the active class for the first one. Notice that the ***data-bs-slide-to*** attribute is associated with the ***number of the carousel item***, starting at 0.

```

<div class="carousel-indicators">
  <button
    type="button"
    data-bs-target="#course-carousel"
    data-bs-slide-to="0"
    class="active"
  ></button>
  <button
    type="button"
    data-bs-target="#course-carousel"
    data-bs-slide-to="1"
  >...
</div>

```

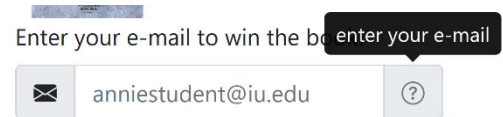


Add the rest of the indicators for the items you have in the carousel.

4 – Bootstrap Tooltips

Tooltips are transitory hints or reminders, and are invoked by hovering over the tooltip source, like an image, a button or other.

Add a tooltip to an image, first and then see how to use a tooltip to improve the useability of a form field.



Tooltips need some JavaScript help as well, this can be copied from the Bootstrap site (<https://getbootstrap.com/docs/5.0/components/tooltips/>) or you can create your own.

```
var tooltipTriggerList = [].slice.call(document.querySelectorAll('[data-bs-toggle="tooltip"]'))
var tooltipList = tooltipTriggerList.map(function (tooltipTriggerEl) {
  return new bootstrap.Tooltip(tooltipTriggerEl)
})
```

Add an image, with a tooltip effect, below the carousel in column 2.

```
<div class="tooltip-area">...</div>
```

```
<div>
  <span
    data-bs-toggle="tooltip"
    data-bs-placement="top"
    title="Cook book">
    
  </span>
</div>
```

Here, the data-bs-toggle is set to Bootstrap's tooltip. The placement can be top, bottom, left, right - controlled by the value of data-bs-placement attribute.

The title text value is what comes up on hovering over the image area, contained by the span tag.



Cook book

Typically, a tooltip is used in a form to give information about fields. Here's another example of using a tooltip, this time to explain what to enter in an input field.

Here, we create an input field with a Bootstrap icon at the end, this is the tooltip element. Add this in after the book image.

```
<form>
  <label for="e-mail" class="form-label">E-Mail Address</label>
  <div class="mb-4 input-group">
    <span class="input-group-text">
      <i class="bi bi-envelope-fill"></i>
    </span>
    <input
      type="email"
      class="form-control"
      id="email"
      placeholder="anniestudent@iu.edu"/>
    <span
      class="input-group-text text-muted"
      data-bs-toggle="tooltip"
      data-bs-placement="top"
      title="enter your e-mail">
      <i class="bi bi-question-circle"></i>
    </span>
  </div>
</form>
```

It should produce this effect,



5 - Bootstrap Accordion

This control saves space on the page (just as the other controls do) by allowing a content area to expand and collapse.

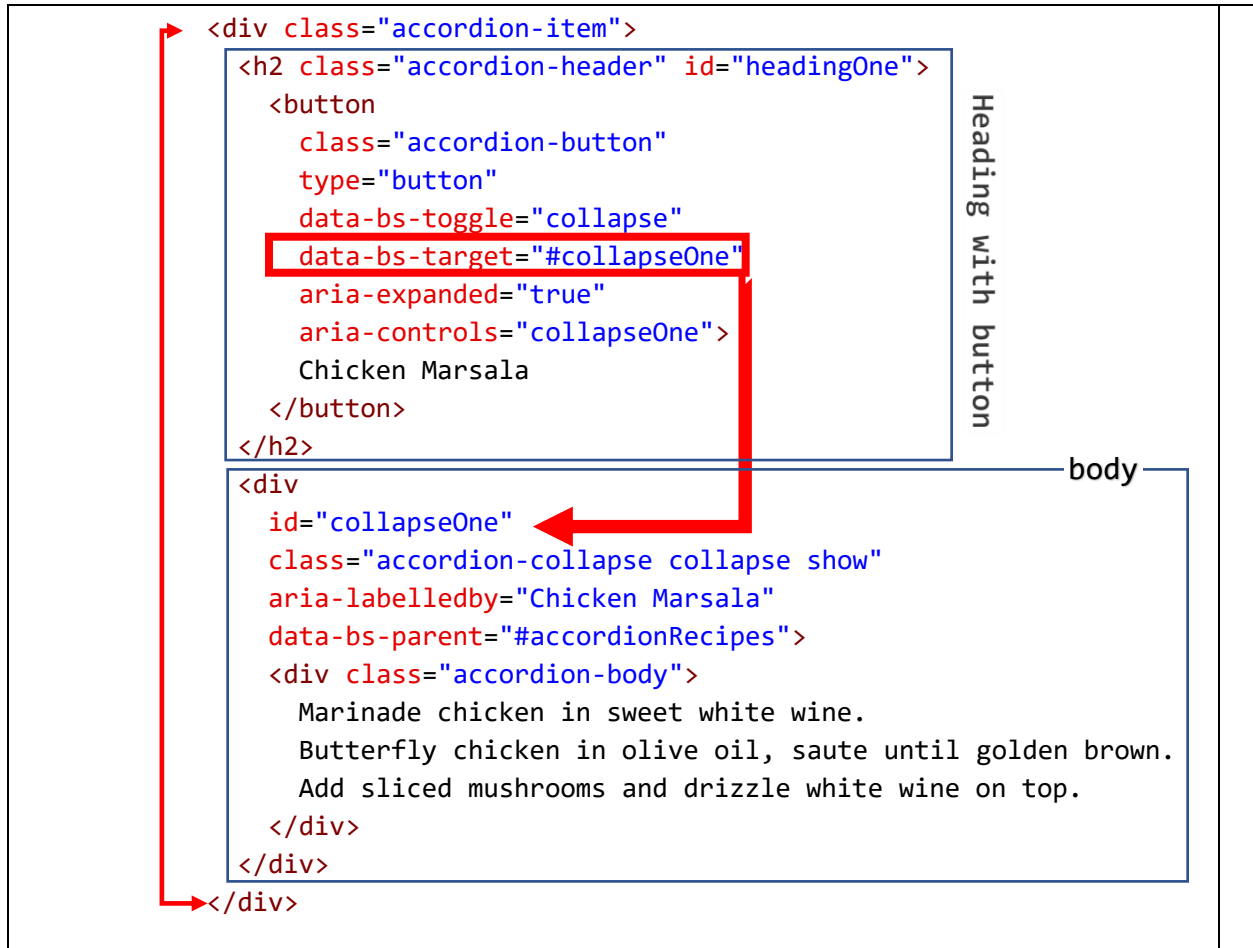
Let's add an accordion in column 3.

```
<div class="col accordion-area">
  <div class="accordion" id="accordionRecipes">
  </div>
```

</div>

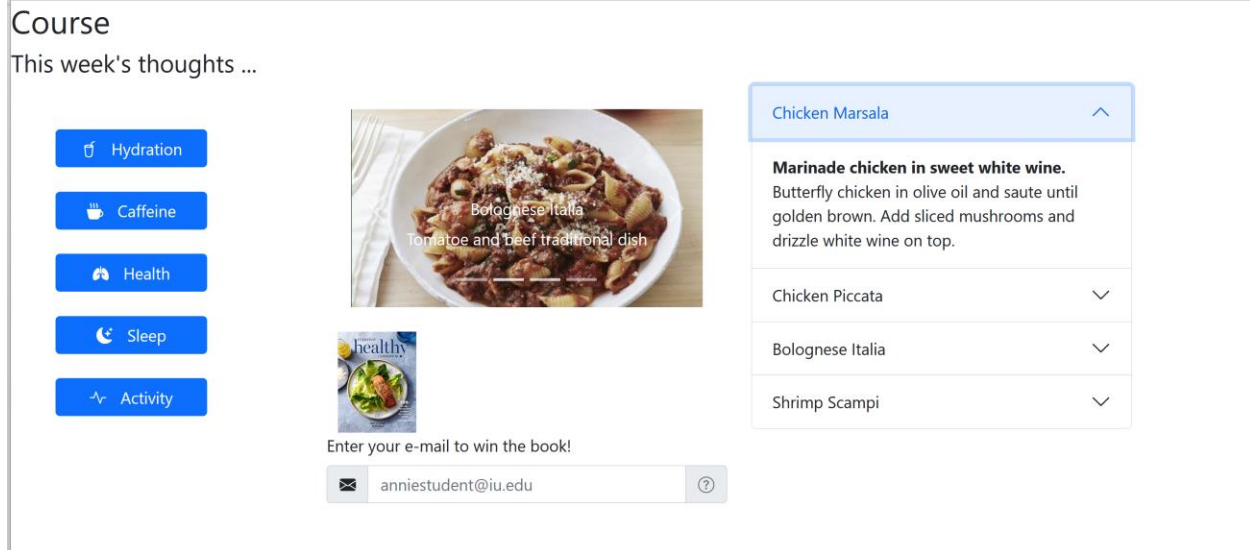
The accordion items should be child elements of the div of class accordion. Each accordion item should have a button (to click for expand/collapse), a title and a body.

Here is the first one



- As in the popovers and tooltips, the `data-bs-target` points to the element to be displayed – this has to be wired carefully
- `data-bs-parent` is set to the id of the accordion

This is what should be rendered



The first item is expanded by default, although you can switch this off.

The first item's button should have the class indicated, of accordion-button.

```
<div class="accordion-item">
  <h2 class="accordion-header" id="headingOne">
    <button
      class="accordion-button"
      type="button" ....
```

All of the other items should include the collapsed class, as in

```
<button
  class="accordion-button collapsed"
  type="button" .....
```

The Bootstrap guide includes an aria- reference, to indicate expanded or collapsed – these should also be changed to reflect initial state.

Here are the other summaries to add for the other accordion items

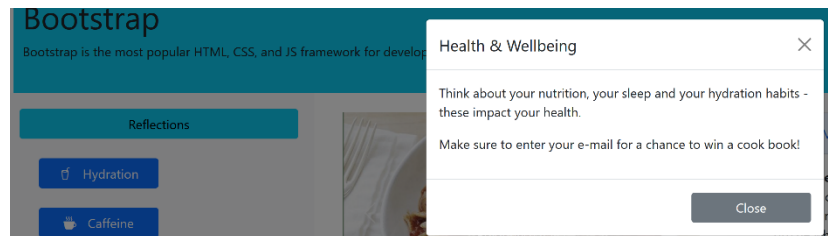
Chicken Piccata	Saute the chicken in olive oil, add capers and sprinkle on paprika
Bolognese Italia	Saute minced beef until browned, add sliced onions, tomato paste, garlic. Season to taste with salt and pepper.

Shrimp Scampi	Saute shrimp (tail off) in oil with lemon seasoning. Add garlic powder. Boil linguini or spaghetti. Fold shrimp and sauces into pasta.
---------------	--

6 - Bootstrap Modal

Finally in this lab, let's add the modal. A modal, also termed dialog box, is an information window that pops up. You cannot do anything with the rest of the application until you close it, hence the term **modal**.

Let's put this button in column 1, before the buttons group. When clicked, the sample modal will reveal.



```
<button type="button" class="btn btn-primary" data-bs-toggle="modal" data-bs-target="#reflectionsModal">
  Reflections
</button>
```

Now we have to wire this to a content area, which is collapsed until we click the toggle button. The markup for this content block can be placed anywhere in the body of the document, but it makes sense to keep it close to the originating button.

```
<div class="modal fade"
  id="reflectionsModal"
  tabindex="-1"
  aria-labelledby="reflectionsModalLabel"
  aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">

      </div>
      <div class="modal-body">

      </div>
      <div class="modal-footer">

      </div>
    </div>
  </div>
</div>
```

```
<h5 class="modal-title"
  id="reflectionsModalLabel">
  Health & Wellbeing
</h5>
<button type="button"
  class="btn-close"
  data-bs-dismiss="modal"
  aria-label="Close">
</button>
```

```
<p>
  Think about your nutrition, your
  sleep and your hydration habits
  - these impact your health.</p>
<p>
  Make sure to enter your e-mail
  for a chance to win a
  cookbook!</p>
```

```
<button type="button"
  class="btn btn-secondary"
  data-bs-dismiss="modal">
  Close
</button>
```

In this lab, you have seen how to use a modal, tooltips, popovers, and a carousel. These are commonly used Bootstrap items, but of course there are more. There are also more sophisticated ways of using these Bootstrap features.

The last remaining task is to add responsiveness. Here is an example of how you might add classes to each column.

```
<div class="container-fluid">  
  <div class="row">  
    <!-- column 1, side buttons - tooltips -->  
    <div class="col-sm-12 col-md-3 button-container bg-light">...  
  </div>  
  <!-- end column 1-->  
  
  <!-- column 2 - carousel and popovers -->  
  <div class="col col-sm-10 col-md-4 carousel-area bg-body bg-gradient">...  
  </div>  
  
  <!-- column 3, accordion -->  
  <div class="col col-sm-12 col-lg-4 accordion-area bg-light">...  
  </div>  
</div>
```