Unit -6 State Management

- **What is state management?**

  PHP allows us to save certain states of the application either on the server itself, or in the user's browser. PHP provides us with two different techniques to manage states in a web application:
- **Sessions**: Server Side State Management
- **Cookies**: Client Side State Management

Hidden Field

- A hidden field let web developers include data that cannot be seen or modified by users when a form is submitted.
- A hidden field often stores what database record that needs to be updated when the form is submitted.
- Syntax

< input type = “hidden” name=” hidden1” value=” php message” >

- Where,

TYPE — specifies that the field is hidden
NAME — specifies the name of the hidden field
VALUE – specifies the value as it appears on the form

- A hidden field is not displayed on the page.
- It simply stores the text value specified in the value attribute.
- Hidden fields are great for passing additional information from the form to the server.

Example:

```
<html>
<body>
<form action="Hiddenout.php" method="POST">
  First name: <input type="text" name="fname"><br>
  <input type="hidden" name="country" value="INDIA">
  <input type="submit" value="Submit">
</form>
<p>Notice that the hidden field above is not shown to a
   user.</p>
</body>
</html>
```

Hiddenout.php

```
<?php
echo "Fname:  ".$_POST["fname"] ; echo "<br/>";
echo "Country:  ".$_POST["country"];
?>
```

Query string

- The information can be sent across the web pages. This information is called **query string**.
- This query string can be passed from one page to another by appending it to the address of the page.
- You can pass more than one query string by inserting the & sign between the query strings.
- A query string can contain two things: the query string ID and its value.
- The query string passed across the web pages is stored in $_REQUEST, $_GET, or $_POST variable.
- Whether you passed the query string by using GET or POST method, it is stored in $_REQUEST variable. If you want to access the query string you can use these variables.
- You should note that whether the query string is passed by the GET or POST method it can be accessed by using the $_REQUEST variable.
- If you want to use $_GET variable to access the query string the form method need to be GET. Also you can use $_POST variable to get the query string if the form method is POST.

Client side State management System: Cookies

- Cookies are used for client-side state management system.
- Cookies are data by the browser, cookies are sent to the web server as a piece of header information with every HTTP request.
- Cookies can contain 1KB (1024B) size of data.
- **Uses of Cookies:**
- To store information about visitors regarding the accessed website's page.
- Number of visit and views.
- Store first visit information and update it in every visit that pointed towards better experience of user.

- *Cookies are not stored on the server, they can be modified and deleted. Cookies are less reliable and secure than sessions.*
- **How to set a cookie**

  PHP provides us with the setcookie() function to create, or set, a cookie.
- **Syntax:**

  setcookie(name, value, expiration, path, domain, secure);

- The first argument, **name**, is mandatory for every cookie. The rest of the arguments are optional, but recommended.

Argument	Usage
name	Required. The name the cookie will be referred to. Must be a string.
value	Optional. The value of the cookie.
expiration	Optional. If an expiration time is not set, the cookie will expire when the browser is closed.
path	Optional. The path on the server the cookie will be available on. The cookie can be set to '/' to be available to the entire domain.
domain	Optional. The (sub)domain that the cookie is available to. Sub-domains of the specified domain are automatically included.
secure	Optional. If set to true, the cookie will only be set for a HTTPS secure connection.

- **Example: set Cookies**

```
<?php // Expires when the browser closes
  setcookie("UserID", 007);
?>
```

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

```
<?php // set the cookie
 setcookie("username", “XYZ”); ?>
 <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>
```

- For **updating cookies** only, we need to change the argument by calling **setcookie()** function. We change name “XYZ” to “ABC”.

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

For **deleting cookies** we need to set expiry time in negative.

```
<?php
Setcookie("username", "xyz", time() - 3600);
?>
```

Note: Drawback of using cookies is it can easily retrieve and also easily deleted. It is not secure.

Server Side State Management:Session

- Session stores server-side information, so that all the information are accessible to all the webpages.
- It is more secure than cookies.
- We know that HTTP is a stateless protocol so that previously performed task cannot be remembered by current request.

- **Uses of Session:**
- It provides login and logout functionality to store and display relevant information.
- It maintains cart of e-commerce.

- **Creation of Session:** session starts from **session_start()** function and data can be set and get by using global variables **$_SESSION**.

```
<?php
   session_start();
  $SESSION["username"] = "abc";
  $SESSION["userid"] = "1";
?>
<html>
<body>
   <?php
    echo "Session variable is set";
    ?>
<button type="submit" href="logout.html"></button>
</body>
</html>
```

- Retrieve information to another pages:

```
<?php
echo "Username:" . " " . $username;
echo "Userid:" . " " . $userid;
?>
```

- For updating the value of session variable

```
$_SESSION["userid"]="1024";
```

- To destroy session in PHP, first unset all the session variable using **session_unset()** and call **session_destroy()** methods.

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

session_unset();
session_destroy();
?>
</body>
</html>
```

- **Form Value using Session**

```
<?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>
```