Lab 9: Introduction to PHP

Learning Outcomes

At the end of this lesson, students should be able to:

a) use the PHP Basic syntax for variable types and calculations.
b) write the PHP program using conditional statements, arrays, looping, and functions to build interactive Web sites.

Introduction

PHP, an open-source product is among the most stable and efficient tools available for creating dynamic, interactive web pages. PHP is available for multiple operating systems and Web servers. When a browser requests an HTML document that includes PHP script, the Web server that provides the document calls the PHP processor. The server determines that a document includes PHP script by the filename extension. For example, if it is *.php, *.php3 or *.phtml, it has embedded the PHP. This chapter provides the students an overview of absolute basics for PHP server-side scripting. Within these practical exercises, the student will find out how to create dynamic Web page using PHP scripting, how to display PHP output and working with PHP‟s built in variables, data types and operators.

Because PHP is a server-side technology, it is able to create, read, and write files on the server system. In fact, PHP can deal with files residing on any server system on the Internet, using both HTTP and FTP protocols. Cookies provide a means for a Web server to induce a client to store information about itself which can subsequently be called up by the Web server when required. Cookies cannot be fully depended because some users set their browsers to reject all cookies. Furthermore, most browser have limit on the number of cookies that will be accepted from a particular server site. Session tracking is an alternative to cookies for storing information on the client. Session can store unlimited amount of information, whereas a cookie can store just one name/value pair. Session objects live only as long the session lasts, that is only as long as the session in which they are created.

1.1 Hands-on Exercises

1.1.1 Testing the Installation of PHP

1. Create a sub folder in the server root directory of your local server. For example, if you have installed the local server using XAMPP, the server root directory is ‘C:\Program Files\Apache Group\Apache2\htdocs’. The folder you are creating will be your working folder to store all the files you create during the lab.

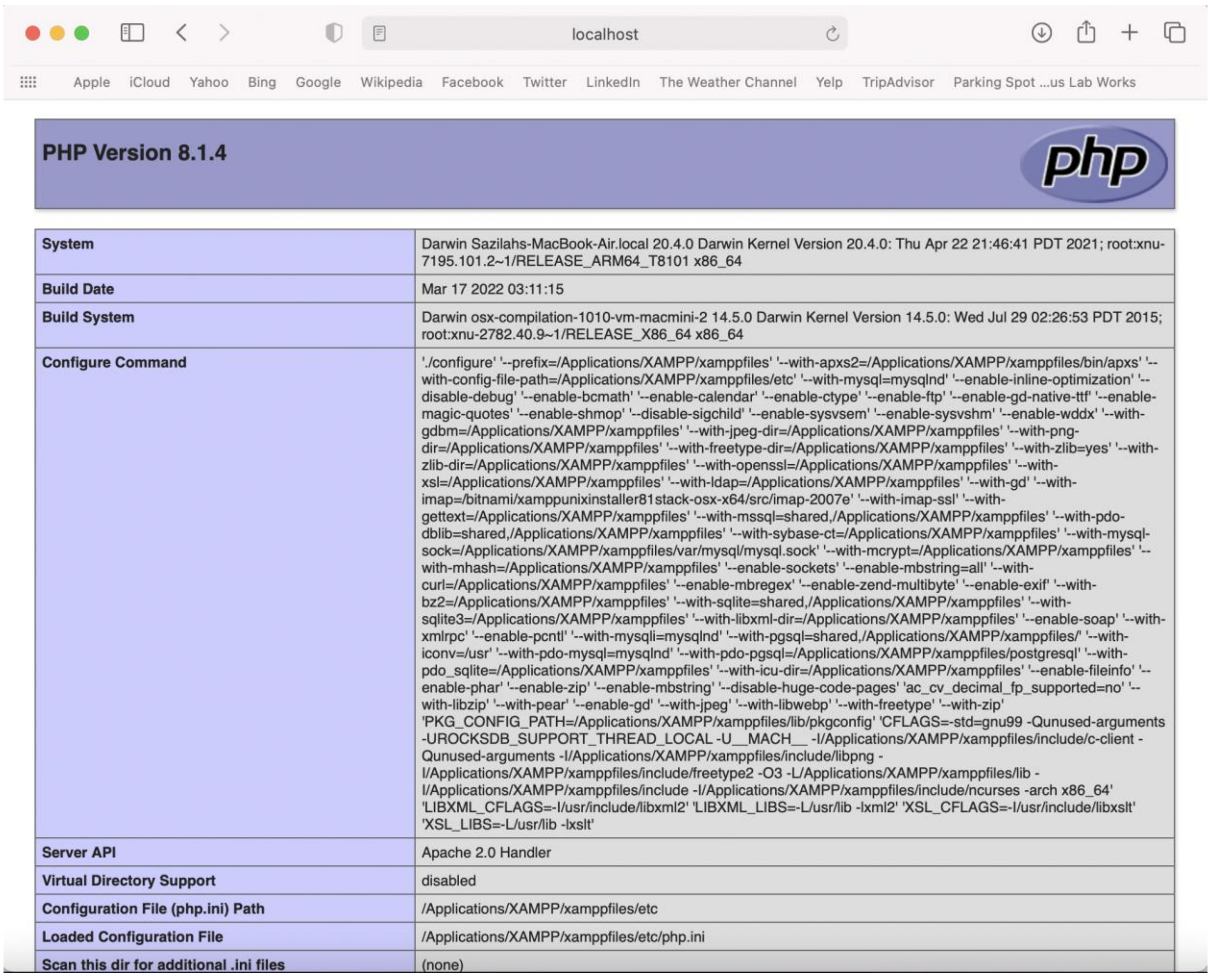
2. Open a text editor and enter the following code:

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

3. Save the php file as ‘test.php’ in your work folder (C:\Program Files\Apache Group\Apache2\htdocs\yourfolder).

4. Open browser with URL (http://localhost:8080/your_folder_name/test.php) to test your PHP installation.

5. The following page shows the PHP configuration with Apache is successful.



PHP Version 8.1.4

System	Darwin Sazilahs-MacBook-Air.local 20.4.0 Darwin Kernel Version 20.4.0: Thu Apr 22 21:46:41 PDT 2021; root:xnu-7195.101.2~1/RELEASE_ARM64_T8101 x86_64
Build Date	Mar 17 2022 03:11:15
Build System	Darwin osx-compilation-1010-vm-macmini-2 14.5.0 Darwin Kernel Version 14.5.0: Wed Jul 29 02:26:53 PDT 2015; root:xnu-2782.40.9~1/RELEASE_X86_64 x86_64
Configure Command	'./configure' '--prefix=/Applications/XAMPP/xamppfiles' '--with-apxs2=/Applications/XAMPP/xamppfiles/bin/apxs' '--with-config-file-path=/Applications/XAMPP/xamppfiles/etc' '--with-mysql=mysqlnd' '--enable-inline-optimization' '--disable-debug' '--enable-bcmath' '--enable-calendar' '--enable-ctype' '--enable-ftp' '--enable-gd-native-ttf' '--enable-magic-quotes' '--enable-shmop' '--disable-sigchild' '--enable-sysvsem' '--enable-sysvshm' '--enable-wddx' '--with-gdbm=/Applications/XAMPP/xamppfiles' '--with-jpeg-dir=/Applications/XAMPP/xamppfiles' '--with-png-dir=/Applications/XAMPP/xamppfiles' '--with-freetype-dir=/Applications/XAMPP/xamppfiles' '--with-zlib=yes' '--with-zlib-dir=/Applications/XAMPP/xamppfiles' '--with-openssl=/Applications/XAMPP/xamppfiles' '--with-xsl=/Applications/XAMPP/xamppfiles' '--with-ldap=/Applications/XAMPP/xamppfiles' '--with-gd' '--with-imap=/bitnami/xamppunixinstaller81stack-osx-x64/src/imap-2007e' '--with-imap-ssl' '--with-gettext=/Applications/XAMPP/xamppfiles' '--with-mssql=shared,/Applications/XAMPP/xamppfiles' '--with-pdo-dblib=shared,/Applications/XAMPP/xamppfiles' '--with-sybase-ct=/Applications/XAMPP/xamppfiles' '--with-mysql-sock=/Applications/XAMPP/xamppfiles/var/mysql/mysql.sock' '--with-mcrypt=/Applications/XAMPP/xamppfiles' '--with-mhash=/Applications/XAMPP/xamppfiles' '--enable-sockets' '--enable-mbstring=all' '--with-curl=/Applications/XAMPP/xamppfiles' '--enable-mbregex' '--enable-zend-multibyte' '--enable-exif' '--with-bz2=/Applications/XAMPP/xamppfiles' '--with-sqlite=shared,/Applications/XAMPP/xamppfiles' '--with-sqlite3=/Applications/XAMPP/xamppfiles' '--with-libxml-dir=/Applications/XAMPP/xamppfiles' '--enable-soap' '--with-xmlrpc' '--enable-pcntl' '--with-mysqli=mysqlnd' '--with-pgsql=shared,/Applications/XAMPP/xamppfiles/' '--with-iconv=/usr' '--with-pdo-mysql=mysqlnd' '--with-pdo-pgsql=/Applications/XAMPP/xamppfiles/postgresql' '--with-pdo_sqlite=/Applications/XAMPP/xamppfiles' '--with-icu-dir=/Applications/XAMPP/xamppfiles' '--enable-fileinfo' '--enable-phar' '--enable-zip' '--enable-mbstring' '--disable-huge-code-pages' 'ac_cv_decimal_fp_supported=no' '--with-libzip' '--with-pear' '--enable-gd' '--with-jpeg' '--with-libwebp' '--with-freetype' '--with-zip' 'PKG_CONFIG_PATH=/Applications/XAMPP/xamppfiles/lib/pkgconfig' 'CFLAGS=-std=gnu99 -Qunused-arguments -UROCKSDB_SUPPORT_THREAD_LOCAL -U__MACH__ -I/Applications/XAMPP/xamppfiles/include/c-client -Qunused-arguments -I/Applications/XAMPP/xamppfiles/include/libpng -I/Applications/XAMPP/xamppfiles/include/freetype2 -O3 -L/Applications/XAMPP/xamppfiles/lib -I/Applications/XAMPP/xamppfiles/include -I/Applications/XAMPP/xamppfiles/include/ncurses -arch x86_64' 'LIBXML_CFLAGS=-I/usr/include/libxml2' 'LIBXML_LIBS=-L/usr/lib -lxml2' 'XSL_CFLAGS=-I/usr/include/libxslt' 'XSL_LIBS=-L/usr/lib -lxslt'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/Applications/XAMPP/xamppfiles/etc
Loaded Configuration File	/Applications/XAMPP/xamppfiles/etc/php.ini
Scan this dir for additional .ini files	(none)

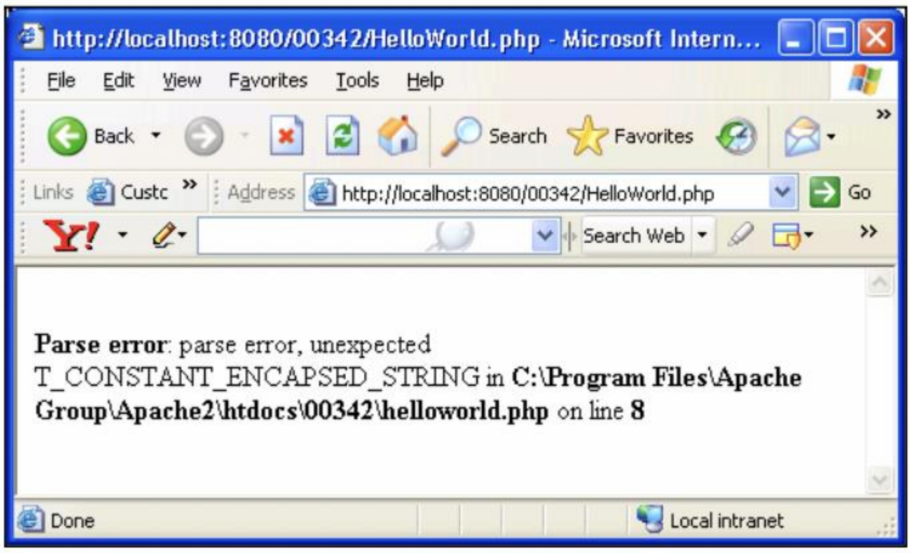
1.1.2 Embedding PHP syntax and variables in a HTML Document

1. In this step you will write a simple PHP code that will displays a simple text output. Open your text editor and enter the following code. In this exercise, PHP code embedded within extension. Save your document as 'helloworld.php' in your work folder. (C:\Program Files\Apache Group\Apache2\htdocs\yourfolder).

```
<html>
<head>
<title> Hello World </title>
</head>
<body>
      <?php
         # Using comments in PHP code..
         echho "\"Hello Students!\"";
      ?>
</body>
</html>
```

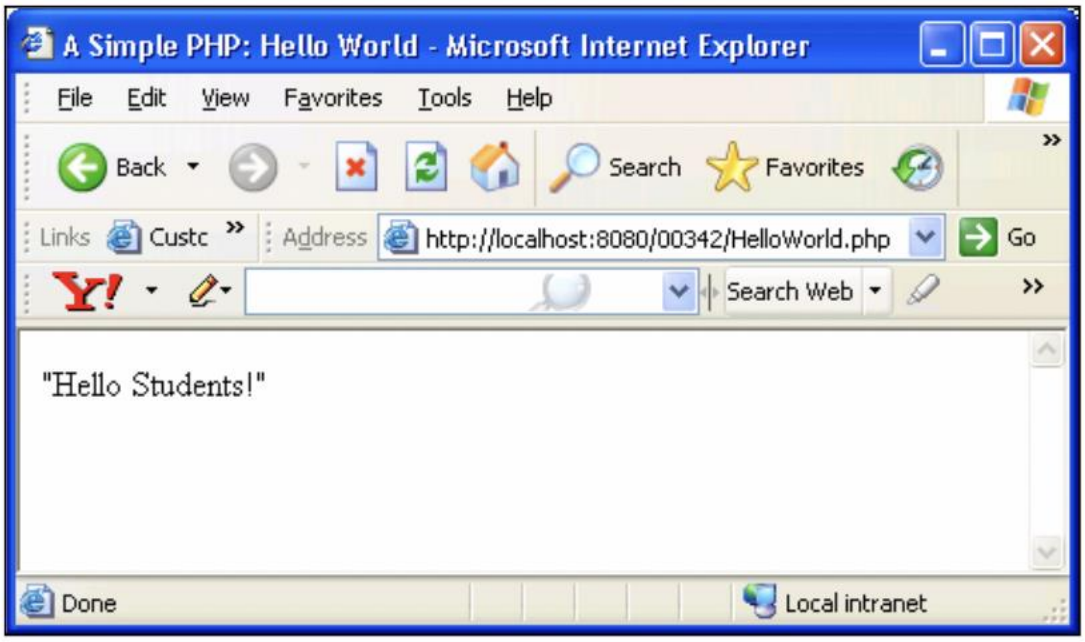
2. On your browser, type http://localhost:8080/your_folder_name/helloworld.php as the URL. (Note: 8080 here is the local server port number). The page will display parsing error message.

3. For fatal errors like misspelled keywords, function names, or symbols, PHP will terminate the program and output an error message returned as an HTML document. In this exercise, we purposely misspelled `'echo'`. Note that the name of the file and the line number are displayed in the figure below.


http://localhost:8080/00342/HelloWorld.php - Microsoft Intern...
File Edit View Favorites Tools Help
Back Search Favorites
Links Custc Address http://localhost:8080/00342/HelloWorld.php Go
Search Web
Done Local intranet

Parse error: parse error, unexpected T_CONSTANT_ENCAPSED_STRING in **C:\Program Files\Apache Group\Apache2\htdocs\00342\helloworld.php** on line **8**

4. Now, change the `'echho'` to `'echo'` to correct the error in the PHP script. Save the PHP script and open it again in the browser. The page appears as shown in next figure.


A Simple PHP: Hello World - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Links Custc Address http://localhost:8080/00342/HelloWorld.php Go
Search Web
Done Local intranet

"Hello Students!"

5. PHP code is identified to the Apache server with the special HTML <?php .... ?> tag combination. These tags are needed to tell the web server to process PHP code in the file.
6. Insert the comment in a single line of PHP code, you preface the comment with either symbol #. Apart from this symbol, you can use the two forward slashes // or /*.......*/ tags to enclose the multiple lines of code.
7. If you want to send PHP reserved characters such as double quotations to the web browser within the `echo` command, you must use the backslash character as shown in the example code.
8. PHP display the text in your browser using the `echo` function. Each code line in PHP must end with a semicolon ';'.

The common two ways to create output in the browser: `echo` *and* `print`.

- *The* `echo` *function can be called with or without parenthesis around its parameters. If parentheses are included, only a single string parameter is acceptable; otherwise any number of parameters can be included. For example,*
  `echo "Roses<br/>","Carnations<br/>";`
- *The* `print` *function is like* `echo`*, except it can never be called with more than one parameter, and it returns a value. The parentheses are optional. The returned value indicates whether the operation was completed (1 if it is succeeded and 0 if it is failed)*

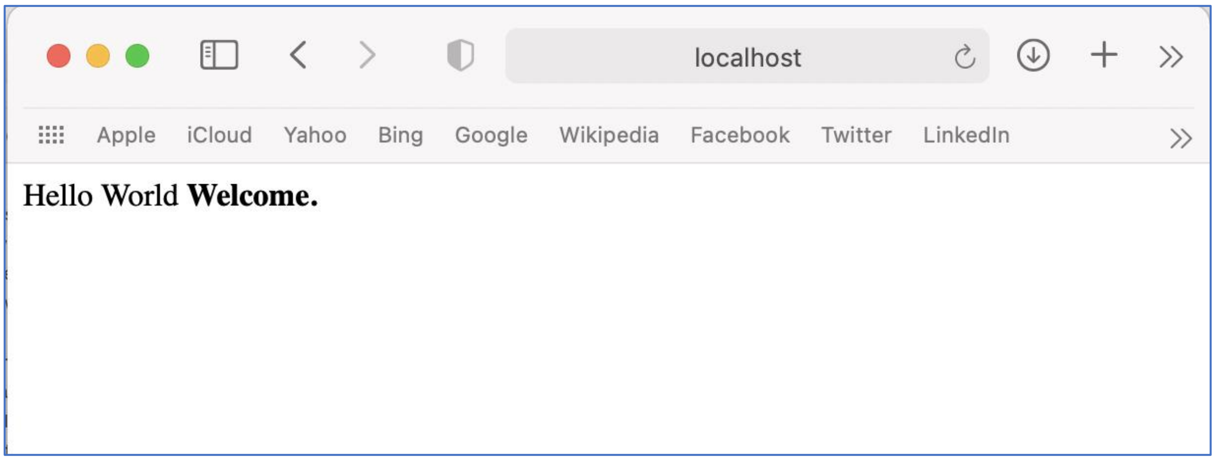
9. Modify helloworld.php by typing this code.

```
<html>
<body>
<head>
<title> Hello World </title>
```

```
</head>
<?php
$txt1="Hello";
$txt2="World";
echo $txt1 . " " . $txt2 . " <b>Welcome.</b>";
?>
</body>
</html>
```

10. PHP display the text in your browser using the `echo` function. Each line in PHP must end.

Output:



```
Hello World Welcome.
```

1.1.3 Including Files

1. We probably want our pages that more than a single page has a common look and feel. PHP can make this possible with provide a method called included files.
2. Open a new document in your text editor and type the following code:

```
<html>
   <head>
   <title>PHP Included Files</title>
   </head>
   <body>
   <?php
     /* To concatenate more than one variable together, use the
     dot (.) operator */
     echo "<h2>Malaysia 2 - " . " MyTeam 1 <br>";
     echo "<h3>28 May 2023 Stadium Bukit Jalil";

?>
```

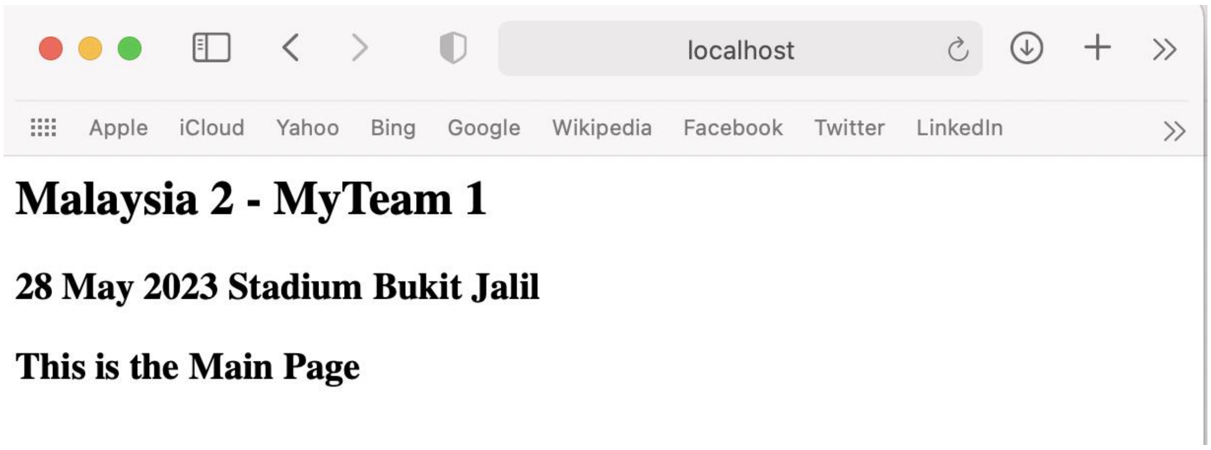
3. Save the file as **header.inc** in your work folder and then close the document.

4. Open a new document in your text editor and enter the following code:

```
   <?php include("header.inc"); ?>
   <p> This is the Main Page </p>
```

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

4. Save the file as '**Mainpage.php**' in your working folder. The *header.inc* that we created in the Step 2 in this exercise is a simple HTML code that includes PHP output statements. The PHP code includes headers signified by `<h2>` and `<h3>` tag combination and called in the '**Mainpage.php**' and included in the PHP file by the Apache web server before being served to the browser.
5. Start your browser and type http://localhost:8080/your_folder_name/Mainpage.php as the URL. The main page appears as shown below :



Malaysia 2 - MyTeam 1

28 May 2023 Stadium Bukit Jalil

This is the Main Page

1.1.4 Accessing Form Variables

1. Depending on your PHP version and setup, you can access the form data via variable in three ways. Let"s say you are going to access the contents of the text field „`Name`", `you` may use one of the following way:

Form	Description
`$_POST['Name']` or `$_GET['Name']`	Supported by version 4.1 and later
`$HTTP_POST_VARS['Name']` or `$HTTP_POST_VARS['Name']`	Supported by version 4.0 and later
`$Name`	To enable this kind of access, you need to turn on the `'register_globals'` in your PHP configuration file. In PHP 4.2 and later this parameter is set to `Off` by default for security reasons.

2. In this exercise, you will use the implicit array for form values, $_POST['...'] for referencing form variables.
3. The following is a basic HTML form that uses the post method to redirect the form to another PHP page that displays the entered information in table format. Open a new document in your text editor and enter the following code:

```
<html>
```

```
<head>
<title>Sample form to take user input</title>
</head>
<body>
<h3>Please fill in all fields and click Submit Form. </h3>
<!-- Post form data to Display.php -->
<form method = "post" action = "Display.php">
<hr />
<!-- create four text boxes for user input -->
<table>
 <tr> <td>First Name</td>
 <td><input type="text" name="fname" /></td> </tr>
 <tr> <td>Last Name</td>
 <td> <input type = "text" name = "lname" /></td> </tr>
 <tr><td>Address</td>
 <td><textarea name = "address" rows=5 cols=25></textarea></td>
   </tr>
   <tr><td>City</td>
 <td><input type = "text" name = "city" /></td> </tr>
 <tr> <td>Phone Number</td>
 <td><input type = "text" name = "phone" /></td> </tr>
   <tr>
   <br />
   <!-- create a submit button -->
   <td><input type="submit" value= "Submit Form"/> </td>
   <td><input type="reset" value="Clear Form"/> </td>
   </tr>
 </table>
 </form>
 </body>
 </html>
```

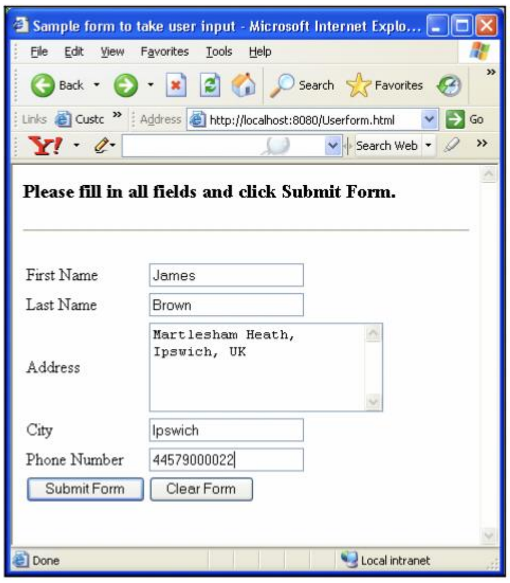
4. Save the file as ‘**UserForm.html’** in your work folder.
5. Now you will create a PHP file to retrieve the values inserted by user in the above

   form. Open a new document and type the following code:

```
        <html>
        <head> <title>Displaying the Information</title>
        </head>
        <body>
        <?php
        // Access HTTP POST Data and assign them to local variables
        $fname_n = $_POST['fname'];
        $lname_n = $_POST['lname'];
        $address_n = $_POST['address'];
        $city_n = $_POST['city'];
           $phone_n = $_POST['phone'];
?>
<h3>Hi <?php echo $fname_n; ?></h3>
```

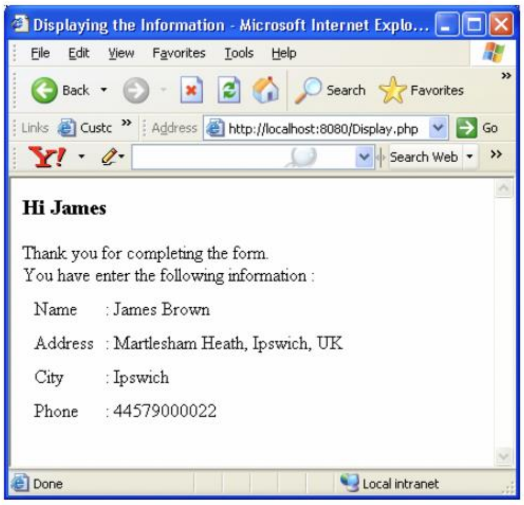
```
Thank you for completing the form.<br />
You have enter the following information :
<table border="0" cellpadding="0" cellspacing="10" >
  <tr>
  <td>Name</td>
  <td> : <?php echo $fname_n, " " , $lname_n ?></td>
  </tr>
  <tr>
  <td>Address</td>
  <td> : <?php echo $address_n ?></td>
  </tr>
  <tr>
  <td>City</td>
  <td> : <?php echo $city_n ?></td>
  </tr>
  <tr>
  <td>Phone</td>
  <td>  : <?php echo $phone_n ?></td>
  </tr>
</table>
</body>
</html>
```

6. Save the file as **Display.php** in your work folder. In the browser, type *http://localhost:8080/your_folder_name/UserForm.html* as the URL. And then enter information on the Web and then click the **Submit** button. The entry form and the results page should resemble figures below.


Please fill in all fields and click Submit Form.

First Name	James
Last Name	Brown
Address	Martlesham Heath, Ipswich, UK
City	Ipswich
Phone Number	44579000022

Sample form to take user input - Microsoft Internet Explo...
http://localhost:8080/Userform.html
Submit Form
Clear Form


Hi James

Thank you for completing the form.
You have enter the following information :

Name	: James Brown
Address	: Martlesham Heath, Ipswich, UK
City	: Ipswich
Phone	: 44579000022

Displaying the Information - Microsoft Internet Explo...
http://localhost:8080/Display.php

There are two methods to specify how the form's data is sent to the Web server which are POST and GET. The `method='post'` *appends form data to the browser request, which contains the protocol HTTP and the requested resource's URL. The other possible value,* `method='get'` *appends the form data directly to the end of the URL.*

7. Let's say the user wants to switch from using **HTTP post** to using **HTTP get**. Make relevant changes to your **'UserForm.html'** and **'Display.php'** documents. Test the pages in the browser. The pages should look like the above figures.

1.1.6 Using String Function and Regular Expression in Server-side Validation

1. On the client side, validation is performed using scripting such as JavaScript. This can be very tricky, because some users might turn off JavaScript support in their browsers before they use your application. In this case, client side validation will not help you much if you try to verify data from a form because your JavaScript code will not be executed or interpreted by the browser.

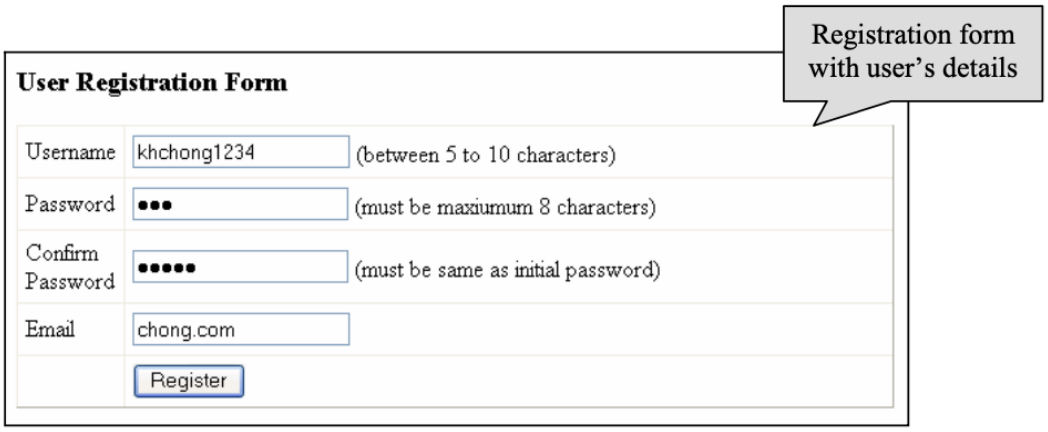
2. To overcome such problem, in this exercise we will write a PHP script to perform server-side validation. The exercise will use a combination of string functions and regular expressions to validate common data submitted via a registration form. **'Regular Expression'** is a function consisting of an expression used for complex string manipulation in PHP.
3. Open a new document in your text editor and then type the following code:

```
<html>
<body>
<h3>User Registration Form</h3>
<?php
$username_input = '';
```

```
$password_input = '';
$password2_input = '';
$email_input = '';
# Only validate form when form is submitted
if(isset($_POST['submit']))
{
 # Get form input
$username_input = $_POST['username'];
$password_input = $_POST['password'];
$password2_input = $_POST['password2'];
$email_input = $_POST['email'];
# Declare a variable to hold error message
$error_msg='';
# Function to check for email address
function checkEmail($mail) {
# check if email is a valid address
$pattern = "/^[A-z0-9\._-]+". "@". "[A-z0-9][A-z0-9-]*". "(\.[A-z0-9_-]+)*". "\.([A-
z]{2,6})$/";
return preg_match ($pattern, $mail);
}
# Function to check String for Length
function checkLength($string, $min, $max) {
$length = strlen ($string);
if (($length < $min) || ($length > $max)) {
return FALSE;
} else {
return TRUE;
}
}
# Password Checking
function checkPass($pass1, $pass2) {
if ($pass1 == $pass2) {
return TRUE;
} else {
return FALSE;
}
}
# Call related function
if(!checkLength($username_input,5,10)){
$error_msg.="Please enter a username between 5 to 10 characters long<br>";
}
if(!checkLength($password_input,1,8)){
$error_msg.="Please enter a password maximum 8 characters long<br>";
}
if(!checkPass($password_input,$password2_input)){
$error_msg.="Both passwords are not matched<br>";
}
if(!checkEmail($email_input)){
$error_msg.="Please enter a valid email address<br>";
}
# Diplay error message if any, if not, proceed to other processing
if($error_msg==""){
echo "<h3><font color=blue>The data you entred were correct. Thank You!<br>";
}
else {
echo "The following errors are generated : <br>";
echo "<font color=red>$error_msg</font>";
}
}
?>
<form method="POST" action="Validate.php">
<table border="1" cellpadding="5" cellspacing="0" style="bordercollapse: collapse"
width="60%">
<tr>
<td width="10%" align="left">Username</td>
<td width="84%">
<input type="text" name="username" size="20" value="<?php echo $username_input; ?>">
(between 5 to 10 characters)</td>
</tr>
<tr>
 <td width="10%" align="left">Password</td>
 <td width="84%">
 <input type="password" name="password" size="20" value="<?php echo $password_input;
?>">(must be maxiumum 8 characters)</td>
</tr>
<tr>
 <td width="10%" align="left">Confirm Password</td>
 <td width="84%">
 <input type="password" name="password2" size="20" value="<?php echo
$password2_input; ?>">(must be same as initial password)</td>
 </tr>
<tr>
 <td width="10%" align="left">Email</td>
 <td width="84%">
 <input type="text" name="email" size="20" value="<?php Echo $email_input; ?>"></td>
</tr>
<tr>
 <td width="10%" align="left"> </td>
 <td width="84%">
 <input type="submit" value="Register" name="submit"></td>
</tr>
</table>
</form>
</body>
</html>
```

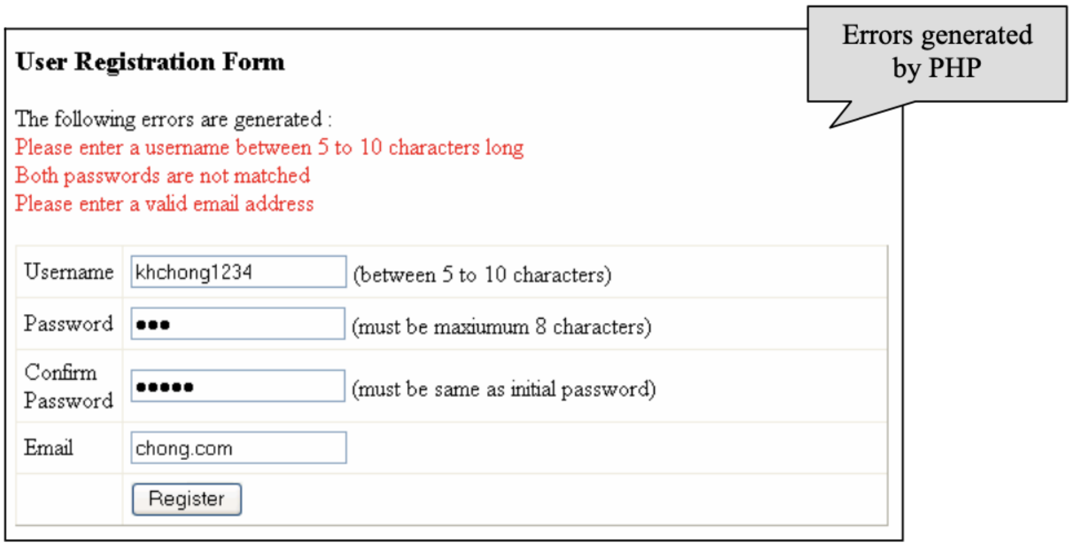
4. Save your file as **‘Validate.php’** in your work folder. Start your browser, type

http://localhost:8080/your_folder_name/V alidate.php.
5. The form and validation page resembles figures below respectively.



User Registration Form

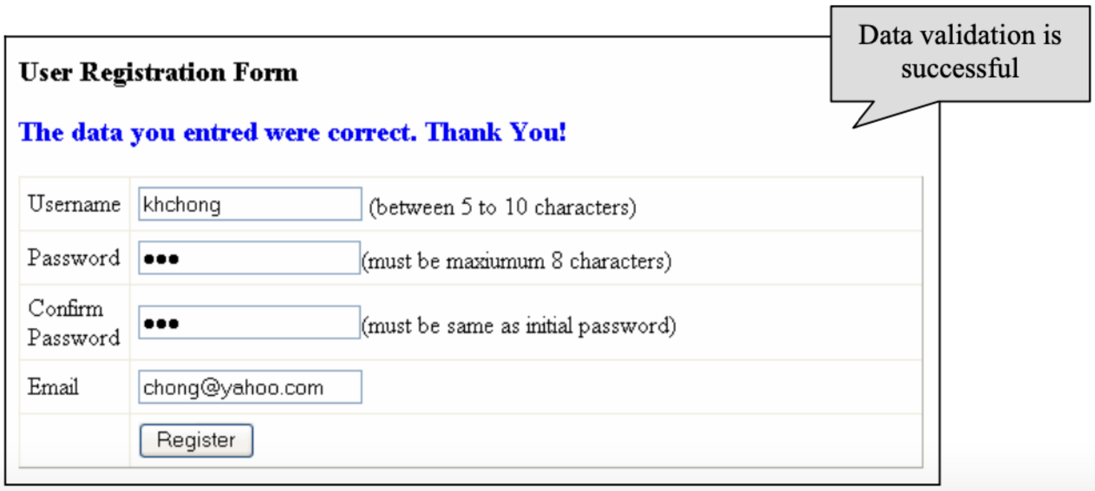
Username	khchong1234	(between 5 to 10 characters)
Password	●●●	(must be maxiumum 8 characters)
Confirm Password	●●●●●	(must be same as initial password)
Email	chong.com	
	Register	



User Registration Form

The following errors are generated :
Please enter a username between 5 to 10 characters long
Both passwords are not matched
Please enter a valid email address

Username	khchong1234	(between 5 to 10 characters)
Password	●●●	(must be maxiumum 8 characters)
Confirm Password	●●●●●	(must be same as initial password)
Email	chong.com	
	Register	



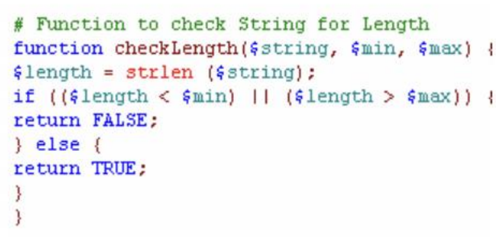
User Registration Form

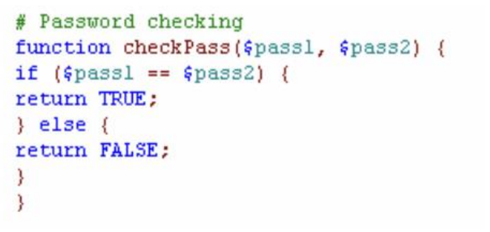
The data you entred were correct. Thank You!

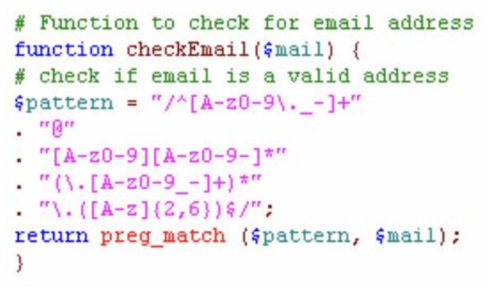
Username	khchong	(between 5 to 10 characters)
Password	●●●	(must be maxiumum 8 characters)
Confirm Password	●●●	(must be same as initial password)
Email	chong@yahoo.com	
	Register	

6. The following table explains the three custom functions included in the validation file for error checking.

Function	**Description**

 `# Function to check String for Length`
`function checkLength($string, $min, $max) {`
`$length = strlen ($string);`
`if (($length < $min) \|\| ($length > $max)) {`
`return FALSE;`
`} else {`
`return TRUE;`
`}`
`}`
	This function will check the length of a string and return `FALSE` if the length is not within a range we specify. For this task, the `strlen()` function is used to determine the length of the string and then compare it to the specified minimum and maximum length.

 `# Password checking`
`function checkPass($pass1, $pass2) {`
`if ($pass1 == $pass2) {`
`return TRUE;`
`} else {`
`return FALSE;`
`}`
`}`
	This function will identify whether the two passwords are matched and return `FALSE` if they are not matched.

 `# Function to check for email address`
`function checkEmail($mail) {`
`# check if email is a valid address`
`$pattern = "/^[A-z0-9\._-]+"`
`. "@"`
`. "[A-z0-9][A-z0-9-]*"`
`. "(\.[A-z0-9_-]+)*"`
`. "\.([A-z]{2,6})$/";`
`return preg_match ($pattern, $mail);`
`}`
	This function tries to find the match for the regular expression pattern in the supplied email string. The regular expression output is assigned as return value of the function. The function invokes the `'preg_match'` function in order to match the objects in the target file with the desired validation parameters.
	The slashes / are delimiters, ^ marks the start of string or line and the Dollar sign $ is the end of the string, or line. The plus-symbol + means required. Then, `[A-z0-9\._-]` is any character A-Z, a-z, 0-9 and _ or - .

1.2 Self-Review Questions

1. Which of the following is **NOT** a valid variable name in PHP?
   A. `$number-in-class`
   B. `$number_in_class`
   C. `$nic`
   D. `$NumberInClass`

2. In PHP, the word 'AND' is used as
   A. a regular variable
   B. a function
   C. it is reserved, but unused
   D. an operator

3. When would you use double quotes rather than single quotes to define a string of text?
   A. When you wanted to include variables and special sequences such as \n within the string
   B. When you wanted the string to be more than one character long
   C. When you wanted every character to be taken as a literal
   D. When you wanted to include special characters such as @ and % within the string

4. PHP server scripts are surrounded by which delimiters?
   A. `<script>...</script>`
   B. `<?php…?>`
   C. `<&>...</&>`
   D. `<%...%?>`

5. How do you get information from a form that is submitted using the ‘get’ method?
   A. `Request.QueryString;`
   B. `$_GET[ ];`
   C. `Request.Form;`
   D. `Request.Get[  ];`

6. What is the correct way to include the file ‘external.inc’ ?
   A. `<? php include (‘external.inc’); ?>`
   B. `<? php require (‘external.inc’); ?>`
   C. `<!--include file (‘external.inc’) -->`
   D. `<% include file= (‘external.inc’) %>`

7. Order the following statements to describe the flow of the following regular expression pattern to verify email address:

```
preg_match('/^[A-z0-9_\-]+[@][A-z0-9_\-]+([.][A-z0- 9_\-]+)+[A-z]{2,4}$/',$email)
```

1	Begin with a delimiter /
	Adding a () around the next subset at the additional [.] tells it to look for more text following the. (.com, .net, .etc)
	Then, just add a ‘@’ after the plus to look for the @ symbol in the e-mail address
	Now all you need is to repeat your previous criteria for matching (text between A and Z or the numbers 0 - 9)
	Finally, the $ indicates the end of the target string
	And the expression is closed with our ending delimiter /.
	Then adding a minimum/maximum bracket {2, 4} tells it to look for an ending that falls within those values (for example, .my, .hk, etc)
	Then indicate the beginning of the line with ^
	Then, [A-Za-z0-9_\-] is any character A-Z, a-z, 0-9 and _ or - .
	Then, indicate that this pattern is one or more with the + symbol.