CHAPTER 4

UNIT IV

Creating and Validating forms

Syllabus

4.1 Creating a webpage using GUI components, Browser Role-GET and POST methods, Server role.

4.2 Form controls: text box, text area, radio button, check box, list, buttons

4.3 Working with multiple forms: A web page having many forms, A forms having multiple submit buttons.

4.4 Web page validation

4.5 Cookies: Use of cookies, attributes of cookies, create cookies, modify cookies value and delete cookies

4.6 Session: use of session, start session, get session variables, destroy session

4.7 sending e-mail.

4.1 Creating a Webpage Using GUI Components

– To run any code on server or to get connected with the PHP on server we need to set up Web pages in a specific fashion.

– HTML controls like text fields, check boxes, radio buttons etc. enclosed in HTML form are used to collect data from user.

– We have to indicate location in the form where the data from the controls will be sent.

– Two methods **"get"** and **"post"** mentioned in form are commonly used to send data from HTML controls to PHP script on server.

– URL is used to specify the location, which helps browser understand where to send the data on server mentioned in the **"action"** attribute of a form.

– PHP script handles both displaying of HTML controls and then reading of data from HTML controls when the user clicks on the Submit button.

– Other than html controls like text fields and check boxes we also need submit button, because only by clicking on submit button data from the form is sent to the PHP script on the server.

– **For example**, a simple form with a text field and submit button, text field is used to get name from users and we have to make the browser know where to send the name data when the user clicks on Submit button.

Example :

```
HTML Form
<html>
<head>
       <title> Sample PHP Program </title>
</head>
<body>
       <form method=”get” action=php_readdata.php>
        <input name=”username” type=”text”>
        <input type=”submit” value=”Submit”>
</form
</body>
</html>
```

4.1.1 Browser Role - GET and POST Methods

Q. Describe term : get and post methods.

Browser used one of the two HTTP (Hypertext Transfer Protocol) methods - GET and POST to communicate with the server. Both methods GET and POST are used to pass the information differently to the server.

4.1.1(A) GET Method

– This is the built in PHP super global array variable that is used to get values submitted via HTTP GET method.
– Data in GET method is sent as URL parameters that are usually strings of name and value pairs separated by ampersands (&).
– URL with GET data look like this :

  http://www.abc.com/dataread.php?name=ram&age=20.
– The name and age in the URL are the GET parameters; ram and 20 are the value of those parameters.
– More than one parameter=value can be embedded in the URL by concatenating with ampersands (&).
– Only simple text data can only be sent through GET method.
– Sensitive information such as the username and password can’t be sent through GET method as it will be visible in the query string and it can store in browser memory as visited page
– Length of the URL is limited as it assigns data to a server environment variable
– Super global variable $_GET is used to access all the information sent through an HTML form using get method.

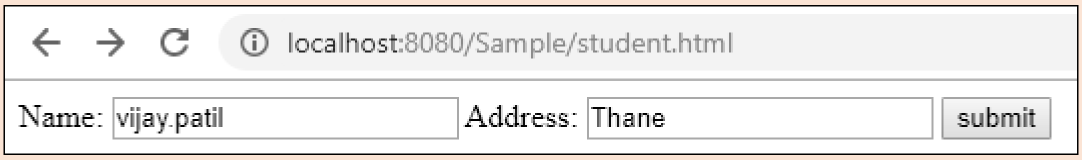
Syntax :

```
<?php
$_GET['variable_name'];
?>
Where "$_GET[...]" is the PHP array and "'variable_name'" is the URL variable name.
```

Example :

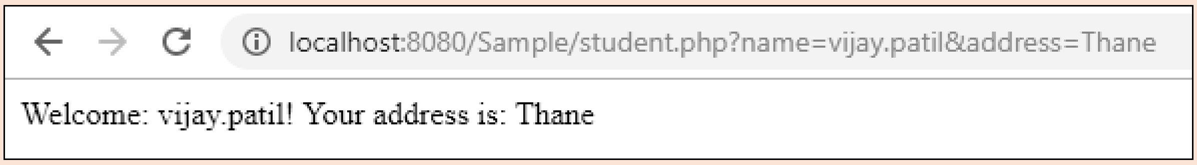
```
Student.html
<html>
<body>
<form action="student.php" method="get">
Name: <input type="text" name="name">
Address: <input type="text" name="address">
<input type="submit">
</form>

</body>
</html>
Student.php
<html>
<body>
Welcome: <?php echo $_GET["name"]; ?>!
Your address is: <?php echo $_GET["address"]; ?>
</body>
</html>
```

Output :



```
localhost:8080/Sample/student.html
Name: vijay.patil Address: Thane submit
```

When the user clicks on the "Submit button", the URL will be something like this :



```
localhost:8080/Sample/student.php?name=vijay.patil&address=Thane
Welcome: vijay.patil! Your address is: Thane
```

Advantages of using the GET Method

- Since the data sent by the GET method are displayed in the URL, it is possible to bookmark the page with specific query string values.
- GET requests can be cached and GET requests remain in the browser history.
- GET requests can be bookmarked.

Disadvantages of using the GET Method

- The GET method is not suitable for passing sensitive information such as the username and password, because these are fully visible in the URL query string as well as potentially stored in the client browser's memory as a visited page.
- Because the GET method assigns data to a server environment variable, the length of the URL is limited. So, there is a limitation for the total data to be sent.

4.1.1(B) POST Method

- This is the built in PHP super global array variable that is used to get values submitted via HTTP POST method.
- Data in POST method is sent to the server in a form of a package in a separate communication with the processing script.
- User entered Information is never visible in the URL query string as it is visible in GET.
- The POST method can be used to send the much larger amount of data and text data as well as binary data (uploading a file).
- Data sent by the POST method is not visible in the URL, so it is not possible to bookmark the page with specific query.
- Super global variable $_POST is used to access all the information sent through an HTML form using the post method.
- A good example of using post method is when submitting login details to the server.

Syntax :

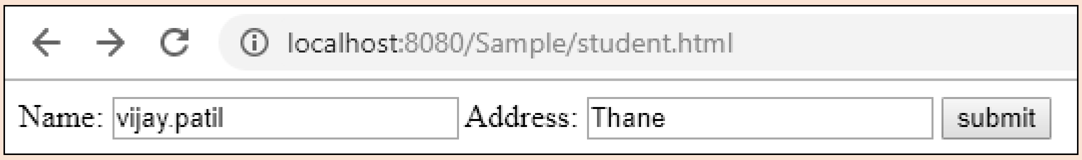
```
<?php
 $_POST['variable_name'];
?>


Where "$_POST[…]" is the PHP array and "'variable_name'" is the URL variable name.
```

Example :

```
Student.html
<html>
<body>
<form action="student.php" method="post">
Name: <input type="text" name="name" id="name">
Address: <input type="text" name="address" id="address">
<input type="submit" value="submit">
```

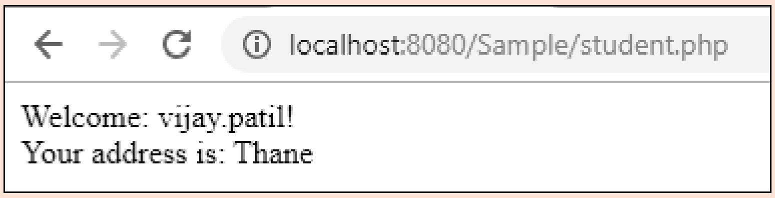
```
</form>
</body>
</html>
student.php
<html>
<body>
Welcome: <?php echo $_POST["name"]; ?>!
<br>
Your address is: <?php echo $_POST["address"]; ?>
</body>
</html>
```

Output :


localhost:8080/Sample/student.html

Name: vijay.patil Address: Thane

When the user clicks on the "Submit button", the URL will be something like this :


localhost:8080/Sample/student.php

Welcome: vijay.patil!
Your address is: Thane

Advantages of using POST Method

– It is more secure than GET because user-entered information is never visible in the URL query string or in the server logs.

– There is a much larger limit on the amount of data that can be passed and one can send text data as well as binary data (uploading a file) using POST.

Disadvantages of using the POST Method

– Since the data sent by the POST method is not visible in the URL, so it is not possible to bookmark the page with specific query.

– POST requests are never cached.

– POST requests do not remain in the browser history.

Difference between get and post methods in PHP :

Q. What is the difference between GET and POST methods in PHP ?

Get method	Post method
Information sent from a form with the GET method is visible to everyone (all variable names and values are displayed in the URL).	Information sent from a form with the POST method is invisible to others (all names/values are embedded within the body of the HTTP request)
GET has limits on the amount of information to send. The limitation is about 2048 characters.	Post has no limits on the amount of information to send.
$_GET is an array of variables passed to the current script via the URL parameters.	$_POST is an array of variables passed to the current script via the HTTP POST method.
Can be bookmarked	Cannot be bookmarked
Can be cached	Not cached
Parameters remain in browser history	Parameters are not saved in browser history
Only ASCII characters allowed	No restrictions. Binary data is also allowed
GET is less secure compared to POST because data sent is part of the URL. So it's saved in browser history and server logs in plaintext.	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
GET method should not be used when sending passwords or other sensitive information.	POST method used when sending passwords or other sensitive information.
Easier to hack for script kiddies	More difficult to hack

4.1.2 Server Role

Q. What are the role of server in PHP ?

– A client (browser) submits an HTTP request using GET and POST method to the server.

– Server processes the request and returns a result, in the form of a response to the client.

– The response from the server contains status information about the request and requested content.

– Using PHP you can create dynamic web sites.

– Here student.html has a static behavior because it delivers same content always. But student.php has a dynamic behavior because the content it delivers changes according to the time of the day. You can improve this file and have different greetings for different time periods.

– When PHP interpreter reads a file, it only processes lines between <?php and ?> tags. It outputs rest of the lines without any processing.

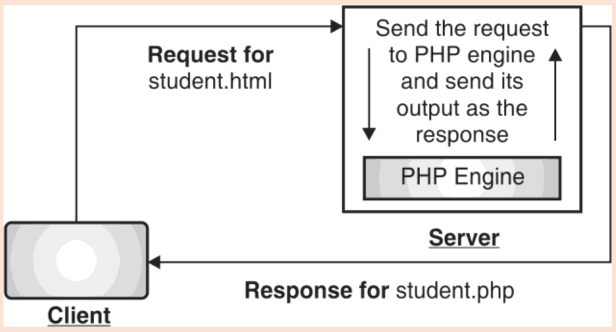


Fig. 4.1.1 : Web request that involves PHP

PHP Script Life cycle consists of following steps :

1. Client Request
2. Server Parse Request and Send Response
3. Client interpret Responded Text/HTML etc…

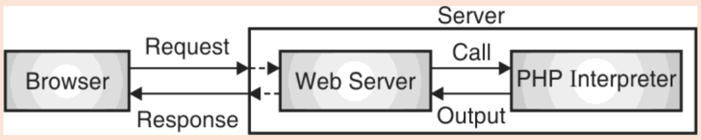


Fig. 4.1.2

– We always start with a browser making a request for a web page. This request is going to hit the web server. The web server will then analyze it and determine what to do with it.

– If the web server determines that the request is for a PHP file (often index.php), it will pass that file to the PHP interpreter. The PHP interpreter will read the PHP file, parse it (and other included files) and then execute it. Once the PHP interpreter finishes executing the PHP file, it will return an output. The web server will take that output and send it back as a response to the browser.

For Example

a.html

```
<html>
<head>
<title> PHP Server </title>
</head>
<body>
<form method="post" action="sample.php">
<label>Mail id:</label>
<input type="text"  name="mailid" id=" mailid">
<input type="submit" value="Submit">
</form>
</body>
```

```
</html>

sample.php    plz add this

<?php
if(isset($_POST["mailid"])){
   echo "<p>Mail id : " . $_POST["mailid"] . "</p>";
}
?>
```

In above example POST method is used to send data to the server as it is POST method data will be invisible to everyone and embedded within the body of the HTTP request. On the server side a super global variable $_POST is used by PHP to create an associative array with an access key ($_POST['name as key']). Data is assessed using key and based on collected data processing takes place and the result is provided as a response to the client.

Embedding PHP in HTML

When we mix PHP and HTML content in a PHP file, we call it as "**embedding PHP in HTML**". We could achieve the same behavior with following PHP file, which contains only PHP content. Embedding PHP blocks in HTML let us get the help of PHP only where it is necessary. We can add rest of the lines to the output by just keeping them outside of PHP blocks instead of outputting each of those lines using PHP. This saves PHP processing resources and lets us have clean and readable files in our development.

Example :

```
info.php     plz add this
<html>
<body>
<h1> Using PHP and HTML together </h1>
Here is your PHP information:
<br>
<?php
phpinfo();
?>
</body>
</html>
```

4.2 Form Controls

The HTML <form> element defines a form which contains form controls that are used to collect user input. Form controls are different types of input elements, like text fields, text area, checkboxes, radio buttons, list, submit buttons and more

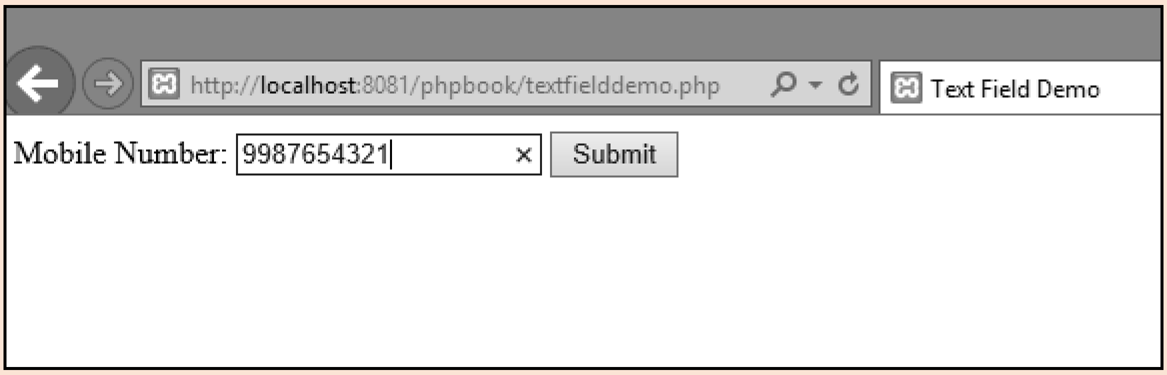
4.2.1 Text Field

- Text Field is used to take a single line input from the user.
- Text Field will be included in <form> element of a web page that will be used to take user input, which will be sent to PHP script available on the server.
- Data on the server will be fetched by any one of the super global variable $_GET and $_POST, after receiving it will be processed and the result will be sent to the user in a form of response.
- **For Example** a Web page textfielddemo.html has a text field, to get a mobile number from the user, on clicking Submit Button request will be sent to PHP Script textdemo.php on the server using GET method, PHP script will use super global variable $_GET to fetch the request and to send a response to the user.

Example 1 :

```
textfielddemo.html
<html>
<head>
<title> Text Field Demo</title>
</head>
<body>
<form method="get" action="textdemo.php">
<label>Mobile Number:</label>
<input type="text" name="mobileno" id=" mobileno">
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Output :



Mobile Number: 9987654321

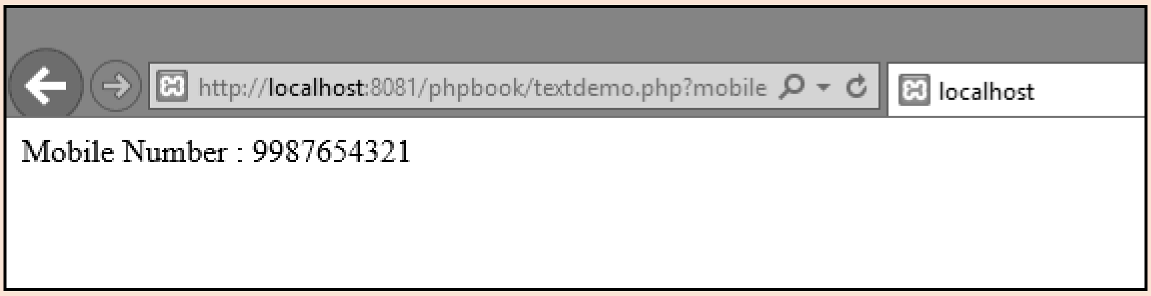
Fig. 4.2.1 : Using text field in an HTML page

Example 2 :

textdemo.php

```
<?php
if(isset($_GET["mobileno"])){
   echo "<p>Mobile Number : " . $_GET["mobileno"] . "</p>";
}
?>
```

Output :



```
Mobile Number : 9987654321
```

Fig. 4.2.2 : Reading data from a text field in PHP

4.2.2 Text Area

- Text Area is used to take multi line input from the user.
- Text Area will be included in <form> element of a web page that will be used to take multi line input like suggestions, address, feedback from users, which will be sent to PHP script available on the server.
- Data on the server will be fetched by any one of the superglobal variable $_GET and $_POST, after receiving it will be processed and the result will be sent to user in a form of response.
- **For Example** a Web page textareademo.html has a text area to get suggestions from users, on clicking Submit Button request will be sent to PHP Script phptextareademo.php on the server using GET method, PHP script will use superglobal variable $_GET to fetch the request and to send a response to the user.

Example 1 :

```
textareademo.html
<html>
<head>
<title>Text Area Demo</title>
</head>
<body>
<form method="get" action="phptextareademo.php">
<label>Suggestion:</label>
<textarea name="data" id="data" cols=”50” rows=”5”>
</textarea>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Output :

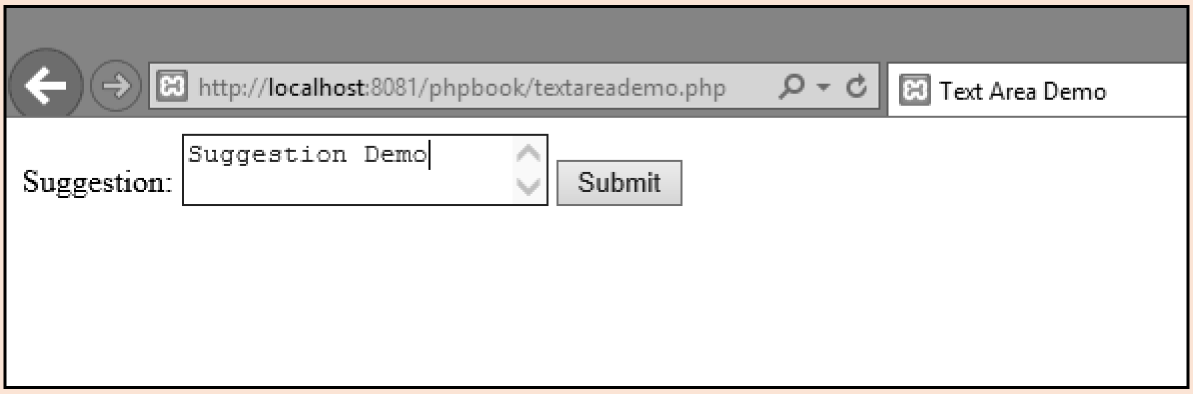
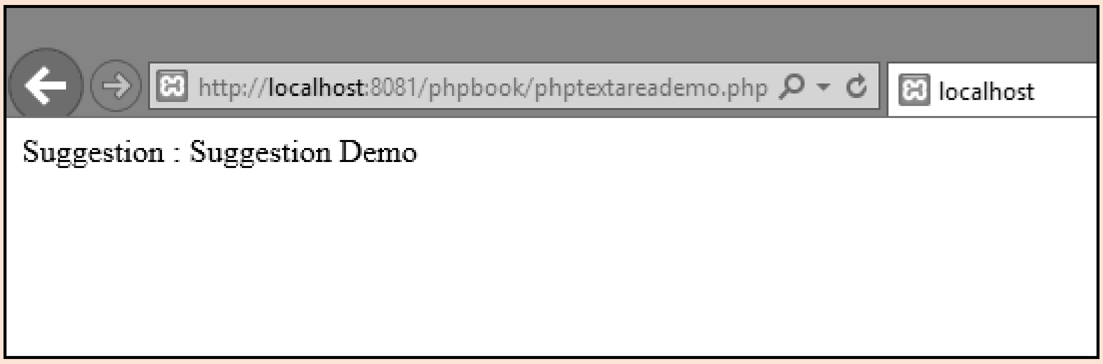


Fig. 4.2.3 : Using text area in an HTML page

Example 2 :

```
phptextareademo.php


<?php
if(isset($_GET["data"])){
   echo "<p>Suggestion : " . $_GET["data"] . "</p>";
}
?>
```

Output :



Suggestion : Suggestion Demo

Fig. 4.2.4 : Reading data from a text area in PHP

4.2.3 Radio Button

- Radio Button is used to make the user select single choice from a number of available choices.
- Radio Button will be included in <form> element of a web page that will be used single choice input from the user, which will be sent to PHP script available on the server.
- Data on the server will be fetched by any of the super global variable $_GET and $_POST, after receiving it will be processed and the result will be sent to the user in a form of response.
- **For Example** a Web page radiobtndemo.html has a radio button to get gender from the user, on clicking Submit Button request will be sent to PHP Script phpradiobtndemo.php on the server using GET method, PHP script will use super global variable $_GET to fetch the request and to send a response to the user.

Example 1 :

```
radiobtndemo.html
<html>
<head>
<title>Radio Button Demo</title>
</head>
<body>
<form method="get" action="phpradiobtndemo.php">
<label>Select your Gender:</label><br/>
<input type="radio" name="gender" value="male" checked> Male<br/>
<input type="radio" name="gender" value="female"> Female<br/>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Output :

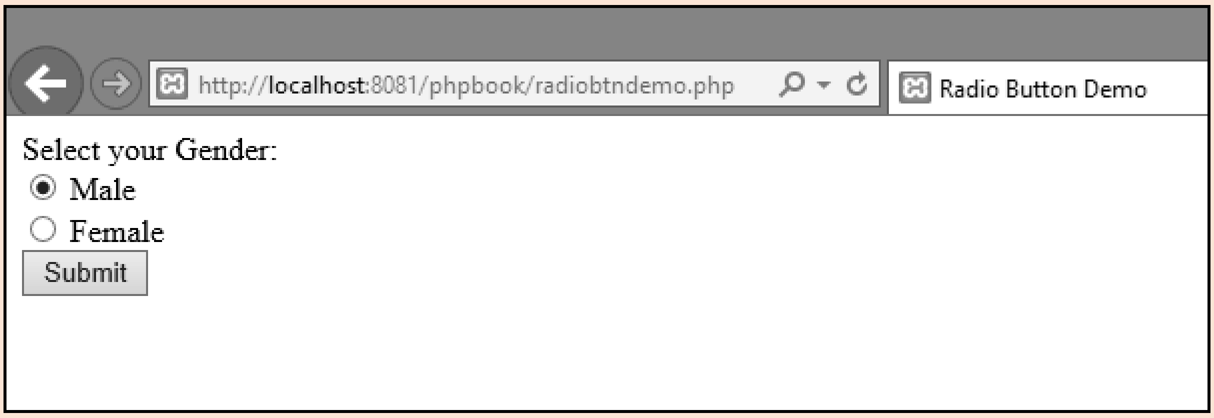
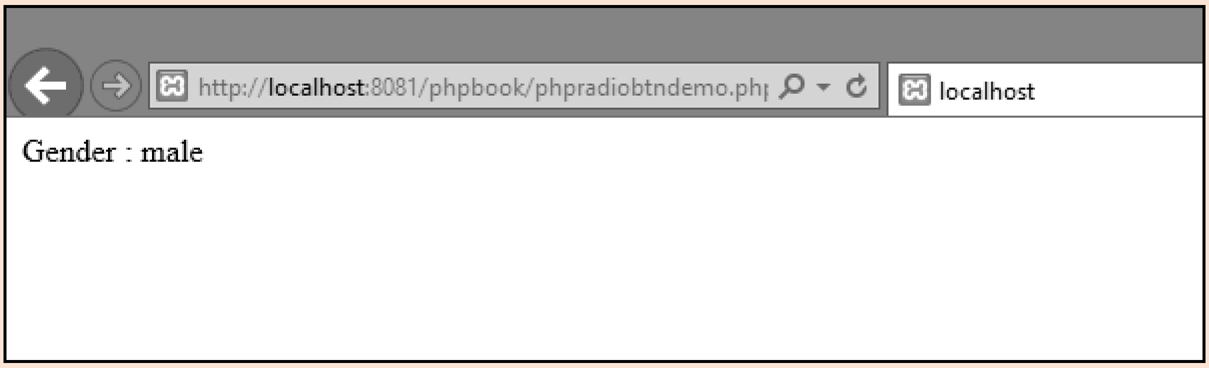


Fig. 4.2.5 : Using radio button in an HTML page

Example 2 :

```
phpradiobtndemo.php
<?php
if(isset($_GET["gender"])){
   echo "<p>Gender : " . $_GET["gender"] . "</p>";
}
?>
```

Output :



```
Gender : male
```

Fig. 4.2.6 : Reading data from a radio buttons in PHP

4.2.4 Check Box

– Check Box is used to select one or more options from available options displayed for selection.

– Check Box will be displayed as square box which will be activated when ticked (checked).

– Check Box will be included in <form> element of a web page that will be used to take user input, which includes multiple options like hobbies, where user can select one or more hobby form the multiple hobbies displayed on a web page, which will be sent to PHP script available on the server.

- Data on the server will be fetched by any one of the super global variable $_GET and $_POST, after receiving it will be processed and the result will be sent to the user in a form of response.
- **For Example** a Web page checkboxdemo.html has a multiple checkbox displaying the hobbies where users will select one or more of its choice and on clicking Submit Button request will be sent to PHP Script phpcheckboxdemo.php on the server using GET method, PHP script will use super global variable $_GET to fetch the request and to send a response to the user.

Example 1 :

```
checkboxdemo.html
<html>
<head>
<title> Text Field Demo</title>
</head>
<body>
<form method="get" action="phpcheckboxdemo.php">
<label>Select your Hobbies:</label>
<input type="checkbox" name="cricket" value="Cricket" checked > Cricket
<input type="checkbox" name="football" value="Football"> Football
<input type="checkbox" name="basketball" value="Basket Ball" > Basket Ball
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Output :

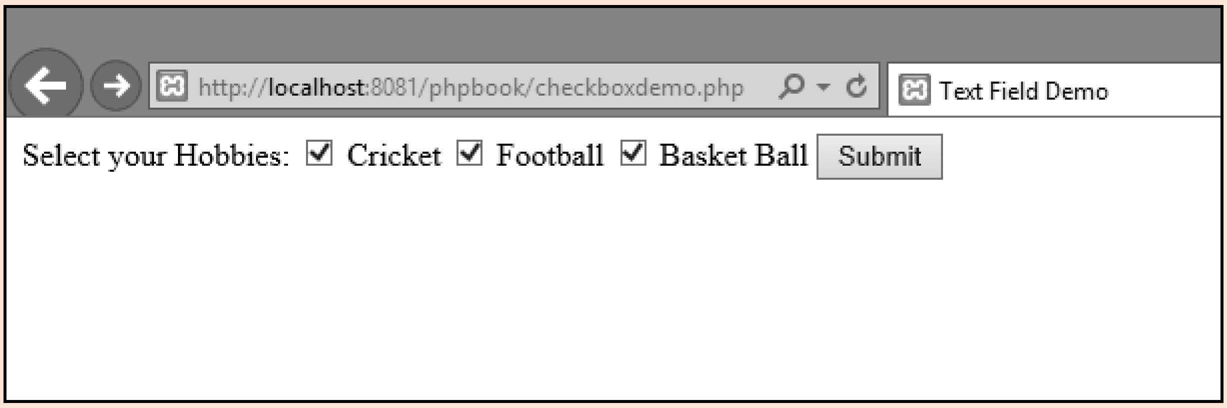
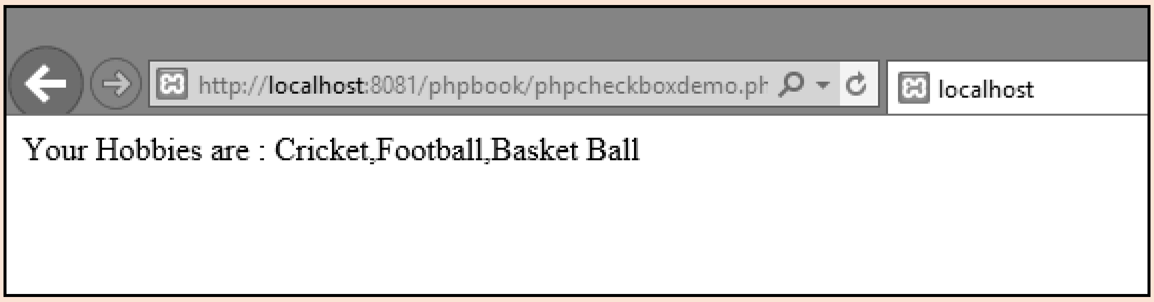

http://localhost:8081/phpbook/checkboxdemo.php
Text Field Demo
Select your Hobbies: ✓ Cricket ✓ Football ✓ Basket Ball
Submit

Fig. 4.2.7 : Using check box in an HTML page

Example 2 :

```
phpcheckboxdemo.php


<?php
   echo "<p>Your Hobbies are : " . $_GET["cricket "] .",". $_GET["football0"] ."," . $_GET["basketball"] . " </p>";


?>
```

Output :



```
http://localhost:8081/phpbook/phpcheckboxdemo.ph
Your Hobbies are : Cricket,Football,Basket Ball
```

Fig. 4.2.8 : Reading data from a check box in PHP

4.2.5 List Box

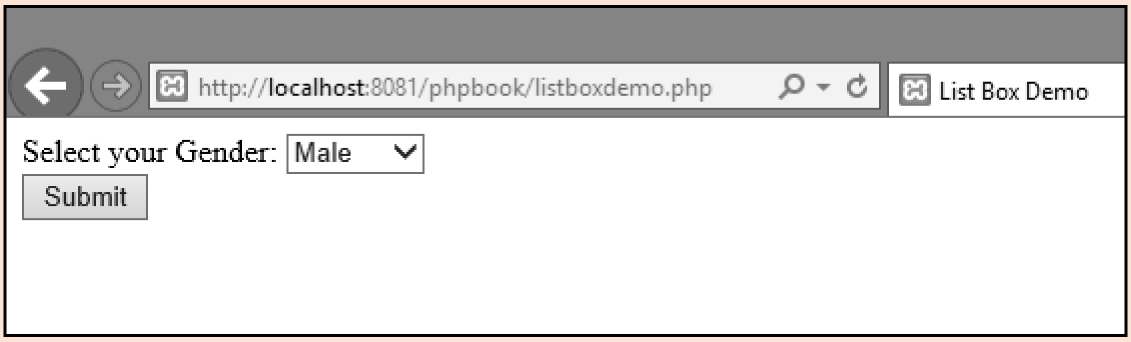
- List Box is used to create drop down list of options.
- HTML <select> tag will be include in <form>element of a web page that will be used to display dropdown list where user can select single or multiple options (when multiple attribute is set), which will be send to PHP script available on server.
- The <option> tag will be used in <select> tag in order to drop list of options.
- Data on the server will be fetched by any one of the super global variable $_GET and $_POST, after receiving it will be processed and the result will be sent to the user as a response.
- **For Example** a Web page selectdemo.html has a drop down list having a list of genders where users have to select one of gender, by clicking Submit Button request will be sent to PHP Script phpselectdemo.php on the server using GET method, PHP script will use super global variable $_GET to fetch the request and to send a response to the user.

Example 1 :

```
selectdemo.html
<html>
<head>
<title> List Box Demo</title>
</head>
<body>
<form method="get" action="phpselectdemo.php">
```

```
<label>Select your Gender:</label>
<select name="gender" id="gender">
<option value="Male">Male</option>
<option value="Female">Female</option>
</select><br/>
<input type="submit" value="Submit">
</form>
</body>
</html>
```

Output :



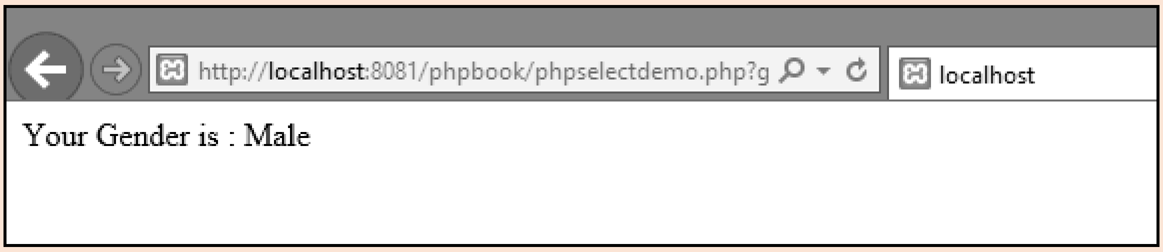
```
http://localhost:8081/phpbook/listboxdemo.php
List Box Demo
Select your Gender: Male
Submit
```

Fig. 4.2.9 : Using list box in an HTML page

Example 2 :

```
phpselectdemo.php
<?php
if(isset($_GET["gender"])){
    echo "<p>Your Gender is : " . $_GET["gender"] . "</p>";
}
?>
```

Output :



```
http://localhost:8081/phpbook/phpselectdemo.php?g
localhost
Your Gender is : Male
```

Fig. 4.2.10 : Reading data from a list box in PHP

4.2.6 Buttons

– Button is created using <button> tag.

- Text and Image can be displayed on the button by placing the text or image between the opening and closing tags of button.
- Buttons has to be added with actions using JavaScript or by associating the button with a form.
- In we are creating button in form <input> tag is used to create a button.
- **For Example** a Web page buttondemo.html has a two buttons “Save Changes” and “Delete”, by clicking any of the Button request will be send to PHP Script phpbuttondemo.php on server using GET method, PHP script will use superglobalvariable $_GET to fetch the request and will display name of the button which is clicked and it will be send as an response to user.

Example :

```
buttondemo.html
<html>
<head>
<title> Button Demo</title>
</head>
<body>
<form method="get" action="phpbuttondemo.php">
    <input type="submit" name="btnSubmit" value="Save Changes" />
    <input type="submit" name="btnDelete" value="Delete" />
</form>
</body>
</html>
```

Output :

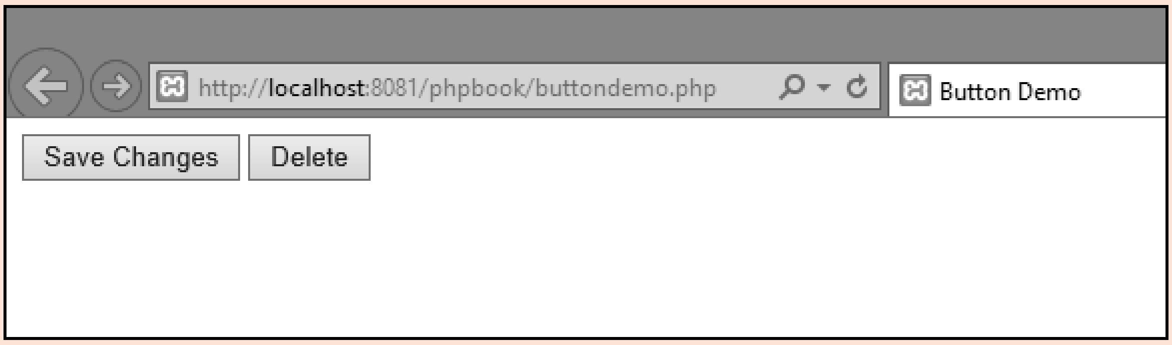
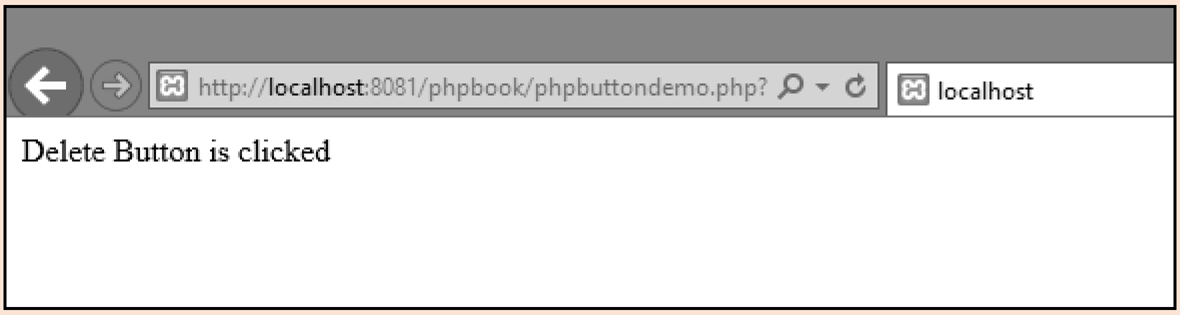


Fig. 4.2.11 : Using buttons in an HTML page

Example 2 :

```
phpbuttondemo.php

<?php
if ($_SERVER['REQUEST_METHOD'] === 'GET') {
    if (isset($_GET['btnDelete'])) {
        echo "Delete Button is clicked";
    } else {
        echo "Save Changes Button is clicked";
    }
}
?>
```

Output :



```
Delete Button is clicked
```

Fig. 4.2.12 : Displaying clicked button in PHP

4.2.7 Hidden Controls

- Hidden Controls are used to store the data in a Web page that user can't see.
- Hidden Controls will be included in <form> element of a web page that will be used to store data that will not be visible to the user, which will be sent to PHP script available on the server.
- Data on the server will be fetched by any one of the superglobal variable $_GET and $_POST, after receiving it will be processed and the result will be sent to the user in a form of response.
- **For Example** a Web page hiddendemo.html has a hidden control which stores user id of the user, on clicking Submit Button request will be sent to PHP Script phphiddendemo.php on the server using POST method, PHP script will use superglobalvariable $_POST to fetch the request and to send a response to the user.

Example 1 :

```
hiddendemo.html
<html>
<head>
<title> Hidden Control Demo</title>
</head>
<body>
<form method="post" action="phphiddendemo.php">
<input type="hidden" name="user_id" id="user_id" value="101">
<input type="submit" value="Submit">
```

```
</form>
</body>
</html>
```

Output :

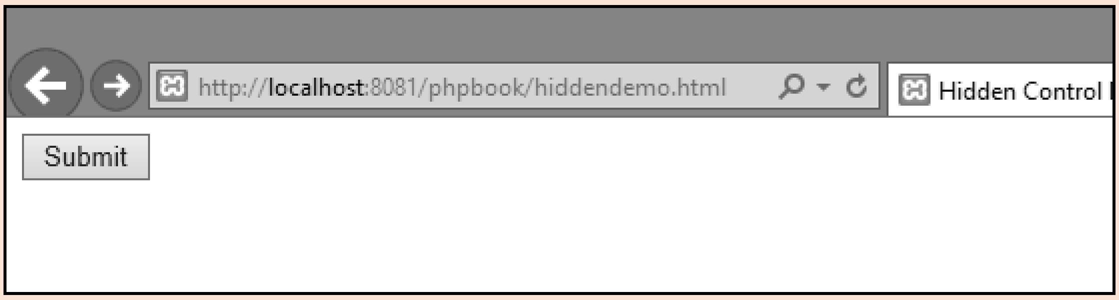
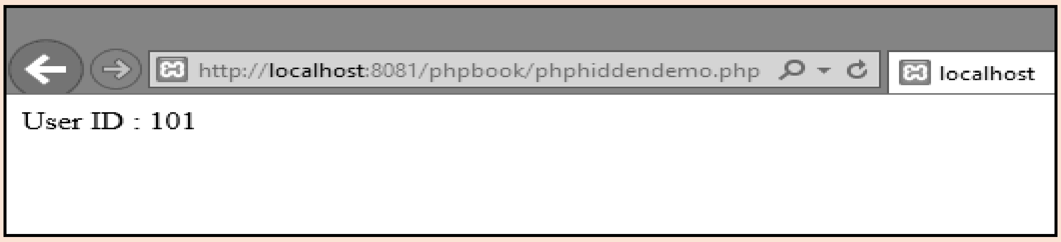


Fig. 4.2.13 : Using hidden control in an HTML page

Example 2 :

```
phphiddendemo.php

<?php
if(isset($_POST["user_id"])){
   echo "<p>User ID : " . $_POST["user_id"] . "</p>";
}
?>
```

Output :



```
User ID : 101
```

Fig. 4.2.14 : Reading data from a hidden control in PHP

4.3 Working with Multiple Forms

4.3.1 A Web Page having Multiple Forms

A web page having multiple forms can be processed in two types :

1. Posting each form to different PHP script file for processing

– Multiple functionality can be provided in a single web page by providing multiple forms in a web page having different functionality.

- Each form on this web page will be given a separate name that will uniquely identify the form in web page with multiple forms.
- Data from each form should be given to separate PHP script file for processing by specifying PHP script filename in the action attribute of the forms.
- Each PHP Script should be written in such a fashion that will handle all the data coming from that form.
- Disadvantage of this method is that we have to write separate PHP script for each form, which creates extra files for handing.
- **For Example,** a Web page multiformdemo.html has two forms, one for sending mail information and another for sending mobile number information, each form is having its own PHP script written to handle its own form elements on the server, on clicking submit button of each form data is sent to its corresponding PHP script which handles the request and generates response for user.

Example :

```
multiformdemo.html
<html>
<head>
<title> Multiple Form Demo</title>
</head>
<body>
    <form name="mailform" method="post" action="phpemaildata.php">
     <input type="text" name="email" id="email" />
     <input type="submit" name="mail-submit" value="Send Mail Information" />
    </form>

    <form name="mobileform" method="post" action="phpmobiledata.php">
     <input type="text" name="mobileno" id="mobileno" />
     <input type="submit" name="mobile-submit" value="Send Contact Information" />
    </form>
</body>
</html>
```

Output :

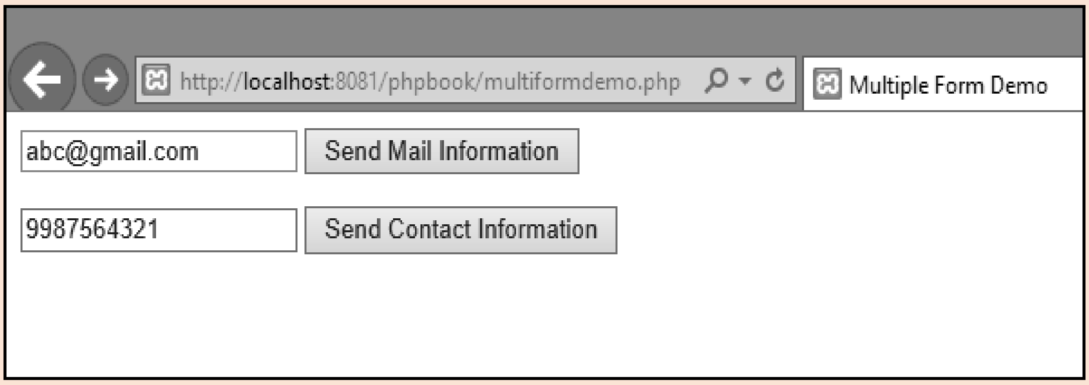
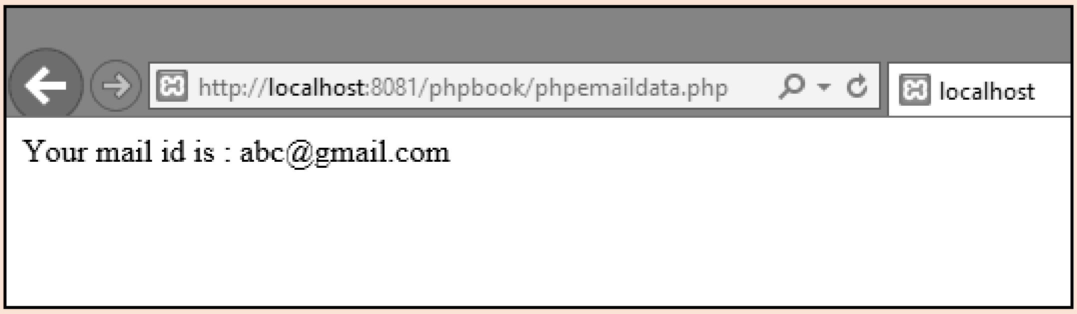


Fig. 4.3.1 : Using two forms to submit data in an HTML page

Example 2 :

```
phpemaildata.php
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
   if (!empty($_POST['mail-submit'])) {
         echo "Your mail id is : ".$_POST['email'];
      }
}
?>
```

Output :



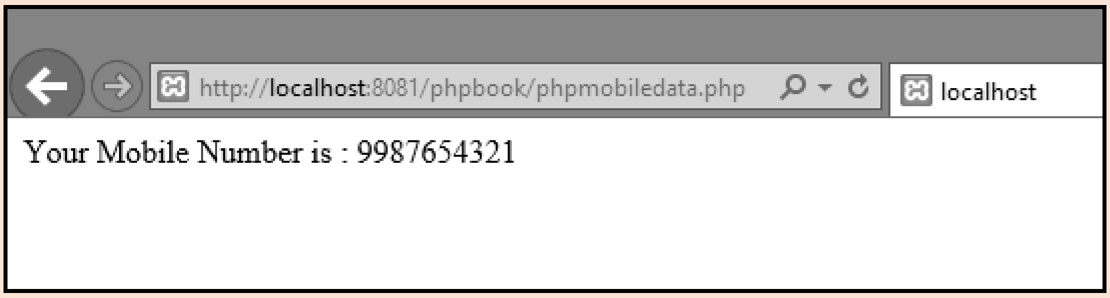
```
http://localhost:8081/phpbook/phpemaildata.php
localhost
Your mail id is : abc@gmail.com
```

Fig. 4.3.2 : Displaying submitted data from mailform in PHP

Example 3 :

```
phpmobiledata.php
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {

      if (!empty($_POST['mobile-submit'])) {
         echo "Your Mobile Number is : ".$_POST['mobileno'];
      }
}
?>
```

Output :



```
http://localhost:8081/phpbook/phpmobiledata.php
localhost
Your Mobile Number is : 9987654321
```

Fig. 4.3.3 : Displaying submitted data from mobile form in PHP

2. Posting all form to single PHP script file for processing

– Multiple functionality can be provided in a single web page by providing multiple forms in a web page having different functionality.

– Each form on this web page will be given a separate name that will uniquely identify the form in web page with multiple forms.

– Data from each form should be given to a single PHP script file for processing by specifying PHP script filename in the action attribute of the forms.

– Each PHP Script should be written in such a fashion that will handle all the data coming from multiple forms.

– Data from multiple forms can be identified by it submit button and the processing each form will be written with help of if, else and else if conditional statements.

– Advantage of this method is that we have to write a single PHP script for processing of all forms, which saves time in the creation and handling of extra files.

– **For Example,** a Web page multiformdemo.html has two forms, one for sending mail information and another for sending mobile number information, both form are having a single PHP script written to handle data of all forms on the server, by clicking submit button of each form data is sent to its PHP script which identify the form by its submit button based on that it fetches the request and a generates a response for the user.

Example :

```
multiformdemo.html
<html>
<head>
<title> Multiple Form Demo</title>
</head>
<body>
    <form name="mailform" method="post" action="phpmultiformdemo.php">
     <input type="text" name="email" id="email" />
     <input type="submit" name="mail-submit" value="Send Mail Information" />
    </form>

    <form name="mobileform" method="post" action="phpmultiformdemo.php">
     <input type="text" name="mobileno" id="mobileno" />
     <input type="submit" name="mobile-submit" value="Send Contact Information" />
    </form>
</body>
</html>
```

Output :

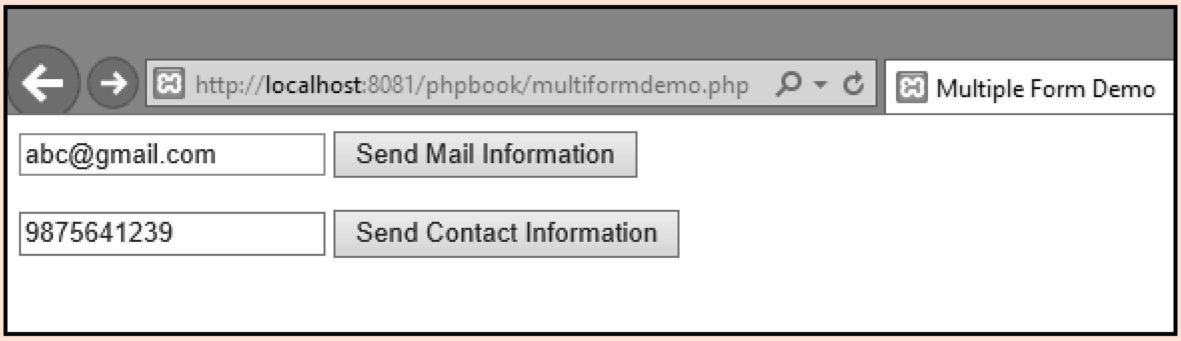
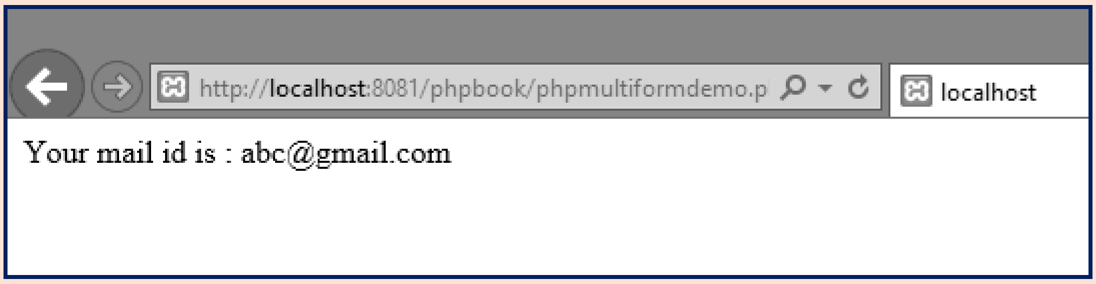


Fig. 4.3.4 : Using two forms to submit data in an HTML page

Example 2 :

```
phpmultiformdemo.php
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
  if (!empty($_POST['mail-submit'])) {
      echo "Your mail id is : ".$_POST['email'];
    }
    else if (!empty($_POST['mobile-submit'])) {
      echo "Your Mobile Number is : ".$_POST['mobileno'];
    }
}
?>
```

Output :



```
Your mail id is : abc@gmail.com
```

Fig. 4.3.5: Displaying submitted data from multiple forms in PHP

4.3.2 A Form having Multiple Submit Buttons

- Multiple operations can be provided on a single form by providing a different buttons for different operation.
- Based on which button is clicked, data in the form is processed differently for the operations mentioned on that button.

- Single PHP Script is sufficient to handle all the operations mentioned on the buttons in the form, PHP Script will identify the button which is being clicked and will carry out the operations according to it.
- Identification of the button is done by its name on the server and corresponding operation is called with the help of if, else and else if conditional statements.
- **For Example** a Web page multibuttondemo.html is having two text fields for accepting two numbers from user and two Submit buttons representing Add and Subtract operation is forms, on clicking each button a corresponding operation mention in PHP script php multibuttondemo.php on the server will be called.

Example 1 :

```
multibuttondemo.html
<html>
<head>
<title> Multiple Form Demo</title>
</head>
<body>
    <form name="mailform" method="post" action="phpmultibuttondemo.php">
     <input type="text" name="no1" id="no1" />
     <input type="text" name="no2" id="no2" />
     <input type="submit" name="addbtn" value="Add" />
     <input type="submit" name="subbtn" value="Subtract" />
    </form>
</body>
</html>
```

Output :

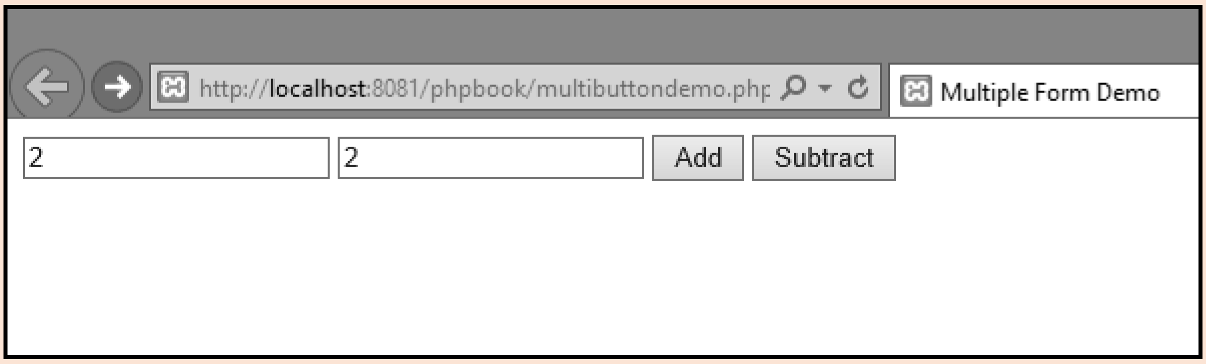
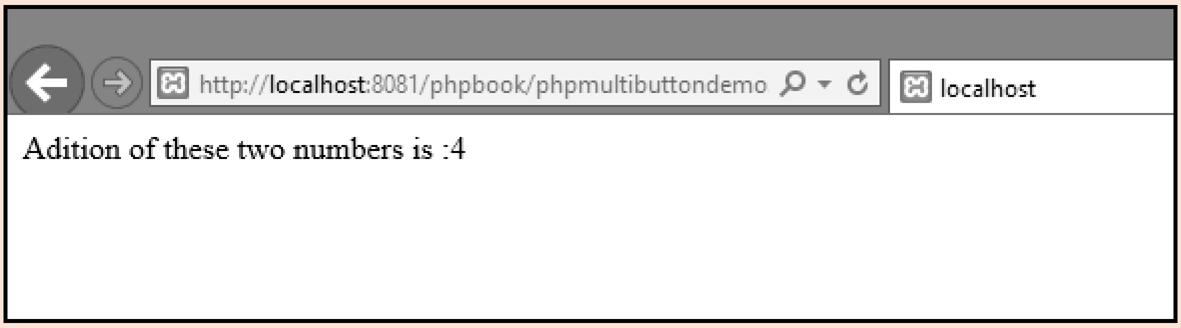


Fig. 4.3.6 : Multiple Submit Buttons in an HTML page

Example 2 :

```
phpmultibuttondemo.php
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
  if (isset($_POST['addbtn'])) {
        echo "Adition of these two numbers is :". ((int) $_POST['no1'] + (int) $_POST['no2']);
    } else if (isset($_POST['subbtn'])){
        echo "Subtraction of these two numbers is :". ((int) $_POST['no1'] - (int) $_POST['no2']);
    }
}
?>
```

Output :



```
Adition of these two numbers is :4
```

Fig. 4.3.7 : Buttons in PHP

4.3.3 Self Processing Form : PHP_Self

PHP_SELF is a variable that returns the current script being executed. This variable returns the name and path of the current file (from the root folder). You can use this variable in the action field of the FORM. the $_SERVER["PHP_SELF"] sends the submitted form data to the page itself, instead of jumping to a different page. This way, the user will get error messages on the same page as the form.

We have the following form in a page named "test.php":

```
<form method="post" action="<?php echo $_SERVER["PHP_SELF"];?>">
```

If a user enters the normal URL in the address bar like "http://www.example.com/test.php", the above code will be translated to :

```
<form method="post" action="test.php">
```

A common use of PHP_SELF variable is in the action field of the <form> tag. The action field of the FORM instructs where to submit the form data when the user presses the “submit” button.

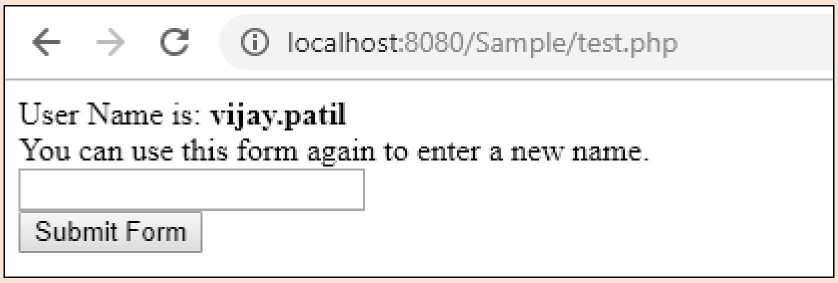
Example :

"test.php"

```
<html>
<body>
<?php
if(isset($_POST['submit']))
```

```
{
   $name = $_POST['name'];
   echo "User Name is:<b> $name </b>";
   echo "<br>You can use this form again to enter a new name.";
}
?>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
<input type="text" name="name"><br>
<input type="submit" name="submit" value="Submit Form"><br>
</form>
</body>
</html>
```

Output :



```
localhost:8080/Sample/test.php
User Name is: vijay.patil
You can use this form again to enter a new name.
```

4.4 Web Page Validation

A web page having multiple forms

- User may by mistakenly submit the data through form with empty fields or in wrong format.
- PHP script must ensure that required fields are complete and submitted data is in valid format.
- PHP provides some inbuilt function using these functions that input data can be validated.
- **empty()** function will ensure that text field is not blank it is with some data, function accepts a variable as an argument and returns **TRUE** when the text field is submitted with empty string, zero, NULL or FALSE value.
- **Is_numeric()** function will ensure that data entered in a text field is a numeric value, the function accepts a variable as an argument and returns **TRUE** when the text field is submitted with numeric value.
- **preg_match()** function is specifically used to performed validation for entering text in the text field, function accepts a “regular expression” argument and a variable as an argument which has to be in a specific pattern. Typically it is for validating email, IP address and pin code in a form.
- **For Example** a PHP page formvalidation.php is having three text fields name, mobile number and email from user, on clicking Submit button a data will be submitted to PHP script validdata.php on the server, which will perform three

different validation on these three text fields, it will check that name should not be blank, mobile number should be in numeric form and the email is validated with an email pattern.

Example 1 :

```
formvalidation.php


<html>
<head>
<title> Validating Form Data</title>
</head>
<body>
        <form method="post" action="validdata.php">
         Name :<input type="text" name="name" id="name" /><br/>
         Mobile Number :<input type="text" name="mobileno" id="mobileno" /><br/>
         Email ID :<input type="text" name="email" id="email" /><br/>
         <input type="submit" name="submit_btn" value="Submit" />
        </form>
</body>
</html>
```

Output :

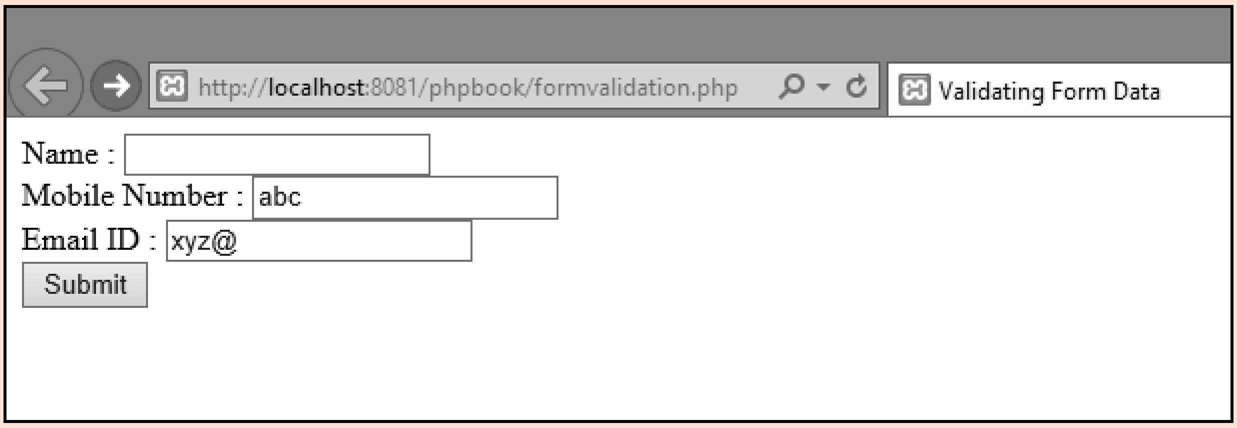


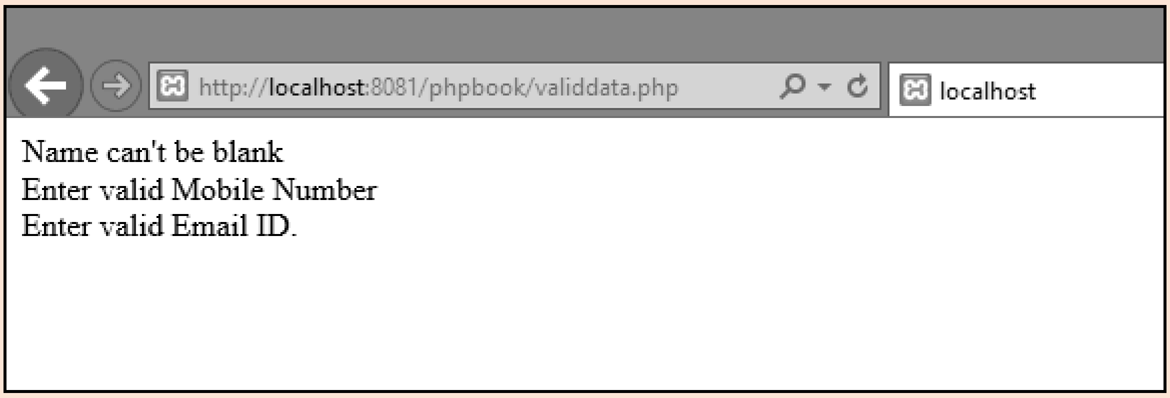
Fig. 4.4.1 : Displaying fields for accepting data for validation

Example 2 :

```
validdata.php


<?php
```

```
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        if(empty($_POST['name']))
        {
         echo "Name can't be blank<br/>";
        }
        if(!is_numeric($_POST['mobileno']))
        {
         echo "Enter valid Mobile Number<br/>";
        }
        $pattern ='/\b[\w.-]+@[\w.-]+\.[A-Za-z]{2,6}\b/';
        if(!preg_match($pattern,$_POST['email']))
        {
         echo "Enter valid Email ID.<br/>";
        }
}
?>
```

Output :



```
Name can't be blank
Enter valid Mobile Number
Enter valid Email ID.
```

Fig. 4.4.2 : Displaying result after validation

4.4.1 Superglobals

Q.	Describe superglobals.

– The superglobals arrays allow you to specify where the input data came from; that is what method was used.

– superglobals are predefined variables in PHP, which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.

– Superglobals are a type of variables that are available from any part of your code. Some are well known like POST and GET that are used to pass form values and COOKIE and SESSION that are used to store specific information for a later use. The superglobals are listed in Table 4.4.1 :

Table 4.4.1

Supreglobal variable	Description
$_SERVER	This is an array containing information such as headers, paths and script locations. The entries in this array are created by the web server. There is no guarantee that every web server will provide any of these
$_REQUEST	An associative array consisting of the contents of $_GET, $_POST, and $_COOKIE.
$_GET	An associative array of variables passed to the current script via the HTTP GET method.
$_POST	An associative array of variables passed to the current script via the HTTP POST method.
$_COOKIE	An associative array of variables passed to the current script via HTTP cookies.
$_SESSION	An associative array containing session variables available to the current script.
$_FILES	An associative array of items uploaded to the current script via the HTTP POST method.
$GLOBALS	Contains a reference to every variable which is currently available within the global scope of the script. The keys of this array are the names of the global variables.
$_PHP_SELF	A string containing PHP script file name in which it is called.
$_ENV	An associative array of variables passed to the current script via the environment method.
$argc	The number of arguments passed to script.
$argv	Array of arguments passed to script.

4.5 Cookies

– PHP cookie is a small piece of information which is stored at client browser. It is used to recognize the user.

– Cookie is created at server side and saved to client browser. Each time when client sends request to the server, cookie is embedded with request. Such way, cookie can be received at the server side. In short, cookie can be created, sent and received at server end.

– A cookie is a small file with the maximum size of 4KB that the web server stores on the client computer.

– A cookie can only be read from the domain that it has been issued from. Cookies are usually set in an HTTP header but JavaScript can also set a cookie directly on a browser.

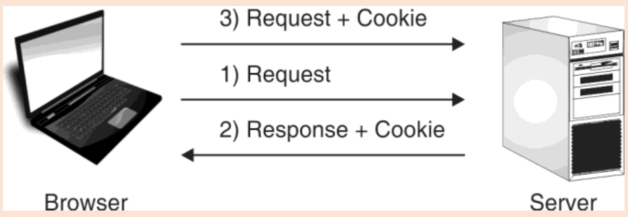


Fig. 4.5.1

There are three steps involved in identifying returning users :

1. Server script sends a set of cookies to the browser. For example : name, age or identification number etc.

2. Browser stores this information on local machine for future use.

3. When next time browser sends any request to web server then it sends those cookies information to the server and server uses that information to identify the user.

4.5.1 Use of Cookies

– Cookie is a small piece of information stored in client browser. It is a technique used to identify a user using the information stored in their browser (if already visited that website)

– When user requests for a page on a Web site data in the cookies which belongs to a same site is send to the server automatically within the request.

– Expiration period of the cookies can be set, it can be set to seconds, minutes, hours, days or for year, it can also be set a cookie to expire once browser applications is closed.

– It more secure than query string.

– It is very much preferable to store non-critical user data on an ongoing basis.

– Tracking the pages visited by a user.

– Personalizing the user experience - this is achieved by allowing users to select their preferences. The page requested that follow are personalized based on the set preferences in the cookies.

4.5.2 Attributes of Cookies

Q. List attributes of cookies? Explain any two.

Attribute of cookies	Description
name	The unique name is given to a particular cookie.
value	The value of the cookie.
expires	The time when a cookie will get expire. When it reaches to its expiration period cookies is deleted from browser automatically. If value is set to zero, it will only last till the browser is running its get deleted when the browser exits.
path	The path where browser to send the cookies back to the server. If the path is specified, it will only send to specified URL else if it is stored with "/" the cookie will be available for all the URL's on the server.
Domain	The browser will send the cookie only for URLs within this specified domain. By default is the server host name.
secure	If this field is set, the cookie will only be sent over https connection. By default it is set to false, means it is okay to send the cookie over an insecure connection.
HttpOnly	This field, if present, tells the browser that it should only make the cookie assessable only to scripts that run on the Web server (that is, via HTTP). Attempts to access the cookie through JavaScript will be rejected.

4.5.3 Create Cookies

Q. How to create cookies in PHP ?

- PHP provides a inbuilt function setcookie(), that can send appropriate HTTP header to create the cookie on the browser.
- While creating the cookie, we have to pass require arguments with it.
- Only name argument is must but it is better to pass value, expires and path to avoid any ambiguity.

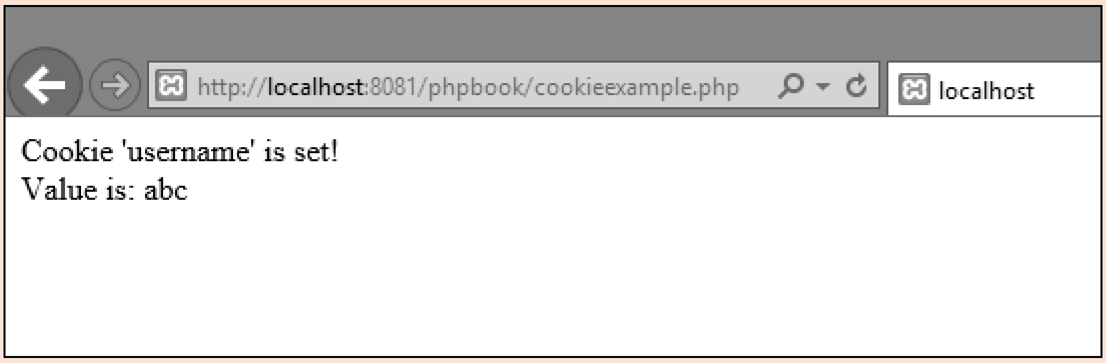
  Syntax : setcookie(*name, value, expire, path, domain, secure, HttpOnly*);
- **For Example :** setcookie() is used to create a cookie storing the user name with name "username", abc is the value stored in the cookie, expires argument uses PHP time() function which returns current time, so the expiry time is 60 * 60 * 24 * 365 seconds after the current time, or one year in future, path argument is set to "/" it means that cookies can be returned to any URL on the server, domain is set to abc.com, secure flag is set to false and HttpOnly is too true.

  setcookie("username", "abc", time() + 60 * 60 * 24 * 365, "/", ".abc.com", false, true);

Example :

```
cookieexample.php
<html>
<body>
<?php
$cookie_name = "username";
$cookie_value = "abc";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day

if(!isset($_COOKIE[$cookie_name])) {
   echo "Cookie name '" . $cookie_name . "' is not set!";
} else {
   echo "Cookie '" . $cookie_name . "' is set!<br>";
   echo "Value is: " . $_COOKIE[$cookie_name];
}
?>
</body>
</html>
```

Output :



```
Cookie 'username' is set!
Value is: abc
```

Fig. 4.5.2 : Cookies Demonstration

4.5.4 Modify Cookies Value

- Cookie value can be retrieved using global variable $_COOKIE.
- We use the isset() function to find out if the cookie is set or not.
- Modification of the cookie can simply be done again by setting the cookie using the setcookie() function.
- Modified cookie can be checked by calling the same cookie with its name to check its modified value.

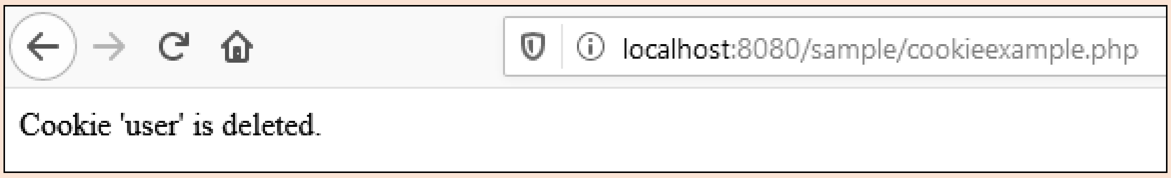
4.5.5 Delete Cookies

- Cookie can be deleted from user browser simply by setting expires argument to any past date it will automatically delete the cookie from user browser.
- Deleted cookie can be checked by calling the same cookie with its name to check if it exists or not.

Example :

```
<html>
<body>
<?php
setcookie("user"," ",time()-3600);

echo "Cookie 'user' is deleted.";

?>
</body>
</html>
```

Output :



Cookie 'user' is deleted.

Fig. 4.5.3

4.5.6 Accessing Cookie Value

For accessing a cookie value, the PHP $_COOKIE superglobal variable is used. It is an associative array that contains a record of all the cookies values sent by the browser in the current request. The records are stored as a list where cookie name is used as the key.

Syntax : $value=$_COOKIE["CookieName"]; //returns cookie value

Example :

```
<?php
setcookie("user", "Vijay");
  ?>
<html>
<body>
<?php
  if(!isset($_COOKIE["user"]))
  {
    echo "Sorry, cookie is not found!";
  } else {
    echo "<br/>Cookie Value: " . $_COOKIE["user"];
  }
  ?>
</body>
</html>
```

Firstly cookie is not set. But, if you *refresh* the page, you will see cookie is set now.

Cookie Value = Vijay

4.5.7 Types of Cookies

There are two types of cookies :

1. Session Based which expire at the end of the session.
2. Persistent cookies which are written on hard disk.

Session cookies expire at the end of the session. This means, when you close your browser window, the session cookie is deleted. This website only uses session cookies. Persistent cookies do not expire at the end of the session. A session cookie may be created when you visit a site or portion of a site. The cookie exists for the duration of your visit. For example, a session cookie is created when you use the Personal Login to access secured pages. Depending on the settings in your browser, you may have the option to deny the session cookie; however, if you deny the cookie you may have trouble using the site which relies on that cookie.

Isset() function

To read data from a cookie, you first have to check if the cookie actually exists. This is achieved through the isset() function. The isset() function is used to check for the existence of a variable, in this case, a cookie variable through the use of the $_COOKIE associative array which stores an array of existing cookies.

Syntax :

```
isset($_COOKIE['nameOfCookie']);
```

If the cookie specified in the isset() function exists, then the function will return true, otherwise it will return false.

Example :

```
<?php
if (isset($_COOKIE['cookie1']))
{
$cookie1 = $_COOKIE['cookie1'];
}
?>
```

In the above example, an if statement checks for the existence of a cookie named cookie1. If it exists, then its value will be passed to the variable *$cookie1*. If it does not, then it will remain empty. The **isset()** function checks for this.

4.5.8 Advantages and Disadvantages of Cookies

Advantages of Cookies :

- Cookies are simple to use and implement.
- Occupies less memory, do not require any server resources and are stored on the user's computer so no extra burden on the server.
- They are easy to implement : The fact that cookies are supported on the client's side means they are a lot easier to implement.
- They are domain-specific : Each domain has its own cookies. There is no domain that shares cookies with other domains. This makes them independent.
- They are simple to use : Cookies are much easier to use. This is the reason why they are enabled and disabled from the client's side.
- The cookies make browsing the Internet faster & easier.

Disadvantages of Cookies :

- They are not secured : As mentioned previously, cookies are not secure as they are stored in the clear text they may pose a possible security risk as anyone can open and tamper with cookies.
- Difficult to decrypt : You can manually encrypt and decrypt cookies, but it requires extra coding and can affect application performance because of the time that is required for encryption and decryption.
- There are limitations in size : Several limitations exist on the size of the cookie text (4kb in general), the number of cookies (20 per site in general). Each site can hold only twenty cookies.
- Can be disabled : User has the option of disabling cookies on his computer from the browser's setting. This means that the user can decide not to use cookies on his browser and it will still work.
- Users can delete cookies : The fact that users can delete cookies from their computers gives them more control over the cookies.

4.6 Session

- Cookies are used to store user data on the client's browser is not the most secure way of storing data, it can be easily hacked.
- Cookie data for a website is uploaded every time a request is sent for specify URL on a server. For example 10 cookies of a site is stored on a client browser then for every request it has to upload 40KB of data for each request done from server as one cookie is of 4KB.
- Both this problem can be solved by using session to store user data
- Session data is stored on the server side and each Session is assigned with a unique Session ID (SID) for that session data.
- As session data is stored on the server there is no need to send any data along with the URL for each request to server.
- More data can be stored in session as compared with cookie because location for storing data is a server.
- PHP stores the session data in a temporary file on the server, the location of the temporary file is specified by the session.save_path directives in the PHP configuration file
- PHP session is used to store and pass information from one page to another temporarily (until user close the website).
- PHP session technique is widely used in shopping websites where we need to store and pass cart information e.g. username, product code, product name, product price etc from one page to another.
- PHP session creates unique user id for each browser to recognize the user and avoid conflict between multiple browsers.

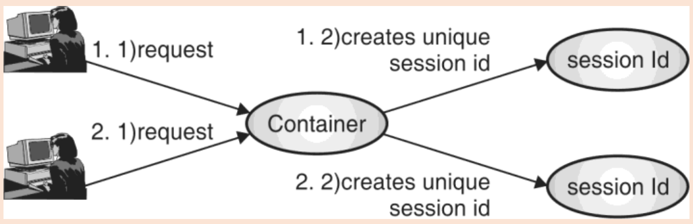

1. 1)request
1. 2)creates unique session id
session Id
2. 1)request
Container
2. 2)creates unique session id
session Id

Fig. 4.6.1

4.6.1 Use of Session

- Session are used to store important information such as the user id more securely on the server where malicious users cannot temper with them.
- To pass values from one page to another.
- Sessions are the alternative to cookies on browsers that do not support cookies.
- You want to store global variables in an efficient and more secure way compared to passing them in the URL
- You are developing an application such as a shopping cart that has to temporary store information with a capacity larger than 4KB.

4.6.2 Start Session

- It very easy to create and start session in PHP.
- PHP session can be started by calling session_start() function.
- If it is a new session then session_start() function will generate a unique SID for the session or else a same function will be used to access existing along with the data store in that session.

```
<?php
session_start();
?>
```

4.6.3 Set Session Variables

- Session variable can be set with a help of a PHP global variable: $_SESSION.
- Data in the session is stored in the form of keys and values pair.
- We can store any type of data on the server, which include arrays and objects.
- For example, we want to store username in the session so it can be assessed whenever it is required throughout the session.

```
<?php
    session_start();
$_SESSION["username"] = "abc";
?>
```

Example :

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>
```

```
<?php
// Set session variables
$_SESSION["Name"] = "Vijay";
$_SESSION["Address"] = "Thane";
echo "Session variables are set.";
?>
</body>
</html>
```

Output :

```
Session variables are set.
```

4.6.4 Get Session Variables

- Session variable can be get with a help of a PHP global variable: $_SESSION.
- While accessing the data using $_SESSION variable we have to mention key in the $_SESSION variable.
- For example, we want retrieve the data stored in session variable.

Example 1 :

```
<?php
	session_start();
echo "User name : "$_SESSION["username"];
?>
```

Output :

```
	User name : abc
```

Example 2 :

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>


<?php
```

```
// Echo session variables that were set on previous page
echo "User Name: " . $_SESSION["Name"] .".<br>";
echo "User Address: " . $_SESSION["Address"] . ".";
?>
</body>
</html>
```

Output :

```
User Name: Vijay.
User Address: Thane.
```

4.6.5 Destroy Session

- Session automatically gets destroyed when user quits browser.
- If someone wants to destroy the session after certain operation example after a logout that can be done using inbuilt PHP functions.
- A PHP function session_unset() is used to remove all session variables and session_destroy() is used to destroy session.

```
<?php
session_unset();                // remove all session variables
session_destroy();        // destroy the session
?>
```

4.6.6 Session_register()

The function session_register() registers Register one or more global variables with the current session. session_register() takes two arguments, the string representing the variable name and the value to be assigned to the variable.

Syntax :

```
<?phpsession_register('username', 'John'); ?>
```

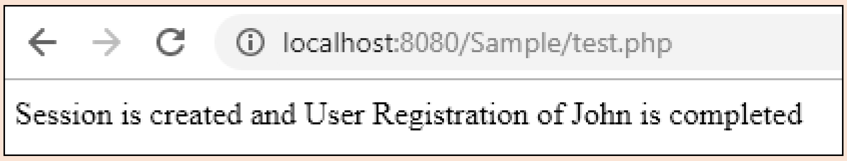
Session variables are accessed by using the variable name as an index key into the $_SESSION array. The session_is_registered() function can also be used to make sure the variable exists before attempting to read the value.

Example :

```
<?php
session_start();
session_register('username', 'JohnW');


function session_register($key, $value)
```

```
{
      $_SESSION[$key] = $value;
      echo "Session is created and User Registration of ".$value." is completed<br/>";
}

function session_is_registered($keyname)
{
      echo "Checking if Session exist or not<br/>";
      if(isset($_SESSION[$keyname])) {
       echo "Session exists<br/>";
       return true;
}else {
          echo "Session does not exists<br/>";
       return false;
   }
}
?>
<html>
```

Output :



Session is created and User Registration of John is completed

4.6.7 Difference between Session and Cookies in PHP

Q. What is the difference between PHP session and Cookie ?

Cookie	Session
Cookies are stored in browser as text file format.	Sessions are stored in server side.
The cookie is a client-side resource.	The session is a server-side resource.
It is stored limit amount of data.	It is stored unlimited amount of data
It is only allowing 4kb[4096bytes].	It is holding the multiple variable in sessions.
It is not holding the multiple variable in cookies.	It is holding the multiple variable in sessions.
We can accessing the cookies values in easily. So it is less secure.	We cannot accessing the session values in easily. So it is more secure.
Remember info until deleted by user or expiry. **PLZ add this**	Remembers info until web site time-out.
Setting the cookie time to expire the cookie.	using session_destory(), we will destroyed the sessions.

The setcookie() function must appear BEFORE the <html> tag.	The session_start() function must be the very first thing in your document. Before any HTML tags.
Usually contains an id string.	Usually contains more complex information.
Specific identifier links to server.	Specific identifier links to user.

4.7 Sending E-mail

- PHP uses Simple Mail Transmission Protocol (SMTP) to send mail.
- Settings of the SMTP mail can be done through "php.ini" file present in the PHP installation folder.
- Any text editor will be used to open and edit "php.ini"file.
- Locate [mail function] in the file.
- After [mail function]in the file following things will be displayed :
  - Don't remove the semi column if you want to work with an SMTP Server like Mercury
  - ; SMTP = localhost;
  - ; smtp_port = 25
- Remove the semi colons before SMTP and smtp_port and set the SMTP to your smtp server and the port to your smtp port. Your settings should look as follows :
  - SMTP = smtp.example.com
  - smtp_port = 25
  - We can get your SMTP settings from your web hosting providers.
  - If the server requires authentication, then add the following lines :
    1. auth_username = example_username@example.com
    2. auth_password = example_password
    3. Save the new changes.
    4. Restart server.

PHP Mail

Q. Describe the mail function.

- PHP mail is an inbuilt PHP function which is used to send emails from PHP scripts.
- It is a cost effective way of notifying users of important events.
- User's gets contact you via email by providing a contact us form on the website that emails the provided content.
- It can also be used to send email, to your newsletter subscribers, password reset links to users who forget their passwords, activation/confirmation links, registering users and verifying their email addresses

  Syntax : mail(to, subject, message, headers, parameters);

The mail function accepts the following parameters :

Parameters	Descriptions
to	Required. Specifies the receiver or receivers email ids.

subject	Required. Specifies the subject of the email. This parameter cannot contain any newline characters.
message	Required. Defines the message to be sent. Each line should be separated with a (\n). Lines should not exceed 70 characters.
headers	Optional. Specifies additional headers, like From, Cc, and Bcc.
parameters	Optional. Additional parameter to the send mail can be mentioned in this section.

Example :

```
<html>
<head>
<title>Email using PHP</title>
</head>
<body>
<?php
        $to = "xyz@abcdomain.com";
        $subject = "This is subject";
        $message = "<b>This is HTML message.</b>";
        $header = "From:abc@xyzdomain.com \r\n";
        $header .= "Cc:pqr@xyzdomain.com \r\n";
        $header .= "MIME-Version: 1.0\r\n";
        $header .= "Content-type: text/html\r\n";
        $retvalue = mail ($to,$subject,$message,$header);
if( $retvalue == true ) {
           echo "Message sent successfully...";
}else {
           echo "Message could not be sent...";
        }
     ?>
</body>
</html>
```

Output :

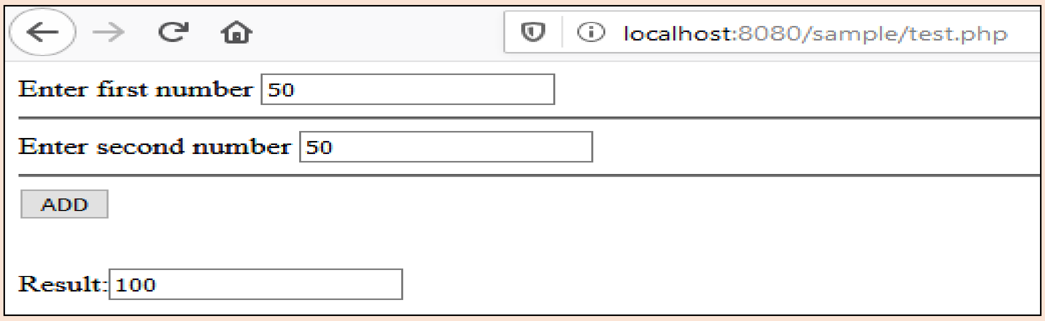
```
Message sent successfully...
```

Programs

1. Write a PHP program to add two numbers using text field.

```
<html>
<body>
<form method="post">
Enter first number <input type="text" name="fnum"/><hr/>
Enter second number <input type="text" name="snum"/><hr/>
<input type="submit"  name="add" value="ADD"/>
</form>
<br>
<?php
if(isset($_POST['add']))
{
$x=$_POST['fnum'];
$y=$_POST['snum'];
$sum=$x+$y;
echo "Result:<input type='text' value='$sum'/>";
}
?>
 </body>
</html>
```

Output :



localhost:8080/sample/test.php

```
Enter first number 50
Enter second number 50
ADD
Result: 100
```

Review Questions

Q. 1 How does the form information get from the browser to the server?

Q. 2 What is the difference between get and post methods?

Q. 3 Explain the superglobals in PHP?

Q. 4 Define session and cookies.

Q. 5 Define coockies. what is the need of it?

Q. 6 Define Coockie. How to create & delete it ? How can we retrieve a Cookie value?

Q. 7 What is the use of isset() function?

Q. 8 How does PHP handle forms with multiple selections?

Q. 9 What is the difference between Session and Cookie?

Q. 10 How can we destroy the cookie?

Q. 11 What is the use of session_start() ?

Q. 12 How to register a variable in PHP session ?

Q. 13 How to validate user input in PHP? Explain

Q. 14 Write a program to create a session, to set a value in session, and to remove data from a session.

Q. 15 Write a PHP code to find greatest of two numbers? Accept numbers from users.

Q. 16 Create Customer form like Customer name, Address, mobile no, item_purchase, Amount using different form input element & display user inserted values in new PHP form

Q. 17 Create Employee form like Employee name, Address, Mobile no, Date of birth, Post & Salary using different form input element & display user inserted values in same PHP form

Q. 18 Explain self processing forms in PHP with example

Q. 19 Write a PHP program that demonstrate form element (input elements).

Q. 20 Write a PHP program that demonstrate passing data using GET method.

Q. 21 Write a PHP program that demonstrate passing data using POST method.

Q. 22 A simple calculator web application that takes 2 numbers and an operator (+,-,*,/,%) from an HTML page and returns the result page with the operation performed on the operands.

Q. 23 Write a program that demonstrate use of cookies.

Q. 24 Write a PHP program that demonstrate use of session.

Q. 25 Write a simple PHP program to check that emails are valid.

❑❑❑