Links

<https://getbootstrap.com/docs/4.0/layout/grid/>

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

<https://blog.teamtreehouse.com/modern-field-guide-responsive-web-design>

<https://www.webfx.com/blog/web-design/information-architecture-101-techniques-and-best-practices/>

<https://developer.mozilla.org/en-US/docs/Web/Accessibility/ARIA>

<https://www.steptwo.com.au/papers/kmc_whatisinfoarch/>

Buttons - <https://css-tricks.com/use-button-element/>

Font Icons - <http://vanseodesign.com/web-design/icon-fonts/>

<https://fontawesome.com/>

<http://lipis.github.io/bootstrap-social/>

<https://gomakethings.com/using-svgs/>

Navbar Examples - <https://www.hongkiat.com/blog/navigation-design-ideas-inspiration/>

<https://www.hongkiat.com/blog/breadcrumb-navigation-examined-best-practices-examples/>

Host Websites - <https://www.webmonkey.com/>

React - <https://reactjs.org/>

Build Website - <https://ionic.io/customers>

<https://www.adobe.com/products/xd.html>

<https://expo.getbootstrap.com/>

<https://www.webdesign-inspiration.com/>

<https://onextrapixel.com/free-bootstrap-wireframing-set-for-powerpoint/>

Tools - <https://www.creativebloq.com/wireframes/top-wireframing-tools-11121302>

Basic Git Commands

At a convenient location on your computer, create a folder named git-test.

Open this git-test folder in your favorite editor.

Add a file named index.html to this folder, and add the following HTML code to this file:

12345678

<!DOCTYPE html>

<html>

<head></head>

<body>

<h1>This is a Header</h1>

</body>

</html>

Initializing the folder as a Git repository

Go to the git-test folder in your cmd window/terminal and type the following at the prompt to initialize the folder as a Git repository:

git init

Checking your Git repository status

Type the following at the prompt to check your Git repository's status:

git status

Adding files to the staging area

To add files to the staging area of your Git repository, type:

git add .

Commiting to the Git repository

To commit the current staging area to your Git repository, type:

git commit -m "first commit"

Checking the log of Git commits

To check the log of the commits to your Git repository, type

git log --oneline

Now, modify the index.html file as follows:

123456789

<!DOCTYPE html>

<html>

<head></head>

<body>

<h1>This is a Header</h1>

<p>This is a paragraph</p>

</body>

</html>

Add a sub-folder named templates to your git-test folder, and then add a file named test.html to the templates folder. Then set the contents of this file to be the same as the index.html file above.

Then check the status and add all the files to the staging area.

Then do the second commit to your repository

Now, modify the index.html file as follows:

12345678910

<!DOCTYPE html>

<html>

<head></head>

<body>

<h1>This is a Header</h1>

<p>This is a paragraph</p>

<p>This is a second paragraph</p>

</body>

</html>

Now add the modified index.html file to the staging area and then do a third commit.

Checking out a file from an earlier commit

To check out the index.html from the second commit, find the number of the second commit using the git log, and then type the following at the prompt:

git checkout <second commit's number> index.html

Resetting the Git repository

To discard the effect of the previous operation and restore index.html to its state at the end of the third commit, type:

git reset HEAD index.html

Then type the following at the prompt:

git checkout -- index.html

You can also use git reset to reset the staging area to the last commit without disturbing the working directory.

git remote add origin <repository URL>

git push -u origin master

git clone <repository URL>

Initializing package.json

* At the command prompt in your **git-test** folder, type

1

npm init



* Follow along the prompts and answer the questions as follows: accept the default values for most of the entries, except set the entry point to index.html
* This should create a *package.json* file in your **git-test** folder.

Installing an NPM Module

* Install an NPM module, lite-server, that allows you to run a Node.js based development web server and serve up your project files. To do this, type the following at the prompt:

1

npm install lite-server --save-dev





* You can check out more documentation on lite-server [here](https://github.com/johnpapa/lite-server).
* Next, open package.json in your editor and modify it as shown below. Note the addition of two lines, line 7 and line 9.

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

{

  "name": "git-test",

  "version": "1.0.0",

  "description": "This is the Git and Node basic learning project",

  "main": "index.html",

  "scripts": {

    "start": "npm run lite",

    "test": "echo \"Error: no test specified\" && exit 1",

    "lite": "lite-server"

  },

  "repository": {

    "type": "git",

    "url": "git+https://jogesh\_k\_muppala@bitbucket.org/jogesh\_k\_muppala/git-test.git"

  },

  "author": "",

  "license": "ISC",

  "homepage": "https://bitbucket.org/jogesh\_k\_muppala/git-test#readme",

  "devDependencies": {

    "lite-server": "^2.2.2"

  }

}





* Next, start the development server by typing the following at the prompt:

1

npm start





* This should open your *index.html* page in your default browser.
* If you now open the *index.html* page in an editor and make changes and save, the browser should immediately refresh to reflect the changes.

Setting up .gitignore

* Next, create a file in your project directory named *.gitignore* (**Note**: the name starts with a period)Then, add the following to the .gitignore file

1

node\_modules





* Then do a git commit and push the changes to the online repository. You will note that the node\_modules folder will not be added to the commit, and will not be uploaded to the repository.

Setting up the Project Folder

* Go to a convenient folder location on your computer and download the ***Bootstrap4-starter.zip*** file using the link provided at the top of this page.
* Unzip the file to see a folder named ***Bootstrap4*** and a sub-folder under it named ***conFusion*** created. Move to the *conFusion* folder.
* Open a cmd window/terminal and move to the conFusion folder.
* At the prompt type

1

npm install





* This will install the lite-server node module to your project.
* Next, initialize a Git repository in the project folder, and then set up a .gitignore file with the contents as shown below:

1

node\_modules





* Now do a commit of your project folder to the Git repository with the message "Initial Setup". You will be doing a commit of your project at the end of each exercise so that you retain the completed files of each exercise.
* Set up an online Git repository and synchronize your project folder with the online repository.

Downloading Bootstrap

* You will use npm to fetch the Bootstrap files for use within your project. Thereafter you need to install JQuery and Popper.js as shown below since Bootstrap 4 depends on these two. At the prompt, type the following to fetch Bootstrap files to your project folder:

1

2

npm install bootstrap@4.0.0 --save

npm install jquery@3.3.1 popper.js@1.12.9 --save





* This will fetch the Bootstrap files and store is in your node\_modules folder in a bootstrap folder. The bootstrap->dist folder contains the precompiled Bootstrap CSS and JS files for use within your project.
* Open your project folder in your editor, and then open the index.html file in the *conFusion* folder. This is your starting web page for the project. We have already created the web page with some content to get you started. We will use Bootstrap to style this web page, and learn Bootstrap features, classes and components along the way.
* Start your lite-server by typing **npm start** at the prompt. The *index.html* file should now be loaded into your default browser.

Getting your Web page Bootstrap ready

* Open the *index.html* file in your favourite text editor. If you are using Visual Studio Code, Brackets, Sublime Text or similar editors, you can open the project folder in the editor and then view index.html.
* Insert the following code in the *<head>* of *index.html* file before the title.

1

2

3

4

5

6

7

    <!-- Required meta tags always come first -->

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <meta http-equiv="x-ua-compatible" content="ie=edge">

    <!-- Bootstrap CSS -->

    <link rel="stylesheet" href="node\_modules/bootstrap/dist/css/bootstrap.min.css">





* This will include Bootstrap CSS into your web page. Note the subtle change in the fonts of the content of the web page. This is the Bootstrap typography effect coming into play. The default Bootstrap typography sets the font to Helvetica Neue and selects the appropriate font size based on the choice of the heading style and paragraph style for the content.
* At the bottom of the page, just before the end of the body tag, add the following code to include the JQuery library, popper.js library and Bootstrap's Javascript plugins. Bootstrap by default uses the JQuery Javascript library for its Javascript plugins. Hence the need to include JQuery library in the web page.

1

2

3

4

    <!-- jQuery first, then Popper.js, then Bootstrap JS. -->

    <script src="node\_modules/jquery/dist/jquery.slim.min.js"></script>

    <script src="node\_modules/popper.js/dist/umd/popper.min.js"></script>

    <script src="node\_modules/bootstrap/dist/js/bootstrap.min.js"></script>





* Now, do a Git commit with the message "Intro. to Bootstrap". You may push the commit to your online repository.

Bootstrap Grid System and Responsive Design

Bootstrap is designed to be mobile first, meaning that the classes are designed such that we can begin by targeting mobile device screens first and then work upwards to larger screen sizes. The starting point for this is first through media queries. We have already added the support for media queries in the last lesson, where we added this line to the head:

1

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">





The *viewport* meta tag ensures that the screen width is set to the device width and the content is rendered with this width in mind. This brings us to the second issue, designing the websites to be responsive to the size of the viewport. This is where the Bootstrap grid system comes to our aid. Bootstrap makes available four sizes, xs for extra small, sm for small, md for medium and lg for large screen sizes. We have already seen the basics of responsive design. In this exercise, we will employ the Bootstrap grid classes to design the websites. We would like our website to have the content stacked on extra small devices, but become horizontal within each row for smaller devices and beyond. Towards this goal, we will make use of the classes .col-\*, .col-sm-\*, col-md-\*, and .col-lg-\* for defining the layouts for the various device sizes. We can specify how many columns each piece of content will occupy within a row, all adding up to 12 or a multiple thereof.

Using a Container class

* We use the container class to keep content within a fixed width on the screen, determined by the size of the screen. The alternative is to use the container-fluid class to make the content automatically to span the full width of the screen. We will discuss further about this when we discuss the Bootstrap grid system in the next lecture. Add the container class to the first div right after the </header> in the file as follows.

1

<div class="container"> ...





Dividing the content into rows

* Let us now add the class *row* to the first-level inner *div* elements inside the container. This organizes the page into rows of content. In the next exercise, we will see how we can add other classes to the rows.

1

    <div class="row"> ...





Creating a Jumbotron

* Let us add the class jumbotron to the header class as shown below. This turns the header element into a Bootstrap component named Jumbotron. A jumbotron is used to showcase key content on a website. In this case we are using it to highlight the name of the restaurant.

1

        <header class="jumbotron"> ...





* In the header add a **container** class to the first inner div and a row class to the second inner div.

Creating a footer

* Finally, in the footer add a **container** class to the first inner div and a row class to the second inner div.

Applying column classes within each row

* In the header row, we will display the restaurant name and the description to occupy 6 columns, while we will leave six columns for displaying the restaurant logo in the future. Let us go into the jumbotron and define the classes for the inner divs as follows:

1

2

3

                <div class="col-12 col-sm-6"> ... </div>

                <div class="col-12 col-sm"> ... </div>





* For the remaining three div rows that contain the content, let us define the classes for the inner divs as follows:

1

2

3

            <div class="col-12 col-sm-4 col-md-3"> ... </div>

            <div class="col col-sm col-md"> ... </div>





* For the footer, let us define the classes for the inner divs as follows:

1

2

3

4

5

6

7

            <div class="col-4 col-sm-2"> ... </div>

            <div class="col-7 col-sm-5"> ... </div>

            <div class="col-12 col-sm-4"> ... </div>

            <div class="col-auto"> ... </div>





Now you can see how the web page has been turned into a mobile-first responsive design layout.

Using Order and Offset with column layout classes

* In the content rows, we would like to have the title and description to alternate so that it gives an interesting look to the web page. For extra small screens, the default stacked layout works best. This can be accomplished by using the .order-sm-last and .order-sm-first for the first and the third rows as follows:

1

2

3

4

            <div class="col-12 col-sm-4 order-sm-last col-md-3"> ... </div>

            <div class="col col-sm order-sm-first col-md"> ... </div>





* For the div containing the <ul> with the site links, update the class as follows:

1

                <div class="col-4 offset-1 col-sm-2">





* After saving all the changes, you can do a Git commit with the message "Bootstrap Grid Part 1" and push your changes to the online repository.

List styles

* You can use several list styles to display lists in different formats. In this exercise, we will use the unordered list style *list-unstyled* to display the links at the bottom of the page without the bullets. To do this, go to the links in the footer and update the ul as follows

1

                    <ul class="list-unstyled"> ... </ul>





Using Custom CSS classes

We can define our own custom CSS classes in a separate CSS file, and also customize some of the built-in CSS classes. We will now attempt to do this in this part of the exercise.

* Create a folder named ***css***. Then create a file named *styles.css* in the ***css*** folder. Open this file to edit the contents. Add the following CSS code to the file:

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

.row-header{

    margin:0px auto;

    padding:0px;

}

.row-content {

    margin:0px auto;

    padding: 50px 0px 50px 0px;

    border-bottom: 1px ridge;

    min-height:400px;

}

.footer{

    background-color: #D1C4E9;

    margin:0px auto;

    padding: 20px 0px 20px 0px;

}





* Include the *styles.css* file into the head of the *index.html* file as follows:

1

    <link href="css/styles.css" rel="stylesheet">





* Then add these classes to the corresponding rows in the *index.html* file as follows. See the difference in the *index.html* file in the browser. The first one is for the row in the <header>, the next three for the rows in the content, and the last one directly to the <footer> tag.

1

2

3

4

5

6

7

8

9

10

    <div class="row row-header"> ... </div>

    <div class="row row-content"> ... </div>

    <div class="row row-content"> ... </div>

    <div class="row row-content"> ... </div>

    <footer class="footer"> ... </footer>





* Our next set of customization is to the jumbotron and the address. Add the following to *styles.css* file:

1

2

3

4

5

6

7

8

9

10

11

12

13

.jumbotron {

    padding:70px 30px 70px 30px;

    margin:0px auto;

    background: #9575CD ;

    color:floralwhite;

}

address{

    font-size:80%;

    margin:0px;

    color:#0f0f0f;

}





Vertically Centering the Content

* In the content section, update all the rows as follows:

1

        <div class="row row-content align-items-center">





* In the footer, update the third column div that contains the social media links as follows:

1

                <div class="col-12 col-sm-4 align-self-center">





Horizontally Centering the Content

* Update the copyright paragraph as follows:

1

2

           <div class="row justify-content-center">

                <div class="col-auto">





* Update the inner div containing the social media links as follows:

1

                    <div class="text-center">





Create a basic navigation bar

* We will now add a simple navigation bar to the web page so that it provides links to the other pages on the website. Start by adding the following code to the body just above the header jumbotron.

    <nav class="navbar navbar-dark navbar-expand-sm bg-primary fixed-top">

        <div class="container">

           <a class="navbar-brand" href="#">Ristorante Con Fusion</a>

                <ul class="navbar-nav mr-auto">

                    <li class="nav-item active"><a class="nav-link" href="#">Home</a></li>

                    <li class="nav-item"><a class="nav-link" href="./aboutus.html">About</a></li>

                    <li class="nav-item"><a class="nav-link" href="#">Menu</a></li>

                    <li class="nav-item"><a class="nav-link" href="#">Contact</a></li>

                </ul>

        </div>

    </nav>





In the above code, we can see the use of the nav element to specify the navigation information for the website. This nav element is styled by the *navbar* that declares it as a navigation bar, and the *navbar-dark* class to specify that the page should use the dark navigation bar. You will now notice the addition of a link with the name of the restaurant. This is the brand name for the website. You can replace this with the logo for the website. This is created by the *<a class="navbar-brand">* tag. In addition the inner *ul* is used to specify the items to be put in the navigation bar. This *ul* is styled with *navbar-nav* class to specify that the items should be displayed inline inside the navigation bar. We also use the container class inside the navigation bar.

Creating a responsive navigation bar

* We would like the navigation bar elements to collapse for shorter screens, to be replaced by a toggle button so that the items can be toggled on or off when required on small and extra small screens. This can be achieved by adding the following code to the navigation bar, just below the container div

            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#Navbar">

                <span class="navbar-toggler-icon"></span>

            </button>





This creates a button with three horizontal lines. For medium to extra large screens, this button is hidden. For small and extra small screens, this button becomes visible. This button will act as the toggle for the navbar items.

* To hide the items from the navigation bar for the small screens, we need to enclose the *ul* within another div as follows:

            <div class="collapse navbar-collapse" id="Navbar">

                <ul class="navbar-nav mr-auto"> ... </ul>

            </div>





By doing this, we are specifying that this div with *collapse* and *navbar-collapse* classes and with the id N*avbar* will be collapsed on small and xs screens, but can be toggled on or off when the toggle button is clicked. Note the use of *data-toggle="collapse" data-target="#Navbar"* within the button above. This specifies that the menu items are collapsed on small and xs screens when the toggle button is visible. They can be displayed or hidden by clicking the toggle button.

* Copy and paste the entire navbar code also into *aboutus.html* to add the navigation also to that page. Make sure to change the <li> corresponding to "About" to *active*, and remove the *active* class from the Home link. Also, update the home link to take you back to *index.html*. Update the navbar-brand tag also to take you back to *index.html*.

Modifications to the CSS styles

* We would like to have the navigation bar displayed in darker purple color, instead of the current color. In addition, when we use the fixed navigation bar, we should give the body of the page an upper margin of 50px, so that the top 50px of the page does not get hidden under the navigation bar. We accomplish these by adding these CSS customisations to the *styles.css* file

1

2

3

4

5

6

7

8

9

body{

    padding:50px 0px 0px 0px;

    z-index:0;

}

.navbar-dark {

     background-color: #512DA8;

}





* Remember to delete the *bg-primary* class from the <nav> element in both *index.html* and *aboutus.html.*
* We are already beginning to see the page format close to the final format for this module.

Adding Breadcrumbs

* To add breadcrumbs to our pages, we take the help of the breadcrumb and breadcrumb-item classes to add the following to the row containing the About Us title in *aboutus.html*.

            <ol class="col-12 breadcrumb">

                <li class="breadcrumb-item"><a href="./index.html">Home</a></li>

                <li class="breadcrumb-item active">About Us</li>

            </ol>



Using Icon Fonts and Other CSS classes

* One of the most popular icon font toolkit is Font Awesome. Go to its website <http://fontawesome.io/> to check out more details about this icon font. You can get Font Awesome using npm by typing the following at the prompt:

pm install font-awesome@4.7.0 --save





* Another module that we install is Bootstrap Social that enables the addition of Social buttons to our site. You can find more information about it at <https://lipis.github.io/bootstrap-social/>. To install it using npm, type the following at the prompt:

npm install bootstrap-social@5.1.1 --save





* We now need to include the CSS files for font awesome and bootstrap-social in the index.html file. Add the following code to the head of the file after the links for importing Bootstrap CSS classes. Do the same change to aboutus.html file:

    <link rel="stylesheet" href="node\_modules/font-awesome/css/font-awesome.min.css">

    <link rel="stylesheet" href="node\_modules/bootstrap-social/bootstrap-social.css">





* Let us now use some font icons in our web page and decorate it. Update the navbar's ul list items as follows in index.html:

                    <li class="nav-item active"><a class="nav-link" href="#"><span class="fa fa-home fa-lg"></span> Home</a></li>

                    <li class="nav-item"><a class="nav-link" href="./aboutus.html"><span class="fa fa-info fa-lg"></span> About</a></li>

                    <li class="nav-item"><a class="nav-link" href="#"><span class="fa fa-list fa-lg"></span> Menu</a></li>

                    <li class="nav-item"><a class="nav-link" href="#"><span class="fa fa-address-card fa-lg"></span> Contact</a></li>





* Similarly update the navbar's ul list items as follows in aboutus.html:

                    <li class="nav-item"><a class="nav-link" href="./index.html"><span class="fa fa-home fa-lg"></span> Home</a></li>

                    <li class="nav-item active"><a class="nav-link" href="#"><span class="fa fa-info fa-lg"></span> About</a></li>

                    <li class="nav-item"><a class="nav-link" href="#"><span class="fa fa-list fa-lg"></span> Menu</a></li>

                    <li class="nav-item"><a class="nav-link" href="#"><span class="fa fa-address-card fa-lg"></span> Contact</a></li>





* Next, in both index.html and aboutus.html, go down to the address in the footer of the page and replace the "Tel.", "Fax" and "Email" with the corresponding font awesome based icons as follows:

                      <i class="fa fa-phone fa-lg"></i>: +852 1234 5678<br>

                      <i class="fa fa-fax fa-lg"></i>: +852 8765 4321<br>

                      <i class="fa fa-envelope fa-lg"></i>:

                      <a href="mailto:confusion@food.net">confusion@food.net</a>





* Finally, let us use the bootstrap-social CSS classes to create the social buttons in the footer in both index.html and aboutus.html, by replacing the social sites' links with the following code:

                    <div class="text-center">

                        <a class="btn btn-social-icon btn-google" href="http://google.com/+"><i class="fa fa-google-plus"></i></a>

                        <a class="btn btn-social-icon btn-facebook" href="http://www.facebook.com/profile.php?id="><i class="fa fa-facebook"></i></a>

                        <a class="btn btn-social-icon btn-linkedin" href="http://www.linkedin.com/in/"><i class="fa fa-linkedin"></i></a>

                        <a class="btn btn-social-icon btn-twitter" href="http://twitter.com/"><i class="fa fa-twitter"></i></a>

                        <a class="btn btn-social-icon btn-google" href="http://youtube.com/"><i class="fa fa-youtube"></i></a>

                        <a class="btn btn-social-icon" href="mailto:"><i class="fa fa-envelope-o"></i></a>

                    </div>





### Adding a Button Bar

* We are now going to add content to *contactus.html* file to learn more about buttons and button bars. Go to the div where we specify "Button group goes here", and replace it with the following code to create a button bar containing three buttons:

                <div class="btn-group" role="group">

                    <a role="button" class="btn btn-primary" href="tel:+85212345678"><i class="fa fa-phone"></i> Call</a>

                    <a role="button" class="btn btn-info"><i class="fa fa-skype"></i> Skype</a>

                    <a role="button" class="btn btn-success" href="mailto:confusion@food.net"><i class="fa fa-envelope-o"></i> Email</a>

                </div>





Note how we define the button bar using the *btn-group* class, and then add the three buttons using the *<a>* tag. In this case, the three buttons are hyperlinks that cause an action and have an *href* associated with them. So we decided to use the *<a>* tag instead of the *<button>* tag. Note how the *<a>* tags have been styled using the *btn* class.

Adding a Basic Form

* We will add a simple form to the page at the location identified by "Form goes here". Add the following code to page to create a simple horizontal form with two fields:

                <form>

                    <div class="form-group row">

                        <label for="firstname" class="col-md-2 col-form-label">First Name</label>

                        <div class="col-md-10">

                            <input type="text" class="form-control" id="firstname" name="firstname" placeholder="First Name">

                        </div>

                    </div>

                    <div class="form-group row">

                        <label for="lastname" class="col-md-2 col-form-label">Last Name</label>

                        <div class="col-md-10">

                            <input type="text" class="form-control" id="lastname" name="lastname" placeholder="Last Name">

                        </div>

                    </div>

                </form>





This creates a form with two elements in the form. Note that the class *row* in the form enables us to use the Bootstrap grid system. Hence we can style the contents using the column classes as appropriate.

* Let us add fields to seek user's telephone number and email:                    <div class="form-group row">

                        <label for="telnum" class="col-12 col-md-2 col-form-label">Contact Tel.</label>

                        <div class="col-5 col-md-3">

                            <input type="tel" class="form-control" id="areacode" name="areacode" placeholder="Area code">

                        </div>

                        <div class="col-7 col-md-7">

                            <input type="tel" class="form-control" id="telnum" name="telnum" placeholder="Tel. number">

                        </div>

                    </div>

                    <div class="form-group row">

                        <label for="emailid" class="col-md-2 col-form-label">Email</label>

                        <div class="col-md-10">

                            <input type="email" class="form-control" id="emailid" name="emailid" placeholder="Email">

                        </div>

                    </div>





Adding a Checkbox and Select

* We now see the addition of a checkbox and a select element to the form. Note the styling of these elements using Bootstrap classes:

                    <div class="form-group row">

                        <div class="col-md-6 offset-md-2">

                            <div class="form-check">

                                <input type="checkbox" class="form-check-input" name="approve" id="approve" value="">

                                <label class="form-check-label" for="approve">

                                    <strong>May we contact you?</strong>

                                </label>

                            </div>

                        </div>

                        <div class="col-md-3 offset-md-1">

                            <select class="form-control">

                                <option>Tel.</option>

                                <option>Email</option>

                            </select>

                        </div>

                    </div>





Adding a textarea

* Next we add a textarea for the users to submit their feedback comments as follows:

                    <div class="form-group row">

                        <label for="feedback" class="col-md-2 col-form-label">Your Feedback</label>

                        <div class="col-md-10">

                            <textarea class="form-control" id="feedback" name="feedback" rows="12"></textarea>

                        </div>

                    </div>





Adding the Submit Button

* Finally, we add the submit button to the form as follows:

                    <div class="form-group row">

                        <div class="offset-md-2 col-md-10">

                            <button type="submit" class="btn btn-primary">Send Feedback</button>

                        </div>

                    </div>





Note the declaration of the type for the button to *submit*.

### Anchors

[Anchors](http://www.w3.org/TR/html5/text-level-semantics.html#the-a-element) (the <a> element) represent hyperlinks. What's a hyperlink? A [hyperlink](http://www.w3.org/TR/html5/links.html#hyperlink) is a resource a person can navigate to or download in a browser. If you want to allow your user to move to a new page or download a file, then use an anchor.

### Inputs

An [input](http://www.w3.org/TR/html5/forms.html#the-input-element) (<input>) represents a data field. The type attribute tells the browser which type of data the input controls. There are five input types related to buttons.

* <input type="submit">: This is the most common button input. It's a button that, when clicked, submits a form.
* <input type="image">: Like an <input type="submit">, this input submits a form. However, it also takes a src attribute and is displayed as an image.
* <input type="file">: This control is used to upload files and is shown as a label and a button. There's no good cross-browser way to style file inputs, so you'll usually set its display to hidden and use a second button to trigger it.
* <input type="reset">: This is a button that resets a form.
* <input type="button">: This is a button with no default behavior. You can use to it add non-standard behavior to a form using JavaScript.

### Buttons

The [<button>](http://www.w3.org/TR/html5/forms.html#the-button-element) element represents a button! Buttons do the same things as the inputs mentioned above. Buttons were introduced into HTML as an alternative to inputs that are much easier to style. Unlike inputs, a button's label is determined by its content. This means you can nest elements within a button, such as images, paragraphs, or headers. Buttons can also contain ::before and ::after pseudo-elements.

Like an input, a button has a type attribute. This attribute can be set to submit, reset or button and does the same thing as the input's type. By default, the type is submit. **If you place a button in a form and don't set its type, when it's clicked it will submit that form!** If you don't want this behavior, set the type to button.

One nifty feature of inputs and buttons is they support the disabled attribute. This makes it easy to turn them on and off. Sadly, anchors don't have this capability.

Bootstrap Tables

* In this part, we will add a new row of content after the Corparate Leadership row in the page. We first start by adding a row and columns to the page as follows:

        <div class="row row-content">

            <div class="col-12 col-sm-9">

                <h2>Facts &amp; Figures</h2>

            </div>

             <div class="col-12 col-sm-3">

            </div>

       </div>





* Inside the first column of this row, insert the table as follows:

                <div class="table-responsive">

                    <table class="table table-striped">

                        <thead class="thead-dark">

                            <tr>

                                <th>&nbsp;</th>

                                <th>2013</th>

                                <th>2014</th>

                                <th>2015</th>

                            </tr>

                        </thead>

                        <tbody>

                            <tr>

                                <th>Employees</th>

                                <td>15</td>

                                <td>30</td>

                                <td>40</td>

                            </tr>

                            <tr>

                                <th>Guests Served</th>

                                <td>15000</td>

                                <td>45000</td>

                                <td>100,000</td>

                            </tr>

                            <tr>

                                <th>Special Events</th>

                                <td>3</td>

                                <td>20</td>

                                <td>45</td>

                            </tr>

                            <tr>

                                <th>Annual Turnover</th>

                                <td>$251,325</td>

                                <td>$1,250,375</td>

                                <td>~$3,000,000</td>

                            </tr>

                        </tbody>

                    </table>

                </div>





Note the use of *table-responsive* class to create a responsive table, and the *table-striped* and *thead-inverse* classes for styling the table.

Bootstrap Cards

* Next we add a card to the second div in the first content row as follows, updating the div first by adding the classes col-12 col-sm-6 to it and then adding the card:

            <div class="col-12 col-sm-6">

                <div class="card">

                    <h3 class="card-header bg-primary text-white">Facts At a Glance</h3>

                    <div class="card-body">

                        <dl class="row">

                            <dt class="col-6">Started</dt>

                            <dd class="col-6">3 Feb. 2013</dd>

                            <dt class="col-6">Major Stake Holder</dt>

                            <dd class="col-6">HK Fine Foods Inc.</dd>

                            <dt class="col-6">Last Year's Turnover</dt>

                            <dd class="col-6">$1,250,375</dd>

                            <dt class="col-6">Employees</dt>

                            <dd class="col-6">40</dd>

                        </dl>

                    </div>

                </div>

            </div>





* Next, we add a Bootstrap card and include a quotation in the card using the blockquote typography style:            <div class="col-12">

                <div class="card card-body bg-light">

                    <blockquote class="blockquote">

                        <p class="mb-0">You better cut the pizza in four pieces because

                            I'm not hungry enough to eat six.</p>

                        <footer class="blockquote-footer">Yogi Berra,

                            <cite title="Source Title">The Wit and Wisdom of Yogi Berra,

                            P. Pepe, Diversion Books, 2014</cite>

                        </footer>

                    </blockquote>

                </div>

            </div>





Note the use of the *<blockquote>* tag to create a block quote in the card. We can use a *<footer>* inside the block quote to specify the attribution of the quote to its origin.

Adding the Restaurant Logo

* We will now add the restaurant logo to the Jumbotron. In index.html go to the header row inside the jumbotron and replace the second <div> column with the following code:

                <div class="col-12 col-sm align-self-center">

                    <img src="img/logo.png" class="img-fluid">

                </div>





You will immediately notice the restaurant logo being displayed in the jumbotron.

* Next, we will add the logo to the navbar where we display the restaurant brand. Go to the navbar and replace the code there for the <a> tag with the "navbar-brand" class with the following code:

            <a class="navbar-brand mr-auto" href="#"><img src="img/logo.png" height="30" width="41"></a>





Note the inclusion of the logo in the navbar.

* Repeat the above two steps for the *aboutus.html* and the *contactus.html* page also to update their navbars and jumbotrons.

Adding Media Objects

* Next we will work with the content on the web page and use the media object classes to style the content in the content rows.
* Go to the first content row, and replace the content in the second column containing the description of Uthappizza with the following code:                <div class="media">

                    <img class="d-flex mr-3 img-thumbnail align-self-center"

                            src="img/uthappizza.png" alt="Uthappizza">

                    <div class="media-body">

                        <h2 class="mt-0">Uthappizza</h2>

                        <p class="d-none d-sm-block">A unique combination of Indian Uthappam (pancake) and

                            Italian pizza, topped with Cerignola olives, ripe vine

                            cherry tomatoes, Vidalia onion, Guntur chillies and

                            Buffalo Paneer.</p>

                    </div>

                </div>





Note the use of the *media* class and the related Bootstrap classes to style the content.

* Next, we will go to the third row and replace the contents of the second column containing the description about Alberto with the following content:

                <div class="media">

                    <img class="d-flex mr-3 img-thumbnail align-self-center"

                            src="img/alberto.png" alt="Alberto Somayya">

                    <div class="media-body">

                        <h2 class="mt-0">Alberto Somayya</h2>

                        <h4>Executive Chef</h4>

                        <p class="d-none d-sm-block">Award winning three-star Michelin chef with wide

                            International experience having worked closely with

                            whos-who in the culinary world, he specializes in

                            creating mouthwatering Indo-Italian fusion experiences.

                            </p>

                    </div>

                </div>





* Finally, do a Git commit with a message "Images and Media".

Adding Badges

* We will continue to edit the *index.html* file. In this file, we will add a badge *HOT* next to the name of the dish Uthappizza in the first content row. To do this, add the following code inside the *<h2>* containing the name of the dish:

<span class="badge badge-danger">HOT</span>





* Next we will add a badge as a badge-pill right next to the earlier tag in the web page. Add the following code to the *<h2>* tag:

<span class="badge badge-pill badge-secondary">$4.99</span>