We innovate. We build. We serve.

Our Mission

The University shall primarily provide advanced instruction and professional training in the arts, philosophy, social sciences, agriculture and fishery, forestry, science and technology engineering, education, law and other related fields. It shall also undertake research and extension services and provide progressive leadership in its areas of specialization.

Section 2, R.A. 9313

Our Vison

A transformative University committed to technology, innovation, and service excellence

Board Resolution No. 41, s. 2014

BACHELOR OF SCIENCE IN INFORMATION TECHNOLOGY

WD 101 - WEB DEVELOPMENT & APPLICATION

Sonny Boy R. Ellema, Jr.

College of Arts & Sciences

Unit 7
PHP Advance

7.0 Learning Outcomes

After completing this chapter, you will be able to:

a. Describe how to use Session in PHP which very vital in creating login form.
b. Create a program that can handle PHP Session

7.1 Introduction

Although you can store data using cookies but it has some security issues. Since cookies are stored on user's computer it is possible for an attacker to easily modify a cookie content to insert potentially harmful data in your application that might break your application.

Also every time the browser requests a URL to the server, all the cookie data for a website is automatically sent to the server within the request. It means if you have stored 5 cookies on user's system, each having 4KB in size, the browser needs to upload 20KB of data each time the user views a page, which can affect your site's performance.

You can solve both of these issues by using the PHP session. A PHP session stores data on the server rather than user's computer. In a session based environment, every user is identified through a unique number called session identifier or SID. This unique session ID is used to link each user with their own information on the server like emails, posts, etc.

7.2 PHP Session

An alternative way to make data accessible across the various pages of an entire website is to use a PHP Session.

A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit.

The location of the temporary file is determined by a setting in the **php.ini** file called **session.save_path**. Before using any session variable make sure you have setup this path.

When a session is started following things happen −

- PHP first creates a unique identifier for that particular session which is a random string of 32 hexadecimal numbers such as 3c7foj34c3jj973hjkop2fc937e3443.
- A cookie called **PHPSESSID** is automatically sent to the user's computer to store unique session identification string.
- A file is automatically created on the server in the designated temporary directory and bears the name of the unique identifier prefixed by sess_ ie sess_3c7foj34c3jj973hjkop2fc937e3443.

When a PHP script wants to retrieve the value from a session variable, PHP automatically gets the unique session identifier string from the PHPSESSID cookie and then looks in its

temporary directory for the file bearing that name and a validation can be done by comparing both values.
A session ends when the user loses the browser or after leaving the site, the server will terminate the session after a predetermined period of time, commonly 30 minutes duration.

Starting a PHP Session

A PHP session is easily started by making a call to the **session_start()** function.This function first checks if a session is already started and if none is started then it starts one. It is recommended to put the call to **session_start()** at the beginning of the page.
Session variables are stored in associative array called **$_SESSION[]**. These variables can be accessed during lifetime of a session.
The following example starts a session then register a variable called **counter** that is incremented each time the page is visited during the session.
Make use of **isset()** function to check if session variable is already set or not.
Put this code in a test.php file and load this file many times to see the result −

```
<?php
  session_start();

  if( isset( $_SESSION['counter'] ) ) {
    $_SESSION['counter'] += 1;
  }else {
    $_SESSION['counter'] = 1;
  }

  $msg = "You have visited this page ".  $_SESSION['counter'];
  $msg .= "in this session.";
?>

<html>

  <head>
    <title>Setting up a PHP session</title>
  </head>

  <body>
    <?php  echo ( $msg ); ?>
  </body>

</html>
```

It will produce the following result − You have visited this page 1in this session.

Destroying a PHP Session

A PHP session can be destroyed by **session_destroy()** function. This function does not need any argument and a single call can destroy all the session variables. If you want to destroy a single session variable then you can use **unset()** function to unset a session variable. Here is the example to unset a single variable −

```
<?php
  unset($_SESSION['counter']);
?>
```

Here is the call which will destroy all the session variables −

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

Turning on Auto Session

You don't need to call start_session() function to start a session when a user visits your site if you can set **session.auto_start** variable to 1 in **php.ini** file.

Start a PHP Session

A session is started with the session_start() function.
Session variables are set with the PHP global variable: $_SESSION.
Now, let's create a new page called "demo_session1.php". In this page, we start a new PHP session and set some session variables:

```
<?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>
```

Session variables are set.

Note: The session_start() function must be the very first thing in your document. Before any HTML tags.

Get PHP Session Variable Values

Next, we create another page called "demo_session2.php". From this page, we will access the session information we set on the first page ("demo_session1.php").
Notice that session variables are not passed individually to each new page, instead they are retrieved from the session we open at the beginning of each page (session_start()).
Also notice that all session variable values are stored in the global $_SESSION variable:

```
<?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>
```

```
Favorite color is .
Favorite animal is .
```

Another way to show all the session variable values for a user session is to run the following code:

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

<?php
print_r($_SESSION);
?>

</body>
</html>
```

```
Array ( )
```

How does it work? How does it know it's me?

Most sessions set a user-key on the user's computer that looks something like this: 765487cf34ert8dede5a562e4f3a7e12. Then, when a session is opened on another page, it scans the computer for a user-key. If there is a match, it accesses that session, if not, it starts a new session.

Modify a PHP Session Variable

To change a session variable, just overwrite it:

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

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

</body>
</html>
```

```
Array ( [favcolor] => yellow )
```

Destroy a PHP Session

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

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

<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();

echo "All session variables are now removed, and the
session is destroyed."
?>

</body>
</html>
```

```
All session variables are now removed, and the session is destroyed.
```

7.3 References

W3Schools HTML5 (http://www.w3schools.com/)
W3Schools CSS3 (http://www.w3schools.com/)
W3Schools PHP (http://www.w3schools.com/)
Ian Lunn. CSS3 Foundations, A John Wiley and Sons, Ltd, Publication 2013
Matt West. HTML5 Foundations, A John Wiley and Sons, Ltd, Publication 2013
Learning web design : a beginner's guide to HTML, CSS, Javascript, and web graphics by Niederst Robbins, Jennifer, Ltd, Publication 2014
Tutorialspoint (https://www.tutorialspoint.com/index.htm)

7.4 Acknowledgement

All the figures and information presented in this module were taken from the references enumerated above.

SAMAR STATE UNIVERSITY
ARTECHE BOULEVARD, CATBALOGAN CITY
6700 SAMAR

www.ssu.edu.ph

email: info@ssu.edu.ph
telefax: (055) 543-8394