**Lab No. 02 –Cookies and Sessions**

In this lab, we try to demonstrate the use of cookies and sessions.

**Part1. Cookies and Sessions**

1. **Cookie**

A cookie is a small piece of information that scripts can store on a client-side machine.

* You can set a cookie on a user’s machine by calling the function **setcookie** before any HTML output is sent.
* Cookie variables are stored in the super global array **$\_COOKIE**.
* The function **setcookie** is defined as follow:

**setcookie(name, value, [expire]);**

Where,

* **Name** argument represents a unique name for a particular cookie.
* **Value** argument is a string value attached to the created cookie.
* **Expire** argument is optional and represents the expiration time of the cookie.
* Two types of cookies are possible: temporary and persistent.
* Temporary cookie expires when the client closes the browser, for example:

**setcookie(“user”,”ahmad”);**

* Persistent cookie expires after a predefined time, for example:

**setcookie(“user”,”ahmad”,time()+360);**

* This cookie will expire after 6 minutes from now.
* To delete a cookie, we can set the cookie with a date that you are sure has already expired, for example:

**setcookie(“user”,”ahmad”,time()-60);**

1. **Session**

* A session can be defined as a series of related interactions between a single client and the Web server.
* A session allows easily creating multi page forms, saving user authentication information from page to page, and storing persistent user preferences on a site.
* Using session, we store only a **session ID** on the client and store state values on the server.
* Session variables are stored in the super global array **$\_SESSION**.

The basic steps of using sessions are:

1. Starting a session
2. Registering session variables
3. Using session variables
4. Deregistering variables and destroying session

PHP scripting language provides the following useful functions to manipulate sessions:

|  |  |
| --- | --- |
| **Function** | **Description** |
| **session\_start()** | Enables a session for a page. This function assigns a new session ID to the new session. This function must be called before any HTML output is sent because session is a header data. |
| **session\_register()** | Registers a variable with the session by passing the name of the variable. When a session is started, you can store any number of variables in the $\_SESSION super global array and then access them on any session enabled page. |
| **session\_id( )** | Returns the current session ID |
| **session\_unregister(name)** | unregisters the global variable named *name* from the current session |
| **session\_unset()** | Unset all of the session variables |
| **session\_destroy( )** | Removes the data store for the current session without removing cookie from the browser cache. |

**Remark.** There are two ways used to pass variables through the navigation between web pages, which are Cookies, sessions. For example to pass variables between two web pages, you can put the variable into session or cookie in the first page and get it back from session or cookie in the next page.

**Question1.** To demonstrate the use of session, we’ll implement a set of three web pages. On the first page, we’ll start a session and register the variable **$\_SESSION[‘sess\_var’]**. The code of the script **page1.php** used to create the first page will look like this:

<?php

session\_start();

$\_SESSION['sess\_var']="Hello World!";

echo "the content of sess\_var is ",$\_SESSION['sess\_var'],"<br>";

?>

<a href="page2.php">Page2</a>

The expected output of **page1.php**:

The content of sess\_var is Hello World!

Page2

Now, we create the script **page2.php** to start a new session, access the previously stored value of the variable sess\_var and unset it. The code of **page2.php** is given as follow:

<?php

session\_start();

echo "the content of sess\_var is ",$\_SESSION['sess\_var'],"<br>";

unset($\_SESSION[‘sess\_var’] ) ;

?>

<a href="page3.php">Page3</a>

The expected output of **page2.php**:

The content of sess\_var is Hello World!

Page3

Now, we pass along to **page3.php** to destroy the whole session. The code of **page3.php** is given as follow:

<?php

session\_start();

echo "the content of sess\_var is ",$\_SESSION['sess\_var'],"<br>";

session\_destroy();

?>

<a href="page1.php">Page1</a>

The expected output of **page3.php**:

The content of sess\_var is

Page1

**Part 2 –MYSQL and PHP**

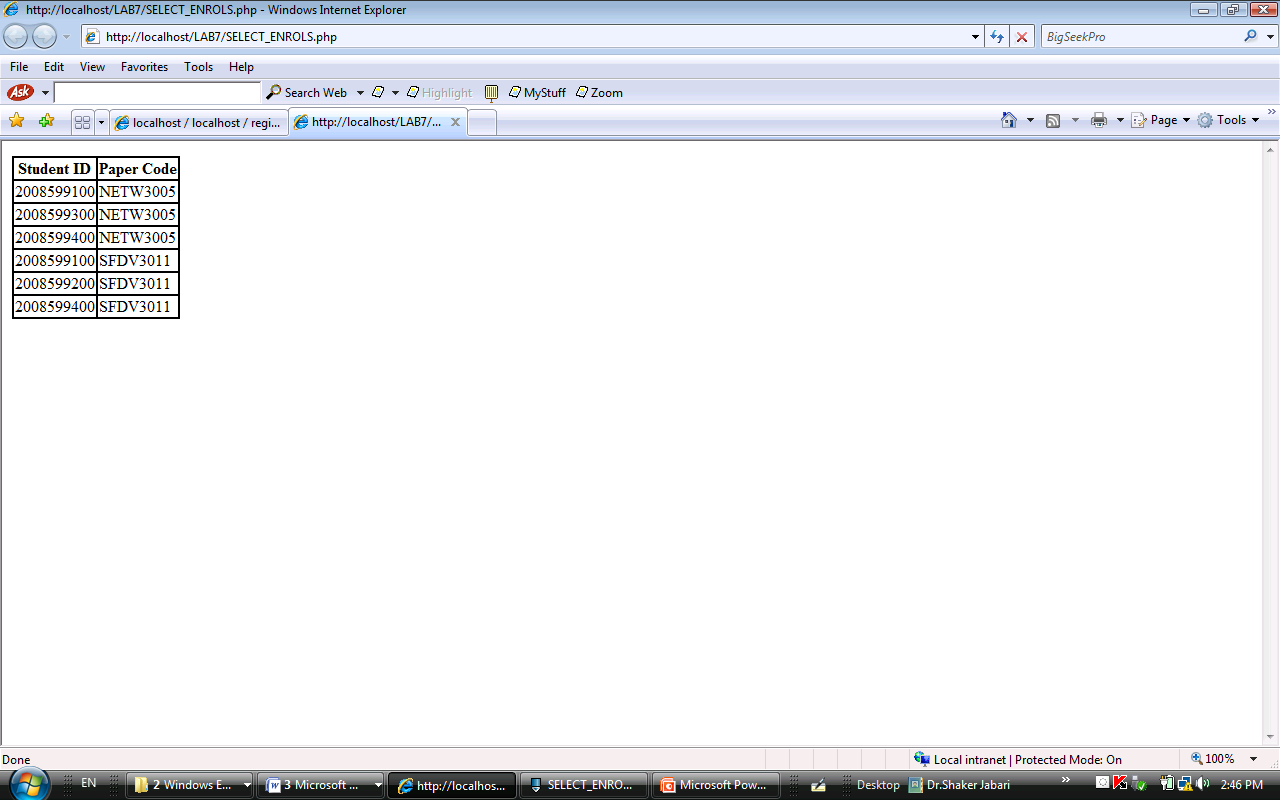
The goal of this task is to be able to understand how to connect php with database. However, the reference is not provided here – therefore you are recommended to watch the videos and then do this part of the lab.

Requirement of Question 1

Create a database with a table having Student ID and Paper Code Tables.

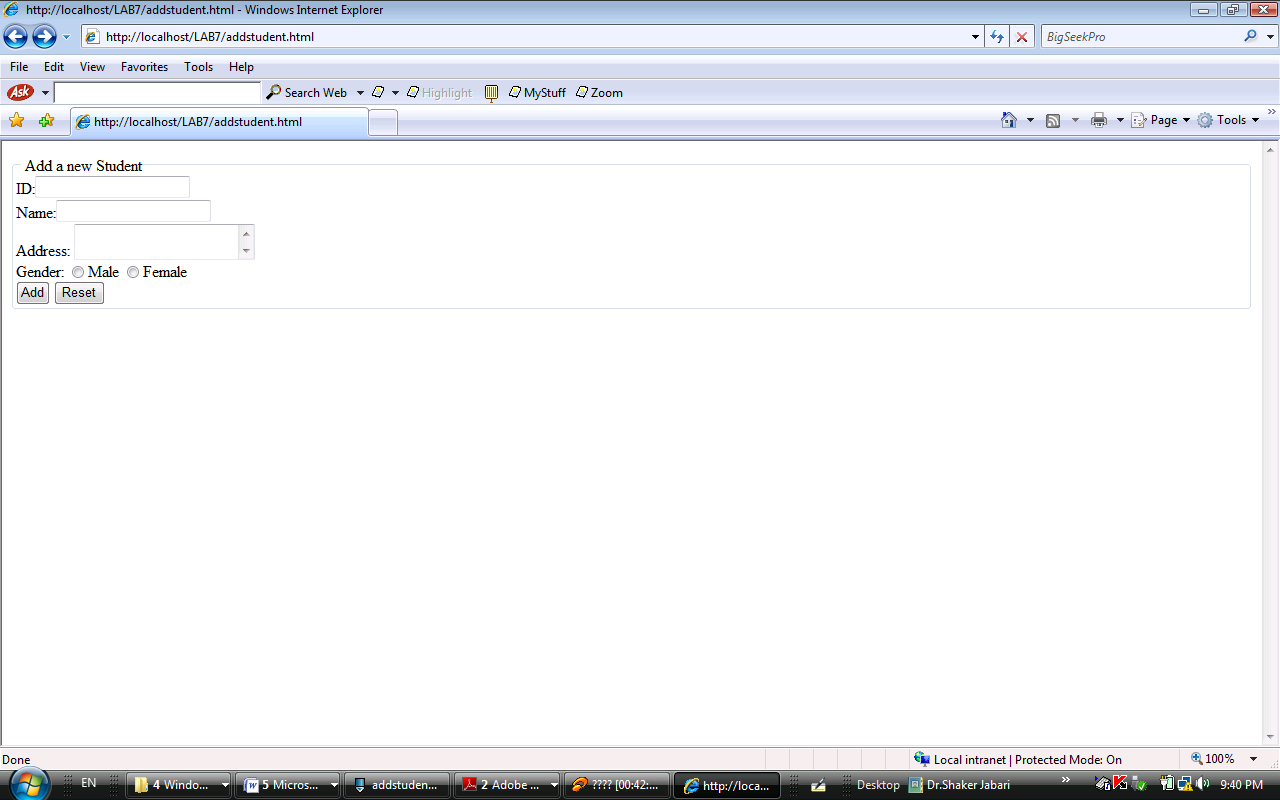
Inser the data that you see in the following html file and then do Q1.

**Question1.** Write a PHP script (**display.php**) to display the list of all enrolments. (Display the result as an HTML table as shown below).



**Question2.**

1. Create a form (**addstudent.html**) to insert a new student as shown below.



* The ID attribute of the input field should have the value “id”
* The Name attribute of the input field should have the value “name”
* The Address attribute of the textarea field should have the value “address”
* The Gender attribute of the radio field should have the value “gender”
* The action of the form should be ***addstudent.php***
* The method of the form should be ***POST***.

1. Now, write the PHP script (**addstudent.php**) to process the form created in question 1 (insert the record in the table student).

Now read the following tutorial

<http://wiki.hashphp.org/PDO_Tutorial_for_MySQL_Developers>

and then do the Q3.

**Question3.** Re-write the previous PHP scripts (**display.php** and **addstudent.php**) using PDO.