Savitribai Phule Pune University

LABCOURSE III

SECTION I

Web Technologies II

Course Type: DSEC
(COURSECODE: CS-368)

T.Y.B.Sc. (COMPUTER SCIENCE)

SEMESTER-II

Name_____

College Name_____

Roll No._____Division_____

Academic Year_____

Internal Examiner: ----------External Examiner: ----------

BOARD OF STUDIES

1. Dr. Bedekar Smita
2. Dr. Dhole Sanjay
3. Dr. Bharambe Manisha
4. Dr. Ponde Poonam
5. Dr. Sardesai Anjali
6. Dr. Mulay Prashant
7. Dr. Sayyad Razzak
8. Dr. Wani Vilas
9. Dr. Shinde Sahebrao
10. Dr. Kolhe Satish
11. Dr. Patil Ranjeet
12. Dr. Sonar Deepak
13. Dr. Yadav Jyoti
14. Dr. Kumbhojkar Nilesh
15. Dr. Dasari Abhay

Co-ordinators

- **Dr.Mulay P. P.**, Annassaheb Magar College, Hadapsar, Pune

  Member Board of Study,Computer Science SPPU Pune

- **Dr. A.B.Nimbalkar**, Annassaheb Magar College, Hadapsar, Pune

Editor:

Dr. A.B.Nimbalkar, Annassaheb Magar College, Hadapsar, Pune

PREPARED BY

Prof. Shaikh A.M.	H.P.T. Arts & R.Y.K. Science College, Nashik
Prof. Kadam S.A.	BaburaojiGholap College , Sangvi, Pune
Prof. Byagar S.	Indira College of Commerce and Science, Pune
Prof. Malani P.S.	K.K.Wagh Arts,Commerce,Science & Computer Science College, Nashik

About The WorkBook

Objectives –

1. The scope of the course.
2. Bringing uniformity in the way course is conducted across different Colleges.
3. Continuous assessment of the students.
4. Providing ready references for students while working in the lab.

How to use this book?

This book is mandatory for the completion of the laboratory course. It is a

Measure of the performance of the student in the laboratory for the entire duration of the course.

Instructions to the students

1) Students should carry this book during practical sessions of Computer Science.
2) Students should maintain separate journal for the source code and outputs.
3) Students should read the topics mentioned in reading section of this Book before coming for practical.
4) Students should solve all exercises which are selected by Practical in-charge as a part of journal activity.
5) Students will be assessed for each exercise on a scale of 5

1	Note done	0
2	Incomplete	1
3	Late complete	2
4	Needs improvement	3
5	Complete	4
6	Well-done	5

Instructions to the practical in-charge

1. Explain the assignment and related concepts in around ten minutes using white board if required or by demonstrating the software.
2. Choose appropriate problems to be solved by student.
3. After a student completes a specific set, the instructor has to verify the outputs and sign in the provided space after the activity.
4. Ensure that the students use good programming practices.
5. You should evaluate each assignment carried out by a student on a scale of 5 as specified above ticking appropriate box.
6. The value should also be entered on assignment completion page of respected lab course.

PHP Semester – II

Assignment Completion Sheet

Sr. No.	Assignment Name	Marks(out of 5)	Sign
1	Cookies and Session		
2	XML documents and DOM		
3	JAVASCRIPT and jQuery		
4	AJAX		
5	PHP Framework CODEIGNITER		
	Total out of 25		
	Total out of 05		

This is to certify that Mr/Ms _____

University Exam Seat Number_____ has successfully completed the course work for *CS - 368 Programing in Web Technology II* and has scored _____Marks out of 15.

Practical In-Charge **Head of Department**

Internal Examiner **External Examiner**

Assignment No 1. Cookies and Session

State Management in PHP

HTTP is a stateless protocol which means every user request is processed independently and it has nothing to do with the requests processed before it. Hence there is no way to store or send any user specific details using HTTP protocol.

But in modern applications, user accounts are created and user specific information is shown to different users, for which we need to have knowledge about who the user (or what he/she wants to see etc.) is on every webpage.

PHP provides for two different techniques for state management of your web application, they are:

1. Server Side State Management
2. Client Side Server Management

Server Side State Management

In server side state management we store user specific information required to identify the user on the server. And this information is available on every webpage.

In PHP we have **Sessions** for server side state management. PHP session variable is used to store user session information like username, user id etc. and the same can be retrieved by accessing the session variable on any webpage of the web application until the session variable is destroyed.

Client Side State Management

In client side state management the user specific information is stored at the client side i.e. in the bowser. Again, this information is available on all the webpages of the web application.

In PHP we have **Cookies** for client side state management. Cookies are saved in the browser with some data and expiry date(till when the cookie is valid).

One drawback of using cookie for state management is the user can easily access the cookie stored in their browser and can even delete it.

PHP Cookies

Cookie is a small piece of information stored as a file in the user's browser by the web server. Once created, cookie is sent to the web server as header information with every HTTP request.

User can use cookie to save any data but it should not exceed **1K(1024 bytes)** in size.

Real world Use of Cookies

1. To store user information like when he/she visited, what pages were visited on the website etc, so that next time the user visits your website you can provide a better user experience.
2. To store basic website specific information to know this is not the first visit of user.
3. You can use cookies to store number of visits or view counter.

Types of Cookies

There are two types of cookies :

1. **Session Cookie**: This type of cookies are temporary and are expire as soon as the session ends or the browser is closed.
2. **Persistent Cookie**: To make a cookie persistent we must provide it with an expiration time. Then the cookie will only expire after the given expiration time, until then it will be a valid cookie.

Creating a Cookie in PHP

In PHP user can create/set a cookie using the `setcookie()` function.

syntax :

setcookie(name, value, expire, path, domain, secure)

The first argument which defines the **name** of the cookie is mandatory, rest all are optional arguments.

Argument	**What is it for?**
name	Used to specify the name of the cookie. It is a mandatory argument. Name of the cookie must be a string.
value	Used to store any value in the cookie. It is generally saved as a pair with name. For example, name is *userid* and value is *7007*, the userid for any user.
expire	Used to set the expiration time for a cookie. if you do not provide any value, the cookie will be treated as a session cookie and will expire when the browser is closed.

path	Used to set a web URL in the cookie. If set, the cookie will be accessible only from that URL. To make a cookie accessible through a domain, set '/' as cookie path.
domain	The domain of your web application. It can be used to limit access of cookie for sub-domains.
secure	If you set this to 1, then the cookie will be available and sent only over HTTPS connection.

So if we want to create a cookie to store the name of the user who visited your website, and set an expiration time of a week, then we can do it like this,

```
<?php
setcookie("username", "iamabhishek", time()+60*60*24*7);
?>
```

To access a stored cookie we use the $_COOKIE global variable, and can use the isset() method to check whether the cookie is set or not.

Example :

```
<?php
// set the cookie
setcookie ("username", "iamabhishek", time()+60*60*24*7);
?>
<html>
   <body>

   <?php
   // check if the cookie exists
   if(isset($_COOKIE["username"]))
   {
      echo "Cookie set with value: ".$_COOKIE["username"];
   }
   else
   {
      echo "cookie not set!";
   }
   ?>

   </body>
```

```
</html>
```

So by providing the name of the cookie inside the square brackets with the global variable $_COOKIE[ ] we can access the cookie.

NOTE: setcookie() function should be placed before the starting HTML tag(<html>).

Updating Cookie in PHP

To update/modify a cookie, simply set it again. For example, if we want to update the username stored in the cookie created above, we can do it using setcookie() method again,

```
<?php

// updating the cookie

setcookie("username", "IamNOTabhishek", time()+60*60*24*7);

?>

<html>

   <body>

   <?php

   // check if the cookie exists

   if(isset($_COOKIE["username"]))

   {

      echo "Cookie set with value: ".$_COOKIE["username"];

   }

   else

   {

    echo "cookie not set!";

   }

   ?>

   </body>

</html>
```

To update the **value** of **username** cookie from *iamabhishek* to *iamNOTabhishek*.

Delete a Cookie in PHP

To delete/remove a cookie, we need to expire the cookie, which can be done by updating the cookie using the setcookie() function with expiration date in past.

```
<?php

// updating the cookie

setcookie("username", "iamNOTabhishek", time() - 3600);

?>

<html>

   <body>

      <?php

      echo "cookie username is deleted!";

   ?>

  </body>

</html>
```

PHP Sessions for State Management

To store information accessible accross web pages, we use sessions. **Session** is not stored on the user browser like cookie, hence it is a more secure option.

As we know HTTP is a stateless protocol, if a user visits a webpage and perform some action, there is no way to remember what he did when the user navigates to the next webpage.

For example, when you log into your facebook account, by providing your email address and password, until and unless you logout, the web application remembers who you are and display what your friends are posting and liking on your News Feed, you can update your profile, send someone message, join a group etc, this is accomplished by **Session**.

When a user logs into their account on any web application, a session is created for them, and in the session their *username* or *userid* or some other unique identifier is stored, which is then used on the consecutive webpages to show information specific to that user. On logout, the session is destroyed.

Session is not limited by any size limit, you can store any information in the session, irrespective of its size.

Real world Use of Session

1. Web applications which require a user to login, use session to store user information, so that on every webpage related information can be displayed to the user.
2. In eCommerce web stores, shopping cart is generally saved as part of session.

Start a Session in PHP

In PHP we can start a session by using the session_start() function. And data is stored in the session using session variable, which can be assigned different values using global variable $_SESSION

Using the function session_start() we initialize the session, in which we can store information using the session variable $_SESSION.

Example :

first_page.php

```
<?php
// start the session
session_start();
// set the session variable
$_SESSION["username"] = "iamabhishek";
$_SESSION["userid"] = "1";
?>
<html>
   <body>
   <?php
   echo "Session variable is set.";
  ?>
   <a href="second_page.php">Go to Second Page</a>
   </body>
</html>
```

NOTE: The function session_start() should be the first statement of the page, before any HTML tag.

Getting PHP Session Variable Values

In the code above, we have started a session and set two session variables. Above webpage will also have a link to navigate to Second page **second_page.php**.

Below is the code for **second_page.php**, in which we fetch values from the session variable which are set in the **first_page.php**.

```
<?php
// start the session
```

```
session_start();

// get the session variable values

$username = $_SESSION["username"];

$userid = $_SESSION["userid"];

?>

<html>

   <body>

   <?php

   echo "Username is: ".$username."<br/>";

   echo "User id is: ".$userid;

   ?>

   </body>

</html>
```

Output :

Username is: iamabhishek

User id is: 1

session_start () function is used to initialize a new session and to fetch the ongoing session(if already started), and then, using the $_SESSION global variable, we can either set new values into the session or get the saved values.

If there are too many values stored in the session, and you don't know which one do you want to get, you can use the below code to print all the current session variable data.

Example :

```
<?php

// start the session

session_start();

?>

<html>

   <body>

  <?php
```

```
print_r ($_SESSION);
?>
</body>
</html>
```

Output :

Array (

[username] => iamabhishek,

[userid] => 1

)

Update Session Variable in PHP

To update any value stored in the session variable, start the session by calling session_start() function and then simply overwrite the value to update session variable.

```
<?php
// start the session
session_start ();
// update the session variable values
$_SESSION ["userid"] = "1111";
?>
<html>
<body>
<?php
echo "Username is: ".$username."<br/>";
echo "User id is: ".$userid;
?>
</body>
</html>
```

Username is: iamabhishek

User id is: 1111

We just updated the **value** of **userid** in the session variable from *1* to *1111*.

Destroy a Session in PHP

To clean the session variable or to remove all the stored values from the session variable we can use the function session_unset () and to detroy the session, we use session_destroy () function.

```
<?php
// start the session
session_start ();
?>
<html>
   <body>
   <?php
   // clean the session variable
   session_unset ();
   // destroy the session
   session_destroy ();
   ?>
   </body>
</html>
```

Set A

1. Write a PHP script to keep track of number of times the web page has been access.
   [Use session and cookies]
2. Write a PHP script to change the preferences of your web page like font style, font size, font color, background color using cookie. Display selected setting on next web page and actual implementation (with new settings) on third page.

Set B

1. Write a PHP script to accept username and password. If in the first three chances, username and password entered is correct then display second form with “Welcome message” otherwise display error message. [Use Session]
2. Write a PHP script to accept Employee details (Eno, Ename, Address) on first page. On second page accept earning (Basic, DA, HRA). On third page print Employee information (Eno, Ename, Address, Basic, DA, HRA, Total) [Use Session]

Set C

1. Crete a form to accept customer information (Name, Addr, MobNo). Once the customer information is accepted, accept product information in the next form (ProdName, Qty,Rate). Generate the bill for the customer in the next form. Bill should contain the customer information and the information of the products entered.

Signature of the instructor : _____ Date:_____

0 : Not Done ☐ 1 : Incomplete ☐ 2: Late Complete ☐

3 Needs Improvement ☐ 4: Complete ☐ 5 Well Done ☐

Assignment No 2. XML documents and DOM

Link CSS to XML :

Syntax:

```
<?xml-stylesheet type="text/css" href="university.css"?>
```

university.css :

```
uname
{
        color:black;
        font-family:copperplate Gothoic Light;
        font-size:16 pt;
        font:bold;
}
city
{
      color:yellow;
      font-family:Arial;
      font-size:12 pt;
      font:bold;
}
rank
{
      color:yellow;
      font-family:Arial;
      font-size:16 pt;
      font:bold;
}
```

university.xml :

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/css" href="university.css"?>
<university>
        <univ>
                <uname>Pune University</uname>
                <city>Pune</city>
                <rank>2</rank>
        </univ>
        <univ>
                <uname>Kolhapur University</uname>
                <city>Kolhapur</city>
                <rank>4</rank>
        </univ>
</university>
```

Create xml file through PHP code :

```
<?php
        $str =<<<XML
      <?xml version="1.0"encoding="ISO-8859-1"?>
      <BookStore>
                  <Books>
                        <PHP>
                        <Title>Programming in PHP</Title>
                        <Publication>O'RELLY</Publication>
                  </PHP>
                  <PHP>
                        <Title>Beginners PHP</Title>
                        <Publication>2000</Publication>
                  </PHP>
                  </Books>
      </BookStore>
XML;
      $fname="bookstore.xml";
      $fp = fopen($fname,"w");
      fwrite($fp, $str);
      fclose($fp);
      echo "created filename is:".$fname;
?>
```

What is DOM?

The W3C DOM provides a standard set of objects for HTML and XML documents, and a standard interface for accessing and manipulating them.

The W3C DOM is separated into different parts (Core, XML, and HTML) and different levels (DOM Level 1/2/3):

- Core DOM - defines a standard set of objects for any structured document
- XML DOM - defines a standard set of objects for XML documents
- HTML DOM - defines a standard set of objects for HTML documents

XML Parsing

To read and update - create and manipulate - an XML document, you will need an XML parser.

There are two basic types of XML parsers:

- **Tree-based parser**: This parser transforms an XML document into a tree structure. It analyzes the whole document, and provides access to the tree elements

  eg. DOM(Document object Model)

- **Event-based parser :** Views an XML document as a series of events. When a specific event occurs, it calls a function to handle it.

  Eg. Expat parser, Simple XML Parser

The DOM parser is an tree-based parser.

Parse XML document using DOM

We want to initialize the XML parser, load the xml, and output it:

1. Initialize the XML parser (DOMDocument())
2. Load the xml file (load())
3. Read the contents from the XML file (saveXML())

DomDocument Example :

Create XML File " Book.xml "

```
<?xml version="1.0" encoding="UTF-8"?>

<BookInfo>

        <book>

                <bookno>1</bookno>

                <bookname>JAVA</bookname>

                <authorname> Balguru Swami</authorname>

                <price>250</price>

                <year>2006</year>

        </book>

        <book>

                <bookno>2</bookno>

                <bookname>PHP</bookname>

                <authorname> S.Kadam</authorname>

                <price>350</price>

                <year>2009</year>

        </book>

        <book>

                <bookno>3</bookno>

                <bookname>C</bookname>
```

```
        <authorname> Denis Ritchie</authorname>
        <price>500</price>
        <year>1971</year>
    </book>
    <book>
        <bookno>4</bookno>
        <bookname>C++</bookname>
        <authorname> Bjarne Straustrup</authorname>
        <price>400</price>
        <year>1994</year>
    <book>
        <bookno>5</bookno>
        <bookname>DATABASE</bookname>
        <authorname> Rogers Presman</authorname>
        <price>700</price>
        <year>2001</year>
    </book>
</BookInfo>
```

PHP file Book.php

Example : to read name of books from book.xml using Dom Document

```
<?php
    $dom=new DomDocument();
    $dom->load("book.xml");
    echo "<h2>Name of books</h2>";
    $bname=$dom->getElementsByTagName("bookname");
    foreach($bname as $b)
    {
        echo "<b>$b->textContent<b><br><br>";
    }
```

?>

What is SimpleXML?

SimpleXML is new in PHP 5. It is an easy way of getting an element's attributes and text, if you know the XML document's layout.

Compared to DOM or the Expat parser, SimpleXML just takes a few lines of code to read text data from an element.

SimpleXML converts the XML document into an object, like this:

- Elements - Are converted to single attributes of the SimpleXMLElement object. When there's more than one element on one level, they're placed inside an array
- Attributes - Are accessed using associative arrays, where an index corresponds to the attribute name
- Element Data - Text data from elements are converted to strings. If an element has more than one text node, they will be arranged in the order they are found

SimpleXML is fast and easy to use when performing basic tasks like:

- Reading XML files
- Extracting data from XML strings
- Editing text nodes or attributes

PHP SimpleXML Functions

PHP: indicates the earliest version of PHP that supports the function.

Function	Description	PHP
__construct()	Creates a new SimpleXMLElement object	5
addAttribute()	Adds an attribute to the SimpleXML element	5
addChild()	Adds a child element the SimpleXML element	5
asXML()	Gets an XML string from a SimpleXML element	5
attributes()	Gets a SimpleXML element's attributes	5
children()	Gets the children of a specified node	5
getDocNamespaces()	Gets the namespaces of an XML document	5
getName()	Gets the name of a SimpleXML element	5
getNamespaces()	Gets the namespaces from XML data	5
registerXPathNamespace()	Creates a namespace context for the next XPath query	5
simplexml_import_dom()	Gets a SimpleXMLElement object from a DOM node	5
simplexml_load_file()	Gets a SimpleXMLElement object from an XML document	5
simplexml_load_string()	Gets a SimpleXMLElement object from an XML string	5
xpath()	Runs an XPath query on XML data	5

Simple XML Example:

1. Load the XML file
2. Get the name of the first element
3. Create a loop that will trigger on each child node, using the children() function
4. Output the element name and data for each child node

To read contents of xml file using SimpleXML

Create following Employee.xml file

Employee.php

```
<?xml version="1.0" encoding="UTF-8"?>

<empdetails>

	<employee>

		<empno>1</empno>

		<empname>Sagar</empname>

		<salary>20000</salary>

		<designation>Clerk</designation>

	</employee>

	<employee>

		<empno>2</empno>

		<empname>Shubham</empname>

		<salary>30000</salary>

		<designation>Manager</designation>

	</employee>

	<employee>

		<empno>3</empno>

		<empname>Abhishekh</empname>

		<salary>500000</salary>

		<designation>CEO</designation>

	</employee>

</empdetails>
```

Read contents of Employee.xml through php using simpleXML

Employee.php :

```
<?php
        $xml=simplexml_load_file("Employee.xml");
        foreach($xml->employee as $emp)
        {
                echo "Employee No = $emp->empno “.”<br>”;
                echo “ Employee Name = $emp->empname “.”<br>”;
                echo “Employee Salary = $emp->salary “.”<br>”;
                echo “Employee Designation= $emp->designation".”<br>”;
        }
?>
```

Set A :

1) Write a script to create XML file named “Item.xml”

```
<Item>
        <ItemName> ……. </ItemName>
        <ItemPrice> …….</ItemPrice>
        <Quantity> …… </Quantity>
</Item>
```

Store the details of 5 Items of different Types

2) Link “ Item. Xml” file to the CSS style sheet and get well formatted output as given below

i) ItemName :

```
Color : red;
Font-family : copperplate Gothic Light;
Font-size : 16pt;
Font :bold;
```

ii) ItemPrice and Quantity

```
color:yellow;
font-family:Arial;
font-size:12 pt;
font:bold;
```

3) Write a PHP script to generate an XML file in the following format in PHP.

```
<?xml version="1.0" encoding="UTF-8"?>
<BookInfo>
    <book>
        <bookno>1</bookno>
        <bookname>JAVA</bookname>
        <authorname> Balguru Swami</authorname>
        <price>250</price>
        <year>2006</year>
    </book>
    <book>
        <bookno>2</bookno>
```

```
            <bookname>C</bookname>
            <authorname> Denis Ritchie</authorname>
            <price>500</price>
            <year>1971</year>
        </book>
</BookInfo>
```

Set B

1. Write PHP script to read above created "book.xml" file into simpleXML object. Display attributes and elements .(Hint L simple_xml_load_file() function)

2. Write a PHP script to read "Movie.xml" file and print all MovieTitle and ActorName of file using DOMDocument Parser. "Movie.xml" file should contain following information with at least 5 records with values.

```
MovieInfo
    MovieNo, MovieTitle, ActorName , ReleaseYear
```

Set C

1. Create a XML file which gives details of movies available in "Movie CD Store " from following categories
   a) Classical
   b) Horror
   c) Action

   Elements in each category are in the following format

```
<Category>
    <MovieTitle> ……. </ MovieTitle >
    <ActorName> …….</ActorName>
    <ReleaseYear> …… </ReleaseYear>
</Category>
```

   Save the file with name "movies.xml"

Signature of the instructor: _____ Date:_____

0 : Not Done ☐ 1 : Incomplete ☐ 2: Late Complete ☐

3 Needs Improvement ☐ 4: Complete ☐ 5: Well Done ☐

ASSIGNMENT NO. : 4 JAVASCRIPT and jquery

Javascript is basically designed to create interactivity with HTML pages. It enables you to read and change the content of HTML controls and also enables you to load a specific page depending upon the client's request. It helps you to do certain validations on client side.

Data types: Number, String, Boolean, Undefined, NULL

Variables: Value of a variable can change during the script. No need to declare a variable.

Operators: Operators available in Javascript are same as that of PHP, C programming.

Conditional statements and loops: In Javascript , the syntax of Conditional statements and loops are same as that of PHP, C programming.

Javascript Object
Objects in Javascript are divided into three categories.
1. Built-in objects
2. Browser Objects
3. User-defined objects.

Built-in Objects:
Array, string, math, Date are commonly used built-in objects.
Array: Using Keyword **new,** you can create an instance of the object.
For ex. Varmyarray=new Aarray();
String: String object is used to manipulate a stored piece of text.
Var text =”PHP and Javascript”
Document.write(text.length);
Math: This object is used to perform common mathematical tasks .
Javascript provides eight mathematical values that can be accessed from the
math object.
These are
Math.E
Math.PI
Math.SQRT2
Math.SQRT 1_2
Math.LN2
Math.LN10
Math.LOG2E
Math.LOG10E
Ex. Round method is used to round a number
Document.write(Math.round(4.7))
Date: This object works with date and time
VarmyDate=new Date()
myDate.setFullYear(2015, 0, 20)

Browser Objects :
BOM is a collection of objects that interact with the browser window. These objects include the **Window object, history object, location object, navigator object, screen object and document object.**The window object method is the top object in BOM hierarchy. The window object is used to move, resize windows , create a new

windows. Window object is also used to create dialogue boxes such as alert boxes. Some commonly used **methods** of window object are **open, close, confirm, alert, prompt etc.**

Document Object Model

The document Object Model (DOM) is a tree- based representation of a document. The DOM was created by **World Wide Web Consortium(w3c)** for XML and HTML/XHTML. The DOM provides a set of objects for representing the structure of the document, as well as for accessing those objects.

Methods available in DOM for accessing objects:

1) getElementById() method:

This method returns the element with the specified ID.
Refer the examples given below.

2) getElementByTagName() method:

This method returns all elements with the specified tag name.
Refer the examples given below.

1) simplejavascript example

```
<!DOCTYPE html>
<html>
<body>
<script type=”text/javascript”>
Document.write(“Hello World!”)
</script>
</body>
</html>
```

2) javascript example using 'alert box ' window object.

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p>My first paragraph.</p>
<script>
window.alert(5 + 6);
</script>
</body>
</html>
```

3)javascript example using 'document.write method'.

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p>My first paragraph.</p>
<button onclick="document.write(5 + 6)">Try it</button>
</body>
</html>
```

Popup boxes : JavaScript has three kind of popup boxes: Alert box, Confirm box, and Prompt box.

Alert Box: An alert box is often used if you want to make sure information comes through to the user.

When an alert box pops up, the user will have to click "OK" to proceed.

Syntax

window.alert("sometext");

The window.alert() method can be written without the window prefix.

Example

```
<!DOCTYPE html>
<html>
<body>
<p>Click the button to display an alert box:</p>
<button onclick="myFunction()">Try it</button>
<script>
functionmyFunction() {
        alert("I am an alert box!");
        }
</script>
</body>
</html>
```

Confirm Box :A confirm box is often used if you want the user to verify or accept something. When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed. If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

Syntax

window.confirm("sometext");

The window.confirm() method can be written without the window prefix.

Example

```
var r = confirm("Press a button");
if (r == true) {
        x = "You pressed OK!";
        } else {
        x = "You pressed Cancel!";
        }
```

Prompt Box :A prompt box is often used if you want the user to input a value before entering a page. When a prompt box pops up, the user will have to click either "OK" or
"Cancel" to proceed after entering an input value. If the user clicks "OK" the box returns the input value. If the user clicks "Cancel" the box returns null.

Syntax

window.prompt("sometext","defaultText");

The window.prompt() method can be written without the window prefix.

Example

```
var person = prompt("Please enter your name", "Harry Potter");
if (person != null) {
        document.getElementById("demo").innerHTML =
        "Hello " + person + "! How are you today?";
        }
```

HTML Events

An HTML event can be something the browser does, or something a user
does.

Here are some examples of HTML events:

An HTML web page has finished loading

An HTML input field was changed

An HTML button was clicked

Often, when events happen, you may want to do something.

JavaScript lets you execute code when events are detected.
HTML allows event handler attributes, with JavaScript code, to be added to HTML elements.
With single quotes:
<some-HTML-element some-event='some JavaScript'>
With double quotes:
<some-HTML-element some-event="some JavaScript">

Common HTML Events : Here is a list of some common HTML events:
Event Description
Onchange : An HTML element has been changed
Onclick :The user clicks an HTML element
Onmouseover : The user moves the mouse over an HTML element
Onmouseout : The user moves the mouse away from an HTML element
Onkeydown : The user pushes a keyboard key
Onload : The browser has finished loading the page

In the following example, an **onclick** attribute (with code), is added to a button element:
Example

```
<!DOCTYPE html>
<html>
<body>
<button onclick="getElementById('demo').innerHTML=Date()">The time is?</button>
<p id="demo"></p>
</body>
</html>
```

4)javascript example using 'document.getElementById(id) method.'

```
<!DOCTYPE html>
<html>
<body>
<h1>My First Web Page</h1>
<p>My First Paragraph</p>
<p id="demo"></p>
<script>
document.getElementById("demo").innerHTML = 5 + 6;
</script>
</body>
</html>
```

This is the easiest way to create a JavaScript Object. Using an object literal, you both define and create an object in one statement.
An object literal is a list of name: value pairs (like age:50) inside curly braces {}.
The following example creates a new JavaScript object with four properties:

```
<!DOCTYPE html>
<html>
<body>
<p>Creating a JavaScript Object.</p>
<p id="demo"></p>
<script>
```

```
var person = {
firstName : "John",
lastName : "Doe",
age : 50,
eyeColor : "blue"
};
document.getElementById("demo").innerHTML =
person.firstName + " is " + person.age + " years old.";
</script>
</body>
</html>
```

Using the JavaScript Keyword new

The following example also creates a new JavaScript object with four properties:

```
<!DOCTYPE html>
<html>
<body>
<p id="demo"></p>
<script>
var person = new Object();
person.firstName = "John";
person.lastName = "Doe";
person.age = 50;
person.eyeColor = "blue";
document.getElementById("demo").innerHTML =
person.firstName + " is " + person.age + " years old.";
</script>
</body>
</html>
```

Using an Object Constructor

The examples above are limited in many situations. They only create a single object.

Sometimes we like to have an "object type" that can be used to create many objects of one type. The standard way to create an "object type" is to use an object constructor function:

Example

```
function person(first, last, age, eye) {
this.firstName = first;
this.lastName = last;
this.age = age;
this.eyeColor = eye;
}
Var myFather = new person("John", "Doe", 50, "blue");
Var myMother = new person("Sally", "Rally", 48, "green");
```

JavaScript Function Definitions

JavaScript functions are defined with the function keyword.You can use a function declaration or a function expression.

Function Declarations

Earlier in this tutorial, you learned that functions are declared with the following syntax:

```
functionfunctionName(parameters) {
```

code to be executed
}

Declared functions are not executed immediately. They are "saved for later use", and will be executed later, when they are invoked (called upon).

Example

```
<!DOCTYPE html>
<html>
<body>
<p>This example calls a function which performs a calculation,
and returns the result:</p>
<p id="demo"></p>
<script>
functionmyFunction(a, b) {
return a * b;
}
document.getElementById("demo").innerHTML =
myFunction(4, 3);
</script>
</body>
</html>
```

jQuery

What is jQuery?

jQuery is a lightweight, "write less, do more", JavaScript library. The purpose of jQuery is to make it much easier to use JavaScript on your website. jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.
jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.
The jQuery library contains following features:

- HTML/DOM manipulation
- CSS manipulation
- HTML event methods
- Effects and animations
- AJAX
- Utilities

The jQuery library is a single JavaScript file, and you reference it with the HTML <script> tag (notice that the <script> tag should be inside the <head> section):

```
<head>
<script src="jquery-3.5.1.min.js"></script>
</head>
```

jQuery CDN

If you don't want to download and host jQuery yourself, you can include it from a CDN (Content Delivery Network).Google is an example of someone who host jQuery:

Google CDN:

```
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
</head>
```

But this will execute only when you are **online.** So if you have to run your scripts **offline** you have to use the library option, ie

```
<head>
<script src="jquery-3.5.1.min.js"></script>
</head>
```

jQuery Syntax

The jQuery syntax is tailor-made for selecting HTML elements and performing some action on the element(s).

Basic syntax is: $(selector).action()

- A $ sign to define/access jQuery
- A (selector) to "query (or find)" HTML elements
- A jQuery action() to be performed on the element(s)

Examples:

$(this).hide() - hides the current element.

$("p").hide() - hides all <p> elements.

$(".test").hide() - hides all elements with class="test".

$("#test").hide() - hides the element with id="test".

jQuery Selectors

jQuery selectors allow you to select and manipulate HTML element(s). jQuery selectors are used to "find" (or select) HTML elements based on their name, id, classes, types, attributes, values of attributes and much more. It's based on the existing CSS Selectors, and in addition, it has some own custom selectors.

All selectors in jQuery start with the dollar sign and parentheses: $().

The element Selector

The jQuery element selector selects elements based on the element name.
You can select all <p> elements on a page like this:
$("p")

Example

When a user clicks on a button, all <p> elements will be hidden:

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-3.5.1.min.js">
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
</script>
</head>
<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me to hide paragraphs</button>
</body>
</html>
```

The #id Selector

The jQuery #*id* selector uses the id attribute of an HTML tag to find the specific element.
An id should be unique within a page, so you should use the #id selector when you want to find a single, unique element.
To find an element with a specific id, write a hash character, followed by the id of the HTML element:
$("#test")
When a user clicks on a button, the element with id="test" will be hidden:
Example

```
$(document).ready(function(){
  $("button").click(function(){
    $("#test").hide();
  });
});
```

The .class Selector

The jQuery .class selector finds elements with a specific class.
To find elements with a specific class, write a period character, followed by the name of the class:
$(".test")

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-3.5.1.min.js"></script>
```

```
<script>
$(document).ready(function(){
  $("button").click(function(){
    $(".test").hide();
  });
});
</script>
</head>
<body>
<h2 class="test">This is a heading</h2>
<p class="test">This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body>
</html>
```

More Examples of jQuery Selectors

Syntax	Description
$("*")	Selects all elements
$(this)	Selects the current HTML element
$("p.intro")	Selects all <p> elements with class="intro"
$("p:first")	Selects the first <p> element
$("ulli:first")	Selects the first element of the first
$("ulli:first-child")	Selects the first element of every
$("[href]")	Selects all elements with an href attribute
$("a[target='_blank']")	Selects all <a> elements with a target attribute value equal to "_blank"

$("a[target!='_blank']")	Selects all <a> elements with a target attribute value NOT equal to "_blank"
$(":button")	Selects all <button> elements and <input> elements of type="button"
$("tr:even")	Selects all even <tr> elements
$("tr:odd")	Selects all odd <tr> elements

$("*")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("*").hide();
  });
});
</script>
</head>
<body>
<h2>This is a heading</h2>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<button>Click me</button>
</body> </html>
```

$(this)

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $(this).hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>
```

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Click me</button>

</body>
</html>
```

$("p.intro")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p.intro").hide();
  });
});
</script>
</head>
<body>

<h2 class="intro">This is a heading</h2>

<p class="intro">This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Click me</button>

</body>
</html>
```

$("p:first")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p:first").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

```
<button>Click me</button>

</body>
</html>
```

$("ulli:first")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("ulli:first").hide();
  });
});
</script>
</head>
<body>

<p>List 1:</p>
<ul>
<li>Coffee</li>
<li>Milk</li>
<li>Tea</li>
</ul>

<p>List 2:</p>
<ul>
<li>Coffee</li>
<li>Milk</li>
<li>Tea</li>
</ul>

<button>Click me</button>

</body>
</html>
```

$("ulli:first-child")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("ulli:first-child").hide();
  });
});
</script>
```

```
</head>
<body>

<p>List 1:</p>
<ul>
<li>Coffee</li>
<li>Milk</li>
<li>Tea</li>
</ul>

<p>List 2:</p>
<ul>
<li>Coffee</li>
<li>Milk</li>
<li>Tea</li>
</ul>

<button>Click me</button>

</body>
</html>
```

$("[href]")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("[href]").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<p><a href="https://www.w3schools.com/html/">HTML Tutorial</a></p>
<p><a href="https://www.w3schools.com/css/">CSS Tutorial</a></p>

<button>Click me</button>

</body>

</html>
```

$("a[target='_blank']")

```
<!DOCTYPE html>

<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("a[target='_blank']").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<p><a href="https://www.w3schools.com/html/" target="_blank">HTML Tutorial</a></p>
<p><a href="https://www.w3schools.com/css/">CSS Tutorial</a></p>

<button>Click me</button>

</body>
</html>
```

$("a[target!='_blank']")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("a[target!='_blank']").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<p><a href="https://www.w3schools.com/html/" target="_blank">HTML Tutorial</a></p>
<p><a href="https://www.w3schools.com/css/">CSS Tutorial</a></p>

<button>Click me</button>
</body></html>
```

$(":button")

```
<!DOCTYPE html>
<html>
<head>
<script src=“jquery-3.5.1.min.js”></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $(":button").hide();
  });
});
</script>
</head>
<body>

<h2>This is a heading</h2>

<p>This is a paragraph.</p>
<p>This is another paragraph.</p>

<button>Click me</button>

</body>
</html>
$("tr:even")
$("tr:odd")
```

DOM Manipulation Methods in jQuery

jQuery provides various methods to add, edit or delete DOM element(s) in the HTML page.
The following table lists some important methods to add/remove new DOM elements.

Method	Description
append()	Inserts content to the end of element(s) which is specified by a selector.
before()	Inserts content (new or existing DOM elements) before an element(s) which is specified by a selector.
after()	Inserts content (new or existing DOM elements) after an element(s) which is specified by a selector.
prepend()	Insert content at the beginning of an element(s) specified by a selector.
remove()	Removes element(s) from DOM which is specified by selector.
replaceAll()	Replace target element(s) with specified element.
wrap()	Wrap an HTML structure around each element which is specified by selector.

jQuery after() Method

The jQueryafter() method inserts content (new or existing DOM elements) after target element(s) which is specified by a selector.
Syntax:
$('selector expression').after('content');
First of all, specify a selector to get the reference of target element(s) after which you want to add the content and then call after() method. Pass the content string as a parameter. Content string can be any valid HTML element.

```
<!DOCTYPE html>
<html>
<head>
	<script src="jquery-3.5.1.min.js"></script>
	<script>
		$(document).ready(function () {

			$('#div1').after('<div style="background-color:yellow"> New div </div>');

		});
	</script>
	<style>
		div{
			border: 1px solid;
			background-color:red;
			margin: 2px 0 2px 0;
		}
	</style>
</head>
<body>
	<h1>Demo: jQueryafter() method </h1>

	<div id="div1">div 1
	</div>

	<div id="div2">div 2
	</div>
</body>
</html>
```

jQuery before() Method

The jQuerybefore() method inserts content (new or existing DOM elements) before target element(s) which is specified by a selector.
Syntax:
$('selector expression').before('content');
Specify a selector to get the reference of target element(s) before which you want to add the content and then call before() method. Pass the content string that can be any valid HTML element as parameter.

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-3.5.1.min.js"></script>
<script>
```

```
    $(document).ready(function () {

      $('#div1').before('<div style="background-color:yellow"> New div </div>');

    });
</script>
        <style>
                div{
                        border: 1px solid;
                        background-color:red;
                        margin: 2px 0 2px 0;
                }
        </style>
</head>
<body>
        <h1>Demo: jQuerybefore() method </h1>
        <div id="div1">div 1
        </div>
```

jQuery append() Method

The jQueryappend() method inserts content to the end of target element(s) which is specified by a selector.
Syntax:
$('selector expression').append('content');
First specify a selector expression to get the reference of an element(s) to which you want to append content, then call append() method and pass content string as a parameter.

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-3.5.1.min.js"></script>
<script>
    $(document).ready(function () {

      $('p').append('World!');

    });
</script>
</head>
<body>
        <h1>Demo: jQueryappend() method </h1>
        <p>Hello </p>
</body>
</html>
```

jQuery prepend() Method

The jQueryprepend() method inserts content at the beginning of an element(s) specified by a selector.
Syntax:
$('selector expression').prepend('content');
First specify a selector expression to get the reference of an element(s) to which you want to prepend the content, then call prepend() method and pass content string as a parameter.

```
<!DOCTYPE html>
<html>
<head>
```

```
        <script src="jquery-3.5.1.min.js"></script>
<script>
    $(document).ready(function () {

      $('div').prepend('<p>This is prepended paragraph</p>');

    });
</script>
</head>
<body>
      <h1>Demo: jQueryprepend() method </h1>
      <div>
            <label>This is div.</label>
      </div>
</body>
</html>
```

jQuery remove() Method

The jQueryremove() method removes element(s) as specified by a selector.
Syntax:
$('selector expression').remove();
First specify a selector expression to get the reference of an element(s) which you want to remove from the document and then call remove() method.

```
<!DOCTYPE html>
<html>
<head>
      <script src="jquery-3.5.1.min.js"></script>
      <script>
    $(document).ready(function () {

      $('label').remove();

    });
</script>
</head>
<body>
      <h1>Demo: jQueryremove() method </h1>
      <div>This is div.
            <label>This is label.</label>
      </div>
</body>
</html>
```

jQueryreplaceAll() Method

The jQueryreplaceAll() method replaces all target elements with specified element(s).
Syntax:
$('content string').replaceAll('selector expression');
Here, syntax is different. First specify a content string as replacement element(s) and then call replaceAll() method with selector expression to specify a target element(s).

```
<!DOCTYPE html>
```

```
<html>
<head>
        <script src="jquery-3.5.1.min.js"></script>  <script>
      $(document).ready(function () {

        $('<span>This is span</span>').replaceAll('p');

      });
</script>
</head>
<body>
        <h1>Demo: jQueryreplaceAll() method </h1>
        <div>
                <p>This is paragraph.</p>
        </div>
        <p>This is another paragraph.</p>
</body>
</html>
```

jQuery wrap() Method

The jQuerywrap() method wrap each target element with specified content element.
Syntax:
$('selector expression').wrap('content string');
Specify a selector to get target elements and then call wrap method and pass content string to wrap the target element(s).

```
<!DOCTYPE html>
<html>
<head>
        <script src="jquery-3.5.1.min.js"></script>
        <script>
      $(document).ready(function () {

        $('span').wrap('<p></p>');

      });
</script>
</head>
<body>
        <h1>Demo: jQuerywrap() method </h1>
        <div>
                <span>This is span.</span>
        </div>
        <span>This is span.</span>
</body>
</html>
```

Example of appending in paragraph text and in unordered list in a given HTML document using jQuery selectors.

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-3.5.1.min.js"></script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("p").append(" <b>Appended text</b>.");
  });

  $("#btn2").click(function(){
    $("ul").append("<li>Appended item</li>");
  });
});
</script>
</head>
<body>
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
<ul>
<li>List item 1</li>
<li>List item 2</li>
<li>List item 3</li>
</ul>
<button id="btn1">Append text</button>
<button id="btn2">Append list items</button>
</body>
</html>
```

Inserting Text before and after a paragraph element.

```
<!DOCTYPE html>
<html>
<head>
<script src="jquery-3.5.1.min.js"></script>
<script>
$(document).ready(function(){
  $("#btn1").click(function(){
    $("p").before("<b>Before</b>");
  });

  $("#btn2").click(function(){
    $("p").after("<i>After</i>");
  });
});
</script>
</head>
<body>

<p>This is a paragraph for trial. Text will be inserted….
      Before and after this!!!!
</p>
```

```
<button id="btn1">Insert before</button>
<button id="btn2">Insert after</button>

</body>
</html>
```

SET A:

1) Write a javascript to display message 'Exams are near, have you started preparing for?' using alert, prompt and confirm boxes. Accept proper input from user and display messages accordingly.
2) Add or append in paragraph text and also in the numbered(ordered) list in a given HTML document using jQuery selectors.
   [Hint : Use Append( ) method]

SET B:

1) Write a javascript function to validate username and password for a membership form.
2) To insert text before and after an image using jQuery.
   [Hint : Use before( ) and after( )]

SET C:

1) Write a Javascript program to accept name of student, change font color to red, font size to 18 if student name is present otherwise on clicking on empty text box display image which changes its size (Use onblur, onload, onmousehover, onmouseclick, onmouseup)
2) Remove div section elements after clicking on button using jQuery.
   [Hint : Use #id selector]

Signature of the instructor:---------- Date:----------

Assignment Evaluation

0:Not Done ☐	1: Incomplete ☐	2: Late Complete ☐
3:Needs Improvement ☐	4:Complete ☐	5:Well Done ☐

AJAX = Asynchronous JavaScript and XML.

AJAX is not a new programming language, but a new way to use existing standards.

AJAX is the art of exchanging data with a server, and updating parts of a web page - without reloading the whole page.

XMLHttpRequest is a JavaScript object capable of calling the server and capturing its response. It is used to send HTTP or HTTPS requests to a web server and load the server response data back into the script.

Create an XMLHttpRequest Object

All modern browsers (IE, Firefox, Chrome, Safari, and Opera) have a built-in XMLHttpRequest object.

Syntax for creating an XMLHttpRequest object:

xmlhttp=new XMLHttpRequest();

When a request to a server is sent, we want to perform some actions basedon the response.

The onreadystatechange event is triggered every time the readyState changes.

The readyState property holds the status of the XMLHttpRequest.

Three important properties of the XMLHttpRequest object:

Property	Description
onreadystatechange	Stores a function (or the name of a function) to be called automatically each time the readyState property changes
readyState	Holds the status of the XMLHttpRequest. Changes from0 to 4: 0: request not initialized 1: server connection established 2: request received 3: processing request 4: request finished and response is ready
Status	200: "OK" 404: Page not found

In the onreadystatechange event, we specify what will happen when the server response is ready to be processed.

The onreadystatechange event is triggered five times (0-4), one time for each change in readyState.

To get the response from a server, use the responseText or responseXML property of the XMLHttpRequest object. The responseText property returns the response as a string.

To send a request to a server, we use the open() and send() methods of the XMLHttpRequest object:

Method	Description
open(*method,url,async)*	Specifies the type of request, the URL, and if the request should be handled asynchronously or not. *method*: the type of request: GET or POST *url*: the location of the file on the server *async*: true (asynchronous) or false (synchronous)
send(*string*) send()	Sends the request to the server (used for GET) Sends the request to the server (used for POST)

Example 1:

Write a Ajax program to search Student Name according to the character typed and display list using array.

Create name.php file

```
<?php
    $n=$_GET['n'];
    $a=array();
    $a[]="sonal";
    $a[]="sanjay";
    $a[]="anant";
    $a[]=”anushka”
    $a[]=”kajal”
    echo "List of Names=<br>";
    foreach($a as $v)
    {
```

```
		$s=substr($v,0,strlen($n));
		if($s===$n)
			echo "$v<br>";
	}
?>
```

Create Ajax.php file:

```
<html><body>
<script type="text/javascript">
function display()
{
var x= new XMLHttpRequest();
var n= document.getElementById('n').value;
x.open("GET", "name.php?n="+n, true);
x.send( );
x.onreadystatechange = function()
{
	if(x.readyState == 4 && x.status==200)
	{
document.getElementById("show").innerHTML = x.responseText;
	}
}
}
</script>
Search Box: <input type="text" name='n' id='n' onkeyup="display()"> <br>
<h1 id="show"> </h1>
</body>
</html>
```

Output :

Run ajax.php file on browser enter character in textbox and see output

Example 2:

Write AJAX program to print movie details by selecting an Actor's name. Create tables Movie and Actor with 1:M cardinality as follows:

Movie (mno,mname, release_year)

Actor (ano,aname)

[Use PostgreSQL]

Solution :

Ajax file : create majax.php

```
<html>
<body>
<script type="text/javascript">
function display()
{

var x= new XMLHttpRequest();
var n= document.getElementById('n').value;
x.open("GET", "actor.php?n="+n, true);
x.send( );
x.onreadystatechange = function()
{
        if(x.readyState == 4 && x.status==200)
        {
document.getElementById("show").innerHTML = x.responseText;
        }
}
}
</script>
Actor Name: <input type="text" name='n' id='n'"> <br>
```

```
<button onclick="display()" > Print</button>
<h1 id="show"> </h1>
</body>
</html>
```

Create php file : actor.php

```
<?php
	$aname=$_POST['aname'];
	$con=pg_connect("host=localhost dbname=root user=root") or die("Could not connect");
	$qry="select mname,release_year from movie,actor where movie.mno=actor.mno and aname=$aname";
	$rs=pg_query($con,$qry) or die("Unable to execute query");
	if($rs!=null)
	{
		echo"<table border=0>";
		echo"<tr><td>Movie Name</td><td>Release Year</td></tr>";
		while($row=pg_fetch_row($rs))
		{
			echo"<tr>";
			echo"<td>".$row[0]."</td>";
			echo"<td>".$row[1]."</td>";
			echo"</tr>";
		}
		echo"</table>";
	}
	else
		echo"NO records found";
	pg_close($con);
?>
```

Output :

Run majax.php file on browser enter actor name in text box and see output

Set A

1. Write AJAX program to read contact.dat file and print the contents of the file in a tabular format when the user clicks on print button. Contact.dat file should contain srno, name, residence number, mobile number, Address. [Enter at least 3 record in contact.dat file]
2. Write AJAX program where the user is requested to write his or her name in a text box, and the server keeps sending back responses while the user is typing. If the user name is not entered then the message displayed will be, "Stranger, please tell me your name!". If the name is Rohit, Virat, Dhoni, Ashwin or Harbhajan , the server responds with "Hello, master <user name>!". If the name is anything else, the message will be "<user name>, I don't know you!".

Set B

1. Create TEACHER table as follows TEACHER(tno, tname, qualification, salary). Write Ajax program to select a teachers name and print the selected teachers details.
2. Write Ajax program to print Order details by selecting a Customer's name. Create table Customer and Order as follows with 1 : M cardinality CUSTOMER (cno, cname, city) and ORDER(ono, odate, shipping address)

Set C

1. Write Ajax program to fetch suggestions when is user is typing in a textbox. (eg like google suggestions. Hint create array of suggestions and matching string will be displayed)
2. Write Ajax program to get book details from XML file when user select a book name. Create XML file for storing details of book(title, author, year, price).

Signature of the instructor:---------- Date:----------

Assignment Evaluation

0: Not Done ☐ 2: Late Complete ☐ 4: Complete ☐

1: Incomplete ☐ 3: Needs Improvement ☐ 5: Well Done ☐

(Designed by: Ms. Sarita Byagar, Indira College of Commerce and Science, Pune)

ASSIGNMENT 5: PHP Framework CODEIGNITER

CodeIgniter is a powerful PHP framework with a very small footprint, built for developers who need a simple and elegant toolkit to create full-featured web applications.

CodeIgniter is based on the **Model-View-Controller (MVC) development pattern**. MVC is a software approach that separates application logic from presentation. In practice, it permits your web pages to contain minimal scripting since the presentation is separate from the PHP scripting.

- The **Model** represents your data structures. Typically, your model classes will contain functions that help you retrieve, insert and update information in your database.
- The **View** is information that is being presented to a user. A View will normally be a web page, but in CodeIgniter, a view can also be a page fragment like a header or footer. It can also be an RSS page, or any other type of "page".
- The **Controller** serves as an intermediary between the Model, the View, and any other resources needed to process the HTTP request and generate a web page.

Pre-requisites for installing CodeIgniter

CodeIgniter requires PHP version 5.4 or newer and MySQL 5.1 or newer with mysqli and pdo drivers so make sure that your system meets CodeIgniter system requirements.

Steps for Installation and Configuration (for Ubuntu OS)

Note: The steps may differ for other Operating system.

1. **Download codeigniter**

   The next thing you need to do is to navigate to your server's directory root and download the current version of CodeIgniter.

   cd /var/www/wget

   https://github.com/bcit-ci/CodeIgniter/archive/3.0.1.zip

2. **Unzip the archive and set up directory ownership**

   Unzip the archive you have downloaded using the command:

   unzip 3.0.1.zip

Rename the directory to be more user-friendly:

```
mv /var/www/CodeIgniter-3.0.1 /var/www/codeigniter
```

Change the ownership of that directory and files:

```
chown -R www-data: /var/www/codeigniter
```

3. Edit Apache virtual host file

Now edit the virtual host file and change the document root to point to /var/www/codeigniter.

```
nano /etc/apache2/sites-enabled/000-default
```

Edit the following lines to match your needs:

```
<VirtualHost *:80>
 ServerAdmin admin@yourdomain.com
 DocumentRoot /var/www/codeigniter/
 ServerName yourdomain.com
 ServerAlias www.yourdomain.com
 <Directory /var/www/codeigniter/>
Options Indexes FollowSymLinks MultiViews
AllowOverride All
Order allow,deny
allow from all
 </Directory>
 ErrorLog /var/log/httpd/yourdomain.com-error_log
 CustomLog /var/log/httpd/yourdomain.com-access_log common
</VirtualHost>
```

4. Restart Apache web server

Save and close the file once you are done. Then restart Apache by executing the command below:

```
service apache2 restart
```

5. Create new database

Now, create a MySQL database for CodeIgniter:

```
mysql> CREATE DATABASE codeigniter_db;
mysql> GRANT ALL PRIVILEGES on codeigniter_db.* to
'codeigniter_user'@'localhost' identified by 'YoUrPaS$w0rD';
mysql> FLUSH PRIVILEGES;
mysql> exit
```

Once you create the MySQL database you need to change the database connectivity senttigs to the settings needed to access your newly created database.

6. Configure database config file

Edit the following file:

```
nano /var/www/codeigniter/application/config/database.php
```

Find the following lines:

```
$db['default'] = array(
        'dsn'   => '',
        'hostname' => 'localhost',
        'username' => '',
        'password' => '',
        'database' => '',
);
```

Here, you need to enter your database connectivity settings. Save the file and close it.

Also, you need to edit the following file:

```
nano /var/www/codeigniter/application/config/config.php
```

and find the following line to set your base URL:

```
$config['base_url'] = 'http://yourdomain.com';
```

Once you enter your domain name, save the file and close it.

7. Access CodeIgniter via web browser

Next thing you need to do is to open your web browser, enter your domain name in the search field and you will be able to access your CodeIgniter installation.
Further instructions about how to use this PHP web application framework you can find at:

http://www.codeigniter.com/user_guide/

Set A

1. Create a CSS file to apply the following styling for an HTML document. Background color: blue,

   H1

   {

   Color : red,

   Font-family : Verdana,

   Font-size : 8

   }

   P

   {

   Color : green,

   Font-family : Monospace,

   Font-size :10

   }

2. Add a Javascript file in codeigniter. The javascript code should check whether a number is positive or negative.

Set B

1. Create a table student having attributes (rollno, name, class). Assume appropriate data types for the attributes. Using Codeigniter , connect to the database and insert minimum 5 records in it.
2. For the above table student, display all its records using Codeigniter.

Set C

1. Create a form to accept personal details of customer (name, age, address). Once the personal information is accepted, on the next page accept order details such as (product name, quantity). Display the personal details and order details on the third page. (Use cookies)

2. Write a PHP script to accept Employee details (Eno, Ename, address) on first page. On second page accept earning (Basic, Da, HRA). On third page print Employee information(ENO,Ename, Address, BASIC, DA, HRA, TOTAL) (Use Session)

Signature of the instructor: ---------- Date: ----------

Assignment Evaluation

0:Not Done		2: Late Complete		4:Complete	
1: Incomplete		3: Needs Improvement		5: Well Done	

(Designed by: Ms. Sarita Byagar, Indira College of Commerce and Science, Pune)