CMR
INSTITUTE OF
TECHNOLOGY

USN | | | | | | | | | | |

Internal Assessment Test 2 Answer Key– June. 2022									
Sub:	**ADVANCES IN WEB TECHNOLOGIES**					Sub Code:	20MCA41	Branch:	MCA
Date:	4/06/2022	Duration:	90 min's	Max Marks:	50	Sem	IV		

Note : Answer FIVE FULL Questions, choosing ONE full question from each Module

	PART I	MARKS	OBE	
			CO	RBT
1	a) What is Bootstrap? Explain why Bootstrap is preferred for website development. b) Explain Bootstrap file structure and basic HTML Template. **OR**	[10]	CO5	L1
2	a) Explain the typography in Bootstrap. b) Explain Grid sizing in Bootstrap	[10]	CO5	L2
	PART II			
3	a) What is Bootstrap Grid System? Give an example of a basic grid structure in Bootstrap. b) Explain Containers. **OR**	[10]	CO5	L1
4	What are the steps for creating inline or horizontal forms? Explain with examples.	[10]	CO5	L3
	PART III			
5	a) Explain Extended Form Controls with examples. b) Explain Form Control Sizing. **OR**	[10]	CO5	L2
6	a) Explain various ways open() method can be used for XMLHttpRequest. b) Explain send().	[10]	CO2	L2
	PART IV			
7	a) Describe using the Status Property b) Describe using readyState Property **OR**	[10]	CO2	L2
8	a) Explain AJAX application model. b) How do Synchronous and Asynchronous Requests differ?	[10]	CO2	L3
	PART V			
9	Write a program to display a button with the caption "Fetch the message." When you click the button, the text in a file named data.txt should fetch from the server. **OR**	[10]	CO2	L3
10	Explain how to send data to the server using POST method with example.	[10]	CO2	L3

1a) **what is Bootstrap? Explain why Bootstrap is preferred for website development.**

Bootstrap is an open source or Cascading Style Sheets framework from Twitter that is extensively used for front-end prototyping as well as building various mobile friendly, robust websites that are compatible with a

wide range of browsers. It is also used for developing web-based interface by employing various mobile responsive designs. It utilizes JavaScript (that includes jQuery), CSS, as well as HTML. In other words, Bootstrap can be defined as a grid-based responsive framework that is used to develop responsive, mobile-centric projects.

Below are some of the benefits that developers find useful:

1) Quickly create a layout (fixed, fluid, and responsive)

2) Quickly create a form

3) Working grid system

4) Tables

5) Buttons

Reasons behind why Bootstrap best for web designing

1. Development Speed: Bootstrap is easy to use and ready-made blocks of code that enables the web developer to create a new website quickly. No need to do code from scratch. There is numerous ready-made bootstrap theme available online.

2. Massive Support Community: Having large support community web developers can get help when an issue occurs. Bootstrap itself is working continuously on updating and users also put a time to time update. The community on Github consists of over 500 contributors and over 9000 comments.

3. Come predesign with CSS, HTML and JS Elements: On designing a website, there are several elements which we can need to focus on. Heading, table, anchor, buttons, images, and typography each hold ample importance in a compelling website designing. For that, we need to compressive understanding of CSS, JavaScript and HTML codes.

“Bootstrap comes predesign with all these elements”

4. Customization as per project requirement: The Bootstrap customization page comes with commons JavaScript, CSS Elements, and other utilities. Just do as customization as per project specifications.

5. Compatible with Browser: Bootstrap support all major web browser Google Chrome, Mozilla Firefox, Internet Explorer, Opera, and Safari.

6. Bundle Plugin: Bootstrap provides some JavaScript plugin. If we need functionalities like sliders, dropdowns, carousel etc., then just add few lines of JS code and all done on the website.

1b) Explain Bootstrap file structure and basic HTML Template.

Bootstrap File Structure

```
bootstrap/
├── css/
│   ├── bootstrap.css
│   └── bootstrap.min.css
├── js/
│   ├── bootstrap.js
│   └── bootstrap.min.js
└── img/
```

```
│   ├── glyphicons-halflings.png
│   ├── glyphicons-halflings-white.png
└── README.md
```

The Bootstrap download includes three folders: css, js, and img. For simplicity, add these to the root of your project. Minified versions of the CSS and JavaScript are also included. It is not necessary to include both the uncompressed and the minified versions. For the sake of brevity, I use the uncompressed version during development and then switch to the compressed version in production.

Basic HTML Template

Normally, a web project looks something like this:

```
<!DOCTYPE html>
<html>
 <head>
 <title>Bootstrap 101 Template</title>
 </head>
 <body>
 <h1>Hello, world!</h1>
 </body>
</html>
```

With Bootstrap, we include the link to the CSS stylesheet and the JavaScript:

```
<!DOCTYPE html>
<html>
 <head>
 <title>Bootstrap 101 Template</title>
 <link href="css/bootstrap.min.css" rel="stylesheet">
 </head>
 <body>
 <h1>Hello, world!</h1>
 <script src="js/bootstrap.min.js"></script>
 </body>
</html>
```

2a) Explain the typography in Bootstrap.

Headings

All six standard heading levels have been styled in Bootstrap (see Figure 2-1), with the <h1> at 36 pixels tall, and the <h6> down to 12 pixels (for reference, default body text is 14 pixels tall). In addition, to add an inline subheading to any of the headings, simply add <small> around any of the elements and you will get smaller text in a lighter color. In the case of the <h1>, the small text is 24 pixels tall, normal font weight (i.e., not bold), and gray instead of black:

```
h1 small {
 font-size:24px;
 font-weight:normal;
 line-height:1;
 color:#999;
 }
```

Lead Body Copy To add some emphasis to a paragraph, add class="lead" (see Figure 2-2). This will give you larger font size, lighter weight, and a taller line height. This is generally used for the first few paragraphs in a section, but it can really be used anywhere:

Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter swine boudin meatloaf shoulder short ribs cow drumstick beef jowl. Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto. Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump.

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

Lead Body Copy
To add some emphasis to a paragraph, add class="lead" (see Figure 2-2). This will give you larger font size, lighter weight, and a taller line height. This is generally used for the first few paragraphs in a section, but it can really be used anywhere:

** Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter swine boudin meatloaf shoulder short ribs cow drumstick beef jowl. Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto. Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump. **

Lead Example

Bacon ipsum dolor sit amet tri-tip pork loin ball tip frankfurter swine boudin meatloaf shoulder short ribs cow drumstick beef jowl. Meatball chicken sausage tail, kielbasa strip steak turducken venison prosciutto. Chuck filet mignon tri-tip ribeye, flank brisket leberkas. Swine turducken turkey shank, hamburger beef ribs bresaola pastrami venison rump.

Emphasis
In addition to using the <small> tag within headings, as discussed above, you can also use it with body copy. When <small> is applied to body text, the font shrinks to 85% of its original size.
Bold
To add emphasis to text, simply wrap it in a tag. This will add fontweight:bold; to the selected text.
Italics
For italics, wrap your content in the tag. The term "em" derives from the word "emphasis" and is meant to add stress to your text.

Emphasis Classes

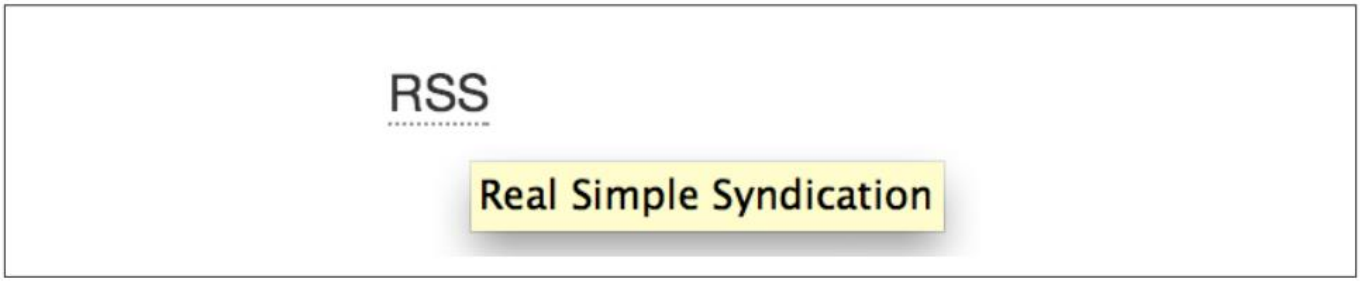
Along with and , Bootstrap offers a few other classes that can be used to provide emphasis (see Figure 2-3). These could be applied to paragraphs or spans:

```
<p class="muted">This content is muted</p>
<p class="text-warning">This content carries a warning class</p>
<p class="text-error">This content carries an error class</p>
<p class="text-info">This content carries an info class</p>
<p class="text-success">This content carries a success class</p>
<p>This content has <em>emphasis</em>, and can be <strong>bold</strong></p>
```

Abbreviations

The HTML <abbr> element provides markup for abbreviations or acronyms, like WWW or HTTP (see Figure 2-4). By marking up abbreviations, you can give useful information to browsers, spell checkers, translation systems, or search engines. Bootstrap styles <abbr> elements with a light dotted border along the bottom and reveals the full text on hover (as long as you add that text to the <abbr> title attribute):

```
<abbr title="Real Simple Syndication">RSS</abbr>
```



Add .initialism to an <abbr> for a slightly smaller font size (see Figure 2-5):
<abbr title="rolling on the floor, laughing out loud">That joke had me ROTFLOL</abbr>

Addresses

Adding `<address>` elements to your page can help screen readers and search engines locate any physical addresses and phone numbers in the text (see Figure 2-6). It can also be used to mark up email addresses. Since the `<address>` defaults to `display:block;` you'll need to use `<br>` tags to add line breaks to the enclosed address text (e.g., to split the street address and city onto separate lines):

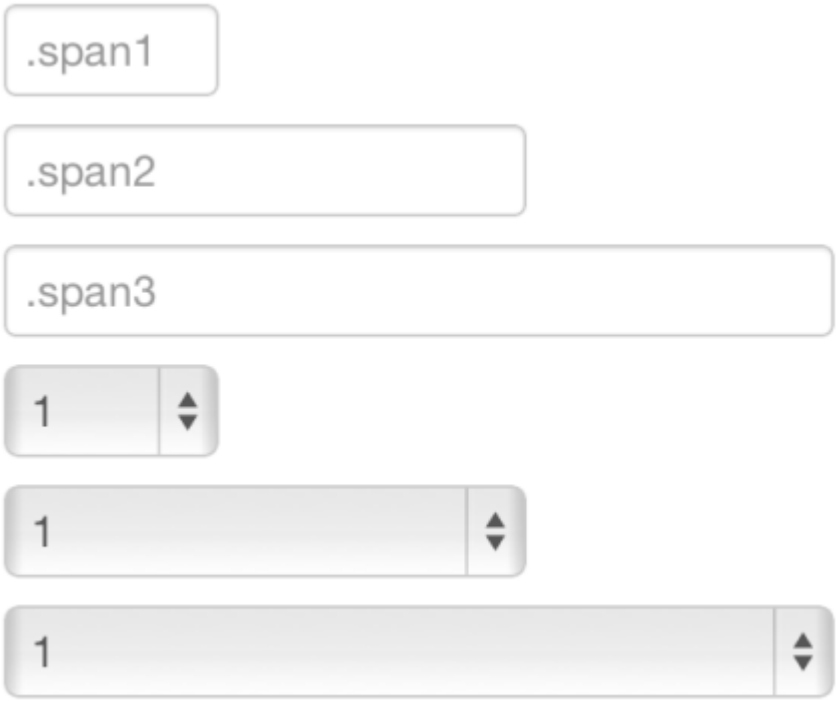
```
<address>
  <strong>O'Reilly Media, Inc.</strong><br>
  1005 Gravenstein HWY North<br>
  Sebastopol, CA 95472<br>
  <abbr title="Phone">P:</abbr> <a href="tel:+17078277000">(707) 827-7000</a>
</address>

<address>
  <strong>Jake Spurlock</strong><br>
  <a href="mailto:#">flast@oreilly.com</a>
</address>
```

2b) Explain Grid sizing in Bootstrap

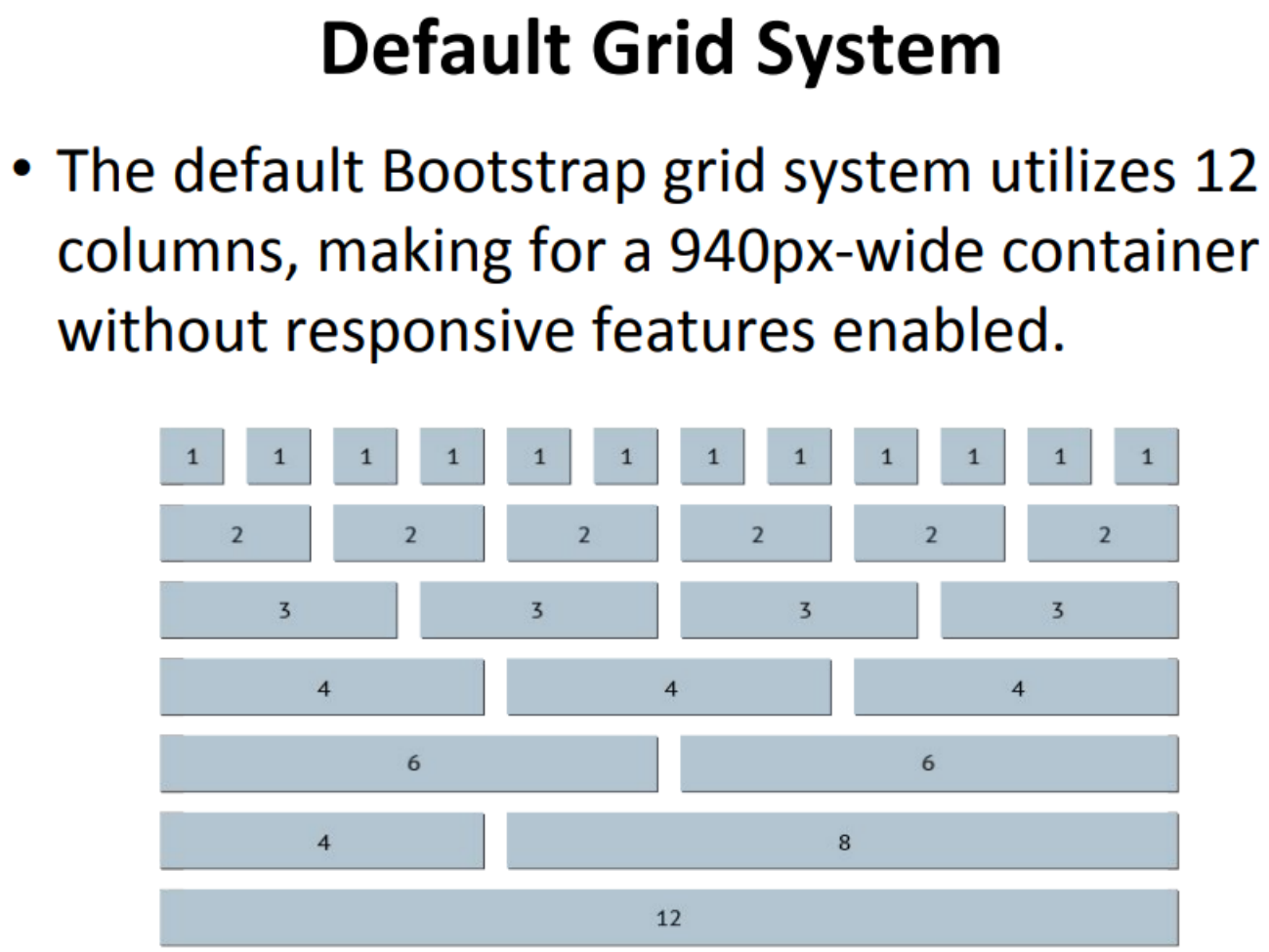
Grid sizing

You can use any `.span` from `.span1` to `.span12` for form control sizing

```
<input class="span1" type="text" placeholder=".span1">
<input class="span2" type="text" placeholder=".span2">
<input class="span3" type="text" placeholder=".span3">
<select class="span1">
  ...
</select>
<select class="span2">
  ...
</select>
<select class="span3">
  ...
</select>
```



3a) What is Bootstrap Grid System? Give an example of a basic grid structure in Bootstrap.



Basic Grid HTML

```
<div class="row">
  <div class="span8">...</div>
  <div class="span4">...</div>
</div>
```

3b) Explain Containers.

Container Layouts
** ... **
** ... **

4) What are the steps for creating inline or horizontal forms? Explain with examples.

nline form
To create a form where all of the elements are inline and labels are alongside, add the
class .form-inline to the <form> tag (see Figure 2-21). To have the label and the input
on the same line, use this inline form code:

```
<form class="form-inline">
 <input type="text" class="input-small" placeholder="Email">
 <input type="password" class="input-small" placeholder="Password">
22 | Chapter 2: Bootstrap CSS
 <label class="checkbox">
 <input type="checkbox"> Remember me
 </label>
 <button type="submit" class="btn">Sign in</button>
</form>
```

Figure 2-21. Inline form
Horizontal form
Bootstrap also comes with a prebaked horizontal form; this one stands apart from the
others not only in the amount of markup, but also in the presentation of the form.

Traditionally you'd use a table to get a form layout like the one shown in Figure 2-22,
but Bootstrap manages to do it without using tables. Even better, if you're using the responsive CSS, the horizontal form will automatically adapt to smaller layouts by stacking the controls vertically.
To create a form that uses the horizontal layout, do the following:
• Add a class of .form-horizontal to the parent <form> element.
• Wrap labels and controls in a with class .control-group.
• Add a class of .control-label to the labels.
• Wrap any associated controls in a with class .controls for proper alignment.
Email Forms | 23

Password

Remember me Sign in

```
    <div class="controls">
      <input type="text" id="inputEmail" placeholder="Email">
    </div>
  </div>
  <div class="control-group">
    <label class="control-label" for="inputPassword">Password</label>
    <div class="controls">
      <input type="password" id="inputPassword" placeholder="Password">
    </div>
  </div>
  <div class="control-group">
    <div class="controls">
      <label class="checkbox">
        <input type="checkbox"> Remember me
      </label>
      <button type="submit" class="btn">Sign in</button>
    </div>
  </div>
</form>
```

5a) **Explain Extended Form Controls with examples.**

Extended Form Controls In addition to the basic form controls listed in the previous section, Bootstrap offers a few other form components to complement the standard HTML form elements; for example, it lets you easily prepend and append content to inputs. Prepended and appended inputs By adding prepended and appended content to an input field, you can add common elements to the user's input (see Figure 2-28). For example, you can add the dollar symbol, the @ for a Twitter username, or anything else that might be common for your application interface. To add extra content before the user input, wrap the prepended input in a
with class .input-prepend. To append input, use the class .inputappend. Then, within that same

, place your extra content inside a with an .add-on class, and place the either before or after the element:

@ .00

Figure 2-28. Prepend and append If you combine both of them, you simply need to add both the .input-prepend and .input-append classes to the parent

```
<div class="input-prepend input-append">
 <span class="add-on">$</span>
 <input class="span2" id="appendedPrependedInput" type="text">
 <span class="add-on">.00</span>
</div>
```

Figure 2-29. Using both the append and prepend

Rather than using a , you can instead use <button> with a class of .btn to attach (surprise!) a button or two to the input (see Figure 2-30):

```
<div class="input-append">
 <input class="span2" id="appendedInputButtons" type="text">
 <button class="btn" type="button">Search</button>
 <button class="btn" type="button">Options</button>
</div>
```

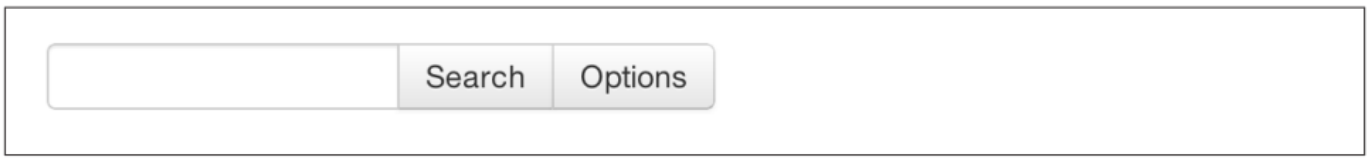


Figure 2-30. Attach multiple buttons to an input

If you are appending a button to a search form, you will get the same nice rounded corners that you would expect (see Figure 2-31):

```
<form class="form-search">
 <div class="input-append">
 <input type="text" class="span2 search-query">
 <button type="submit" class="btn">Search</button>
 </div>
 <div class="input-prepend">
 <button type="submit" class="btn">Search</button>
 <input type="text" class="span2 search-query">
 </div>
</form>
```

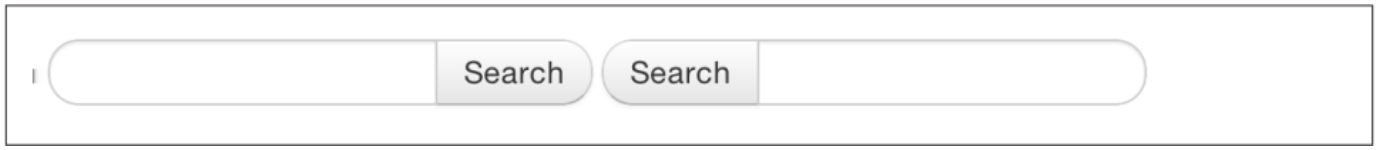
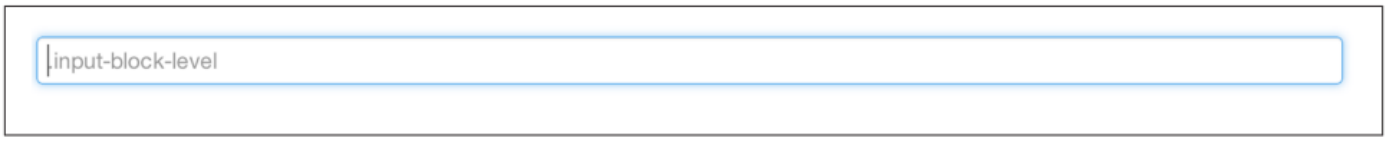


Figure 2-31. Append button to search form

5b) Explain Form Control Sizing.

With the default grid system that is inherent in Bootstrap, you can use the .span* system

for sizing form controls. In addition to the span column-sizing method, you can also use a handful of classes that take a relative approach to sizing. If you want the input to act as a block-level element, you can add .input-block-level and it will be the full width of the container element, as shown in Figure 2-32:
<input class="input-block-level" type="text" placeholder=".input-block-level">



Relative input controls
In addition to using .span* for input sizing, you can also use a few different class names (see Figure 2-33):
<input class="input-mini" type="text" placeholder=".input-mini">
<input class="input-small" type="text" placeholder=".input-small">
<input class="input-medium" type="text" placeholder=".input-medium">
<input class="input-large" type="text" placeholder=".input-large">
<input class="input-xlarge" type="text" placeholder=".input-xlarge">
<input class="input-xxlarge" type="text" placeholder=".input-xxlarge">

6a) Explain various ways open() method can be used for XMLHttpRequest.

Creating Ajax application

- Opening the XMLHttpRequest Object

```
open("method", "URL"[, asyncFlag[, "userName"[, "password"]]])
```

The following table explains what the various arguments mean:

method	The HTTP method used to open the connection, such as GET, POST, PUT, HEAD, or PROPFIND.
URL	The requested URL.
asyncFlag	A Boolean value indicating whether the call is asynchronous. The default is true.
userName	The username of your account.
password	The password used to connect to your account.

```
function getData(dataSource, divID)
{
  if(XMLHttpRequestObject) {

    XMLHttpRequestObject.open("GET", dataSource);
    .
    .
    .
  }
}
```

6b) Explain send().

Creating Ajax application

- Connecting to the Server

```
function getData(dataSource, divID)
{
  if(XMLHttpRequestObject) {
    var obj = document.getElementById(divID);
    XMLHttpRequestObject.open("GET", dataSource);

    XMLHttpRequestObject.onreadystatechange = function()
    {
      if (XMLHttpRequestObject.readyState == 4 &&
        XMLHttpRequestObject.status == 200) {
          obj.innerHTML = XMLHttpRequestObject.responseText;
      }
    }

    XMLHttpRequestObject.send(null);
  }
}
```

7) Describe using readyState Property & Describe using the Status Property

Creating Ajax application

- Getting Ready for the Data Download

```
function getData(dataSource, divID)
{
  if(XMLHttpRequestObject) {

    XMLHttpRequestObject.open("GET", dataSource);

    XMLHttpRequestObject.onreadystatechange = callback()

  }
}
function callback()
{
  .
  .
  .
}
```

```
function getData(dataSource, divID)
{
  if(XMLHttpRequestObject) {

    XMLHttpRequestObject.open("GET", dataSource);

    XMLHttpRequestObject.onreadystatechange = function()
    {
       .
       .
       .
    }

  }
}
```

- Is the download complete? Are we ready to use that data? You can determine that with two properties of the XMLHttpRequest object: readyState and status.

0	Uninitialized
1	Loading
2	Loaded
3	Interactive
4	Complete

200	OK
201	Created
204	No Content
205	Reset Content
206	Partial Content
400	Bad Request
401	Unauthorized
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Timeout
411	Length Required

413	Requested Entity Too Large
414	Requested URL Too Long
415	Unsupported Media Type
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported

Creating Ajax application

- Using the readyState Property

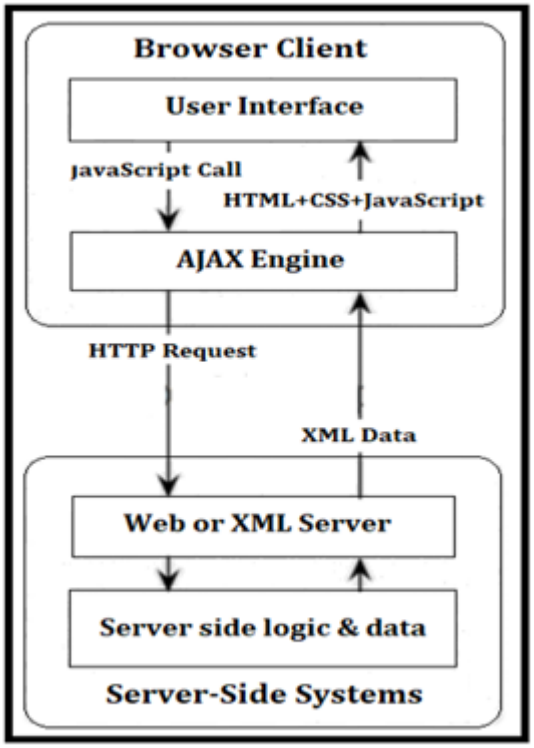
```
function getData(dataSource, divID)
{
  if(XMLHttpRequestObject) {

    XMLHttpRequestObject.open("GET", dataSource);

    XMLHttpRequestObject.onreadystatechange = function()
    {
      if (XMLHttpRequestObject.readyState == 4
        .
        .
        .
      }
    }
```

Creating Ajax application

- Using the status Property

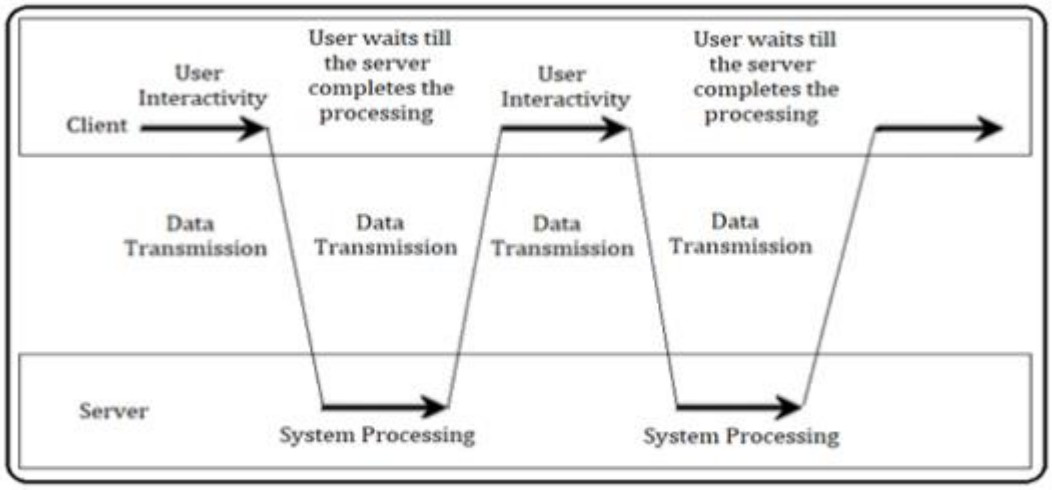
```
function getData(dataSource, divID)
{
  if(XMLHttpRequestObject) {

    XMLHttpRequestObject.open("GET", dataSource);

    XMLHttpRequestObject.onreadystatechange = function()
    {
      if (XMLHttpRequestObject.readyState == 4 &&
        XMLHttpRequestObject.status == 200) {
        .
        .
        .
      }
```

8a) Explain AJAX application model.

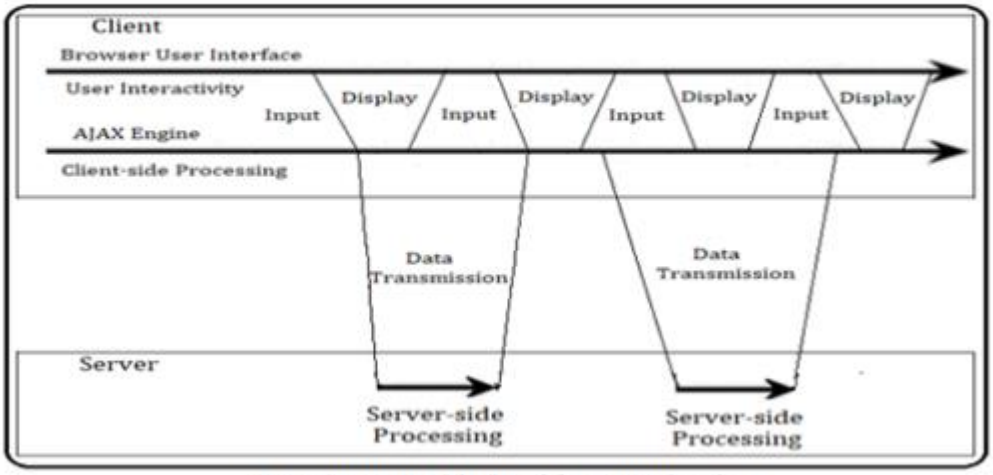


AJAX Web Application Model

8b) How do Synchronous and Asynchronous Requests differ?



Synchronous Mode of Communication



Asynchronous Mode of Communication

9) Write a program to display a button with the caption "Fetch the message." When you click the button, the text in a file named data.txt should fetch from the server.

```
<html>
	<head>
		<title>An Ajax example</title>
		<script language = "javascript">
			var XMLHttpRequestObject = false;
				if (window.XMLHttpRequest) {
					XMLHttpRequestObject = new XMLHttpRequest();
				} else if (window.ActiveXObject) {
					XMLHttpRequestObject = new ActiveXObject("Microsoft.XMLHTTP");
				}
			function getData(dataSource, divID){
				if(XMLHttpRequestObject) {
					var obj = document.getElementById(divID);
					XMLHttpRequestObject.open("GET", dataSource);
					XMLHttpRequestObject.onreadystatechange = function()
					{
						if (XMLHttpRequestObject.readyState == 4 &&
							XMLHttpRequestObject.status == 200) {
							obj.innerHTML = XMLHttpRequestObject.responseText;
						}
					}
					XMLHttpRequestObject.send(null);
				}
			}
		</script>
	</head>
	<body>
		<H1>An Ajax example</H1>
		<form>
			<input type = "button" value = "Fetch the message"
				onclick = "getData('data.txt', 'targetDiv')">
		</form>
		<div id="targetDiv">
			<p>The fetched message will appear here.</p>
		</div>
	</body>
</html>
```

10) Explain how to send data to the server using POST method with example.

```
<html>
	<head>
		<title>An Ajax example</title>
		<script language = "javascript">
			var XMLHttpRequestObject = false;
				if (window.XMLHttpRequest) {
					XMLHttpRequestObject = new XMLHttpRequest();
				} else if (window.ActiveXObject) {
					XMLHttpRequestObject = new
ActiveXObject("Microsoft.XMLHTTP");
				}
			function getData(dataSource, divID, data){
				if(XMLHttpRequestObject) {
					var obj = document.getElementById(divID);
					XMLHttpRequestObject.open("POST", dataSource);
					XMLHttpRequestObject.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded');
					XMLHttpRequestObject.onreadystatechange = function()
					{
						if (XMLHttpRequestObject.readyState == 4 &&
							XMLHttpRequestObject.status == 200) {
							obj.innerHTML =
XMLHttpRequestObject.responseText;
						}
					}


					XMLHttpRequestObject.send("data="+data);
				}
			}
		</script>
	</head>
	<body>
		<H1>An Ajax example</H1>
		<form>
			<input type = "button" value = "Fetch the first message"
				onclick = "getData('dataresponder.php','targetDiv',1)">
				<input type = "button" value = "Fetch the second message"
				onclick = "getData('dataresponder.php','targetDiv',2)">
		</form>
		<div id="targetDiv">
			<p>The fetched message will appear here.</p>
		</div>
	</body>
</html>
<?php
 if ($_GET["data"] == "1") {
 echo 'The server got a value of 1';
 }
 if ($_GET["data"] == "2") {
 echo 'The server got a value of 2';
 }
?>
```