Full Stack Web Development: The Big Picture

Objectives and Outcomes

Full Stack Web Development: The Big Picture: Objectives and Outcomes

This lesson gives you a big picture view of the Full Stack Web Development and positions the current course in the overall structure of the specialization. This provides you the context for doing this course and sets the path forward in this course. The first video gives you an overview of full stack web development. The second video gives you an overview of this course and the various topics to be covered in this course. The third video explains how to make the best use of the learning resources available in the course. At the end of this lesson, you will be able to:

Understand what is meant by full stack in the context of web development
Distinguish between front-end, back-end and full stack web development
Understand the position of this course in the context of this specialisation

1. What is Full Stack Web Development

What is Full Stack Web Development? Let me start by clarifying a few terms, so that we start with a common understanding about this area. First, we hear people talking about Front end and Back end, or sometimes the Client-side and the Server-side. Now when we talk about Front end we are primarily talking about presenting the data to the users. So this is where the users access the web through the browser. And so we would be using technologies like HTML, CSS and JavaScript to target the end users.
0:39
The Back end is more concerned with the business logic and the data aspect of our
0:45
web application, so this might be implemented in one of several languages.
0:53
We often hear people talking about the Three Tier Architecture for web development and implementation. Now in this approach, which is a common client-server software architecture pattern, the entire development is divided into three layers. The Presentation layer, which is concerned with the UI related issues and how to present the data to the end-user. Then we have the Business Logic layer, which is more concerned with
1:24
data validation and processing the data in order to generate information for the users. And then we have the Data Access layer, which deals with data persistence and storage of data behind the scenes, for example, in a database.
1:51
Exploring this further, let's look at what is implemented in each of these three layers in more detail. Now, the Business Logic layer is typically implemented, traditionally, in a server, where we use languages like Ruby, Python, C++, Java, on ASP.net, to implement a server. This server in turn, is talking to a Back end database management system to exchange data and in turn, will be generating
2:30
the Front end display of data to the user
2:36
by Server-side rendering of the HTML, CSS and JavaScript. This has been the traditional approach that that has been used successfully in the development. Now in this approach, we need specialists in each

of these three layers. So, you have a Front end specialist who would be well versed in HTML, CSS, and JavaScript. And then you have the Back end specialists, one who would be well-versed in Server-side implementations, say, for example, one of the languages used for Server-side implementation. And then you would have a data specialist who would be mostly concerned with the databases and data persistence aspect of our web development.
3:27
There is an increasing trend towards Full Stack Web Development where a single language is used for implementation over the entire stack. So you could be having the front implemented as a single page application, using one of the JavaScript frameworks like AngularJS. Then the Server-side could be implemented using technologies like NodeJS and NodeJS modules, which again, depends on JavaScript. And then, the Data Access or data storage part implemented using Technologies like MongoDB, which works based on storing JSON data, and serving up information in the form of JSON data. Now, JSON has become the standard format for data interchange among the three layers. Server side increasingly is delivering a REST API, so that you can target multiple platforms through the REST API. So, essentially, the server is serving up data in the form of JSON, which can then be rendered either on a standard web browser, or on a mobile device using one of the multiplied form, hybrid mobile application development environments.
4:56
So, in this particular specialization we are targeting the entire stack. We'll look at HTML, CSS, and JavaScript and then we will look at web UI development using Bootstrap. Then we'll look at Front end JavaScript frameworks like AngularJS. Then we'll look at NodeJS and NodeJS Modules, and also review Back end as a service, and also review data support through MongoDB on the Server side.

2. Course Overview

As you begin this course you must be wondering, what exactly will be covered in this course? I am now going to explain to you exactly what we are going to deal with in this course, and how it is positioned in the bigger scheme of things in this full stack web development specialization.
0:27
Going back to our picture of the full stack web development specialization, we are now still dealing with the front end development. And in particular, in this course, we're gonna concentrate on the UI framework, which is bootstrap.
0:44
So in this scheme, this is the module dealing with the bootstrap on the front-end side, the presentation layer side.
0:56
In this course, we are going to cover bootstrap in a lot of detail. We'll also introduce you to web tools, in particular Node.js and NPM, and Bower. The remaining web tools we will see in the next course.
1:18
This course itself, is organized into four modules, each module corresponding to one week of work.
1:28
In the first module we'll give you the big picture view of Full Stack Web Development, then we'll introduce you to Bootstrap. Then we'll talk about the responsive web design, and the Bootstrap grid system. We'll also look at an important component in Bootstrap, which is the navigation bar.
1:49
And at the end of this module, you will have your first assignment.
1:54
As you go through this course, you will be building up a website to help you to consolidate all the knowledge that you are acquiring in this course.

2:10
In the second module we will be dealing with Bootstrap's CSS components. These components are pure CSS components, which means that they don't need JavaScript for them to work. In particular, we'll look at user input components, like buttons and forms. Then we'll look at displaying content, using tables, panels and wells. Then we'll move on to image and media, dealing with images, thumbnails and media objects that are included in your webpage. Then we'll look at alerting users by using things like labels, badges, alerts and progress bars. At the end of the second module you'll have the second assignment 2. The third module deals exclusively with Bootstrap's JavaScript supported components. So we'll look at the overview of Bootstrap's JavaScript components, and we'll explore tabs, pills, and tabbed navigation. Then we'll look at collapse, accordion, ScrollSpy, and Affix. Then, we look at Tooltips, Popovers and Modals. This pretty much covers all the JavaScript components. We'll also look at Carousel as an important component there. At the end of this module, we'll have assignment 3 due.
3:41
The final module rounds out our discussion on bootstrap JavaScript competence, by dealing with Bootstrap and jQuery, and how they work together. And we'll also see how Bootstrap's JavaScript competence can be controlled through the JavaScript methods that are available for this competence. Then we'll look at Node.js and Node Package Manager. We'll also look at CSS preprocessors, Less and Sass.
4:14
Then we'll encounter our first web tool, which is Bower. At the end of this module, you'll have assignment 4.
4:25
This completes this course. I hope you have a good time going through this course and take the maximum benefit out of learning the material in this course. This course will set up the foundation upon which we'll build in the following courses.

How to Use the Learning Resources

As you go through the course you must be wondering how to make the best use of the learning resources that are available to you. Let me take you through a quick journey of the learning resources that are available to you within this course, and how you can maximize the benefit from making use of these learning resources.
0:25
The course content itself is organized into multiple modules. A module has an overarching theme that guides the content within that module.
0:37
A module itself is organized into several lectures.
0:42
At the end of each module, you will have an assignment that is due. Now, when you look at the organization of the content, typically a module corresponds to one calendar week of work.
0:57
When you look at a typical lecture within a module, a lecture will be organized into several parts. At the beginning of the lecture, you will have objectives and outcomes that are clearly specified. Thereafter, there will be an introduction to some concepts and technologies that are related to that particular lecture. Then it is followed by a review quiz.
1:22
After that you will have a hands on exercise where you will practice what you've just learned.
1:28
And then finally, we will provide you with a set of additional resources that you can follow through in

order to further enhance your understanding of the subject.
1:41
To help us understand this more concretely, let's pay a visit to your course website and then have a look at the contents that are available there.
1:52
So as you can see the course website, the course content is organized under this section here. You will see that the course content is organized into weeks. Each week corresponds to a module of work.
2:10
Now, when you go into a particular week, when you select a week, you would notice that the week itself is organized into a set of lectures with a common theme for the lectures. The whole module itself has a common overarching theme that guides the design of the module. As an example, let's take a look at this lecture on responsive design and Bootstrap grid system. As you enter the lecture, it will start out with the objectives and outcomes for this lecture. I would strongly suggest that you read through this document in order to understand what you're going to encounter in this lecture and what are the outcomes at the end of this lecture. So at the completion of this lecture set you should be able to do all the outcomes that are listed in this document.
3:07
Following this, you will see a set of lectures that are available to you. So you can watch all of these lectures and then after that, you will move on to
3:21
a hands-on exercise that will be delivered to you. So, the hands-on exercise is delivered in two parts. One, So we have a video of the exercise itself, guiding you through every step of the exercise. You also have a written document that contains the instructions for the exercise. So, you should be watching the video in order to understand how the exercise progresses. I will be explaining in more detail in the video. But also together with that, you have this written document where I have given the step by step exercise instructions.
4:11
So you can follow through and even the source code that I am using in the exercises are available to you. So in case you decide to cut and paste the source code, you can always obtain the source code right here within this written document.
4:29
At the end of the hands on exercise you will have a quiz that will help you to review the concepts that you have just seen in this lecture.
4:41
And at the end of each lecture we also have a set of additional resources. These are pointers to further reading that you can pursue in order to enhance what you have just learned. It's completely voluntary.
4:58
You can find some these to be very informative for you to read in your own spare time.
5:06
So to summarize the learning resources that are available to you within this course include video lectures, then we have practice quizzes, then hands on exercise, both in the video format as well as written instructions. And I would again remind you that you should be watching the video of the exercise in order to understand how the exercise progresses step by step, and then pointers to additional resources.
5:36
The reason the learning resources are organized the way they are is that the best way to learn is not just focus on the concepts but also have hands-on practice so that it reinforces your understanding of the concepts. So that is why we have the hands-on exercises following the introduction to the concepts and the technologies.

6:01
Now this is where I would remind you of the traditional Chinese proverb which says that what I hear, I forget, what I see, I remember, what I do, I understand. So let me again emphasize the importance of going through the hands-on exercises. That is the only way you will completely understand the concepts and technologies that we cover in this course.
6:29
Another related point that I wanted to remind is from another proverb which says, he who learns does not think is lost, he who thinks but does not learn is in great danger.
6:44
Or rather, learning without thinking begets ignorance. Thinking without learning is very dangerous. Now, this is where let me emphasize that as you go through the concepts and the technologies, and as you do the hands on exercise, ask yourself the question, why are you doing these steps. What exactly is it intended to convey to you and what is the reason for doing the steps the way they are being done? That is the only way you will completely understand what you're going through. It is very easy to go through this sort of exercises and concepts without thinking and at the end you will end up learning nothing.
7:32
Similarly, if you just skip all the learning part and assumed that you've learned everything then that is equally as dangerous. So please, spare some time to think as you learn and to understand.
7:49
We always have the discussion forums where any questions related to the materials can be clarified in more detail.
8:00
I hope you have a very good learning experience as you go through this course.

Introduction to Bootstrap

Objectives and Outcomes

In this lesson, you will be given a quick overview of front-end UI frameworks, and an introduction to Bootstrap. The exercises will introduce you to getting started with Bootstrap for your web project. At the end of this lesson, you will be able to:

Identify the purpose of using front-end UI frameworks in web design and development
Set up a project with Bootstrap support
Configure a web project to use Bootstrap
Become familiar with the basic features of Bootstrap

3. Front-end Web UI Frameworks: An Introduction

Front-end web UI frameworks are increasingly becoming the approach for designing and implementing websites quickly. Let's examine some details of front-end web UI frameworks in this lecture.
0:22
We last covered service. What are front-end web UI frameworks? Then we'll look at why we would be interested in using a front-end framework and finally a list of the popular front-end web UI frameworks.
0:40
What exactly do front-end web UI frameworks provide for you? Now typically a front-end web UI

framework is a collection Of HTML, CSS, and Javascript templates that enable you to quickly design and implement your website. So they will provide a collection of components like Buttons, Tables, Navigation bars, Modals, Accordions, Carousels and Dropdowns and many other features. And also addressing Typography and color schemes for your website.
1:23
Why should we be interested using front-end web UI frameworks? One of the primary reasons is responsive web design. Now, we will examine responsive web design in more detail in the next lecture. Here, let me quickly summarize what we mean by this.
1:44
We realize that people are accessing websites using various different devices of different screen sizes. Now how do we target these different screen sizes by using a single website design and make our website automatically adapt to different screen sizes. So that is one of the primary issues being addressed by responsive web design. In particular, we see an increasing trend towards mobile first. Meaning that first, design your website with the mobile screen size in mind, and then adapt to larger screen sizes from there. We'll also look at mobile first design in more detail in the next lecture. A second reason why you might be interested in front-end frameworks, is that different browsers have different quirks in them. So we need a way of commonly targeting all the different browsers so that our website renders uniformly independent of which browser a user uses to access our website. Now addressing the quirks of different browsers is not that straightforward. Fortunately for us, these frameworks have already taken care of addressing these issues so that it becomes straightforward for us to concentrate on our website design. And if we make use of the front-end frameworks, they will automatically take care of the problems of addressing the different browser issues.
3:22
A third and equally important reason for using a front-end framework is increased productivity. Now designing websites, from the scratch, is quite cumbersome. Now, because these front-end frameworks provide us with a lot of features and competence, it becomes more straightforward to design our website with a consistent look and feel.
3:47
Finally, let's not underestimate the community support that a front-end framework enjoys. Now, the larger the community that participates in a front-end framework, the larger the resources that we have available to us that we can leverage to address issues that we might encounter in using a front-end framework. You might find a lot of examples out there, you might find people posting blog entries addressing various features. That is very very important when you need to quickly get started implementing a website.
4:29
Finally let's look at some of the popular front-end frameworks. Now this is a small list. There are a large number of front-end frameworks out there.
4:44
I'm giving you the list of the most popular among them. Now, again, I might have missed out your favorite front-end web framework. Don't get too worked up about it. People have different opinions on this. Now this is one of the most opinionated areas of website design. People tend to hang on to their favorite web UI frameworks and insist that that is the best way to address the problem. Again, we should all take this with a big grain of salt. Maybe I should say a big rock of salt after all, but anyway. Based upon looking at several different plug entries, this is the list that I feel comfortable with in presenting to you. One of the most popular of all the front-end frameworks is of course Bootstrap. We'll look at Bootstrap in more detail In the subsequent lectures.
5:45
Foundation by Zurb is yet another popular framework, has a pretty large following. Then we have Semantic UI which is gaining a lot of ground. Pure by Yahoo, UI Kit, and many more. Now, when you

look at these frameworks, you will notice that they have a lot in common, and each one of these frameworks have some unique features. Now, your choice of a framework is dictated more by what you're trying to achieve than trying to stick to one single framework for all your needs. Again, look at it as possibilities that you have for you. Getting started with any one of these frameworks is not that difficult once you have a quick introduction to one of those frameworks. Many of them have common features as a set. So it should not be very difficult to switch from one framework to another if you'll find that a second one is better suited for your purpose.
6:50
So now we will go onto the next set of lectures where we'll examine Bootstrap in more detail.

Introduction to Bootstrap

Let me now quickly introduce you to Bootstrap, the front-end UI framework that we're gonna deal with in this course.
0:15
Bootstrap, as described on its own website, says that it is the most popular HTML, CSS, and JavaScript based framework for developing responsive, mobile first projects on the web. This is definitely true that Bootstrap is one of the most widely used frameworks for web design.
0:42
Now, when you look at the bootstrap framework itself, it is a framework designed for enabling people to design websites faster and easier. It provides a lot of interesting HTML, CSS, and JavaScript based templates. Which you will encounter as we go through this course. And enables us to design websites very quickly using these components. And Bootstrap has built-in support for responsive design using the mobile first approach. We will see in the next few lessons, how the responsive mobile-first approach of Bootstrap can be leveraged to design responsive websites. Bootstrap was first released in 2011 by Mark Otto and Jacob Thornton.
1:44
They were Twitter employees that designed Bootstrap for their own use within the company. And then, later, decided to make it open source.
1:54
Currently Bootstrap is in version 3.3.5. 4.0 is already on the horizon, will be released soon. Bootstrap has maintained backward compatibility as far as possible, but Bootstrap 3 was a big step forward in supporting responsive web design in a much better manner than earlier versions of Bootstrap. So, definitely it is worth considering and worth learning because it was one of the first comprehensive frameworks that was available for use. So we'll move on to the next lesson, where we will first get familiar with Bootstrap and how to make use of Bootstrap within our project.

4. Getting Started with Bootstrap

Let us now learn how to get started using Bootstrap in a web project. In a exercise that follows this lecture you will get some hands on experience of using Bootstrap in a real web project.
0:19
To get started using Bootstrap we have three possible approaches. First, you can go to the Bootstrap website and then download the Bootstrap files.
0:30
Second, you can get hold of the Bootstrap sources In either Less or Sass preprocessor format. We will examine these in more detail later on in this course. The third alternative for you is to use the precompiled Bootstrap available through Bootstrap CDN. Here, you don't need to download any files, but instead, simply link to the Bootstrap files that are available through a content distribution network

site.
1:04
We will follow the first approach. In the exercise that follows this lecture. We will go to the Bootstrap site and then download the Bootstrap files and then unzip the downloaded zip file containing the Bootstrap files. When you unzip the zip file, you will find that
1:24
it creates a folder and subfolders inside the folder. The structure of the Bootstrap folder is shown on the right side, here. You have subfolders, CSS that contains some Bootstrap CSS files. Then you have a subfolder named JS that contains the JavaScript, part of the bootstrap sources. And then you have a subfolder contain, called fonts, which contains fonts that you use in bootstrap. We need to include these files into our Bootstrap project.
2:00
How do we get started using Bootstrap within our project?
2:05
In your HTML5, in the head of your HTML5, you are required to include these three lines.
2:15
Later on, we will examine what these three lines do for your project.
2:22
Then, you need to input the bootstrap CSS files into your web page. To do that, we will include these two lines into the head of an HTML page. Now, we are importing the Bootstrap file itself and then, the Bootstrap theme CSS file into our project using these two lines. You notice that I have specified the files as bootstrap.min.css and bootstrap-theme.min.css. The minfied format is what we use when we design our project. Now if you are editing a Bootstrap or a CSS file, you will write the CSS file in normal format. Then you can strip off all the unnecessary blank spaces and
3:16
the comments from your CSS file and then create a minified file. So, the verified file is much smaller in size and so easier for people to download when they view your web page.
3:33
In addition, Bootstrap also uses Javascript.
3:37
Bootstrap makes use of the jQuery Javascript library for providing much of its Javascript functionality. So Bootstrap has JavaScript plugins
3:52
designed using the jQuery library as the basis. So that is the reason you need to include the jQuery library. In addition, you will include Bootstraps own JavaScript classes, this has to be done just before the end of the body of your web page. The primary reason why we include the JavaScript files towards the end of the body of your web page, is so that when you view the web page, the web page can be rendered even before the JavaScript is actually processed. So, when your web page is being rendered by your browser, it'll first display all the content and then use JavaScript to then restyle the content. Now, this makes the web page loading a lot more faster.
4:48
As I mentioned earlier, you can also include Bootstrap directly from its Bootstrap CDN. This is the set of lines that you will include within your HTML page in order to use Bootstrap directly from the content distribution network. So the first two lines you will include in the head to import the CSS files. The last two lines will be included at the bottom of the body of your HTML page to import the JavaScript libraries.
5:27
Now, when you design a web page, we will often use a class called as a container to hold all the contents of our web page. The container defines the fixed width region within your web page. Inside

which the contents will be displayed for your web page. So this is one way of restricting your contents to display in certain part of the web page.
5:56
The container size will vary depending upon the size of the screen from which you are viewing your website. We will deal with the container more in the next set of lessons.
6:14
Normally, we would use the container as the outermost div element to wrap the contents of our website.
6:22
This is required in order to make the Bootstrap grid work correctly. What exactly is the Bootstrap grid? That we will deal with in the next set of lessons where we will deal with responsive web design.
6:36
You can also use another class card as container fluid to use a full width container. The container fluid class enables a full width container window, which spans the entire size of your screen.
6:55
To understand the container class better, let me show you a web page. You're going to see this web page as part of your exercises later. Now, within this web page, all the contents in here are enclosed inside a container so that they will be displayed inside a fixed region of your screen.
7:23
Bootstrap also provides another class called the row. We use the row class to divide the page contents into multiple rows. Each row acts as a horizontal grouping for information that has to be displayed in a single group. A row may contain multiple columns. We will deal with that in the next lesson.
7:49
In Bootstrap, the rows must always be contained inside containers for the Bootstrap grid to work correctly.
7:58
Another element that we're going to examine in the exercise is a Jumbotron. A Jumbotron is a lightweight
8:08
Bootstrap component that is used to showcase key content on your web page. So for example, this can be used to show the name of your company, some brief description about your company, and typically at the top of your page.
8:27
When you use a Jumbotron within your web page, you can use the Jumbotron outside a container and then include a container inside a Jumbotron. So that the contents of the Jumbotron can be within a fixed region of the Jumbotron but this enables your Jumbotron to span the entire screen width. Returning to our web page example, the top blue region is using a Jumbotron and inside that we are displaying the contents of our website. The name and some brief description of the website.
9:08
Now, we will move on to our first hands on exercise where you will use the classes the you just encountered. You will start by including Boostrap into your HTML page and then we will use the container and the rule classes to structure our web page and we also see the use of Jumbotron within our web page.

Setting up your Development Environment

Software Requirements

Text editor of your choice: Any text editor that you are already familiar with can be used for editing the project files. I will be using Brackets (http://brackets.io/) as the editor. You may also consider using

Sublime Text (http://www.sublimetext.com/).
Browser of your choice: You may use your preferred browser. I will be using Chrome as the browser in all the exercises. All the exercises and assignments in this course have been tested using Chrome v. 46. Please note that not all browsers may support all the HTML5 features to the same extent. You might encounter problems when using other browsers. I strongly urge you to use the latest Chrome browser for the exercises and assignments in this course so that any problems are minimized.
Command line shell: Familiarity with the command line shell will be essential for the exercises. In Windows a cmd window or power shell with admin privileges would be needed. On a Mac or in Linux, a terminal window can be used. Please get familiar with the "sudo" command in OS X and Linux.
Files required for the exercises: We will provide additional starter files for the exercises. Links to download the files will be provided inline in the exercise instructions (reading). The links are also available through the Resources (reading) of the specific lesson.
Note: Please remember to retain the folders and all the files that you create in the exercises. Further exercises will build upon the files that you create in the preceding exercises. DO NOT DELETE the files at the end of the exercises, unless otherwise instructed.

Exercise (Instructions): Getting Started with Bootstrap

Objectives and Outcomes

This exercise introduces you to the basic features and some classes of Bootstrap. At the end of this exercise, you will be able to:

Understand how to set up a web project to use Bootstrap
Include the Bootstrap CSS and JS classes into a web page
Design a web page structure using a few basic Bootstrap classes
Note: Please remember to retain the folder and all the files that you create in this exercise. Further exercises will build upon the files that you create in this exercise. DO NOT DELETE the files at the end of the exercise.

Setting up the Project Folder

Go to a convenient folder on your computer and create a folder named conFusion. This will be the name of the project that we will implement in the set of exercises to follow.
Downloading Bootstrap

Go to the Bootstrap website http://getbootstrap.com and click on the download button to download the zip file containing Bootstrap files.
Move the bootstrap-3.3.5-dist.zip file to the conFusion folder you created above and unzip it. You should now see a folder named bootstrap-3.3.5-dist. Go to this folder and move the three folders there (css, fonts and js) to conFusion folder above. You can now delete the zip file and bootstrap-3.3.5-dist folder. Now you are all set to use Bootstrap to design your web project.
Download index.html file

Download index.html file to 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.
Getting your Web page Bootstrap ready

Open the index.html file in your favourite text editor. If you are using Brackets, you can open the conFusion folder from within Brackets and then open the index.html file as a working file. You can do the same with Sublime Text and some other text editors too.
Open the index.html file in your favourite browser so that you can preview the formatted web page in the browser. If you are using Brackets, you can use the Live Preview button to load and automatically refresh the page in Chrome as you edit the files. A similar set up can be done with Sublime Text after installing some plug-ins.
Insert the following code in the head of index.html file before the title.

```
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<!-- The above 3 meta tags *must* come first in the head; any other head
    content must come *after* these tags -->
```

Add the following code in the head after the title. This will include Bootstrap CSS into your web page.

```
<!-- Bootstrap -->
<link href="css/bootstrap.min.css" rel="stylesheet">
<link href="css/bootstrap-theme.min.css" rel="stylesheet">
<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media
  queries -->
<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->
<!--[if lt IE 9]>
  <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"
    ></script>
  <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script
    >
<![endif]-->
```

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 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.

```
<!-- jQuery (necessary for Bootstrap's JavaScript plugins) -->
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min
  .js"></script>
<!-- Include all compiled plugins (below), or include individual files as
  needed -->
<script src="js/bootstrap.min.js"></script>
```

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 top most div in the file as follows.

 ...
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.

 ...
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.

<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.
Conclusion

We have now understood how to set up a web project to use Bootstrap. In the next lecture, we will explore further on responsive design and Bootstrap's grid system.

Exercise (Video): Getting Started with Bootstrap

Strap on your boots and let's get started with our first hands on exercise. Sorry, just couldn't resist that. Okay, let's get started with the first exercise. Where we are going to learn how you would set up a web project to use Bootstrap, how you would include Bootstrap CSS and JavaScript classes into a webpage. And how to apply the Bootstrap classes to various elements in the webpage, to design your webpage.
0:35
First, go to your convenient location on your computer. And create a folder named conFusion. Yes, that's right conFusion. That's the name of the website that we're gonna create.
0:49
So let's do that first.
1:00
Once you create that folder, let's move into that folder and then we'll download Bootstrap files.
1:09
Next, go to the Bootstrap website, www.getbootstrap.com, and then download the zip file containing the Bootstrap CSS JavaScript and font classes.
1:22
Then, move the zip file to the folder that we just created, and then unzip the file.
1:30
This will create a folder named bootstrap-3.3.5-dtxt. Move into that folder and move the three folders inside there into the conFusion folder. So let's select them and then move them to the conFusion folder.
1:55
Then move back to the conFusion folder and then now you can delete the Bootstrap distribution files.
2:04
Now our folder is setup with the Bootstrap files, to get started on our web project.

2:13
Next, download the index.html file that we have provided for you and then move it to the conFusion folder.
2:22
Then, you can open the index.html file in your favorite browser, and then have a look at the contents in there. I have already configured this html file with some content, which we're going to style using the Bootstrap classes. This project creates a website for a fictitious restaurant named con Fusion.
2:46
When you look at the current state of the webpage, it is plain and simple, with just the content. Looks like a typical computer science professor's webpage.
2:59
At the end of this module, we're gonna take this webpage and then use Bootstrap classes to style it, and turn this webpage into this webpage.
3:17
You can see that the content has been appropriately styled using various Bootstrap classes, to make it a lot more attractive. And fun to look at.
3:32
So let's get started styling this webpage using Bootstrap.
3:38
I'm going to be using Brackets as the editor of choice for this set of courses. You are welcome to use any text id term that you are familiar with for editing the files.
3:53
On the screen, you already see that I have opened the conFusion folder in Brackets, and opened the index.html file to be displayed in the window there. Now, one reason why I chose to use Brackets, is the availability of the live preview facility in Brackets. If you go to the right side of the screen, you see this button, which looks like a thunderbolt. When you click on that, Brackets will automatically open
4:26
that webpage in a browser. And whenever you make any changes to that code, in the webpage in Brackets. It's automatically going to update the webpage in the browser. This is an interesting feature that makes it easy for us to see the changes immediately in a browser. Now because of the restriction of the screen, I am not able to show you both the browser and of the Brackets window at the same time, simply because it won't be very clear in the video. So I have my webpage open in my browser in a separate window, which I'm gonna switch to occasionally to show you the changes whenever we edit the file in Brackets.
5:20
Let's go into the index.html file, and then paste this code in the head of the html file.
5:32
Now, this code sets up our webpage to be responsive. Now, I'm going to explain the details of this code in the next set of lessons. For the moment, I want to put that code into place. So that our webpage begins being responsive from the start.
5:55
Next after the title in the head of the webpage, I'm going to add in the lines to import the Bootstrap CSS files.
6:12
Finally at the bottom of the page, just before the end of the body tag, I am going to add in the code to import the jQuery library and Bootstraps JavaScript plugins.
6:34
After we make these changes, let's go and have a look at our current state of the webpage. Now when you switch to our webpage, you will already see that the page has changed subtly. Their fonts that are

being used in the page to render content, has now been updated to use the Bootstraps default fonts, which is Helvetica Neue.
7:05
We are now going to add a container class to enclose the content of our webpage. So switching back to the HTML of the page,
7:17
we will now go into the content and to the first div after the header, so you can see that in the body of our webpage, we first have the header. I'm going to style the header a little bit later. But, we'll move to the first div inside the body and then apply the class container to this.
7:45
After this change, let's go back and have a look at our webpage.
7:51
Switching back to our webpage, you already see that
7:56
the content of a webpage is enclosed inside a container here. We'll still have the header and the footer as tied, which we're going to do next.
8:12
Once we upload the container class to the content of the webpage.
8:17
Inside the content, we are going to divide the content into multiple rows. So to do that, for the first inner div's that we're going to encounter inside the container, we will apply the row class to each one of them. So let's go ahead and do that, and then come back and have a look at the webpage again.
8:44
We have a total of three rows in the content of our webpage.
8:50
Switching back to our webpage, let's have a quick look at the change.
8:59
You won't notice much change in the webpage. The content has been enclosed inside the rows, but we haven't applied any other styling to the webpage.
9:10
We will now move to the header, which is the first section of the body of our webpage. So to the header, I am going to apply the class of Jumbotron.
9:26
And the first inner div inside that header, I'm going to apply the class container. And the second inner div and better apply the class row here. After these changes, let's go back and have a look at our webpage.
9:47
Switching back to our webpage. We now see that our header has been styled using a Jumbotron. So, you can see the light grey background applied to the Jumbotron. And you can also see how the content has been rendered in the head of our webpage. You can already see the webpage taking on a good shape.
10:07
Finally, let's move to the footer of the page, and then apply the container class to the first inner div and the row class to the second inner div in the footer.
10:22
After these changes, let's have a look at our webpage. Now you can see that our footer has also been brought into the center part of our webpage.
10:37
Having one complete look at the webpage from the top to the bottom, you can see the Jumbotron at the

top, then you see that the content has been enclosed inside a container. And the footer content has also been enclosed inside another container.
10:52
This completes this exercise. So we have now understood, how to set up our web project to use Bootstrap classes, and we have used some basic classes from Bootstrap to style our webpage.

Responsive Design and Bootstrap Grid System

Objectives and Outcomes

In this lesson, you will be given an overview of responsive web design and an introduction to the Bootstrap grid system. The exercises will concentrate on enhancing your web project using the Bootstrap grid in order to make it responsive. At the end of this lesson, you will be able to:

Understand the reasons for using responsive web design in a web project
Use the Bootstrap grid system to design responsive websites
Add your own custom CSS classes to a Bootstrap based web project

5. Responsive Design

In this lesson, we are gonna deal with responsive design. I have been using this word quite a bit in the previous lesson, but without explaining it in a lot of detail. Now, let's consider responsive design in detail and understand exactly what we mean by responsive design and how would you design a website to be responsive.
0:27
An immediate question that arises in our mind is, why should we consider responsive design? Why should we make our website responsive? When we look at how websites are being accessed these days, you'll notice that people access websites from a variety of devices. All the way from desktops, laptops, tablets, down to mobile phones. So, your website may be viewed by users using screens of different sizes. If you design your website with a single screen size in mind, then it may render improperly or completely unreadable on small screen devices. So if we can't proactively design our website to dramatically adapt to the size of the screen from which the viewer is visiting your website, then that will make it a lot more easy for the visitors to be able to see your website clearly. Now, if they can build this feature into our website itself so that it will automatically adapt to it's layout to the size of the screen from which the user is viewing your website, that that will be a great improvement.
1:49
Responsive design aims to achieve this in practice. To do this, we're going to use the viewport property of our device, in order to adjust our website to the screen of the device.
2:12
The best way for us to understand responsive design is to take a look at an example.
2:18
Let's have a look at the course website itself. So here you see the course website opened in a browser window. Now this window is I can bring the entire width of my screen which is currently 1280 pixels. Suppose I view the same website from a device, which has a much smaller screen say about 700 pixels, how would these website be rendered in those circumstances?
2:46
To help us to understand this better, what I'm going to do is to just shrink the size of my browser so that my website will automatically adapt itself to the size of the browser. So, in some way, I am simulating

accessing the website from the basis of different screen sizes. So at this moment, this would be a 1,200 pixel screen size. Which would be equivalent to a typical laptop these days. Now, let's look at what the website looks like if you were accessing it from a tablet in the landscape position. So, perhaps, that looks more like the size of the screen from a tablet. This is typical tablet landscape sizes are around 700 to sometimes even much higher
3:42
pixel sizes these days, but maybe a seven inch older laptop perhaps has a width of about 800 pixels. So, this is what your website will look like on a tablet. Let's continue shrinking the browser window, and see how the website will look like at even more smaller sizes. So as you keep shrinking the window size, (改 browser window 大小時) notice how the contents of the website is automatically adapting itself to the size of the screen. Now, you notice that when I shrank it below a certain point, there was a sudden change in the way the website was being shown on the screen in the browser window. So let's see, we have here, and then a little step further down you suddenly notice that the website has rendered some more differently you see that the navigation bar at the top of the window has disappeared, instead to be replaced by a button on the left side. Now the navigation bar is still accessible. When you click on the button, you now see a sidebar pop up with the information that used to be in the sidebar of your website for larger screen size. Now this side bar has been hidden at this screen size. So, now I can hide this by simply clicking at any other location. Also note how the contents of the website have adapted themselves to the screen size. So, this would be like a tablet, perhaps, in its portrait position. Let's continue to shrink the website window down. And use you see how the contents of the website are automatically adapting itself.
5:39
And as we scroll through the website, you see that the contents have adapted themselves to the size of the browser window here, or equivalently to the size of the screen.
5:55
How is this possible?
5:58
This is where responsive design comes to our rescue.
6:05
Another term you will often hear people talking about in the context of responsive design is Mobile First. What exactly is Mobile First design? Traditionally, when people were designing their websites for both larger screen windowed devices like laptops and desktops and smaller screens, they would often do the layout for the larger screens first, and then try to adapt their content to be appropriately displayed on the smaller screen. So in this diagram here that you see the traditional approach would have been first to design for a laptop and then adapt the content to fit into the smaller window size.
6:55
Increasingly, people are adopting the mobile first approach. What this means is that you first design your website with the mobile screen size in mind. First, do all your layout for the mobile screen size, and then start adapting your website design for larger screen sizes. So as your screen size expands the content will automatically flow and then adjust itself to occupy the larger width of the screen that becomes available on larger screen devices.
7:28
In order to achieve mobile first design, there are three things that need to come together. First, your front end UI framework should be able to support a grid based system. This has been increasingly been opted by most front end UI frameworks. So look at anyone of the front end frameworks that is in use, and you will see that they will have some kind of support for a grid-based system. We'll look at grid-based systems, in particular the Bootstrap Grid, in more detail in the next lecture.
8:02
The second aspect is to have fluid images, or what we call responsive images, so you can make your

website images automatically adapt to the size of the screen. So this again has to be built-in. Bootstrap has good support for fluid images or responsive images already available within the Bootstrap CSS classes. We'll examine that in more detail in the next module.
8:32
The third kind of support that is required is what we call media queries. If you remember, in the previous slide I had mentioned about the viewport.
8:45
Using the viewport, you can get a sense of the size of the screen that is being being used to view your website. So for mobile devices, using the view port properly, you would be able to realize that the screen size is fairly small, say around 700 pixels or even lesser than that. And for larger screen sizes, obviously, you can adapt. Now, this can be used to our advantage by putting in media queries and using media queries in our CSS class design. How is this achieved in practice?
9:31
Media queries uses an approach where, within your CSS file, you will define slightly different variations for each of the CSS classes depending on the size of the window being used. To conduct a media query, you would include a statement like this in your CSS file. You would say, @media and screen, and you can specify the minimum width. And inside these braces for this media query, you can specify the CSS styles that are applicable only if the minimum width of the screen is about six, 600 pixels. So like this you can design the CSS tiles for different screen sizes. So, for example, in Bootstrap as we will see later, you will have media queries which will redesign the CSS classes, or rather adapt the CSS classes for different screen widths.
10:38
To help us understand better, I have brought up the bootstrap.css file for our project that we had designed I opened the bootstrap.css file in brackets, and you can see that I have scrolled inside this file to a location where a media query is present. So, looking at the code here, you'll see that it says media min-width is 1,200 pixels, and if the minimum width of the screen is 1,200 pixels, then I am going to define my container to be 1,170 pixels. So this is one way you can adapt your CSS classes to the size of the screen. So, a media query like this included your CSS file will enable your CSS classes to be redefined for different screen sizes. This is one way to implement responsive design within your project.
11:47
Now, you're going to ask yourselves, oh that seems so difficult to do. There are so many things to track, and how do I make sure that everything works correctly? Don't worry. As a novice you might want to start with a pre-designed front end UI framework like Bootstrap. They have already done the heavy lifting for you. They have already designed the classes to be responsive, so you can start with that and once you get more comfortable with these CSS queries and various features, then you can go in and tweak the files to suit your specific needs.

6. Bootstrap Grid System

In the discussion on responsive design, I had briefly alluded to the grid system and the fact that many for intent UI frameworks support some form of a grid system. Bootstrap, in particular, supports the bootstrap grid system. In this lecture we are going to examine the Bootstrap grid system in more detail and understand how we can leverage a Bootstrap grid system to design responsive bootstraps.
0:39
When we configured our web project to use bootstrap, one of the first set of statements that we included in our html page was the meta tag with the viewport. The viewport enables our webpage to be able to identify the width of the device, and then appropriately adapt itself to the width of that device. So using this meta tag that we have here, we have specified that the width should be equal to the

device-width. So essentially, we are occupying the entire device width as a return by our device from which we are accessing this webpage. So if you're accessing the webpage from a mobile phone or from a tablet the device would match the screen width of our device.
1:39
For laptops and desktops, we know that our browsers can be any size smaller than the entire screen size. So our browser window can be smaller than the actual screen size of the device. So in those circumstances the device will be set equal to the width of the browser window that is being used to access our website.
2:06
So the viewport meta tag is used to be able to identify and set the device-width, so that the content is appropriately rendered with this width in mind.
2:18
Designing websites to be responsive to the size of the viewport, requires us to use the grid system. In particular in this case, the Bootstrap grid system. We will examine this in more detail next.
2:33
The Bootstrap grid itself is designed to be responsive, mobile first, and fluid. So these attributes help us to design responsive websites using the Bootstrap grid system. We will examine how we go about doing this in the next few slides. And then following that we will have an exercise where we will examine this practically in more detail.
3:02
We have already seen the container class in Bootstrap in the previous exercise and the discussion where we introduced Bootstrap to you.
3:12
The container class is used to enclose the content of our webpage. So the container class enables us to define the width that our content is going to occupy on our webpage, when it is rendered in on the screen,
3:31
so once you specify the container class depending on the screen size. It'll automatically fix itself to an appropriate size to match the screen size or the browser window size through which you are accessing your website. Inside the container, we saw that we would divide our content into multiple rows.
3:57
Rows themselves are going to occupy the entire width of the container.
4:03
So the row will span the entire width of the container in this case, as shown in the picture here.
4:11
Inside each row, Bootstrap divides the row into 12 equal sized columns. So this is where the Bootstrap grid system structure comes into place. So when you divide your row into 12 equal columns then, you can take your content, and then lay it out to occupy any number of these columns within each row.
4:39
So for example, you could lay out one piece of content to occupy the first seven columns of this group. So to do that we will specify it as shown in the picture there. We will specify the class as col-sm-5. I will explain the details of that subsequently. But for the moment you can see how we are able to specify a piece of content to occupy five columns. Then we could have another piece of content occupying the remaining seven columns in the strip. So this way, these two pieces of content are going to be laid out on the screen side by side.
5:20
Between these two pieces of content there will be a small white margin left, what we call as a gutter.
5:27
We look at how Bootstrap configures all these different parameters for

5:33
laying out the content.
5:37
Bootstrap itself supplies four different classes to target four different groups of screen sizes. Extra-small to target the extra-small screen sizes, typically mobile phones. Small for smaller screen sizes, larger than mobile phones, but smaller than say, laptops. So this would be the tablet category, there. Then we have medium, md, which typically targets the laptop scale devices. And then finally, we have large, which targets large screens, screens that are greater than 1200 pixels in width.
6:23
In addition, Bootstrap provides us with these column classes, col-sm- whatever number, col-xs-, and md- and lg-. Correspondingly, these targetting the extra small, small, medium, and large screen sizes.
6:48
Now, when you layout any piece of content, you can specify how many of those 12 columns within a row this piece of content is going to occupy. So, that is why, when you saw the example earlier, you saw that I specified a .col-sm-5, meaning that for small screen sizes, this piece of content is gonna occupy five columns.
7:13
Now, looking at the details of all these sizes, when you examine this, in this table here, we can see that we have four groups of screen sizes. Extra small for any screen that is less than 768 pixels. Small for screen sizes between 768 and 992 pixels. Medium for greater than 992 but less than 1200 pixels. And larger screens would be greater than 1200 pixels. Correspondingly to four different categories of devices that each of these are targeting.
7:51
Now the grid behavior itself for extra small screens, everything is laid out in single stacked be one below the other.
8:03
All the rows, and within the rows you can lay out columns appropriately for small to large screen sizes. It will start collapsed first, but as you target the larger screen sizes, these will be laid out appropriately as specified by you using the column classes. The will work also. You can see that depending on the screen size, the container will be appropriately fixed. So for small screen sizes, it is 750 pixel, all the way up to large screen sizes, which is 1170 pixels, out of the 1200 pixels a mode that the width of the screen is sized to be.
8:53
In addition, for any one of these screen categories the rule is always divided into 12 equal columns.
9:04
The gutter width, again, as I've said between two laid-out pieces of content there will be some whitespace left over, which we call as a gutter. The gutters width will always be 30 pixels. So if you lay out two pieces of content, the first one occupying five and the second one occupying seven columns between these two pieces of content there'll be 30 pixel gutter left over as white space. So that way on the screen you will see these two piece of content sufficiently separated from each other, and clearly laid out.
9:38
The column width itself is calculated depending on the size of the screen. So for example, for small screens, as you see the column, each column will occupy 62 pixels, all the way up to larger screens where each column will occupy at least 97 pixels or above.
9:58
When you lay out content on the screen, you can lay out content to be displayed on the screen in different ways, depending on the screen size. In the example that is shown on the screen here, you can see that for extra small screen sizes, I have laid out the two pieces of content to be stacked one on top

of the other.
10:22
This I am specifying by specifying the classes as col-xs-12 for both the pieces of content. But for small to large screen sizes I want this content to be laid out side by side within a single row. Now this is specified, as you see on the right hand side of the picture above, you can see that for the first one, I specified the columns sm-5, meaning that the first piece of content will occupy five columns, the left most five columns. And the remaining piece of content will occupy seven columns. That's why column sm-7 the seven columns to the right of the first piece of content. So this way, these two pieces of content will be laid out side-by-side on any screen sizes, small, medium, or large. You may be wondering why I did not explicitly specify for medium and large. Bootstrap, by default, if you do not specify the layout for some of the categories, then the earlier one that is specified for the smaller screen size will be automatically applied for larger. So in this case, when I say col-sm-5, which means that for nd and lg also, it's the same specification that will hold, meaning that the one piece of content will occupy five columns and the second one will occupy the remaining seven columns. Now, this is the standard way, if you don't specify any further, this is how the content will be laid out on the screen.
12:03
Bootstrap also provides us with an additional set of classes called pull and push classes. Using the pull and push classes, you can override the standard behavior of the layout. So in this example, you can see that for screen sizes extra small, we are laying out the content one on top of the other. The red one on top of the green one.
12:32
But for small, medium, and large screen sizes, the green one will be laid out to the left side, and the red one will be laid out to the right side. This behavior is specified by using the push and pull classes. So as you can see for the red content, I have specified the column sm-push-7, meaning that this piece of content will be pushed right by seven columns. So essentially, occupying the last five columns in that row. The second piece of content I have specified it as column-sm-pull-7, meaning that this piece of content will be pulled lesswards by 7 units. So this way you can easily customize how the content is going to be laid out on different screen sizes.
13:26
In addition to Push and Pull, we also have yet another Column class type, call this offset. Using offset, we can specify that a piece of content should be offset to the right by a certain number of columns. In this example I'm specifying
13:47
that the piece of content should be offset by one column. Meaning that this piece of content will be laid out starting at column number 2 rather than column number 1. If you don't specify the offset by default, the content will be laid out starting from column number 1. Now, the other piece of content will occupy the remaining seven columns in this row.
14:16
One last piece of modification that we're gonna see is the use of nested columns. Now, when we laid out our content
14:28
to occupy two different number of columns on the screen, then inside any one of those pieces of content, I can again go and insert a row. When I do that inside the column, that particular row will span exactly the number of columns that the piece of content is supposed to span. So in this example I see that the right hand side content is supposed to suspend seven columns, so inside the first row, it is occupying the rightmost seven columns in there. Inside that piece of content, I have introduced another row. When I introduce this row, Bootstrap again will give me 12 columns inside that row.
15:11
So meaning that this content that is inside this content that is occupying the seven right-most columns,

they can now again be subdivided into 7 equal columns. And you can then specify content to occupy any number of those 12 columns inside the row which is inside another piece of content which is occupying a certain number of columns in an outer row. So this kind of nesting gives us a lot of flexibility in laying out contents for screens of different screen sizes.
15:50
So now we have seen how Bootstrap's grid system allows us a lot of flexibility in laying out our screens and specifying how the content should be laid out for different screen sizes.
16:06
So now we will move on to our next hands-on exercise, where we will start with the exercise that we worked on in the previous set of lessons, and we'll continue modifying that webpage, applying Bootstrap's column classes
16:28
to properly layout the contents in that page in addition, we will also look at how we can introduce our own custom CSS classes into the webpage.

Exercise: Responsive Design and Bootstrap Grid System

Objectives and Outcomes

This exercise introduces you to responsive design and Bootstrap support for mobile first responsive design through the use of the grid system. In addition we learn how to customize some of the Bootstrap classes through defining our own modifications in a separate CSS file. At the end of this exercise, you will be able to:

Create responsive websites using the Bootstrap grid system
Customize the CSS classes through your own additions in a separate CSS file
Note: In this exercise we will continue to update the index.html file in the conFusion folder that we created and edited in the previous lecture.

Exercise (Instructions): 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 lecture, where we added this line to the head:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

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 have already divided the content into rows. Each row in Bootstrap grid system is divided into 12 columns. 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-xs-*, .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.

Applying column classes within each row

In the header row, we will display the restaurant name and the description to occupy 8 columns, while we will leave four 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:

```
<div class="col-xs-12 col-sm-8"> ... </div>
<div class="col-xs-12 col-sm-4"> ... </div>
```

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

```
<div class="col-xs-12 col-sm-3"> ... </div>
<div class="col-xs-12 col-sm-9"> ... </div>
```

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

```
<div class="col-xs-6 col-sm-3"> ... </div>
<div class="col-xs-6 col-sm-5"> ... </div>
<div class="col-xs-12 col-sm-4"> ... </div>
<div class="col-xs-12"> ... </div>
```

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

Using Push, Pull 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 .col-sm-push-* and .col-sm-pull-* for the first and the third rows as follows:

```
<div class="col-xs-12 col-sm-3 col-sm-push-9"> ... </div>
<div class="col-xs-12 col-sm-9 col-sm-pull-3"> ... </div>
```

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

```
<div class="col-xs-5 col-xs-offset-1 col-sm-2 col-sm-offset-1">
```

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

```
<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 file named mystyles.css in the css folder. Open this file to edit the contents. Add the following CSS code to the file:

```
.row-header{
    margin:0px auto;
    padding:0px auto;
}
.row-content {
    margin:0px auto;
    padding: 50px 0px 50px 0px;
    border-bottom: 1px ridge;
    min-height:400px;
}
.row-footer{
    background-color: #AfAfAf;
    margin:0px auto;
    padding: 20px 0px 20px 0px;
}
```

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

```
    <link href="css/mystyles.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.

```
    <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="row-footer"> ... </footer>
```

Our last set of customization is to the jumbotron and the address. Add the following to mystyles.css file:

```
.jumbotron {
    padding:70px 30px 70px 30px;
    margin:0px auto;
    background: #7986CB ;
    color:floralwhite;
}
address{
    font-size:80%;
    margin:0px;
    color:#0f0f0f;
}
```

Now we begin to see the web page take a form closer to our final design for this module.

Conclusion

In this exercise, we reviewed responsive design and the Bootstrap grid system. We also learnt how to customize using our own CSS classes.

Exercise (Video): Responsive Design and Bootstrap Grid System

We are now moving to the next hands-on exercise where we are going to begin with the web page that we completed in the previous exercise. And we are now going to introduce the bootstrap grid classes into it. In order to appropriately lay out the content of the pitch. In addition, we're gonna introduce some custom CSS classes into the webpage. So let's go ahead and then look at the HTML code for the web page one more time.
0:38
Going into our HTML code in brackets we're going to see that at the top of the page in the head, to be have introduced the viewpoint metatag already in previously, so this is what provides us with the responsive behavior for our webpage.
1:02
Now we are gonna go in and apply the column classes to all the different divs inside the body of our webpage.
1:14
Moving to the header of our page inside the body, I'm going to apply
1:21
the grid column classes to the two inner divs Inside the Jumbotron. So to do that for the first div I'm going to apply a column class as col-xs-12. 12 col-sm-8. 之意思: So this content is gonna be displayed
1:48
in extra small devices it will occupy the entire width of the screen. But for small to large screen sizes, it is going to occupy eight columns. You also notice that I have another div in the jumbotron. Currently, that div contains no content. Later on I am planning to introduce a logo for our restaurant into that div. So I'm gonna leave the div empty at the moment, but I'm gonna apply the column classes to the div so that it occupies the remaining number of columns. To this div I'm going to apply column classes XS 12 and SM 4 because we are left with four columns from the small to large screen sizes, empty on the right-hand side of this row. Once we have completed this, let's now move into the divs in the actual content of our webpage. For each one of those divs, I have a headline and then I have the actual content. So the headline part of it I will enclose in a smallish number of columns, but the detailed content, I will have it occupying a larger number of columns. So I'm going to do that by applying two different div classes for each of those columns. For extra small screen sizes, I'm going to lay them out one on top of the other stacked, but for small to large, I'm gonna lay them out side by side.
3:24
So let's go and apply the column classes to those inner divs in those rows there. For the div containing the headline, I'm going to apply column xs 12 and column sm 3 in that case and for the other div I'm going to apply the remaining columns from that. Think about how I'm going to do that, all done,
3:51
let me repeat that for the remaining rows of. The actual content of the webpage.
3:59
For the divs in the footer, I am going to have the first div occupy three columns for the small to large string sizes, the second div to occupy five columns. For the small to medium and large. And the last div to occupy the remaining for columns. But for extra small I'm gonna have the first and second divs occupy six columns each. And the third one will occupy the entire twelve columns.
4:30
And the copyright statement I'm going to let that expand the entire twelve columns. So let me apply those column classes to be as tips and then we'll come back and look at the completed webpage. I have now completed the changes. Let's go and have a look at the page. You can now see how the web page is rendered. You can see that the jumbotron, the restaurant name and the description is occupying left eight columns. And then we have empty space here on the right side where the logo will go in which

will be part of the exercise in the next module. Then, we see that the three pieces of content are now laid out such that the headline occupies three columns on the left side. And the actual long description occupies the remaining nine columns here. Going down to that footer, now you can see that the links occupy three columns, the address occupies five columns. And then the links our social media sites occupies the remaining four columns here. The copyright down below here is spanning the entire twelve columns of that row.

5:55

Let us now examine the responsive behavior of this web page. Now, taking this web page, let me slowly shrink the web page and see how the webpage responds as the screen size is reduced. So this for 1,280 pixels. This is the length of the screen size that I'm using currently. I'm gonna keep shrinking the size and you can see the changes in the layout of the webpage as I reduce the screen size. So this is still for small screen size. So you can see the layout is still like that, but let's go for the extra small and see what happens.

6:46

Now you can see that when I switch from small to extra small, immediately our content layout changes its behavior. So, now I have my description of the restaurant and the name of the restaurant display occupying the entire screen size. These two items now instead of being side by side now are stacked one on top of the other. And then look at the footer. You see that the links and address are displayed side by side.

7:21

And the links to our social media sites are displayed in a separate

7:28

columns, separate row spanning the entire screen size. The copyright remains as such. So now you can see how we have introduced responsiveness to our web page.

7:45

We can see that within the web page, the headline and the description are all laid out in a similar fashion in all the rows. Now I wanna make a slight change whereby for some of the rows I want to swap the headline and the description around specifically for the question in the third row. So that the webpage acquires some more interesting character. How do we go about doing that? This is where the push, pull and offset classes that we discussed are going to come to our aid. Let's apply those changes to those classes and then come back and see the layout of the web page of the point. For small to large I'm going to push the headline to the right by nine units, and pull their description to the left by three units.

8:44

So returning to our HTML page, now for the first and third row, I'm gonna push the headline to the right, and pull the Description to the left. So going into the divs, for the headline div I'm going to apply col-sm-push-9and for the description I'm going to apply col-sm pull-3. Same thing for the third row too.

9:16

Going to the footer for the div that shows the links to our webpages, I'm going to apply an offset of one unit to the right and I'm going to led the two units. So i am going to go in here and I would say, col-xs-6, but I will change to col-xs-5. And then I will say col-xs-offset-1. And col-sm, I'll reduce it to 2. With this I have updated the class definitions for the links row, links column in my footer. And I'll save the changes and let's go and look at the webpage.

10:13

Returning to our webpage, now you see how I have used the push and pull classes to swap things around in the three drawers. In the first drawer you can see that the headline has been pushed to right and the content has been pulled to the left. And similarly for the third row. Now you can see that the links down below here have been pushed right by one column using the offset.

10:42
The remaining ones stay as such. Now if I shrink the webpage to the extra small size, notice that the behavior. The layout remains the same as before.
11:01
Coming down to the footer I notice that my links are being displayed as a bulleted list. I don't want the bulleted list instead I want to be completely left so to do that I will apply a unstyled list class to that list and then you will notice the change there. Going into the code for this UL tag I am going to apply a class list-unstyled. Returning to your webpage you can see now in the footer how the links are being displayed without the bullets.
11:42
Next I'm going to show you how we can introduce a custom CSS file into our web project. To do that, go to the CSS folder and then introduce a new file there. And then we will name it as mystyles.css. And then when the mystyles.css file is opened, I'm going to paste in some CSS code to be applied to some of the elements in my web page. So I'll paste in the CSS code which I have provided for you. This CSS code, I am making use to style my various rows in the web page.
12:31
In here, you can see that for my header and my footer I am giving some margin and padding values, and then similarly for the content rows I am giving some margin and adding values. In addition, from my content rows, I am making every content row to be a minimum height of 400 pixels. So, this way, our webpage is laid out in a lot more cleaner fashion.
12:59
For the footer, I will apply a background color of a light gray color there so that it looks a bit more different from the remaining rows of my web page. After making the changes to the mystyles.css file, I'm gonna save that, and then now I need to include the CSS file into my HTML page.
13:24
Moving back to the html code, in the head right after the import of the bootstrap css files, I am going to add in one more to import the CSS file that I've just created. So I will say css/mystyles.css. That's the page that I want to input, and this is a style sheet.
13:58
So, with that I have included my custom CSS classes into my project. Now I can apply those CSS classes to the different rows. So, let me go ahead and apply those to the header, the content rows. And the footer and then we'll see how the webpage has changed. So I have applied the header formatting to the row in the header. Then I have applied that content row content class to the rows in the content part of our webpage. And finally I'm ready to apply the footer content to the row inside my footer. Or rather I will apply that directly to the footer itself.
14:39
So now that I have completed applying these classes, let me go back and check that web page and see how it has changed.
14:48
Returning to your webpage now you can see the webpage layout has changed slightly. The jumbotron remains as such in the header. Then, but the remaining rules are now occupying a lot more space on the screen. So since I have declared that the minimum width each row should occupy would be 400 pixels. So the height of each of these rows is now set to 400 pixels.
15:17
The footer has now acquired the gray background color and is displayed at the bottom.
15:26
Finally, I'll go into my styles.css file and add in a couple of more custom classes for the jumbotron and the address that I'm using in the footer. So adding in the code, you see that for the jumbotron, I am setting the padding for the jumbotron appropriately. And also I am going to set the background of the

jumbotron to be a light indigo color. And the text to be displayed would be in a off-white color in the jumbotron. The address which appears in the footer of my page. Now, I'm going to reduce the font size of my address to be slightly smaller than the standard size. It'll be about 80%, so that is the modification that I am applying here to that address.
16:28
Switching back to the view of our web page, we now notice that our jumbotron has acquired a light indigo color, and in addition the lettering is now in off-white color in the jumbotron.
16:47
Going down to the footer, you will see that our address is now displayed in a slightly smaller font size than before. This completes the modifications that we are going to affect in this exercise. At the end of this exercise, we have learned how to apply the column classes to the various divs inside of rows in our webpage. In addition, we have also seen how we can introduce custom CSS classes and then apply them to various elements in our webpage.
17:23
This concludes this exercise.

Navigation and Navigation Bar

Objectives and Outcomes

In this lesson, you will be given an overview of navigation design and the importance of providing appropriate navigation support within your website. You will learn about support for navigation design elements available in Bootstrap, including the Navbar and Breadcrumbs. Other navigation aids will be covered in subsequent modules. In addition, the use of icon fonts in web page design will be covered. The exercises will concentrate on adding a responsive navigation bar to the website. At the end of this lesson, you will be able to:

Understand the need for navigation support in a web project
Use the Bootstrap navigation features including the Navbar and breadcrumbs in providing navigation support in websites
Use icon fonts for decorating your website with meaningful graphical elements
Mark as completed

7. Navigation and Navigation Bar

Let's now talk about navigation. What is navigation? Why is it important in the context of websites? And how do we provide navigation within our website hierarchy for visitors to our website?
0:21
When we look at many websites out there, we immediately realize that most websites are not single pages.
0:30
Of course, you do see occasional single-page websites, too. But in general, most websites typically consist of multiple pages. And so when a visitor comes to your main web page, typically indexed .html, they might also need to navigate to other pages within your website. And when they do so, you should have a way of indicating to them which other pages are there on your website, and also if they move to another page, where they are within your web page hierarchy on your website, and also an easy way of returning to the main page of your website. All these features can be provided through various navigational aids that you can build into your websites. Now there are several commonly used navigation patterns that are used extensively on most websites. In particular, navigation bar is an easy

way of providing
1:39
the user with access to the hierarchy of a website.
1:44
When you think about website navigation, one of the important features that you should consider is the information architecture that you use in your website. What exactly is information architecture? This has to deal with the organization, the labeling, and the appropriate navigation methods that you provide through the information that is available on your website. Now, information architecture design is a whole field in itself. And that is beyond the scope of this course. But I have provided you with some additional links in the resources, where you can go and explore more details about information architecture. In particular, we need to consider information architecture when we are designing websites, and that also suggests what kind of navigation we should provide within our website.
2:40
Typically, websites are organized in some kind of a hierarchical fashion.
2:46
You have a top level page from which you might have subpages. And from those, you can go into further down in the hierarchy. You might even have, within a single layer, cross navigation between different entities within that layer. So, any navigation scheme that you design should be able to accommodate all these different patterns of access to these different objects of information within your web page. In general, most websites will have a navigation bar
3:26
designed and available within the web pages of the website. The navigation bar is typically shown either at the top or at the bottom of the website. Sometimes it will be fixed in location, either the top or the bottom of the window, which is displaying the web page. Whatever be the case, a navigation bar has become a de facto standard for providing a way of informing a visitor about various pages that are available within your website and how you can navigate from one to another. In addition, within the navigation bar, you might also indicate where the user currently is within this hierarchy.
4:13
Taking a look at your own course website, you will see the provision of a navigation bar within the course website. In this case, it is provided more like a sidebar. So you can see that on the left-hand side, you have information like where you are currently at the moment. And so for example, right now, we are on the Course Info page of
4:37
our course website.
4:40
There are a few suggestions on how to design a good navigation bar. Here, I have a small list of them. Of course, non-exhaustive list, so you might have more things to consider when you are actually applying a navigation bar to your own website. But when you design navigation, you should typically provide user-friendly terms, but list it within your navigation bar. And also, you have to standardize the navigation. So no matter which page they user is at the moment, they will still be able to navigate to any other location within your website.
5:25
And also, some way of identifying where within the hierarchy the user currently is. So you might highlight the current page that the user is viewing in the navigation bar, so that it's easy for them to see where they are within the hierarchy.
5:45
There are some standard web conventions that people use when they design navigation bars. One such convention that has been widely adopted is that if you put a logo of your website in your navigation bar, clicking on that logo will typically take you back to the home page of that website.

6:05
Now, some things that you should definitely avoid is to have too many items in your navigation bar. How many is too many? If you find it difficult to use your navigation bar, well, that is always already too many.
6:19
It is better to have a smaller subset, targeted subset, of links within your navigation bar. Meaningful links, within your navigation bar. Again, this is a choice of taste for different people that have different criteria for considering what should and should not be in the navigation bar. And also depending on the specific website that has been designed.
6:47
One other thing that you should think about is to avoid generic labels. Things like if you put a product there. What does a product mean? Or a service there. What kind of service are you talking about? It is better to customize those labels to specific items that you are providing within your website.
7:10
Another commonly used pattern for providing navigational information within a website is what we call as a breadcrumb. If you have noticed on many websites, at the top, you might see a set of links being indicated with pointers to sublinks. And the current page that you are on might be highlighted in that set of links. Clicking on any one of those earlier links will take you back within the hierarchy. So, in some sense, the breadcrumbs sort of also reflects some kind of a structure of your web page. Now, there are many different ways of designing breadcrumbs. If you have a website where people are going through a step-by-step process, say, for example, purchasing an airline ticket, the breadcrumbs might indicate the current sequence within the entire set of steps that the user is at the moment. Or if you have a e-commerce website, you might have a current sequence of steps in order to order an item or a set of items from that company. So you might indicate, for example, whether the user is currently in the selection process of the items, or the user is in the payment process, and so on. So, that is what we have addressed, path based navigation support within the breadcrumbs. The second approach would be to provide a location based information within the breadcrumb hierarchy. A third alternative that you might have seen in some websites, where the breadcrumbs are not actually indicated as a linear sequence but more like a set of choices, maybe where you have several set of attributes related to the items that you might be selecting on an e-commerce website. Say, for example, a price range, the color, or the size, or the type, or maybe the manufacturer's information, and so on. So those kind of things that can also form information that
9:29
could be characterized under breadcrumbs. Let's take an example of your own course website
9:39
to see how breadcrumbs are being used there.
9:45
For example, when you visit your own course page,
9:51
as you are going through the various items in the course content, you would notice on the top a breadcrumb based navigation that is already provided for you. In there, we have the upper level links being shown in the breadcrumbs as links and the current location
10:17
being shown just by information, and it is not a link, but just identifying where you are currently in the hierarchy.
10:28
There are several other navigation aids that are used in practice. These include tabs, pills, pagination. Especially when you have a lot of information to be shown to a user, you might actually subdivide it into pages, and then let the user access the information page by page. So you would provide some kind

of pagination support at either the top or the bottom of the page of information that you are displaying. We have dropdowns, accordions, tags, Scrollspy, and Affix, and many others. We will look at some of these in more details in subsequent modules of this course.

8. Bootstrap Navbar

Let us now talk about Bootstraps own support for navigation. In particular, in this lecture, we will talk about the Bootstrap's Navbar.
0:14
Navbar is a very easy way of providing navigational information within your website. Let's look at the details next.
0:26
As I already mentioned, the Navbar is a way of easily providing navigation for your website. The Navbar typically appears at the top of the page or the bottom of the page depending upon the design of your website. Now, Navbars can also support responsive features, meaning that when you're viewing the website on a extra small screen. Maybe you may wish to collapse the Navbar and only display content by toggling the information appropriately.
1:00
For larger screen sizes, it is probably more easier to accommodate a full fledged navigation bar. So we will look at bootstrap support for both situations in more detail next.
1:15
Let's talk in particular, about a simple navigation bar that can be provided within bootstrap and how we go about configuring it. So here I have an example of a simple navigation bar and the corresponding code that is used in Bootstrap to design this navigation bar. Now I am leveraging the Nav HTML element to design my navigation bar. To this nav HTML element, I am using the Navbar classes that are available as part of Bootstrap. The Navbar, which turns that HTML element into a Navbar and the Navbar default. So this is the default structuring of the navigation bar in Bootstrap or the default coloring and so on. Also notice that I am declaring that this Navbar takes on the role of providing navigation within this website. Now let's look at the example Navbar on the screen and then see how this is actually constructed by using the HTML code below there.
2:29
In particular, you see that the list of items being displayed in the Navbar is declared in the code as an unsorted list
2:40
with the class nav navbar-nav. And within these each list item is displayed using li and then you provide the href through the normal HTML links there.
2:57
One particular list item has been identified as the active list item there. So for this example, I have identified the Home as the active list item. What this means is that in the Bootstrap's Navbar, this Home item is highlighted with a darker color for the default Navbar as you can see on the screen. I am now extending the Navbar into Into adding more features. One, is that I am adding in what we call as a brand to the navbar. So within the Navbar on the left side, you can see the name of the website being displayed there. That name will form part of the Navbar header. And inside the navbar header, which is a separate div there, I am declaring a class called navbar-brand to a link and then I am providing that link there. Now you can replace the text within that link by an actual image which might be the logo of your website. So that's another way of providing navigation within the navbar, specifically using the logo navigation. Typically, when you put the logo into the navbar, users expect that when they click on that logo, they are going to be taken back to the home page of the navbar. So when you specify the href for that class there, you should make sure that it is identifying your home page.

4:40
We can also similarly put items to the right side, right into the screen within the navbar. To do that, we are going to use the nav navbar-nav for the unsorted list that we've seen already. In addition, we're going to supply another attribute called navbar-right. When you specify that, there are list items that are gonna be put on the right edge of the navbar. For left edge you, would say navbar-left, for right edge you would say navbar-right.
5:14
We can make the navbar also responsive. Meaning that when you are viewing the website on an extra small screen or any chosen size of the screen. You can collapse the navbar and hide the items within the navbar. And instead replace it with a toggle button as you see in the picture here. When the user clicks on the toggle button, then you will expand the navigation bar to show the link items, like so. When the user again clicks on the toggle button, you're going to hide those navigation items. So this behavior can easily be supported within Bootstrap using the toggle button that can be included in the header, nav-navbar header of your Bootstrap. We will see how that is done in the next slide. Now we are going to examine how to design this responsive navbar.
6:17
First, we need a toggle button within the navbar. So this is usually included in the navbar-header as shown by the code here. So in here, I am declaring a button type. And the class I am using is navbar-toggle and also specified as collapsed. Meaning that the items that this Button is referring to will be in the collapsed mode when you are viewing this page on an extra small device.
6:54
When the user clicks on this button, then the further details are the extended navbar will be toggled and shown on the screen. Now this is implemented by enclosing the items within the list inside a div and giving it an id. This id in this example, the id given is navbar and that id is used as a data target for the button. What exactly is a data target and how is it used? For that, we need to understand that JavaScript
7:35
modules that were implemented in Bootstrap. So I'm going to come back to refer to this one more time in the module where we look at JavaScript within Bootstrap. But here, let's understand that this id is also used to specify the data target for the button. Meaning that that button is going to toggle on and off, whatever is inside this div on the screen. Now for this div in addition, I'm going to give the class as navbar collapse and collapse. Meaning that initially when you are viewing this page on an extra small screen, the navbar will start out collapsed. When the user clicks on the button, then the navbar will be toggled and vice versa.
8:30
Some additional configurations that you can give for navbars, include specifying that the navbar should be fixed to the top of the page.
8:40
Meaning that the navbar is always visible and will be fixed at the top of the page. The content will scroll, but the navbar will remain fixed in that location. You can also similarly fix the navbar at the bottom. The corresponding classes are indicated here. .Navbar-fixed-top and .navbar-fixed-bottom. There are other specification that you can give is to specify the navbar as navbar-static-top. Meaning that this navbar will be at the top of the page, but will scroll when you scroll the page. So the navbar will disappear from the screen when you scroll the page up.
9:19
In Bootstrap, earlier we saw that we declared the navbar as navbar default. The default navbar for bootstrap is a light colored navigation bar. You can also have a dark colored navigation bar in bootstrap by providing the class as navbar-inverse instead of navbar-default. So in the navbar-inverse, the background of the navbar will be dark in color but the text will be lighter in color.
9:49

One last addition to the navbar that is often used, is what we refer to as a dropdown.
9:57
When we talk about a dropdown,
10:00
any item within the navbar can be turned into a dropdown. So that when the user clicks on the item, then you would display a list of dropdown items to the user and then allow them to select one of those dropdown items. Similarly, when you click on the item in the navbar, then we will hide this dropdown, right? So this toggling of this dropdown list of items is done by clicking on the item inside the navbar. In this example, if you click on the Menu item in the navbar, it will display the detailed list there. If you click on it again, then that detail list will disappear. So this is the use of dropdowns in navbar. How do we implement it?
10:56
We'll examine that next.
10:59
So to implement the dropdown, we need to Declared the drop down as a list item within the unsorted list that we used to construct the navbar links there. Now one of the items will be declared as a list item of the type dropdown. The code on the left side indicates how you would declare the list item as a dropdown. And in addition, you specify the class as dropdown-toggle and the data would be specified at dropdown, data-toggle as dropdown. And some additional rows there and in particular, notice that declaration of the span class caret. Now this is used to show these inverted triangle next to the item within the bar. To indicate that when the user clicks on that item, we're ging to show a dropdown list there. Now how do we declare the items within the dropdown list? That will be included as another unsorted list inside this list item. And this unsorted list shown on the right side has the class dropdown menu. And individual list items listed there will be shown in
12:34
the list of items in the dropdown menu. So you can see that these list items correspond with Appetizers, Main Courses and so on. You can also include a separator by declaring the role as a separator and the class as a divider for one of the list items. You can also have a header there by declaring a list item with a class dropdown-header. So that that will not be treated as a link, instead it will be treated as just a header for the items displayed below. So this is yet another feature that we will use within the navbar.
13:15
This completes our discussion on Bootstrap Navbar. In the exercise that follows, we are going to design a navbar for our webpage.
13:27
At the end of the exercise, you will see our webpage having a fixed top navbar as shown in the website details here. And this navbar is colored appropriately for our website. We'll see how we can do that using the CSS class and we also see a dropdown item within the navbar.

9. Icon Fonts

Let me now give you quick overview of icon fonts, what are they, how are they used on websites, and what advantages do they bring when you use them within your websites.
0:16
Icon fonts are a set of icons and cliffs. They can be used just like regular font on your website so just like you use your standard alphabets on your website, you can use icon fonts in any location on your website.
0:36
The advantages that icon fonts bring is that they can be easily styled using CSS. Which means you can control the color, size and background and many other features of icon fonts using CSS. Icon fonts are

a very light weight replacement for graphics on your webpages. So if you have simple graphics, instead of using images, you can use icon fonts as an easy replacement for them. Now when your web site includes icon fonts, they are downloaded without needing additional
1:18
requests being sent to your web site in order to download the images. So that's one of the advantages why icon fonts have gained a lot of popularity in usage on websites. Many different icon fonts are available that are used by website designers. To help us understand how icon fonts are used in websites, let's quickly visit your course web page itself. Now, on the left side in the side bar, you can see that we have a list of links there. In addition to the links, to the left side of the links, we also have small graphical icons included in the web page. All these icons have been created using icon fonts. Let's take a look at few example icon fonts that are used. In particular, in this course, I'm gonna make you use of the Glyphicons. Glyphicons are already included as part of the Bootstrap framework.
2:21
In addition, I'm going to make use of Font Awesome, which is another popular icon font.
2:28
As I mentioned, Glyphicons are already included in Bootstrap. To make use of Glyphicons, on any location in your webpage, you use the span tag. To the span tag you will give a class as glyphicon and then the specific class for the icon that you are using at the location. So in this example you see me using the Glyphicon home as the second class there. Glyphicon home corresponds to the home icon in the glyphicon set.
3:07
Glyphicons included in Bootstrap are about 250 in number.
3:14
Another popular icon font that I mentioned, that we will make use of in this course, is Font Awesome. Now Font Awesome includes a large set of icons. And provide you with a large variety of icons that you can use on your website. Now once you take a look at Font Awesome you're gonna say, awesome man.
3:36
Okay, okay, okay. You can laugh as much as you want. I am definitely not going to pretend to be a teenager.
3:45
Okay coming back to icon fonts. They way you use icon fonts on your website, Is to use the i tag. To the i tag, you would give a class as fa and then fa hyphen the specific icon that you're using at that location. So in this example, you see me using fa fa phone, meaning the phone icon will be substituted in that particular location on your webpage.
4:12
How do you make use of Font Awesome in your web project? You need to download the Font Awesome distribution files, and then you should include the CSS files that are part of Font Awesome, into your web project. And the fonts will automatically be used. Now one way that I use Font Awesome in my web projects is after they download the Font Awesome distribution files, then I will copy the CSS files from the CSS folder of the Font Awesome folder into the CSS folder of my web project. And also I will copy the fonts from the Font Awesome folder into the fonts folder of my web project. That's how I have been using.
5:05
You can include these Font Awesome files in any other way that you choose to. Another CSS class that I am going to make use of in the exercise that follows this lecture is Bootstrap-social. Bootstrap-social is a CSS class that is available to
5:25
style some buttons for social web sites. Bootstrap-social makes use of Font Awesome's icon fonts that

are representing the logos for the various social websites. Now, Bootstrap-social adds additional features to those Font Awesome icons. So for example, it takes the Font Awesome icon and then turns it into a button for the various social media sites. How do you make use of Bootstrap-social? You download Bootstrap-social.CSS file and then include it into your webpage.
6:12
Then wherever you need to include a button to a social website then you will include a A tag and give it a class as button. Button, of course, buttons which we will examine in the next module. Then you specify buttons social icon, and button for the specific social website. So, in this example, you will see me referring to button Facebook, and then In addition, you would be including the Font Awesome icon for that particular social media site. So that's why you see me using there the i class, fa, and the fa-facebook to use the Facebook icon that is included as part of the Font Awesome set. Now we will move on to our next hands on exercise. We will, in this exercise, go back to the website what we have been designing so far and then include the nav bar into our web page. In addition, we'll make use of both Glyphicons and then Font Awesome icons in various locations on our web page. We'll also use Bootstrap-social to include some social website buttons in the footer of our webpage.

Exercise (Instructions): Navbar

Objectives and Outcomes

In this exercise, we will examine the navigation support that we can build into a web page using the Navbar in Bootstrap. We will also learn the basic use of icons in web page design using the built-in glyphicons that are part of Bootstrap, the Font Awesome icons, and bootstrap-social icons. At the end of this exercise, you will be able to:

Create a navigation structure for your website using the Navbar
Use icons within your website to represent various entities making use of the glyphicons, font-awesome icons and bootstrap-social icons
Include additional CSS classes into your project
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-inverse navbar-fixed-top" role="navigation">
  <div class="container">
    <ul class="nav navbar-nav">
      <li class="active"><a href="#">Home</a></li>
      <li><a href="#">About</a></li>
      <li><a href="#">Menu</a></li>
      <li><a 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-inverse class to specify that the page should use the dark navigation bar. In addition the inner ul is used to specify the items to be put in the navigation bar. This ul is styled with nav and navbar-nav classes to specify that the items should be displayed inline inside the navigation bar. We also use the container

class inside the navigation bar. This navigation bar does not use responsive design at the moment.

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 smaller screens. This can be achieved by adding the following code to the navigation bar, just below the container div

```
<div class="navbar-header">
   <button type="button" class="navbar-toggle collapsed" data
    -toggle="collapse" data-target="#navbar" aria-expanded
    ="false" aria-controls="navbar">
      <span class="sr-only">Toggle navigation</span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
      <span class="icon-bar"></span>
   </button>
   <a class="navbar-brand" href="#">Ristorante Con Fusion</a>
</div>
```

You will now notice the addition of a link to the left of the Home 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. The other part is the creation of a button with three horizontal lines. For larger screens, this button is hidden. For smaller screens, this button becomes visible. This button will act as the toggle for the navbar items on small screens. At the moment you still see the items being displayed in the navigation bar even for smaller screens. We will fix this in the next step.

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

```
<div id="navbar" class="navbar-collapse collapse">
   <ul class="nav navbar-nav"> ... </ul>
</div>
```

By doing this, we are specifying that this navbar with the id navbar will be collapsed on smaller screens, but can be toggled on or off when the toggle button is clicked. Note the use of data-toggle="collapse" data-target="#navbar" aria-expanded="false" aria-controls="navbar" within the button. This specifies that the menu items are collapsed on smaller screens when the toggle button is visible. They can be displayed or hidden by clicking the toggle button.

Note that the navbar scrolls when the web page in the browser is scrolled. If we wish to keep the navigation bar always visible at the top of the page when the page is scrolled, then we should change the navbar-static-top to navbar-fixed-top. Let us do this now. Note that after the change, the navigation bar remains visible at the top of the page even when the page is scrolled.

Adding a Dropdown Menu to the Navigation Bar

The next modification adds a Dropdown menu to the navigation bar. Let us target the "Menu" item in the navigation bar and turn it into a dropdown menu item. When this item is clicked, a dropdown menu will be displayed. To do this, modify the list item for the "Menu" item in the navigation bar by replacing it with the following code:

```
<li class="dropdown">
   <a href="#" class="dropdown-toggle" data-toggle
    ="dropdown"
    role="button" aria-haspopup="true" aria-expanded
     ="false">
    Menu <span class="caret"></span></a>
   <ul class="dropdown-menu">
      <li><a href="#">Appetizers</a></li>
      <li><a href="#">Main Courses</a></li>
      <li><a href="#">Desserts</a></li>
      <li><a href="#">Drinks</a></li>
      <li role="separator" class="divider"></li>
      <li class="dropdown-header">Specials</li>
      <li><a href="#">Lunch Buffet</a></li>
      <li><a href="#">Weekend Brunch</a></li>
   </ul>
 </li>
```

In this modification, the "Menu" list item is now replaced with a dropdown-toggle, which when pressed will show the selection menu below the navigation bar. The structure of this code has similarities with the toggle button that we used above for showing and hiding the navbar items for small screens.

Modifications to the CSS styles

We would like to have the navigation bar displayed in darker blue color, instead of black. In addition, we would like the navigation bar to indicate the current page by highlighting the item with a darker background in the navbar. 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 mystyles.css file

```
body{
   padding:50px 0px 0px 0px;
   z-index:0;
}
.navbar-inverse {
    background: #303F9F;
}
.navbar-inverse .navbar-nav > .active > a,
.navbar-inverse .navbar-nav > .active > a:hover,
.navbar-inverse .navbar-nav > .active > a:focus {
   color: #fff;
   background: #1A237E;
}
.navbar-inverse .navbar-nav > .open > a,
 .navbar-inverse .navbar-nav > .open > a:hover,
 .navbar-inverse .navbar-nav > .open > a:focus {
   color: #fff;
   background: #1A237E;
}
.navbar-inverse .navbar-nav .open .dropdown-menu> li> a,
```

```
.navbar-inverse .navbar-nav .open .dropdown-menu {
   background-color: #303F9F;
   color:#eeeeee;
}
.navbar-inverse .navbar-nav .open .dropdown-menu> li> a:hover {
   color:#000000;
}
```

We are already beginning to see the page format close to the final format for this module.

Using Icon Fonts and Other CSS classes

The last part of the exercise is to make use of the glyphicons that are provided as part of Bootstrap. In addition we will also use two additional popular font icons.

One of the most popular iconic font toolkit is Font Awesome. Go to its website https://fortawesome.github.io/Font-Awesome/ and download the zip file and move it to the conFusion folder and unzip it. You will find the font-awesome-4.4.0 folder being created. Go into this folder and move the contents of the css folder to conFusion/css and the contents of the fonts folder to conFusion/fonts folder. Then you can delete the font-awesome-4.4.0 folder.

Download the bootstrap-social.css file to conFusion/css folder. This is a modified version of the bootstrap-social available on the http://lipis.github.io/bootstrap-social/. We added in support for G+ and YouTube buttons.

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:

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

Let us now use some font icons in our web page and decorate it. First use the glyphicons that is part of Bootstrap to add a home icon to the Home link on the navigation bar. Update the home list item as follows:

```
<li class="active"><a href="#"><span class="glyphicon
   glyphicon-home"
     aria-hidden="true"></span> Home</a></li>
```

Next, 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"></i>: +852 1234 5678<br>
<i class="fa fa-fax"></i>: +852 8765 4321<br>
<i class="fa fa-envelope"></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 by replacing the social sites' links with the following code:

```
<div class="nav navbar-nav" style="padding: 40px 10px;">
   <a class="btn btn-social-icon btn-google-plus" 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-youtube" 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>
```

Conclusions

We have learnt how to add navigation support into a web page using the navigation bar. We also learnt about using other CSS class and other icon fonts in a web project.

Exercise (Video): Navbar

In the final exercise for this module, we're gonna start with the webpage that we completed at the end of the previous exercise. And add a few more features first. We will build in a navigation structure for our website. We will do that by including a navigation bar into our webpage and then add in the links to other webpages that are going to form part of our website. Now the remaining pages will be added as we go through other modules of this course. We will put the navigation structure into place at this moment. The second item that we are going to look at in this exercise is the use of Icon font in designing our web pages.
0:57
In particular, we'll look at Font Awesome and the built in glyphicons that come with Bootstrap and also we'll look at adding Bootstrap social buttons, in order to create
1:13
buttons to various social websites on our webpage.
1:21
As I mentioned in the body of a ratio replace just before the header I'm going to paste the code to create the navbar.
1:32
So as you can see I have already pasted the code for the navbar I am using the nav element from HTML5 And I have styled it with the navbar, navbar-inverse and navbar-static-top classes. What this means is that this navbar will appear at the top of the page.
1:52
Taking a look at the web page, you can see that the navbar is displayed at the top of the page and will scroll as I scroll the page. In addition, you notice that that Home link on the navbar is highlighted with a darker background. Now, I am using the active class in order to highlight the. Going back to our navbar code, you notice that inside my navbar I have a container. And within the container I am using a unsorted list. And I am styling it with the class nav navbar-nav. So this is where the links that appear on our navbar are included. The home link, in particular, I have styled using the active class. And hence, it has been highlighted on the screen. Going back to our web page, let's see how the navbar behaves as we
3:02
resize that browser window.
3:07

So you can see that the navbar continues to be displayed at the top, but when we hit the extra small screen size, do you notice that the navbar is now being displayed vertically? Now this is obviously going to look look ugly on small screens. How do we turn this navbar into a responsive navbar? Meaning that when you are displaying it on extra small screens, I want this content of the navbar to be hidden and only shown when I explicitly ask for it to be shown. Returning to our navbar, I have pasted in additional code in order to make it responsive. We have seen this code already in the lecture earlier. So this code is exactly similar to what I have shown you earlier when we discussed about bootstraps navbar. So this makes the navbar responsive. So after this change, let's go and have a look at our web page and see how it responds when the browser window is shut.
4:14
Returning to the web page, we can now see that the navbar has the additional brand shown within the navbar.
4:26
In addition, as I shrink the webpage, you notice that
4:33
when the page reaches the extra small size, the links in the navbar are now hidden, and instead, to be replaced by this button here. When I click on the button, the items in the navbar are displayed, so I can click and hide by toggling, using this button.
4:57
As a next step, I'm going to introduce a drop down into the navbar.
5:03
I'm going to take the Menu link in the navbar, and then when I click on the Menu link, it'll display a list of
5:13
items or categories of items in the menu, so that their visitor can be directly taken to any one of those categories, like appetizers, main courses, desserts, and so on.
5:27
So, let's go ahead and make the change the code, and then review it, to introduce the drop-down Into our navbar. I have replaced the menu link that we had in the unsorted list by this list item, whose class is dropdown. So inside this list item, we have the contents of the dropdown enclosed here. Again inside another unsorted list which is. Styled with the class dropdown-menu. So, this list will be displayed when I click on the menu item. Let's go visit the web page one more time. Taking a look at the web page now, you'll see that the Menu item has now been replaced with a Menu drop-down. You'll see the small inverted triangle there, indicating the caret. Now, when I click on the item, you can see the items on the drop-down being displayed here. The next modification that I'm going to introduce is to turn the navbar from a navbar static top to a navbar fixed top. So I'm gonna go in and change the class from navbar-static-top to navbar-fixed-top in the navigation code. Now in addition I'm going to paste in some additional CSS classes into mystyle.css file. So let me paste that and then we'll review it quickly. The additional code that I pasted into the mystyle.css file allows me to style the navbar in a bit more detail. In particular, I am giving the body a padding of 50 pixels, because now, since the navbar is fixed at the top of the page, the navbar is going to cover the top 50 pixels of the browser window. If our body of our page scrolls, then the top 50 pixels are gonna be hidden underneath the navbar. So, to avoid that, I'm giving a upper margin of 50 pixels, so that the body is displayed below the navbar. And will still be able to scroll without getting hidden by the navbar on top. I have also customized the navbar color itself. I have redefined navbar-inverse to have a background color.
8:04
This reaches a darker shade of indigo.
8:09
In, addition I have customized a few more of the navbar classes. So I have made sure that,

8:19
The navbar, if I am using the active class, that background color is going to be even darker shade of indigo. And also, the color for the text is going to be white.
8:38
So, with this change, whenever I declare any one of the list items in the navbar to be a active link, then that will be colored even darker than the navbar's color. In addition, also for the drop down menu, you saw that drop down was being displayed with Pale background with darker lettering. I have changed that to follow the same color as the navbar itself. So after these changes, let's go and take a quick look at the. Now, going to the web page, you can now notice that my navbar has acquired a dark indigo color.
9:21
This is to contrast this with the lighter indigo colors after.
9:28
In addition you see that the active link is even more darker than the dark indigo that I am using for the navbar. So that the highlighted link is more clearly visible. The Menu items, when I click on that you see that the background color of the drop down has been changed, and the items themselves are now displayed in light color. And as I scroll through the items, the selected item is clearly identified. Now, when you scroll this, Web page, you notice that the scrolling, none of the part of the web page is hidden under the navbar. Our last modification to the webpage is going to introduce Icon fonts into our webpage.
10:23
I'm going to make use of the built-in glyphicons that come with Bootstrap. In addition, I'll download the Font Awesome Icon font by and then include that into our web project. Also, I will use Bootstrap Social to style the social media buttons that I have included in my webpage. So, I had links in the footer of the web page earlier so I'm gonna turn those links into buttons styled with Font Awesome and Bootstrap Associate. So let's go ahead and do that next.
11:05
First, we'll go to the Font Awesome website. And then download the Font Awesome code.
11:18
After you're done with the Font Awesome zip file, move that to our project directory, unzip it, and you will be left with a font-awesome-4.4.3 folder. Go into that folder and then you will see a folder called CSS. From the CSS folder, take the two Font Awesome CSS finds and then move them to the CSS folder of our project.
11:48
Also in the Font Awesome folder, you will find a folder named fonts. Going to that fonts folder and then move all the files from there to our web projects fonts folder.
12:03
Once we are done with that, we can now delete the font-awesome folder. We no longer need the remaining items from that folder.
12:14
Next, download bootstrapsocial.css that we have provided for you. This is a modified version of bootstrapsocial.css file that is available
12:26
on the Internet. We have modified it slightly to introduce additional buttons for Google+ and YouTube.
12:34
Okay, now we have assembled everything but the need to use fonts in our webpage. We are now gonna go to the webpage HTML code, and then add in the icons into our webpage first. We need to include the two CSS files, the font-awesome.min.css, and bootstrap-social.css files into our HTML page. So, let's go ahead and do that first, in the head of the page.

13:10
The first icon that I'm going to introduce is into the home link on our navbar. So I'm going to introduce a home right next to the home link of our navbar. To do that, I'm going to make use of the glyphicons for the home button. So let me replace the home link list item in the navbar and then we'll have a quick look at the cut.
13:39
Taking a quick look at the modification, I see that I am still defining that link as an active link.
13:48
In addition to the home link that I have, I have included this span
13:55
with a class called glyphicon glyphicon-home. That is how we make use of glyphicons in our web page.
14:04
Next we go to the footer of our page. And where we have the address, I'm going to replace the telephone, the fax, and the email links there using fontasm icons. Returning to the base code you see that I have now replaced the telephone, the fax, and email letters from that address by their corresponding photos and icons. So, to use photos and icons, we introduce Aa tag called i, with a class fa fa-phone, to reuse the phone icon. And fax and envelope for the image icons. So, with these changes I have introduced four types of icons into the address in the footer. Our last change is to the social media links that are in the footer. I'm gonna replace those text links with buttons, using Bootstrap Socials tiled Font Awesome icons for the different social media sites. So let me substitute the code in place there and then we'll have a look at the source code. Returning to the source code, you can now see that I have replaced those links with Bootstrap Social paste buttons. How do we do that? Let's go and have a look at the code. Now the div that I have there, I have turned the div into a navbar-nav class. So, we can use navbar stylings in other locations in our webpage, if we wish to. So I'm using that to define this bar containing the icons of all my social media links. So, for each one of those links, let's have a look at the link corresponding to Google+ here. So I am defining a tag a, with the class button. So I am turning that into your button class, and then I am saying button social icon, and button Google+. So this description comes from Bootstrap Social's way of specifying buttons. And then inside there I am using the Font Awesome icon for Google+ enclosed in there. Now the button is styled to be like the Google+'s standard buttons that you would see on many different web pages. Similarly for the remaining social links within that part of the footer. So after these changes let's go and have a look at our web page. Returning to our web page. Now in the navbar, you see that next to the Home link we have an icon which is the glyphicon that we have included there. Now, moving to the footer, we can see that the telephone, fax, and email are now replaced by icons.
17:34
Also, have a look at the links to the social media sites that we have on the right side. They have all now been turned into buttons.
17:51
This completes this exercise. In this exercise we looked at using navbars and using a responsive navbar in our webpage. In addition, we looked at using icon fonts and introduced additional CSS classes to style our webpage.