EXP-7 PHP- PART 2

Roll No.: K041	**Name:** Anish Sudhan Nair
Prog/Yr/Sem: B.Tech. Cybersecurity	**Batch:** K2/A2
Date of Experiment: 11/03/2022	**Date of Submission:** 14/03/2022

PHP - GET & POST Methods

There are two ways the browser client can send information to the web server.

- The GET Method
- The POST Method

Before the browser sends the information, it encodes it using a scheme called URL encoding. In this scheme, name/value pairs are joined with equal signs and different pairs are separated by the ampersand.

```
name1=value1&name2=value2&name3=value3
```

Spaces are removed and replaced with the + character and any other nonalphanumeric characters are replaced with a hexadecimal values. After the information is encoded it is sent to the server.

The GET Method

The GET method sends the encoded user information appended to the page request. The page and the encoded information are separated by the **?** character.

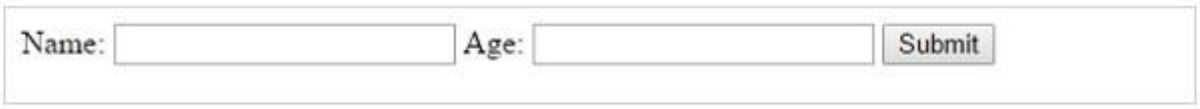
```
http://www.test.com/index.htm?name1=value1&name2=value2
```

- The GET method produces a long string that appears in your server logs, in the browser's Location: box.
- The GET method is restricted to send upto 1024 characters only.
- Never use GET method if you have password or other sensitive information to be sent to the server.
- GET can't be used to send binary data, like images or word documents, to the server.
- The data sent by GET method can be accessed using QUERY_STRING environment variable.
- The PHP provides **$_GET** associative array to access all the sent information using GET method.

Try out following example by putting the source code in test.php script.

```
<?php
   if( $_GET["name"] || $_GET["age"] ) {
      echo "Welcome ". $_GET['name']. "<br />";
      echo "You are ". $_GET['age']. " years old.";
      
      exit();
   }
?>
<html>
   <body>
   
      <form action = "<?php $_PHP_SELF ?>" method = "GET">
         Name: <input type = "text" name = "name" />
         Age: <input type = "text" name = "age" />
         <input type = "submit" />
      </form>
      
   </body>
</html>
```

It will produce the following result −



The POST Method

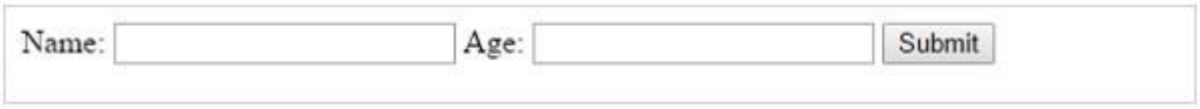
The POST method transfers information via HTTP headers. The information is encoded as described in case of GET method and put into a header called QUERY_STRING.

- The POST method does not have any restriction on data size to be sent.
- The POST method can be used to send ASCII as well as binary data.
- The data sent by POST method goes through HTTP header so security depends on HTTP protocol. By using Secure HTTP you can make sure that your information is secure.
- The PHP provides **$_POST** associative array to access all the sent information using POST method.

Try out following example by putting the source code in test.php script.

```
<?php
   if( $_POST["name"] || $_POST["age"] ) {
      if (preg_match("/[^A-Za-z'-]/",$_POST['name'] )) {
         die ("invalid name and name should be alpha");
      }
      echo "Welcome ". $_POST['name']. "<br />";
      echo "You are ". $_POST['age']. " years old.";

      exit();
   }
?>
<html>
   <body>

      <form action = "<?php $_PHP_SELF ?>" method = "POST">
         Name: <input type = "text" name = "name" />
         Age: <input type = "text" name = "age" />
         <input type = "submit" />
      </form>

   </body>
</html>
```

It will produce the following result −



The $_REQUEST variable

The PHP $_REQUEST variable contains the contents of both $_GET, $_POST, and $_COOKIE. We will discuss $_COOKIE variable when we will explain about cookies.

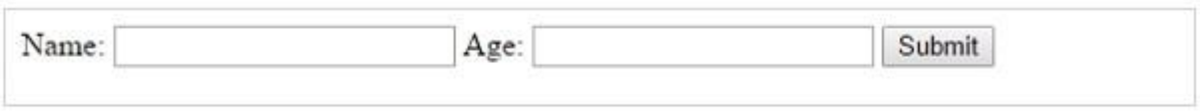
The PHP $_REQUEST variable can be used to get the result from form data sent with both the GET and POST methods.

Try out following example by putting the source code in test.php script.

```
<?php
   if( $_REQUEST["name"] || $_REQUEST["age"] ) {
      echo "Welcome ". $_REQUEST['name']. "<br />";
      echo "You are ". $_REQUEST['age']. " years old.";
      exit();
   }
?>
<html>
   <body>

      <form action = "<?php $_PHP_SELF ?>" method = "POST">
         Name: <input type = "text" name = "name" />
         Age: <input type = "text" name = "age" />
         <input type = "submit" />
      </form>

   </body>
</html>
```

Here $_PHP_SELF variable contains the name of self script in which it is being called.

It will produce the following result −



What is the Difference Between GET and POST Method in PHP?

GET vs POST Method in PHP	
GET is a method that sends information by appending them to the page request.	POST is a method that transfers information via HTTP header.
URL	
The form information is visible in the URL	The form information is not visible in the URL
Information Amount	
Limited amount of information is sent. It is less than 1500 characters.	Unlimited amount of information is sent.
Usage	
Helps to send non-sensitive data	Helps to send sensitive data (passwords), binary data (word documents, images)and uploading files
Security	
Not very secure.	More secure.
Bookmarking the Page	
Possible to bookmark the page	Not possible to bookmark the page

Cookie in PHP

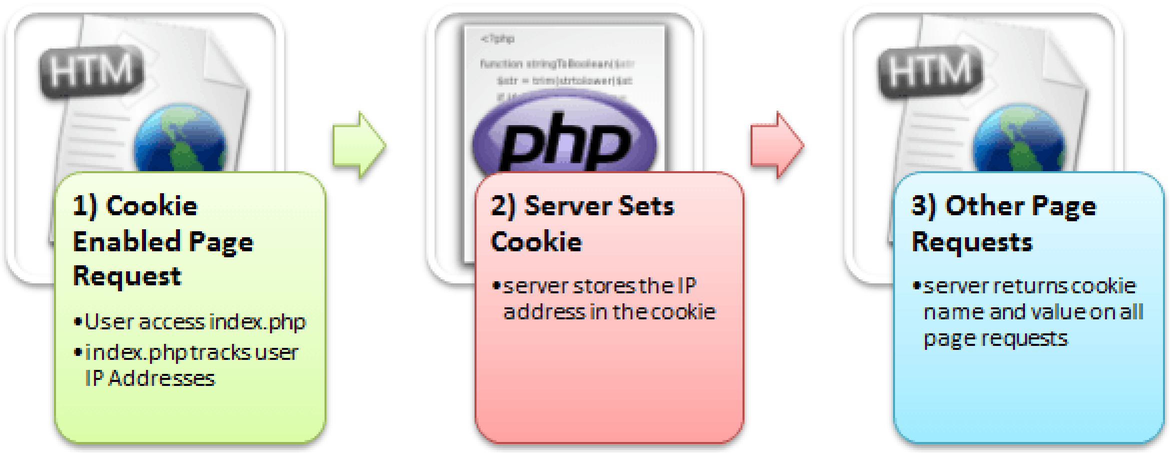
What is Cookie?

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

Once a cookie has been set, all page requests that follow return the cookie name and value.

A cookie can only be read from the domain that it has been issued from.

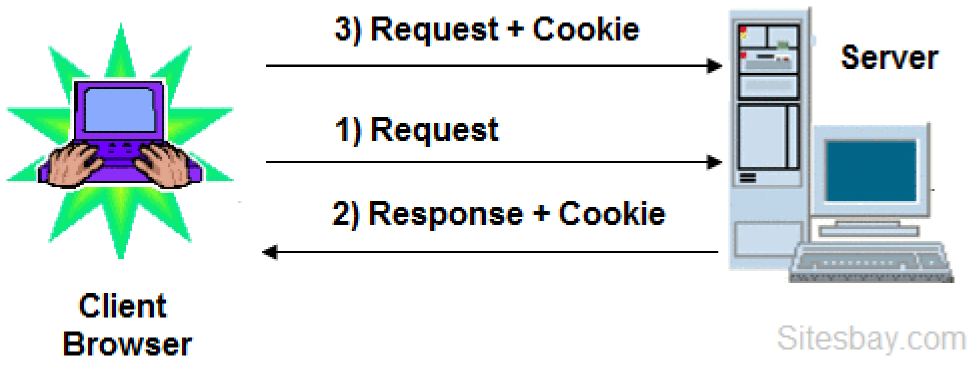
The diagram shown below illustrates how cookies work.



HTM
1) Cookie Enabled Page Request
•User access index.php
•index.php tracks user IP Addresses
php
2) Server Sets Cookie
•server stores the IP address in the cookie
HTM
3) Other Page Requests
•server returns cookie name and value on all page requests

Cookie is generally used to identify a user. Cookies are text files stored on the client computer and they are kept of use tracking purpose. PHP transparently supports HTTP cookies. Using PHP, you can both create and retrieve cookie values.

Note: PHP Cookie must be used before <html> tag.



There are three steps involved in identifying returning users;

- Server script sends a set of cookies to the browser. For example name, age, or identification number etc.
- Browser stores this information on local machine for future use.
- 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.

Create Cookies in PHP

A cookie is created in php using setcookie() function. Here only the name parameter is required. All other parameters are optional.

Syntax

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

Create/Retrieve a Cookie in PHP

The following example creates a cookie named **"user"** with the value **"Hitesh Kumar"**. The cookie will expire after 30 days (86400 * 30). The **"/"** means that the cookie is available in entire website (otherwise, select the directory you prefer).

We then retrieve the value of the cookie "user" (using the global variable $_COOKIE). We also use the isset() function to find out if the cookie is set:

Create Cookie

```
<?php
$cookie_name = "user";
$cookie_value = "Hitesh Kumar";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>

<!DOCTYPE html>
<html>
<body>
<?php
```

```
if(!isset($_COOKIE[$cookie_name]))
{
 echo "Cookie named '" . $cookie_name . "' is not set!";
} else
{
 echo "Cookie '" . $cookie_name . "' is set!
";
 echo "Value is: " . $_COOKIE[$cookie_name];
}

?>
<body>
<html>
```

Modify a Cookie Value

To modify a cookie, just set (again) the cookie using the setcookie() function

Modify Cookie

```
<?php
$cookie_name = "user";
$cookie_value = "Alex Porter";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/");
?>

<!DOCTYPE html>
<html>
<body>
<?php

if(!isset($_COOKIE[$cookie_name]))
{
  echo "Cookie named '" . $cookie_name . "' is not set!";
}
else
{
 echo "Cookie '" . $cookie_name . "' is set!
";
 echo "Value is: " . $_COOKIE[$cookie_name];
}

?>
<body>
<html>
```

Delete a Cookie in PHP

Delete Cookie

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time() - 3600);
?>

<!DOCTYPE html>
<html>
<body>
<?php

echo "Cookie 'user' is deleted.";

?>
<body>
<html>
```

Check Cookies are Enabled or not

To Check Cookies are Enabled or not, First Create a test cookie with the setcookie() function, then count the $_COOKIE array variable;

Check Cookie Enabled ?

```
<?php
setcookie("test_cookie", "test", time() + 3600, '/');
?>

<!DOCTYPE html>
<html>
<body>
```

```
<?php

if(count($_COOKIE) > 0)
{
 echo "Cookies are enabled.";
} else
{
 echo "Cookies are disabled.";
}

?>
<body>
<html>
```

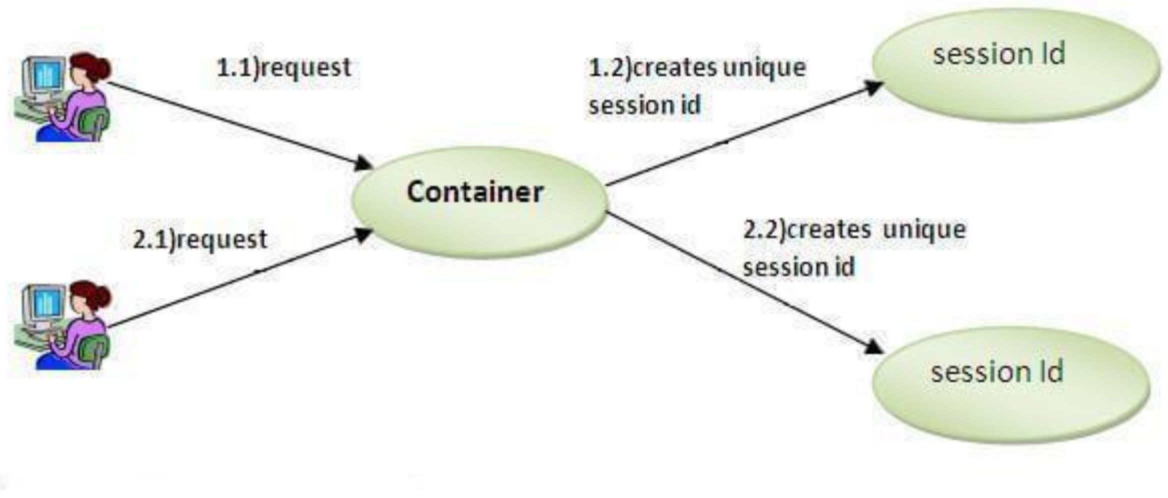
Session in PHP

PHP Session

PHP session is used to store and pass information from one page to another temporarily (until user close the website).

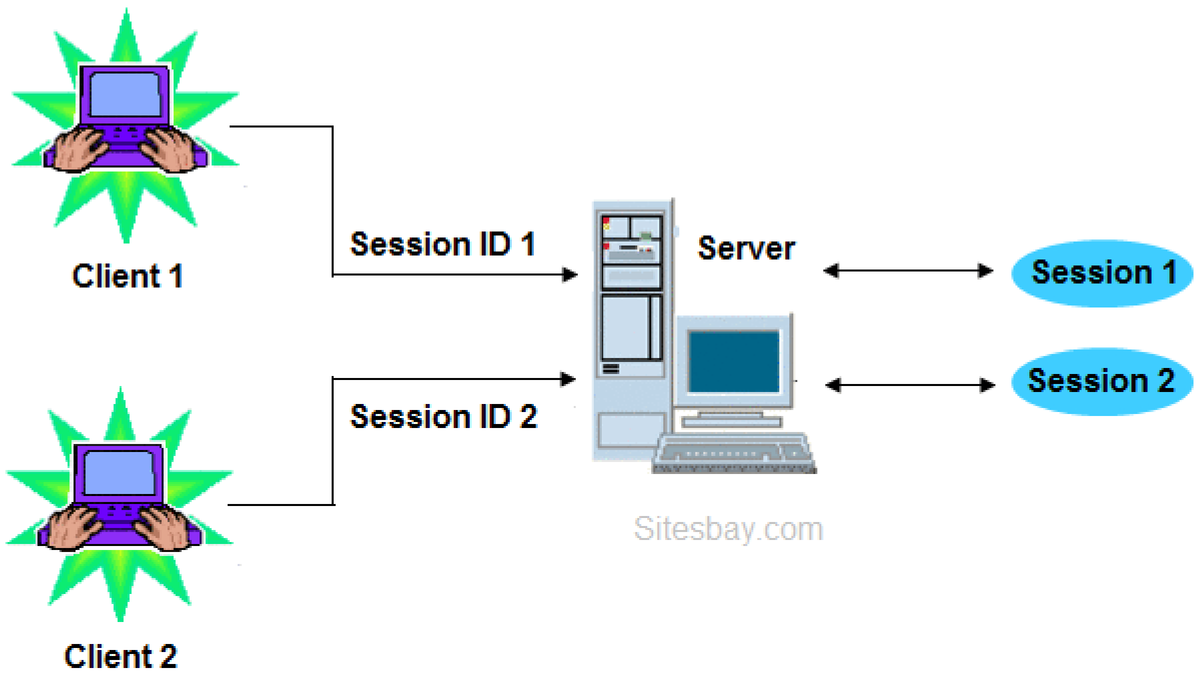
PHP session technique is widely used in shopping websites where we need to store and pass cart information e.g. username, product code, product name, product price etc from one page to another.

PHP session creates unique user id for each browser to recognize the user and avoid conflict between multiple browsers.



Create Session in PHP

Session is used to store and pass information from one page to another temporarily (until user close the website). In case of cookie, the information are store in user computer but in case of session information is not stored on the users computer.



Why use session in PHP

When you work with any application, you open it, do some changes, and then you close it. This is much like a Session. The computer knows who you are. It knows when you start the application and when you end. But on the internet there is one problem; the web server does not know who you are or what you do, because the HTTP address doesn't maintain state.

Session variables solve this problem by storing user information to be used across multiple pages (e.g. username, favorite color, etc). By default, session variables last until the user closes the browser. So; Session variables hold information about one single user, and are available to all pages in one application.

Start PHP Session

A session is started with the session_start() function. Session variables are set with the PHP global variable: **$_SESSION**.

session_start() function in PHP

PHP session_start() function is used to start the session. It starts a new or resumes existing session. It returns existing session if session is created already. If session is not available, it creates and returns new session.

Syntax

```
bool session_start ( void )
```

Example

```
session_start ()
```

Here we create a new web page with name myapp1.php

Start Session

```
<?php
// Start the session
session_start();
?>

<!DOCTYPE html>
```

```
<html>
<body>
<?php

// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";

?>
<body>
<html>
```

Note: The session_start() function must be declare top of your php page, Before any HTML tags.

Get PHP Session Variable Values

Now we create new page myapp2.php for get previous page session information. All session variable values are stored in the global $_SESSION variable

Get PHP Session Variable Values

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

<!DOCTYPE html>
<html>
<body>

<?php

// Echo session variables that were set on previous page
echo "Favorite color is " . $_SESSION["favcolor"] . ".<br>";
echo "Favorite animal is " . $_SESSION["favanimal"] . ".";

?>

<body>
```

```
<html>
```

Another way to get session

Get PHP Session Variable Values

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

<!DOCTYPE html>
<html>
<body>

<?php
print_r($_SESSION);
?>

<body>
<html>
```

The code above will get an error like this

```
Message: Value must be 1 or below
```

Modify a PHP Session Variable

To change a session variable, just overwrite it:

Modify a PHP Session Variable

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

<!DOCTYPE html>
<html>
```

```
<body>

<?php

// to change a session variable, just overwrite it
$_SESSION["favcolor"] = "yellow";
print_r($_SESSION);

?>

<body>
<html>
```

Destroy Session in PHP

To remove all global session variables and destroy the session, use session_unset() and session_destroy() method.

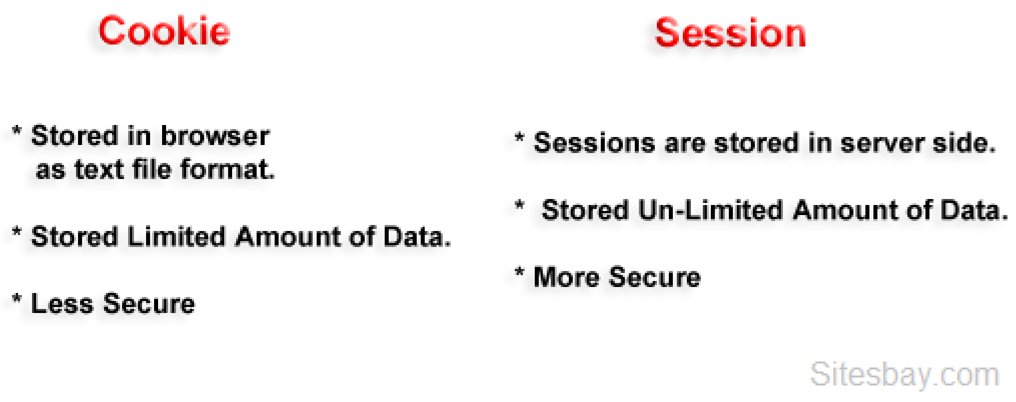
Destroy Session

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

<!DOCTYPE html>
<html>
<body>
<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();
?>
<body>
<html>
```

Difference Between Session and Cookie in PHP

Cookies are stored in browser as a text file format. It is stored limit amount of data.It is only allowing 4kb[4096bytes]. It is not holding the multiple variable in cookies.

Sessions are stored in server side. It is stored unlimited amount of data.It is holding the multiple variable in sessions. we cannot accessing the cookies values in easily.So it is more secure.



Cookie

* Stored in browser
  as text file format.

* Stored Limited Amount of Data.

* Less Secure

Session

* Sessions are stored in server side.

* Stored Un-Limited Amount of Data.

* More Secure

Sitesbay.com

Cookies	Sessions
Cookies are stored in browser as text file format.	Sessions are stored in server side.
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.
setting the cookie time to expire the cookie.	using session_destory(), we 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.

Example of Cookie

Example of Cookie

```
<?php

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

  $cookie_uame = "user";

  $cookie_uvalue= "Hitesh Kumar";

  //set cookies for 1 hour time
  setcookie($cookie_uname,
$cookie_uvalue, 3600, "/");

  //expire cookies
setcookie($cookie_uname,"",-3600);

?>
```

Example of Session

Example of Session

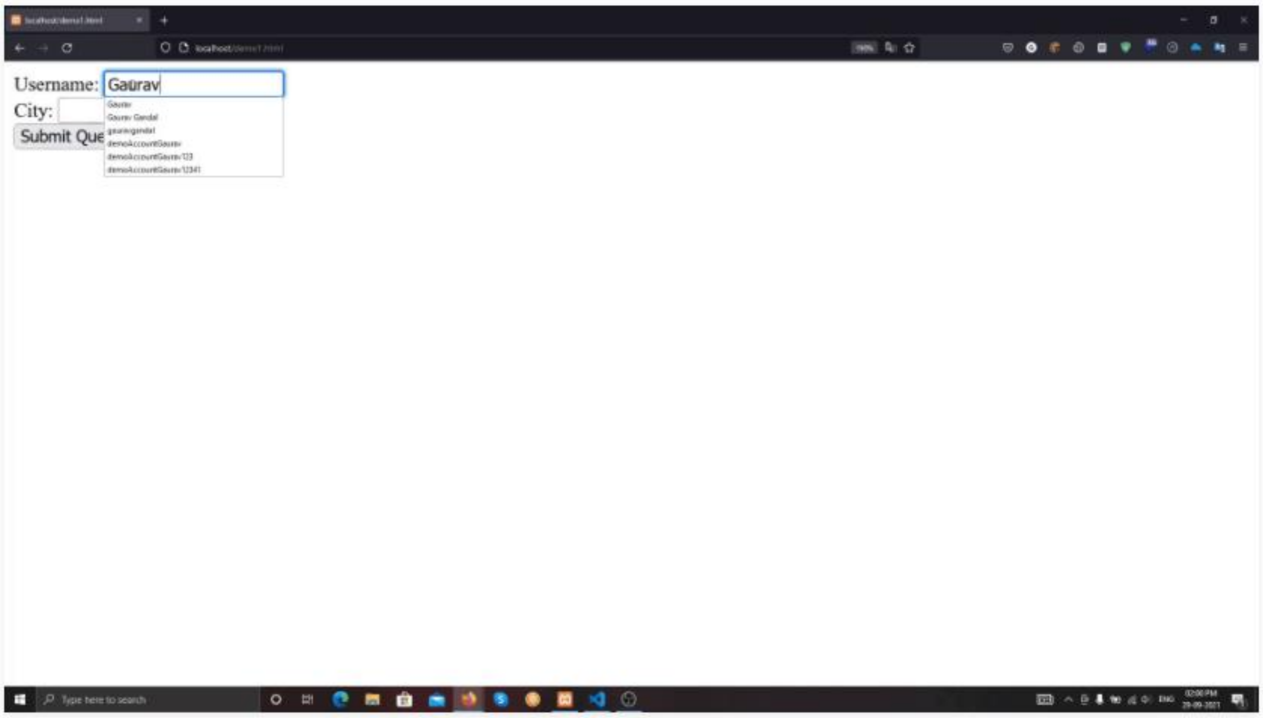
```
<?php

session_start();

//session variable
$_SESSION['user'] = 'Hitesh';

//destroyed the entire sessions

session_destroy();

//Destroyed the session
variable "user".
unset($_SESSION['user']);
?
```

PART B

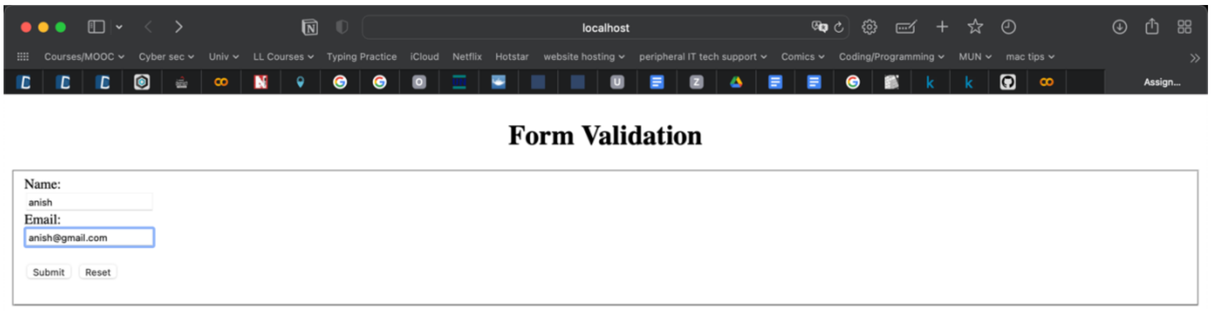
EXCERSICE

1 Using Get And Post Methods. Design a Form As Shown Below And Pass The Data As Per Form In Fig .

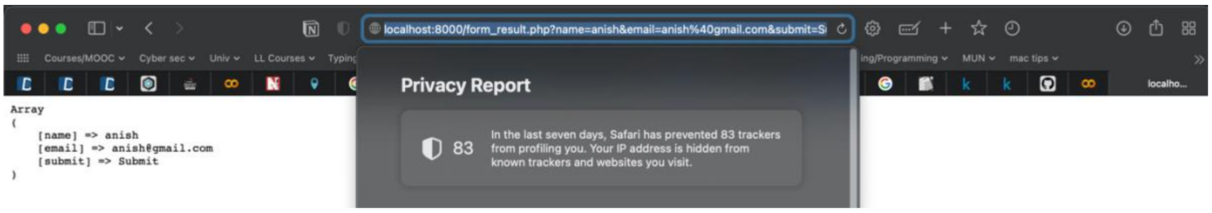
Also Discuss Your Observation While Applying These 2 Different Methods.


Username: Gaurav
City:
Submit Query

GET Method


Form Validation

Name: anish
Email: anish@gmail.com
Submit Reset

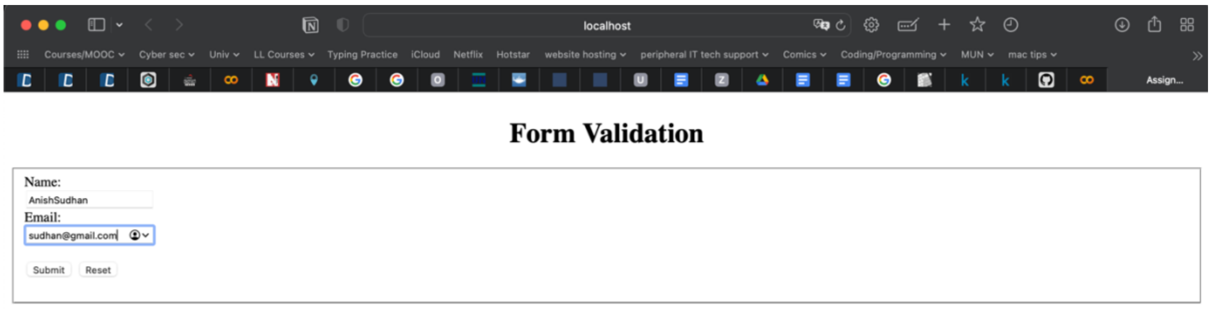

localhost:8000/form_result.php?name=anish&email=anish%40gmail.com&submit=S

```
Array
(
    [name] => anish
    [email] => anish@gmail.com
    [submit] => Submit
)
```

Privacy Report

83 In the last seven days, Safari has prevented 83 trackers from profiling you. Your IP address is hidden from known trackers and websites you visit.

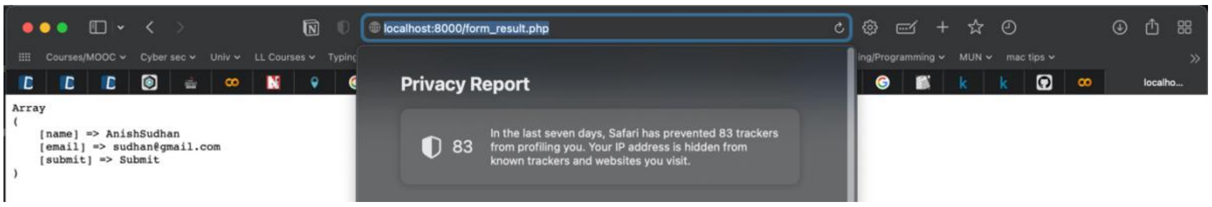
POST Method



Form Validation

Name:
AnishSudhan
Email:
sudhan@gmail.com

Submit Reset



```
Array
(
    [name] => AnishSudhan
    [email] => sudhan@gmail.com
    [submit] => Submit
)
```

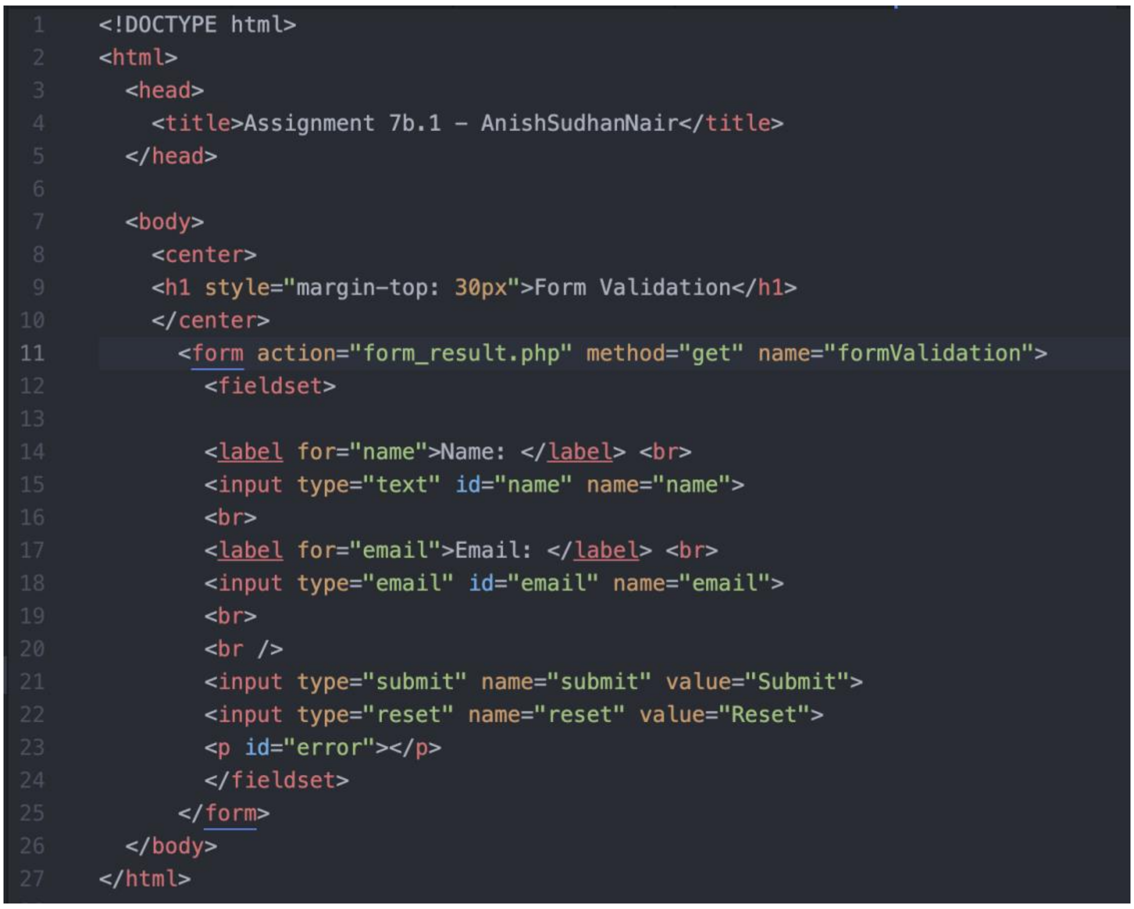
Privacy Report

83 In the last seven days, Safari has prevented 83 trackers from profiling you. Your IP address is hidden from known trackers and websites you visit.

In GET method, the entered data is displayed in the URL while in POST method, it is completely hidden.

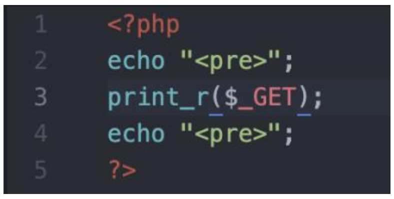
Code:

For Get



```
<!DOCTYPE html>
<html>
  <head>
    <title>Assignment 7b.1 - AnishSudhanNair</title>
  </head>

  <body>
    <center>
    <h1 style="margin-top: 30px">Form Validation</h1>
    </center>
      <form action="form_result.php" method="get" name="formValidation">
        <fieldset>

        <label for="name">Name: </label> <br>
        <input type="text" id="name" name="name">
        <br>
        <label for="email">Email: </label> <br>
        <input type="email" id="email" name="email">
        <br>
        <br />
        <input type="submit" name="submit" value="Submit">
        <input type="reset" name="reset" value="Reset">
        <p id="error"></p>
        </fieldset>
      </form>
  </body>
</html>
```

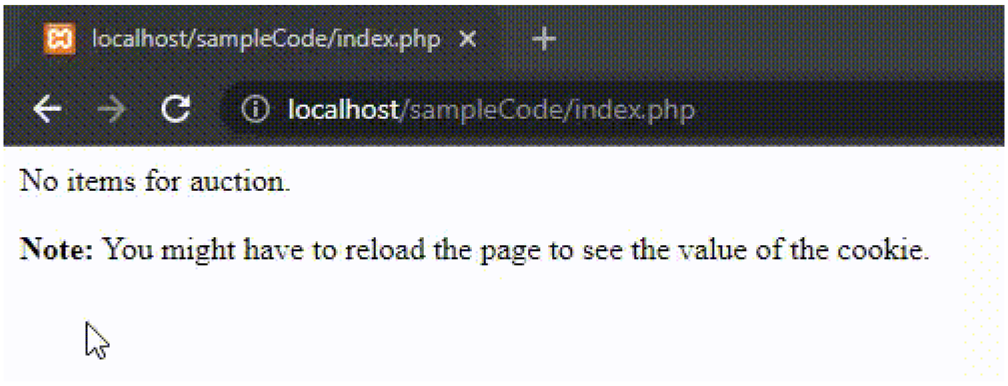
form_result.php



```php
<?php
echo "<pre>";
print_r($_GET);
echo "<pre>";
?>
```

2.

Create a cookie named Auction_Item and assigning the value Luxury Car to it. The cookie will expire after 2 days(2 days * 24 hours * 60 mins * 60 seconds). Once expired following message should display in browser.



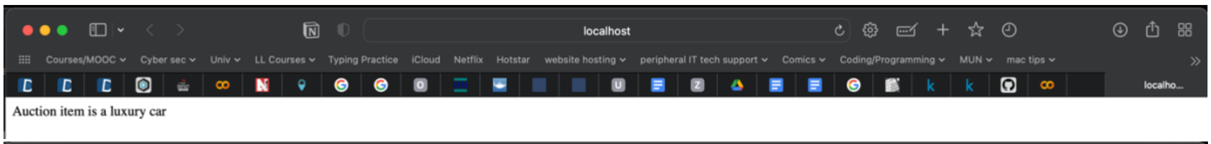
localhost/sampleCode/index.php

No items for auction.

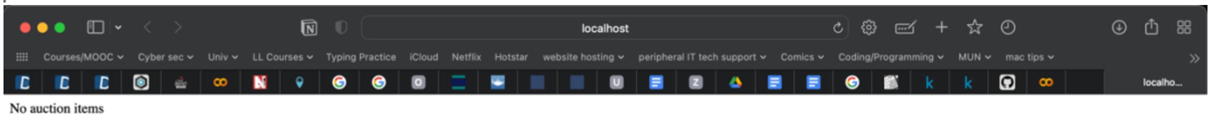
Note: You might have to reload the page to see the value of the cookie.

2.

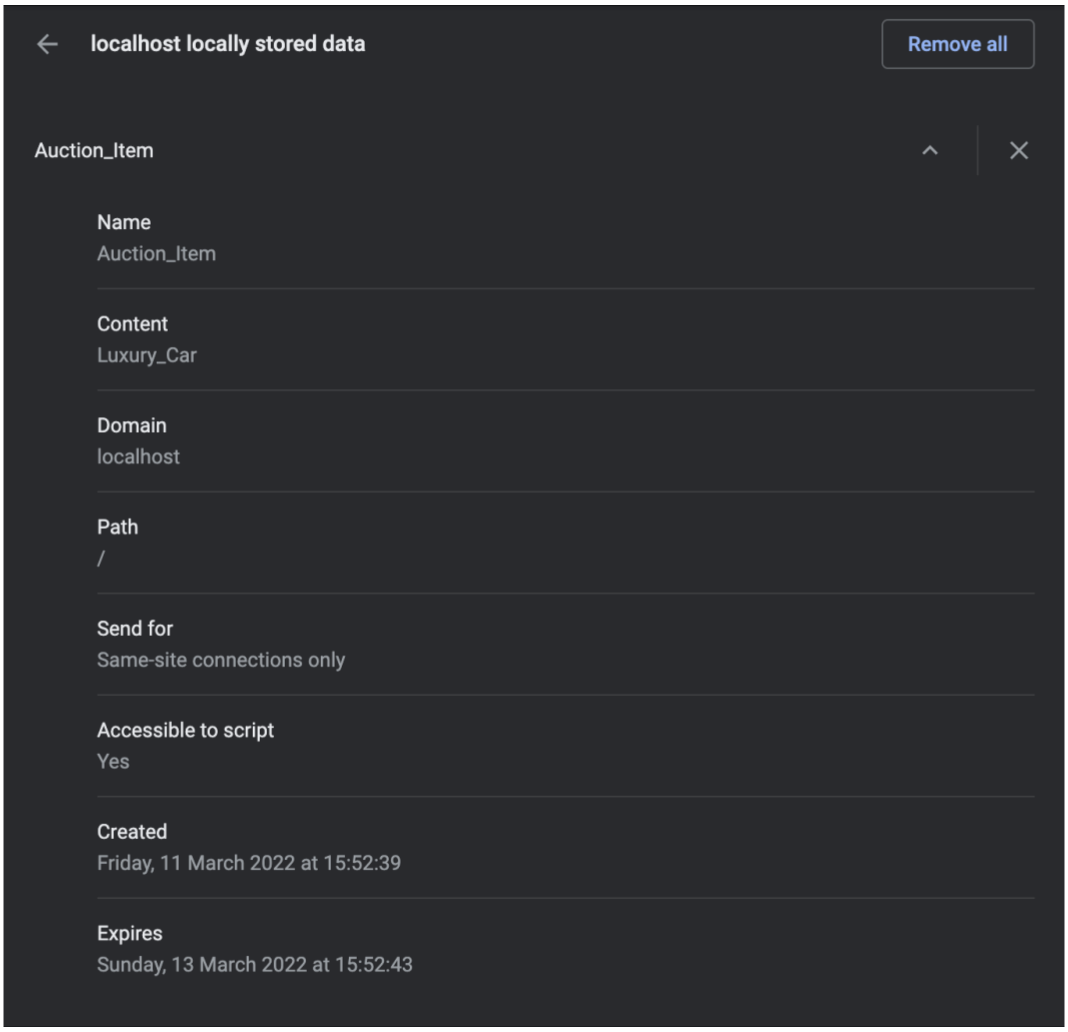
Output:



Auction item is a luxury car



No auction items


localhost locally stored data

Remove all

Auction_Item

Name
Auction_Item

Content
Luxury_Car

Domain
localhost

Path
/

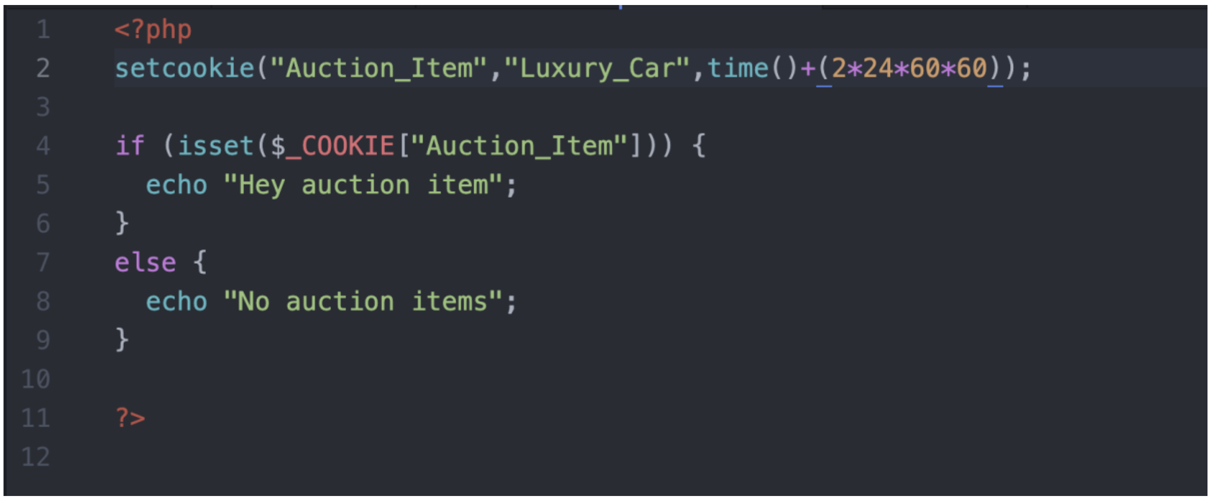
Send for
Same-site connections only

Accessible to script
Yes

Created
Friday, 11 March 2022 at 15:52:39

Expires
Sunday, 13 March 2022 at 15:52:43

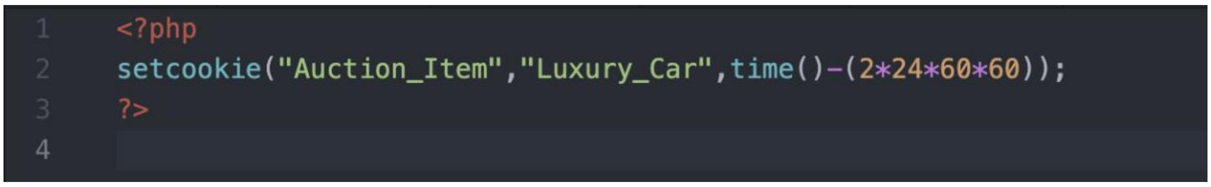
Code:


```
<?php
setcookie("Auction_Item","Luxury_Car",time()+(2*24*60*60));

if (isset($_COOKIE["Auction_Item"])) {
  echo "Hey auction item";
}
else {
  echo "No auction items";
}

?>

```

3. Write a code to delete the cookies defined in above exercise. (delete a cookie named “Auction_Item)

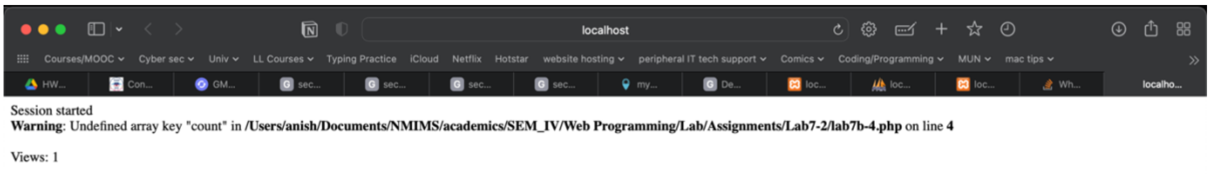
The setcookie() function can be used to delete a cookie. For deleting a cookie, the setcookie() function should be called by passing the cookie name and other arguments or empty strings but however this time, the expiration date is required to be set in the past.

Code:



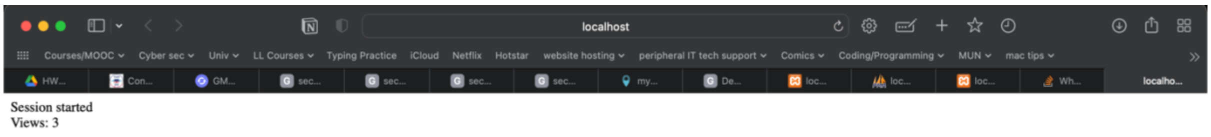
```php
<?php
setcookie("Auction_Item","Luxury_Car",time()-(2*24*60*60));
?>
```

4. Write PHP Code to find number of visitor visited particular webpage using session.



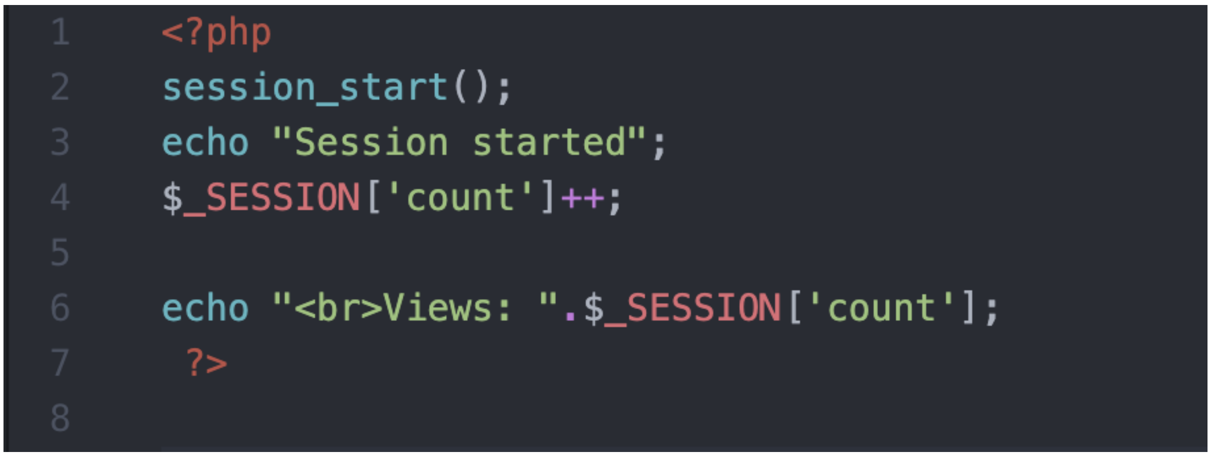
```
Session started
Warning: Undefined array key "count" in /Users/anish/Documents/NMIMS/academics/SEM_IV/Web Programming/Lab/Assignments/Lab7-2/lab7b-4.php on line 4

Views: 1
```

On refreshing twice



```
Session started
Views: 3
```

Code



```php
<?php
session_start();
echo "Session started";
$_SESSION['count']++;

echo "<br>Views: ".$_SESSION['count'];
 ?>

```