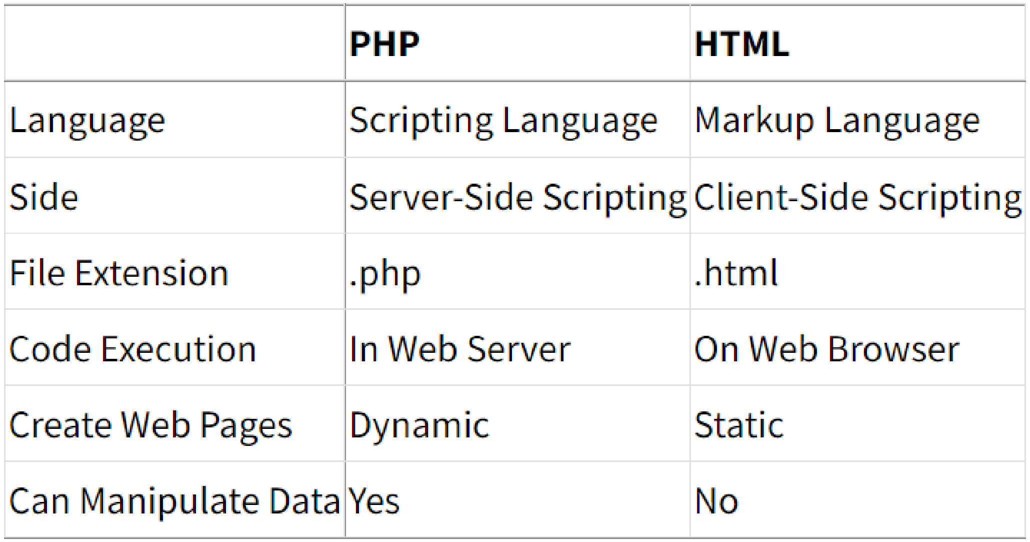
LAB 4

PHP BASICS

PART-I

September 01, 2022

- Real-time dynamic websites can not be created using only HTML and CSS.
- Why?
- HTML itself can’t process the data
- We need a scripting language like PHP, PERL etc.



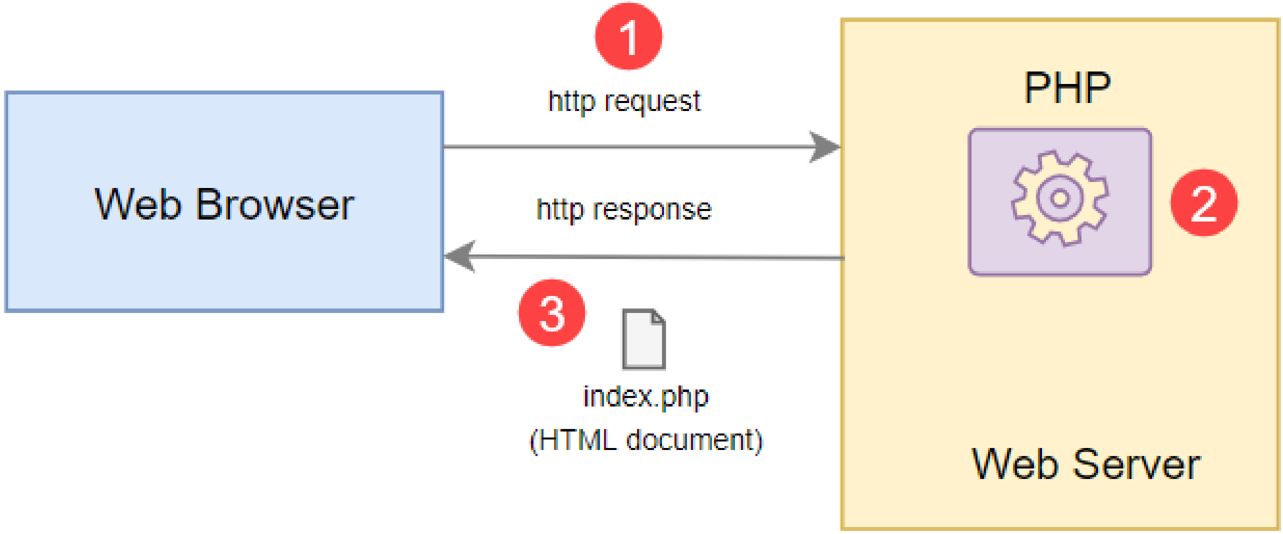
	PHP	HTML
Language	Scripting Language	Markup Language
Side	Server-Side Scripting	Client-Side Scripting
File Extension	.php	.html
Code Execution	In Web Server	On Web Browser
Create Web Pages	Dynamic	Static
Can Manipulate Data	Yes	No

Let’s see “what is PHP?”

PHP

- **P**re-**H**ypertext **P**rocessor or **P**HP: **H**ypertext **P**rocessor
- HTML-embedded server-side scripting language used for Web Development
- PHP scripts are executed on the server
- Open-source software supporting many databases (MySQL, Informix, Oracle, Sybase, PostgreSQL, Generic ODBC, etc.)
- Much of its syntax is borrowed from C, Java and Perl
- Allows web developers to write dynamically generated pages quickly
- Runs on different platforms and is compatible with almost all servers used today

How Does PHP Works?

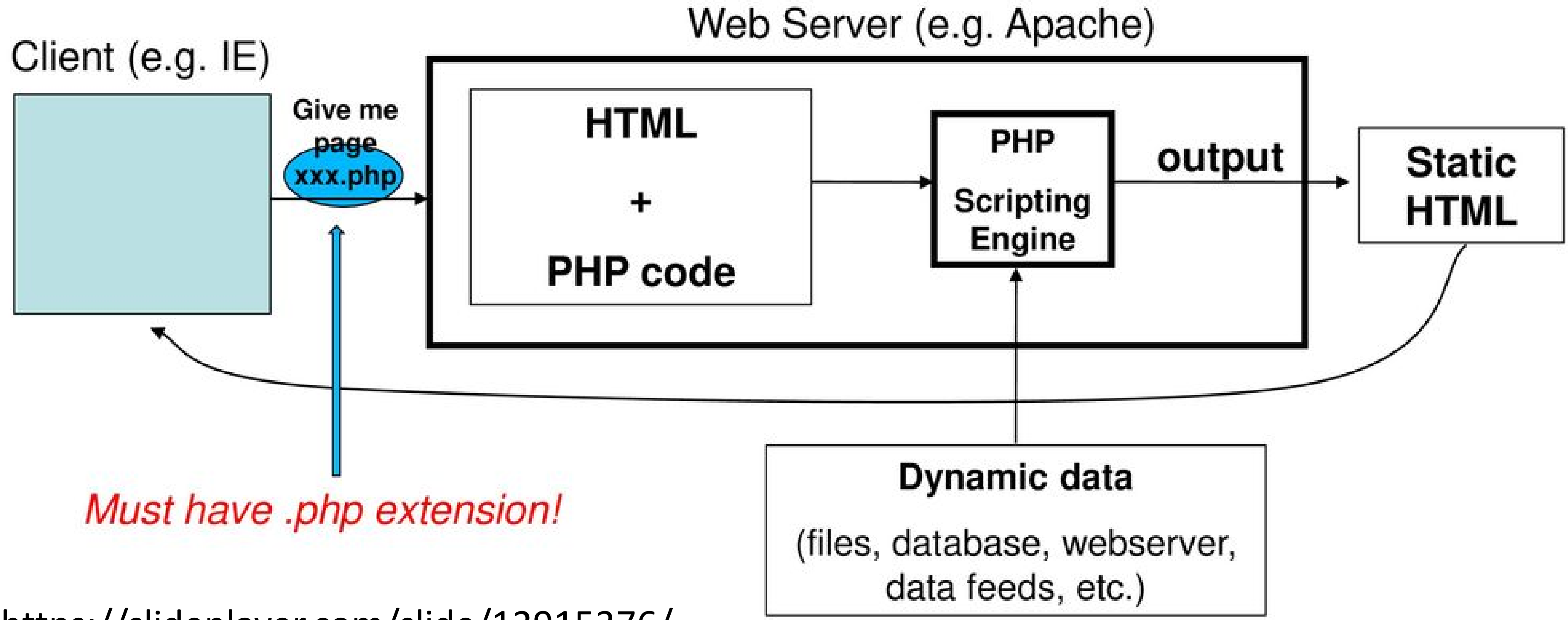


1. The web browser sends an HTTP request to the web server for a website whose first page is index.php.
2. The PHP preprocessor that locates on the web server processes PHP code to generate the HTML document.
3. The web server sends the HTML document back to the web browser, e.g., index.php.

https://www.phptutorial.net/php-tutorial/what-is-php/

How Does PHP Work?

- PHP is a program that performs dynamic HTML scripting
  - It is a language interpreter that directly executes commands then ultimately outputs into HTML/HTTP (rather than a compiler which outputs translates commands into machine code for later execution)
  - The server knows to execute PHP code when the requested file has a .php extension



Web Server (e.g. Apache)

Client (e.g. IE)

Give me page xxx.php

HTML + PHP code

PHP Scripting Engine

output

Static HTML

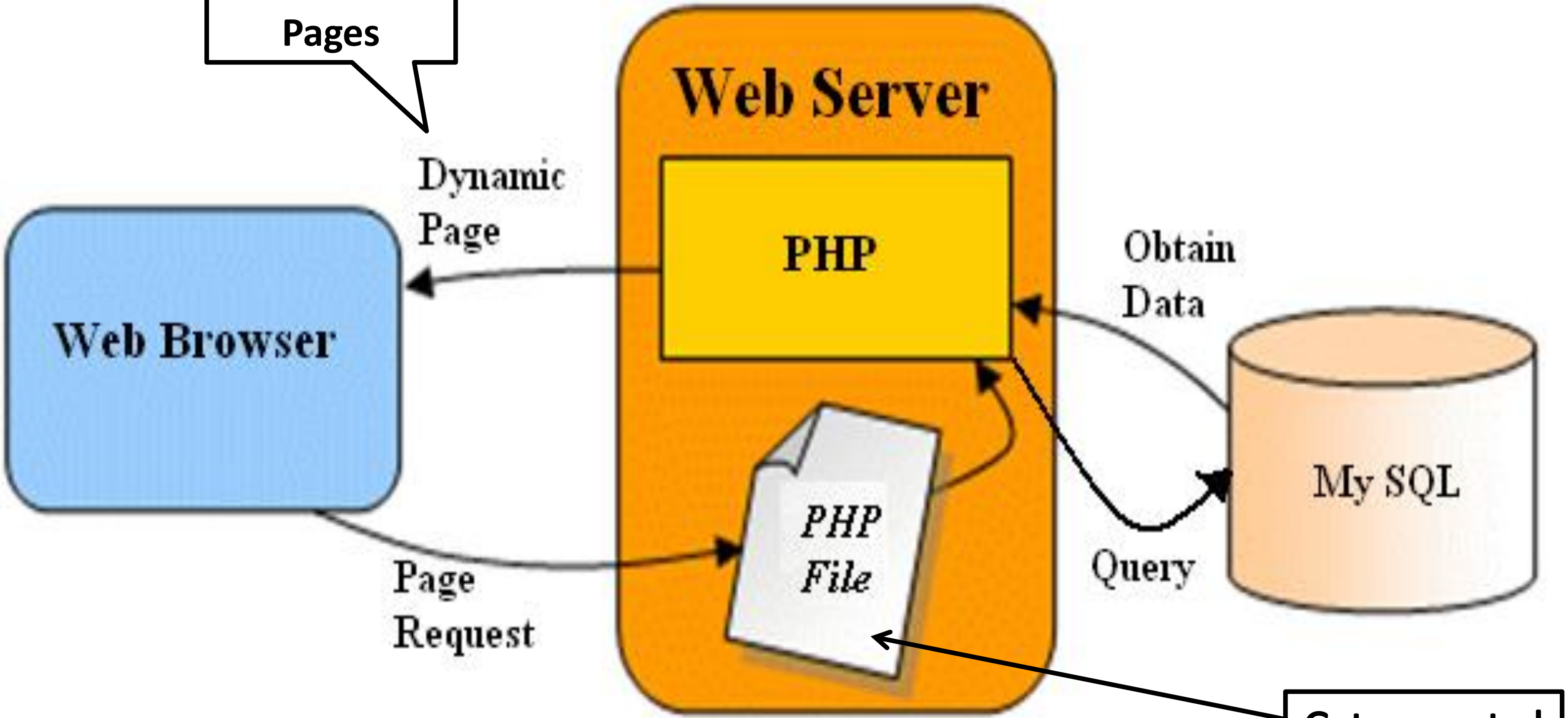
Dynamic data

(files, database, webserver, data feeds, etc.)

Must have .php extension!

https://slideplayer.com/slide/12915376/

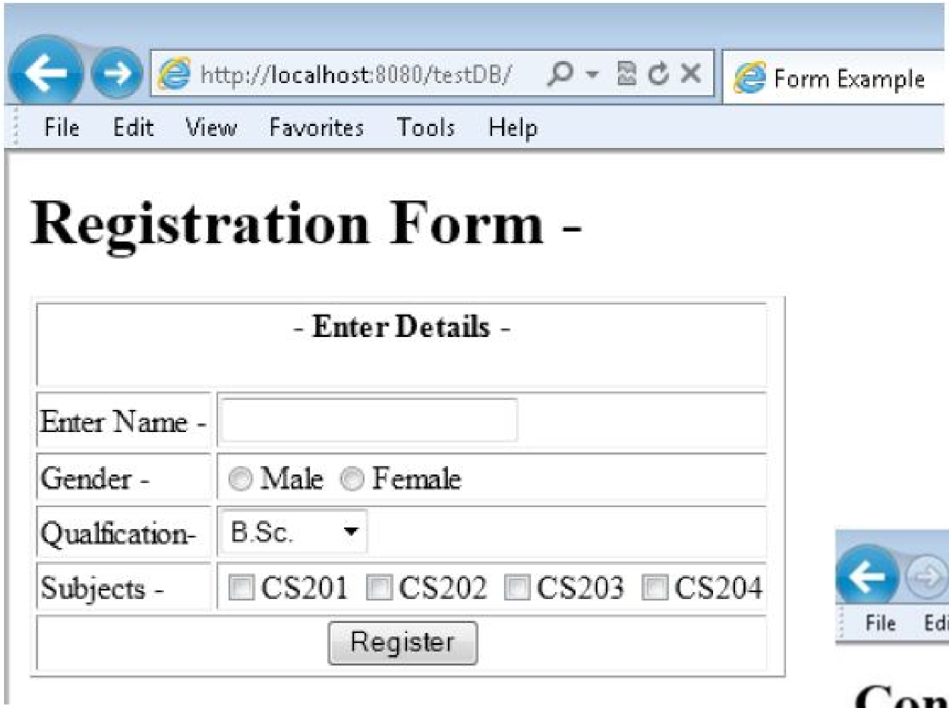
How Does PHP Works



Plain HTML Pages

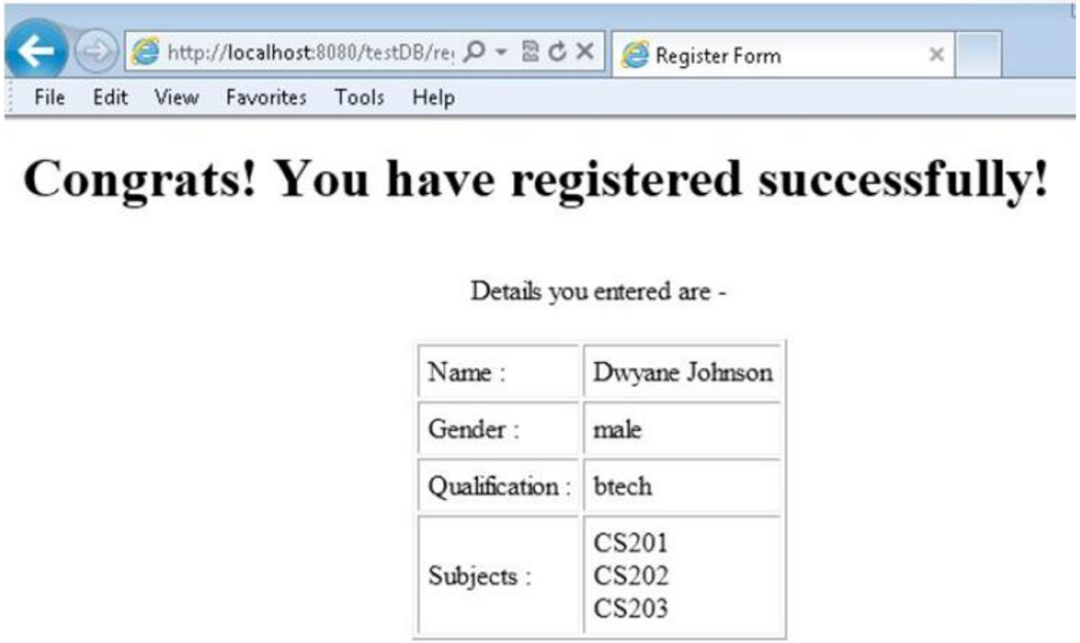
Gets executed on SERVER

Example PHP



Registration Form -

- Enter Details -	
Enter Name -	
Gender -	○ Male ○ Female
Qualfication-	B.Sc. ▾
Subjects -	□ CS201 □ CS202 □ CS203 □ CS204
Register	



Congrats! You have registered successfully!

Details you entered are -

Name :	Dwyane Johnson
Gender :	male
Qualification :	btech
Subjects :	CS201 CS202 CS203

Learn PHP !

- Extension: file_name.php
- Content: text, HTML tags and PHP scripts.
- PHP code must be contained within the tags:

  <?php and **?>**

- PHP statement ends with a semicolon **‘;’**
- Example:

```
<?php
        PHP Statement 1;
                    …
        PHP Statement n;
?>
```

‘Echo’ Command

- Used to output text to the web browser.

PHP Statement	Output
echo "Have a nice day.";	Have a nice day.
echo "<h5>Have a nice day.</h5>";	Have a nice day.
$var_1 = 4; echo $var_1;	4
echo "This ", "string ", “is ", "made ", "with multiple parameters.";	This string is made with multiple parameters.
$str_1 = "Have a nice day"; echo "$str_1 Beena.";	Have a nice day Beena.
$str_1 = "Have a nice day"; echo '$str_1 Beena.';	$str_1 Beena.

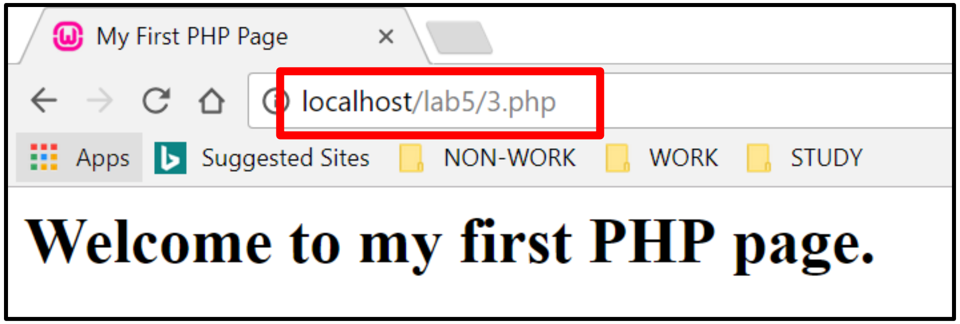
Notice the difference between the usage of single and double quotes.

‘Echo’ Example

```
<html>
   <head>
      <title>
           My First PHP Page
      </title>
    </head>
   <body>
      <?php
           echo "<h1>Welcome to my first PHP page.</h1>";
       ?>
   </body>
</html>
```

Instructions to run file1.php:

1) Files will run on local server (xampp) which is installed on your PCs and can be accessed by localhost.
2) Keep the file in C:\ xampp\htdocs folder.
3) Run files on web browser by typing localhost/file1.php.



Welcome to my first PHP page.

Comments

- Single Line → **// or #**
- Multi Line → **/* … */**

- Example:

```
<?php
   // This is a single line comment.
   # This is another single line comment.
   /* This is a multiple
          line comment. */
?>
```

Variables

PHP variables:

- Starts with the $ sign, followed by the name of the variable
- Name may start with a letter or underscore "_".
- Are comprised of alpha-numeric characters and underscores. a-z, A-Z, 0-9, or _ .
- Are case-sensitive and need not be declared.

Syntax:

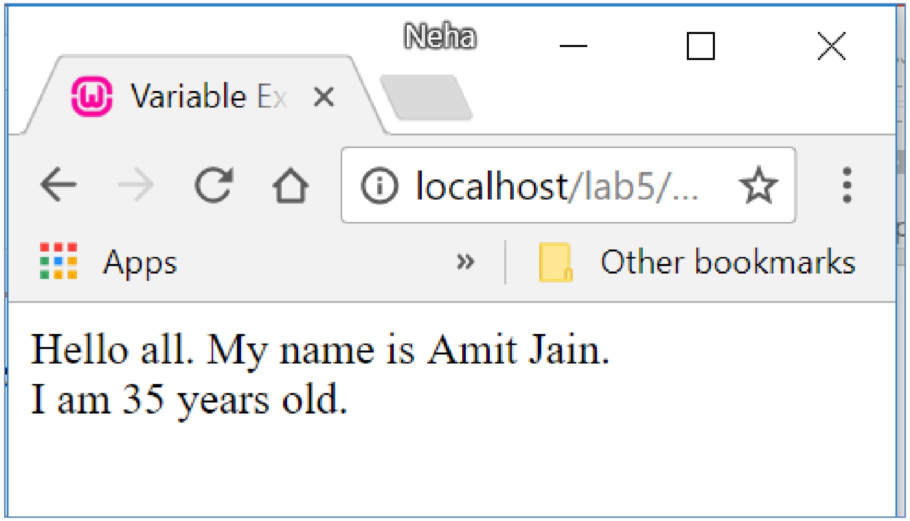
- **$variable_name = Value;**

Example:

- $_hello = "Hello World!";
- $first_number = 4;
- $num2 = 8;

Example - Variables

```
<html>
    <head><title>Variable Example</title></head>
    <body>
       <?php
           $str1 = "Hello all. My name is Amit Jain.";
           $num = 35;
           echo $str1;
           echo "<br>";
    echo "I am $num years old.";
       ?>
</body></html>
```



```
Hello all. My name is Amit Jain.
I am 35 years old.
```

Operators in PHP

Let $a = 5; and $b = 2;

Operator	Meaning	Example	
.	Concatenation	echo "Hello!"."<br />"."How are you.";	
=	Assignment	$var_1 = 4;	
Arithmetic Operators			
+	Addition	$a + $b	= 5 + 2 = 7
-	Subtraction	$a - $b	= 5 - 2 = 3
*	Multiplication	$a * $b	= 5 * 2 = 10
/	Division	$a / $b	= 5 / 2 = 2
%	Modulus	$a % $b	= 5 % 2 = 1

Contd…

Operator	Meaning	Example	Equivalent Operation
+=	Plus Equals	$a += 2;	$a = $a + 2;
-=	Minus Equals	$a -= 2;	$a = $a – 2;
*=	Multiply Equals	$a *= 2;	$a = $a * 2;
/=	Divide Equals	$a /= 2;	$a = $a / 2;
%=	Modulo Equals	$a %= 2;	$a = $a % 2;
.=	Concatenate Equals	$a .= "hello";	$a = $a . "hello";
++	Pre/Post Increment	$a++; or ++$a;	$a = $a + 1;
--	Pre/Post Decrement	$a--; or --$a;	$a = $a - 1;

Comparison Operators

Let $a = 4; and $b = 6;

Operator	Meaning	Example	Result
==	Equal To	$a == $b	False
!= or <>	Not Equal To	$a != $b	True
<	Less Than	$a < $b	True
>	Greater Than	$a > $b	False
<=	Less Than or Equal To	$a <= $b	True
>=	Greater Than or Equal To	$a >= $b	False

Logical Operators

Let $a = 4; and $b = 6;

<table>
<tr><th>Operator</th><th>Meaning</th><th>Example</th><th>Result</th></tr>
<tr><td>&&</td><td>AND</td><td>$a == 4 && $b == 6</td><td>True</td></tr>
<tr><td>||</td><td>OR</td><td>$a != 4 || $b == 6</td><td>True</td></tr>
<tr><td>!</td><td>NOT</td><td>!($a == $b)</td><td>True</td></tr>
</table>

Arrays

- It’s a special variable, which can store multiple variable values (elements) under a single name.
- Each element in the array has its own index so that it can be easily accessed.
- Types:
  - **Numeric Arrays →** index-value pairs.
  - **Associative Arrays →** ID key-value pair.
  - **Multidimensional Arrays →** one or more arrays.

Contd…

- **Numeric Arrays**

```
$name=array(“Amit",“Jain");
            or
$name[0] = “Amit";
$name[1] = “Jain";
```

- **Associative Arrays : (ID key => element_value)**

```
$name=array("First" => “Amit", "Last" => “Jain");
            or
$name['First'] = “Amit";
$name['Last'] = “Jain";
```

Contd…

- **Multidimensional Arrays**

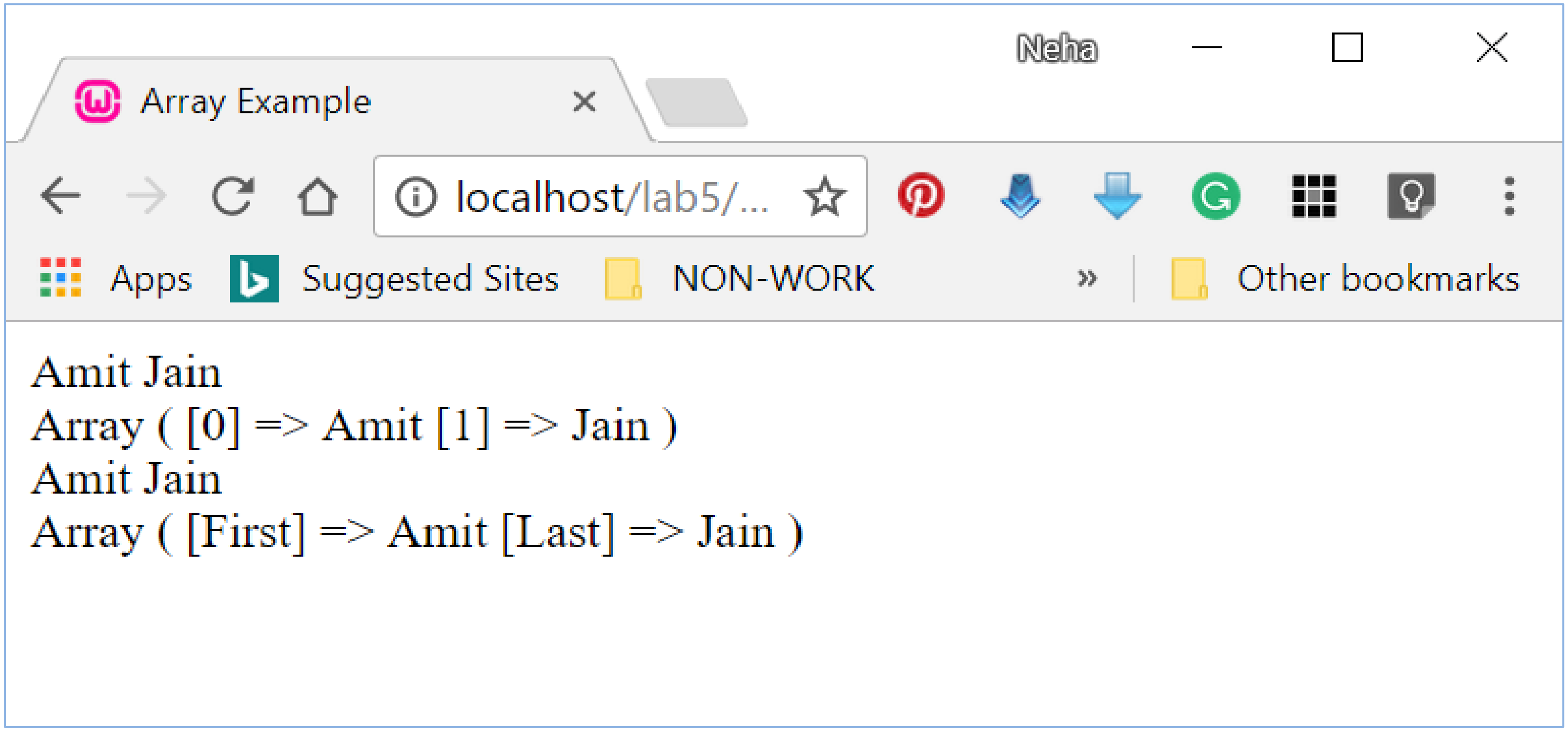
```
$name = array
(
    “Amit" => array (10, 8, 9, 9),
    “Ram" => array (7, 5, 8, 8),
    “Priya" => array (8, 9, 10, 9)
);
```

Example – Arrays

```
<html>
    <head><title>
      Array Example
    </title></head>
    <body>
      <?php
           $name[0] = “Amit";
           $name[1] = “Jain";
           $name1['First'] = “Amit";
           $name1['Last'] = “Jain";
           echo $name[0];
            echo " ";
```

```
            echo $name[1];
            echo "<br>";
            print_r($name);
            echo "<br>";
            echo $name1['First'];
            echo " ";
            echo $name1['Last'];
            echo "<br>";
            print_r($name1);
       ?>
     </body>
</html>
```

Contd…



```
Amit Jain
Array ( [0] => Amit [1] => Jain )
Amit Jain
Array ( [First] => Amit [Last] => Jain )
```

The print_r() function is used to print human-readable information about a variable.

Forms - Revisited

- Forms provides a way to capture user input from a web page and sends it to the server.
- Forms contains other elements like textbox, radio buttons, checkbox, buttons etc. in the web page to capture the user input.
- HTML forms are placed on a web page by using <form> tag. This tag contains other form elements.
- The <form> tag uses the '**action**' attribute to identify to where to send the data and '**method**' attribute to identify how to get the data.
- It contains a **SUBMIT** button to send the data from one web page to server.
- HTML itself can't process the data. Hence the scripting language such as PHP, PERL etc. are needed.

Method

Get:

- The form data is appended to the URL specified in the action attribute when submitted.
- It is ideal for:
  - short forms (such as search boxes)
  - when you are just retrieving data from the web server (not sending information that should be added to or deleted from a database)
- It is useful for form submissions where a user wants to bookmark the result.
- There is a limit to how much data you can place in a URL
- Never use the “get” method to pass sensitive information
- Default if method attribute not used.

Post:

- The form data is not appended to the URL.
- The values are sent in what are known as HTTP headers.
- Use it if your form:
  - allows users to upload a file
  - is very long
  - contains sensitive data (e.g., passwords)
  - adds information to, or deletes information from, a database
- Form submissions with the "post" method cannot be bookmarked.
- It does not have size limitations for sending the data.
- The "post" method is more robust and secure than “get”.

$_GET and $_POST in PHP

- They are predefined associative arrays.
- $_GET and $_POST collect values in a form with method="get" and method="post" respectively.
- Names used in forms serve as the keys in these associative arrays.

Example – GET

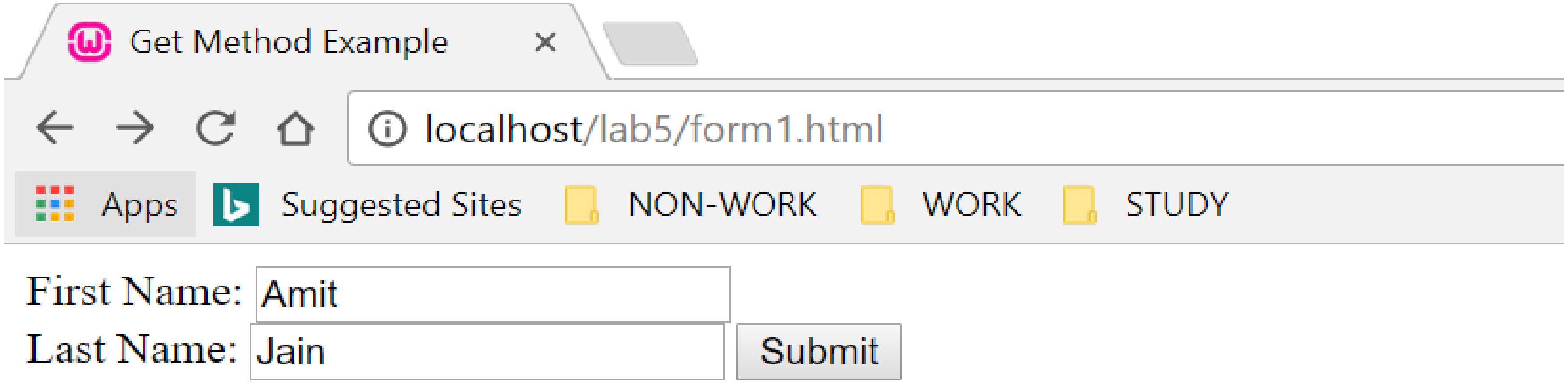
Form.html

```
<html>
   <head><title>
      Get Method Example
   </title></head>
   <body>
      <form action="getMethod.php"
             method="get">
        First Name: <input type="text"
             name="fname" /> <br>
        Last Name: <input type="text"
              name="lname" />
        <input type="submit" />
      </form>
</body></html>
```

getMethod.php

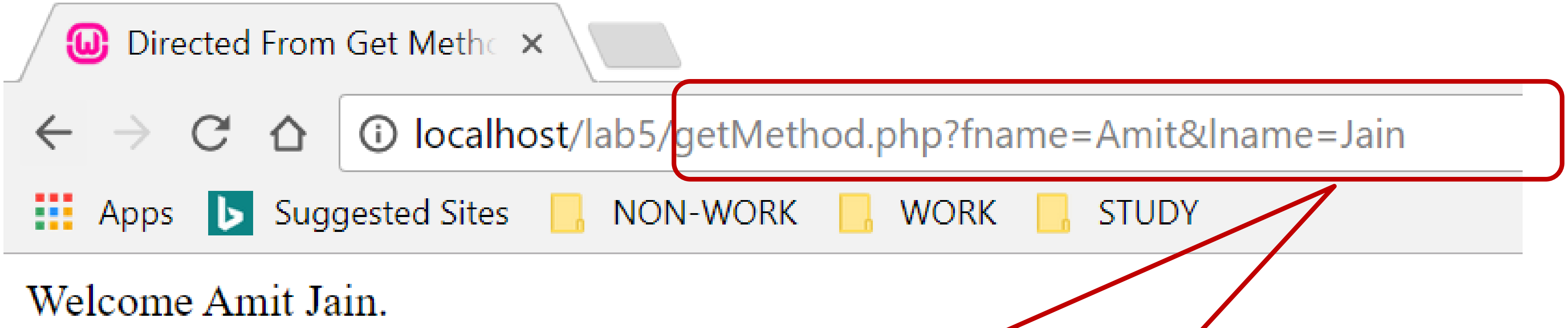
```
<html>
    <head><title>
    Directed From Get Method Example
    </title></head>
    <body>
     Welcome
     <?php
          echo $_GET["fname"];
          echo " ";
          echo $_GET["lname"];
          echo ".";
     ?>
</body></html>
```

Contd…



First Name: Amit

Last Name: Jain



Welcome Amit Jain.

getMethod.php?fname=Amit&lname=Jain

Example – POST

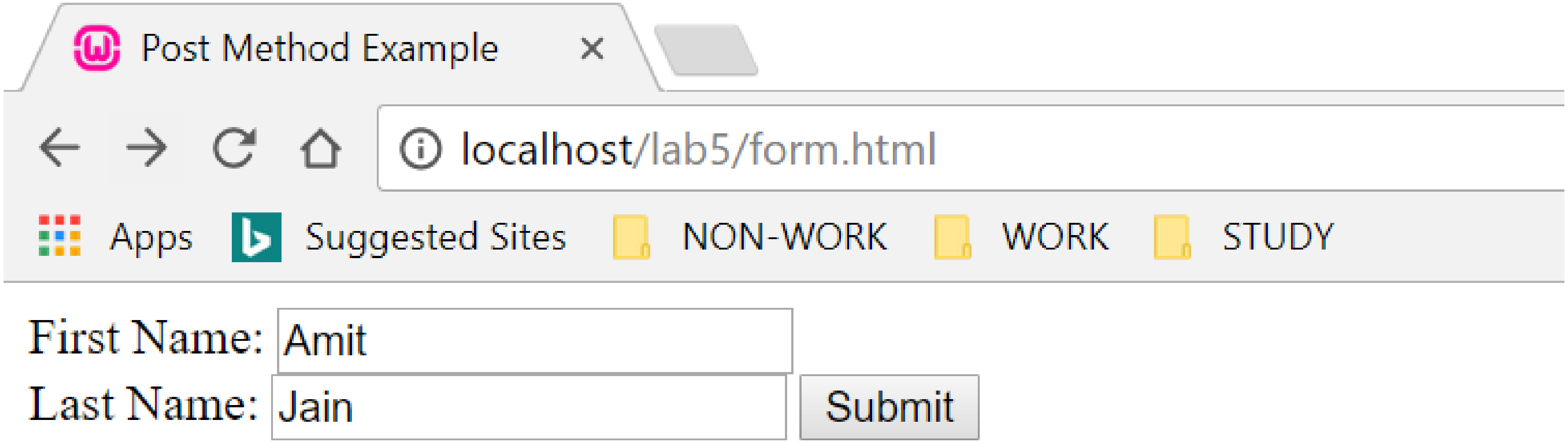
Form.html

```
<html>
  <head><title>
    Post Method Example
  </title></head>
  <body>
    <form action="postMethod.php"
          method="post">
      First Name: <input type="text"
          name="fname" /> <br>
      Last Name: <input type="text"
           name="lname" />
      <input type="submit" />
    </form>
</body></html>
```

postMethod.php

```
<html>
    <head><title>
    Directed From Post Method Example
    </title></head>
    <body>
     Welcome
     <?php
          echo $_POST["fname"];
          echo " ";
          echo $_POST["lname"];
          echo ".";
     ?>
</body></html>
```

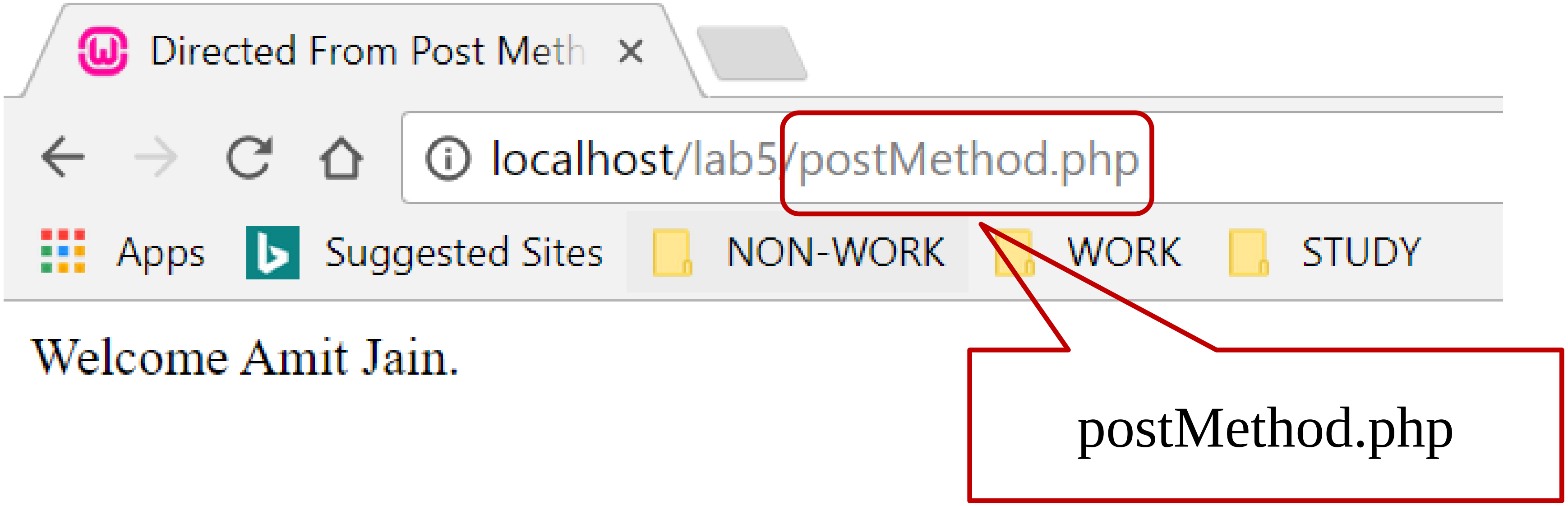
Contd…



localhost/lab5/form.html

First Name: Amit

Last Name: Jain



localhost/lab5/postMethod.php

Welcome Amit Jain.

postMethod.php

Strings in PHP

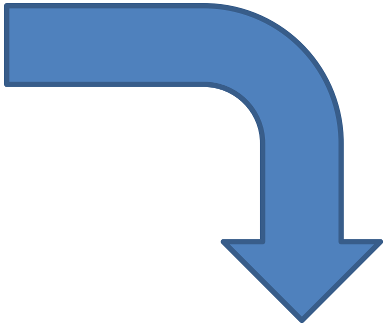
Function	Example	
Creation	$str1 = "Hello!";	
Concatenation	$str1 = "Hello!"; $str2 = “Amit"; echo $str1. " " . $str2;	
String Length	strlen("Hello!");	= 6
Searching	strpos("Hello! Amit",“Amit");	= 7

Contd…

Check postMethod.php and Replace

```
<?php
   echo $_POST["fname"];
   echo " ";
   echo $_POST["lname"];
   echo ".";
?>
```

It works both ways



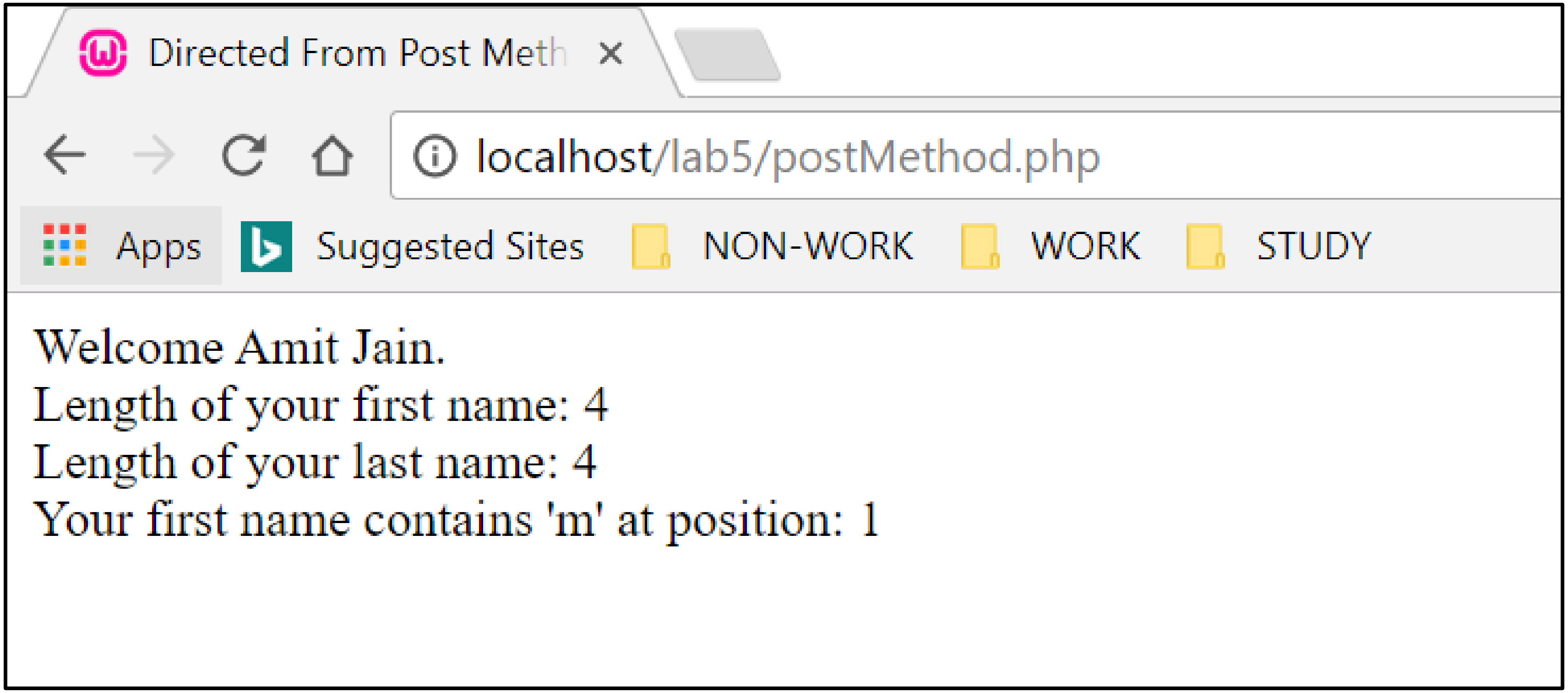
```
<?php
   echo $_POST["fname"]." ".$_POST["lname"].".";
?>
```

Example - String

- Update postMethod.php

```
 <?php
      echo $_POST["fname"]." ".$_POST["lname"].".";
      echo "<br>Length of your first name: ".
                   strlen($_POST["fname"]);
      echo "<br>Length of your last name: ".
                   strlen($_POST["lname"]);
      echo "<br>Your first name contains ‘m' at position:
                   ". strpos($_POST["fname"],“m");
 ?>
```

Contd…



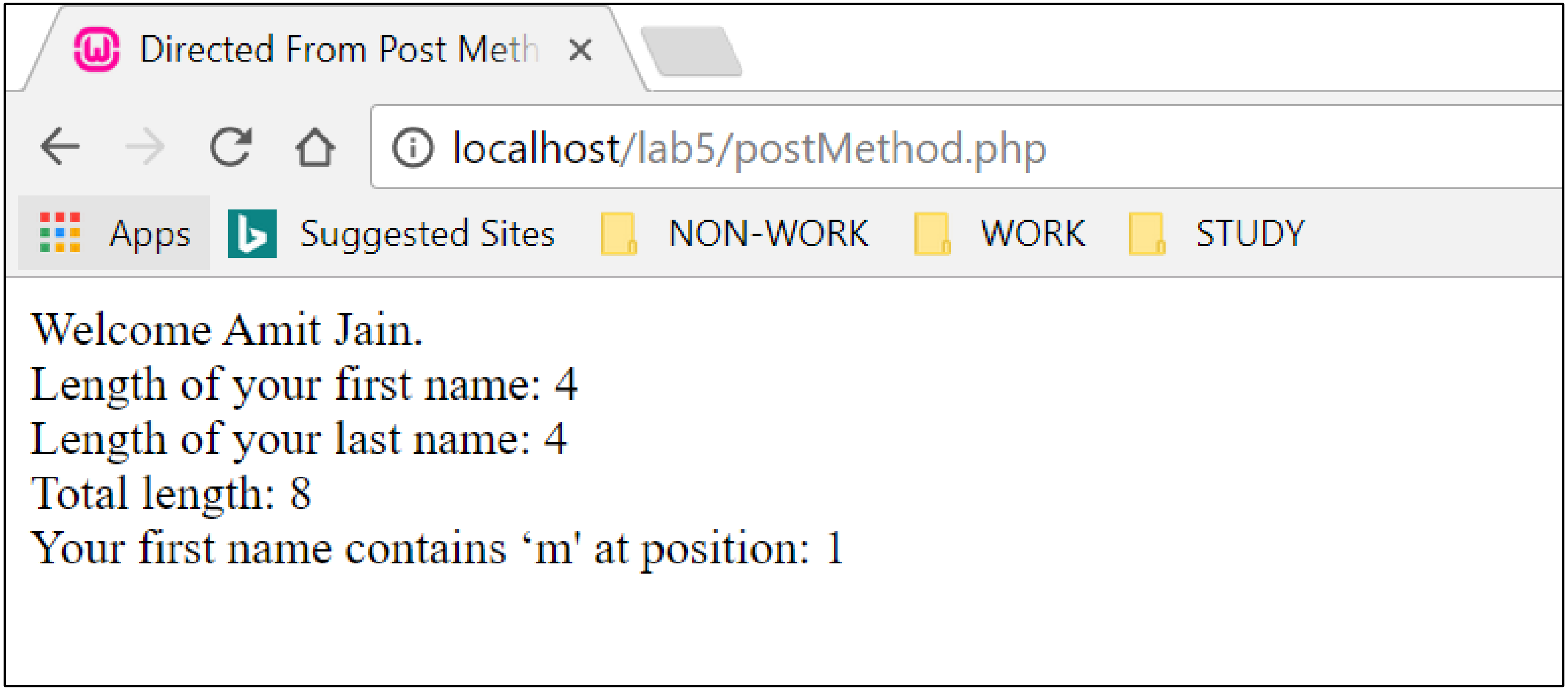
```
Welcome Amit Jain.
Length of your first name: 4
Length of your last name: 4
Your first name contains 'm' at position: 1
```

Example – String Operators

- postMethod.php

```
 <?php
        $fnlen = strlen($_POST["fname"]);
        $lnlen = strlen($_POST["lname"]);
        $len = $fnlen + $lnlen;
        echo $_POST["fname"]." ".$_POST["lname"].".";
        echo "<br>Length of your first name: ".$fnlen;
        echo "<br>Length of your last name: ".$lnlen;
        echo "<br>Total length: ".$len;
        echo "<br>Your first name contains ‘m' at position:
            ".strpos($_POST["fname"],“m");
 ?>
```

Contd…



```
Welcome Amit Jain.
Length of your first name: 4
Length of your last name: 4
Total length: 8
Your first name contains ‘m' at position: 1
```

Control Structure in PHP

- If…Else statement
- Switch statement
- For Loop
- While Loop
- Do-while Loop
- For Loop
- Foreach Loop

Conditional Statements

1. If Statement

Syntax:

```
        if ( condition true )
        {
            PHP statements;
        }
```

2. If / else Statement

Syntax:

```
        if ( condition true )
        {
            PHP statements;
        }
        else // condition false
        {
            PHP statements;
        }
```

Contd…

3. Elseif Statement

Syntax:

```
if ( condition1 true ){
      PHP statements;
}elseif (condition2 true){
      PHP statements;
}…
…
}else{
      PHP statements;
}
```

Contd…

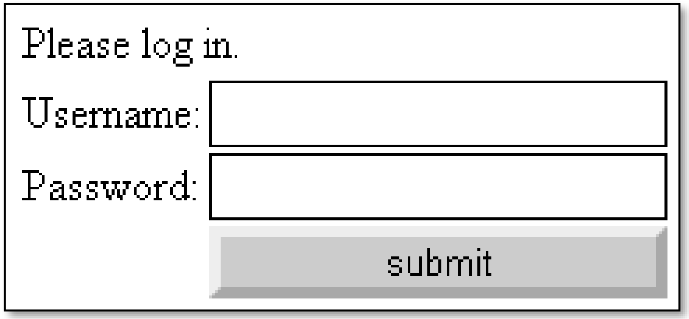
4. Switch Statement

Syntax:

```
switch ($var){
    case label1:
        PHP statements;
        break;
    …
    case labeln:
        PHP statements;
        break;
```

```
default:
    PHP statements;
    break;
}
```

A Simple example using text field element :



```
<html>
<head> <title>Login</title> </head>
<body>
    <form method="post" action="login.php">
        Please log in.<br/>
        Username: <input name="username" type="text" /><br />
        Password: <input name="password" type="password" /><br/>
        <input name="submit" type="submit" />
    </form>
</body> </html>
```

```
<?php
    if($_POST['username'] == “amitjain" && $_POST['password'] ==
    “password123")
    {       echo("Welcome, Amit Jain.");        }
    else
    {       echo("You're not Amit Jain!");        }
?>
```

Looping Statements

while Loop

Syntax:

```
    while ( condition true )
    {
        PHP statements;
    }
```

do…while Statement

Syntax:

```
    do
    {
        PHP statements;
    }
    while ( condition true )
```

Contd…

for Loop

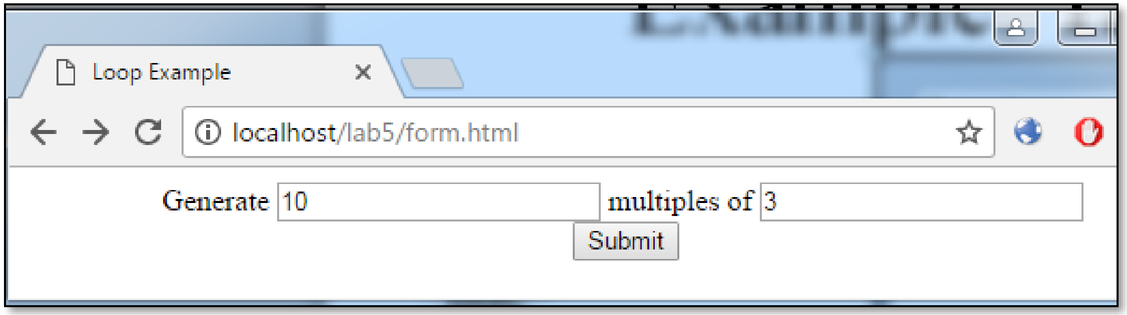
Syntax:

```
for (initial value;
      condition;
      increment)
        {
        PHP statements;
        }
```

Example: Table Generator

Form.html

```
<html>
<head>
   <title>
         Loop Example
   </title>
</head>
<body>
      <center><form action="loop.php" method="post">
         Generate <input type="text" name="mul" />
         multiples of <input type="text" name="num" />
         <br><input type="submit" />
      </form></center>
</body></html>
```

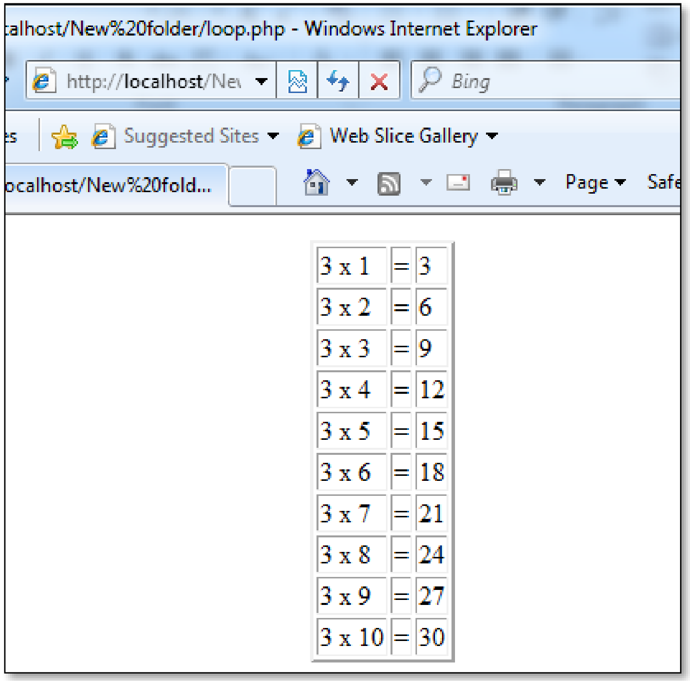


Generate 10 multiples of 3

Example: Table Generator

loop.php

```
<html><body><center><?php
        $mul = $_POST["mul"];
        $num = $_POST["num"];
        echo '<table align=\"center\ border= 2">';
        for ( $i = 1; $i <= $mul; $i++) {
                echo "<tr><td>";
                echo $num." x ".$i;
                echo "</td><td> =
                </td> <td> ";
                echo $num * $i;
                echo "</td></tr>";}
        echo "</table>";
?></center></body></html>
```



3 x 1	=	3
3 x 2	=	6
3 x 3	=	9
3 x 4	=	12
3 x 5	=	15
3 x 6	=	18
3 x 7	=	21
3 x 8	=	24
3 x 9	=	27
3 x 10	=	30

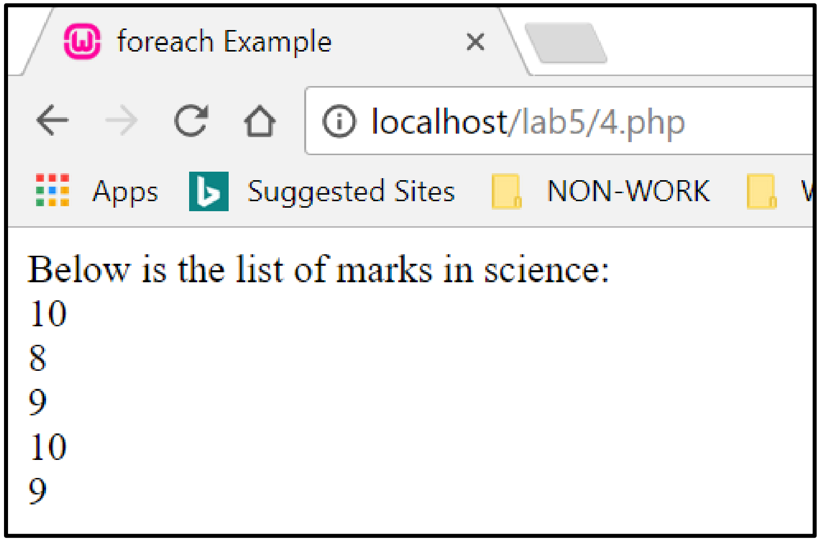
foreach Loop

- Used to loop through arrays.
- Syntax:

```
foreach($array as $value)
{
        PHP statements;
}
```

- For every loop iteration, the value of the current array element is assigned to $value and the array pointer is moved by one, until it reaches the last array element - so on the next loop iteration, we can look at the next array value.

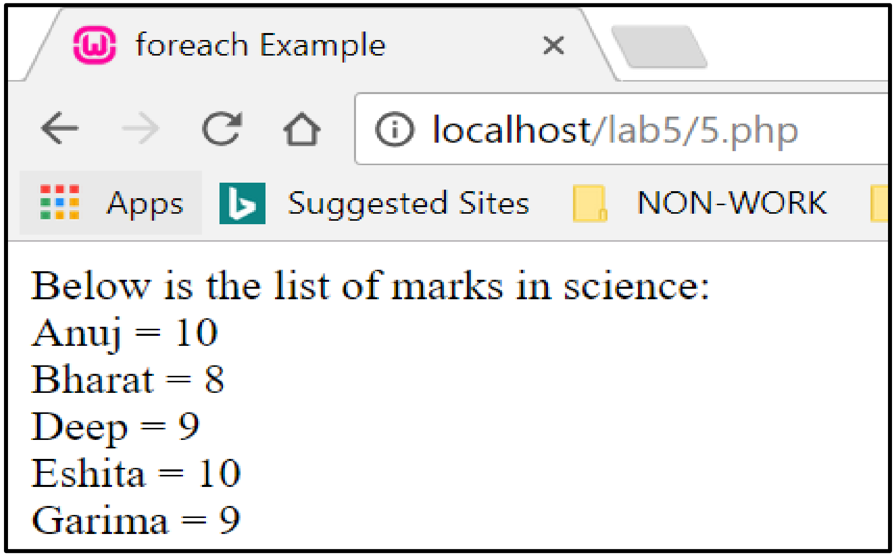
Example of foreach (using numeric array)



```
Below is the list of marks in science:
10
8
9
10
9
```

```
<html>
    <head><title>
        foreach Example
    </title></head>
    <body>
        <?php
            $marks = array(10, 8, 9, 10, 9);
            echo 'Below is the list of marks in science:<br>';
            foreach ( $marks as $i) {
                echo $i."<br>";        }        ?>
</body></html>
```

Example of foreach (using associative array)



```
Below is the list of marks in science:
Anuj = 10
Bharat = 8
Deep = 9
Eshita = 10
Garima = 9
```

```
<html>
    <head><title>
        foreach Example
    </title></head>
    <body>
        <?php
            $marks = array("Anuj"=>10, "Bharat"=>8, "Deep"=>9,
             "Eshita" =>10, "Garima"=>9);
            echo 'Below is the list of marks in science:<br>';
            foreach ( $marks as $name => $i) {
                    echo $name." = ".$i."<br>“;}
        ?></body></html>
```

include

- Takes file name as an input and inserts the contents of the specified PHP file into the issuing PHP script.
- Used to insert same PHP, HTML, or text segment on multiple pages of a website, for example, a menu.
- Saves lot of time.
- Syntax:

```
include("file_name.php");
```

Example - include

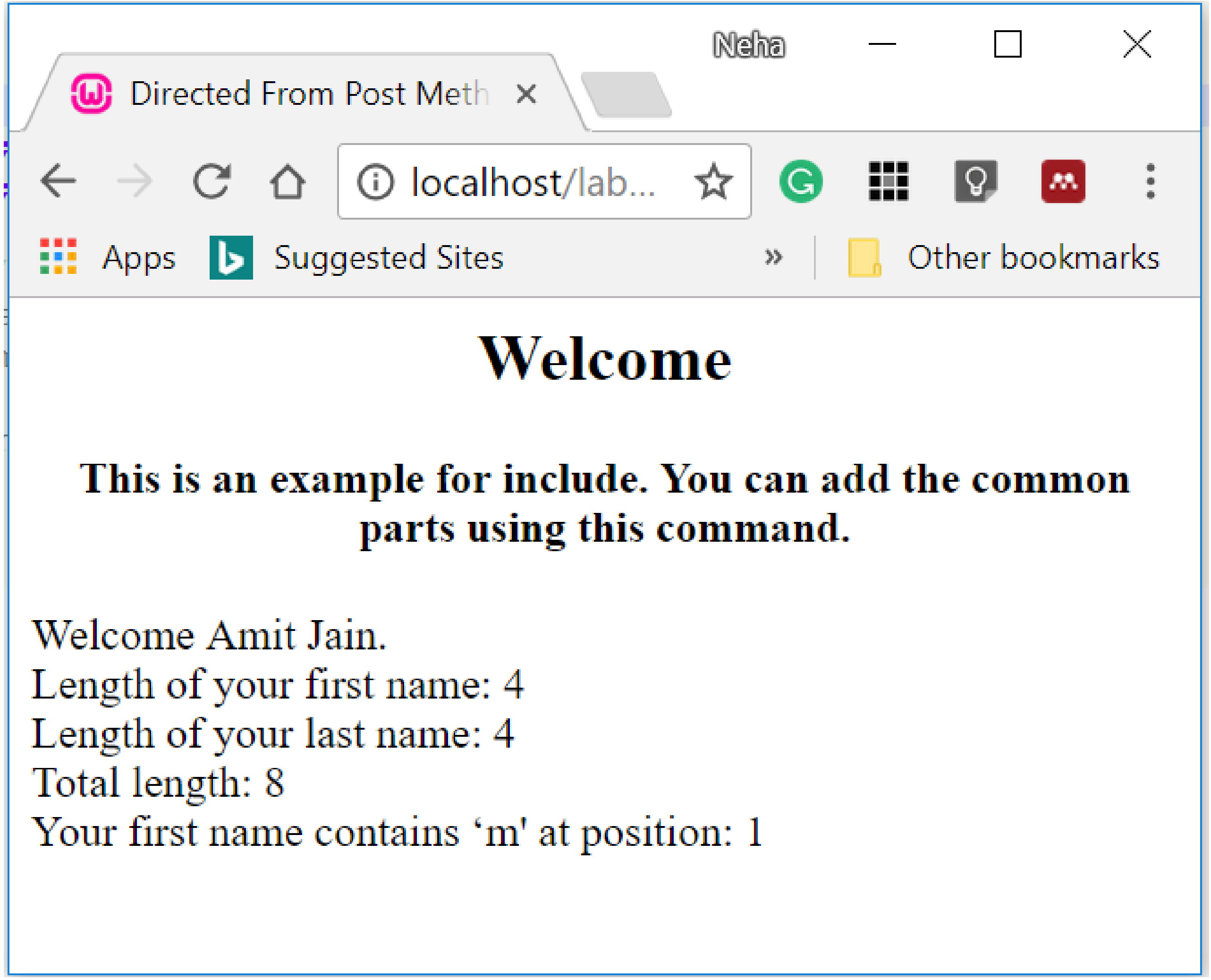
welcome.php

```
<html>
  <body>
    <center><h2> Welcome
            </h2></center>
    <center><h4> This is an
            example for include. You
            can add the common
            parts using this
            command.
            </h4></center>
  </body>
</html>
```

postMethod.php

```
<?php
  include("welcome.php");
  …
  …
?>
```

Contd…



Welcome

This is an example for include. You can add the common parts using this command.

```
Welcome Amit Jain.
Length of your first name: 4
Length of your last name: 4
Total length: 8
Your first name contains ‘m' at position: 1
```

require

- It is same as the include command.
- But it stops the execution of a PHP script if necessary files are missing or misnamed, which doesn’t happen in the case of ***include*** command.
- Syntax:

  require("file_name.php");

HTML FORM ELEMENT INTERACTION

Revisit

- The PHP script
  - running at server,
  - receives the data from the form and
  - uses it to perform an action such as updating database contents, sending database format, user authentication etc.
- To create a form and point it to a PHP document, the HTML tag <form> is used and an action is specified as follows:

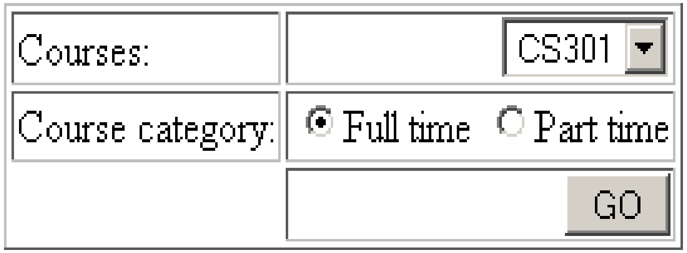
```
<form method="post" action="action.php">
        <!-- Your form here -->
</form>
```

- All fields in the form are stored in the variables $_GET or $_POST, depending on the method used to submit the form
- Difference
  - GET submits all the values in the URL, while
  - POST submits values transparently through HTTP headers.

Form Revisited: Radio Buttons & Drop Down Lists

```
<html>  <body>
   <form action="drop_radio.php" method="post">
   <table border=1>
 <tr><td> Courses : </td>
      <td align = 'right'> <select name = "courses">
           <option value = "CS301">CS301</option>
           <option value = "EC301">EC301</option>
           <option value = "CS611">CS611 </option>
           <option value = "EC612">EC612 </option>
        </select></td>
</tr>

<tr><td> Course Category </td>
      <td align ='right'>
  <input type = "radio" name = "category" value="full-time” checked="checked"/>Full time
  <input type = "radio" name = "category" value="part-time" /> part time</br>
  </td>
</tr>
<tr><td></td>
      <td align = 'right'><input type = "submit" name="submit” value="GO"/></td>
</tr>
</table></form>
</body></html>
```



Courses:	CS301 ▾
Course category:	● Full time ○ Part time
	GO

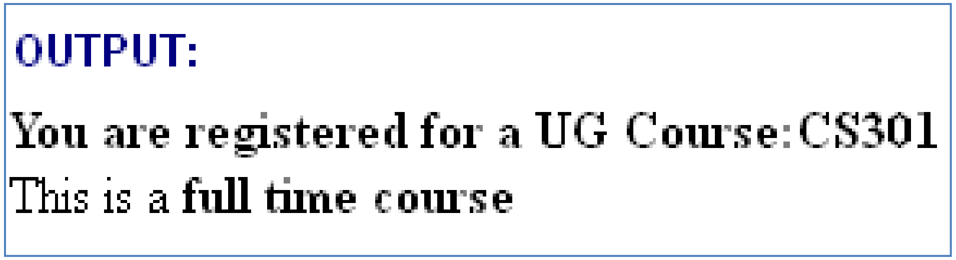
PHP Code that handles the previous form:

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

    if($_POST["courses"]=='CS301' OR $_POST["courses"] == 'EC301')
       { $ctype='UG Course'; }
     else
       { $ctype='PG Course'; }

    if($_POST["category"]=='full-time')
      { $catg='full time';}
    else
      { $catg='part time';}

      echo "</b> You are registered for a ".$ctype.":".$_POST["courses"]."</b></br>";
      echo "This is a <b>".$catg."course</b>";
  ?>
</body>
</html>
```

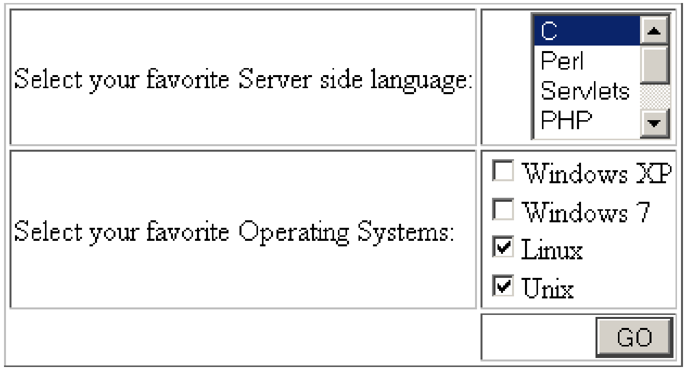


OUTPUT:

```
You are registered for a UG Course:CS301
This is a full time course
```

EX: CHECK BOXES & MULTILISTS

```
<body>
<form action="check_multilist.php" method="post">
<table border=1>
<tr><td> Select your favorite server side language:</td><td align = 'right'>
    <select name = "lang[]" multiple="multiple">
           <option value = "C"> C </option>
           <option value = "Perl"> Perl </option>
           <option value = "Servlets"> Servlets </option>
           <option value = "PHP"> PHP </option>
            <option value = "ASP"> ASP </option>
            <option value = "JSP"> JSP </option>
     </select></td></tr>
<tr><td> Select your favorite Operating Systems: </td> <td align = 'left'>
        <input type = "checkbox" name = "OS[]" value="Windows XP" checked="checked"/>
Windows XP</br>
        <input type = "checkbox" name = "OS[]" value="Windows 7" /> Windows 7</br>
        <input type = "checkbox" name = "OS[]" value="Linux"/> Linux</br>
        <input type = "checkbox" name = "OS[]" value="Unix"/> Unix</br>
</td></tr>
<tr><td></td><td align = 'right'><input type = "submit"
name="submit"value="GO"/></td></tr>
</table>
</form></body>
```



Select your favorite Server side language:	C Perl Servlets PHP
Select your favorite Operating Systems:	☐ Windows XP ☐ Windows 7 ☑ Linux ☑ Unix
	GO

PHP Code that handles the previous form:

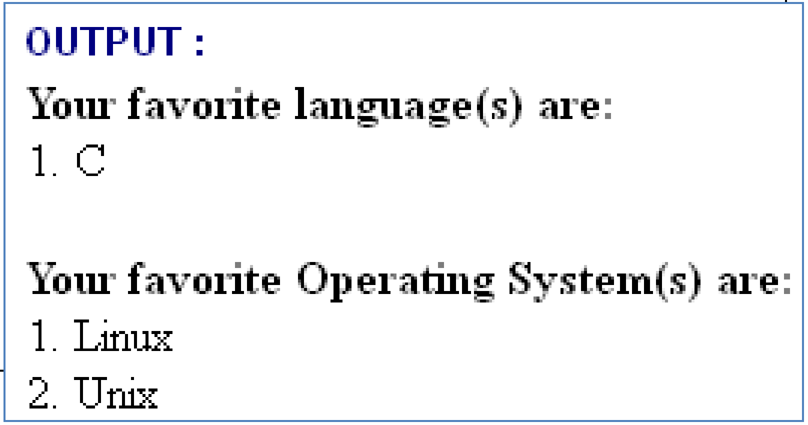
```php
<?php

$lang = $_POST['lang'];
$OS = $_POST['OS'];

$count = count ($lang);
echo "<b> Your Favorite Language(s) are: </b></br>";
for($i = 0; $i< $count; $i++)
{
        echo ($i + 1 . "." . $lang[$i] . "<br/>");
}

echo "</br></br><b> Your Favorite Operating System(s) are: </b></br>";
$count = count($OS);
for($i = 0; $i < $count ; $i++)
{
        echo($i + 1 . "." . $OS[$i] . "<br/>");
}

?>
```



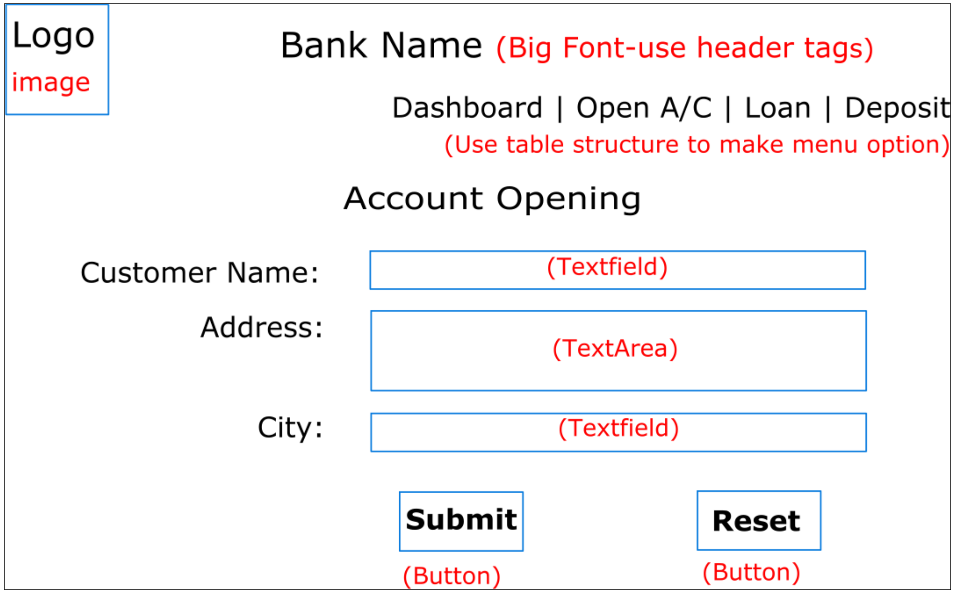
OUTPUT :

```
Your favorite language(s) are:
1. C

Your favorite Operating System(s) are:
1. Linux
2. Unix
```

Ready for Assignments …

Assignment 5



Logo
image

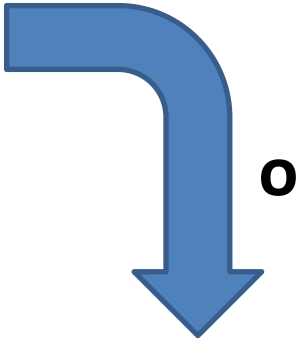
Bank Name (Big Font-use header tags)

Dashboard | Open A/C | Loan | Deposit
(Use table structure to make menu option)

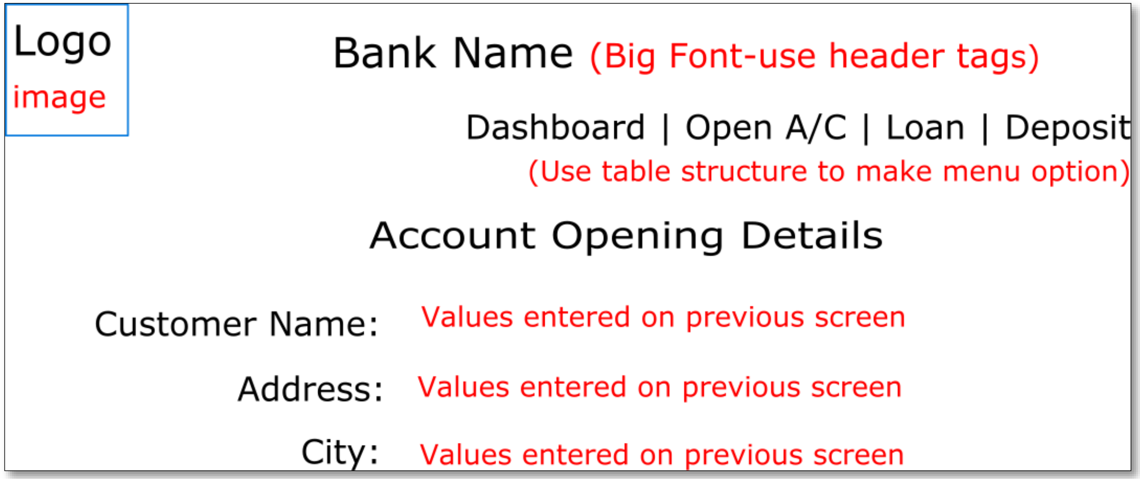
Account Opening

Customer Name:	(Textfield)
Address:	(TextArea)
City:	(Textfield)

Submit (Button) **Reset** (Button)



on submit Form



Logo
image

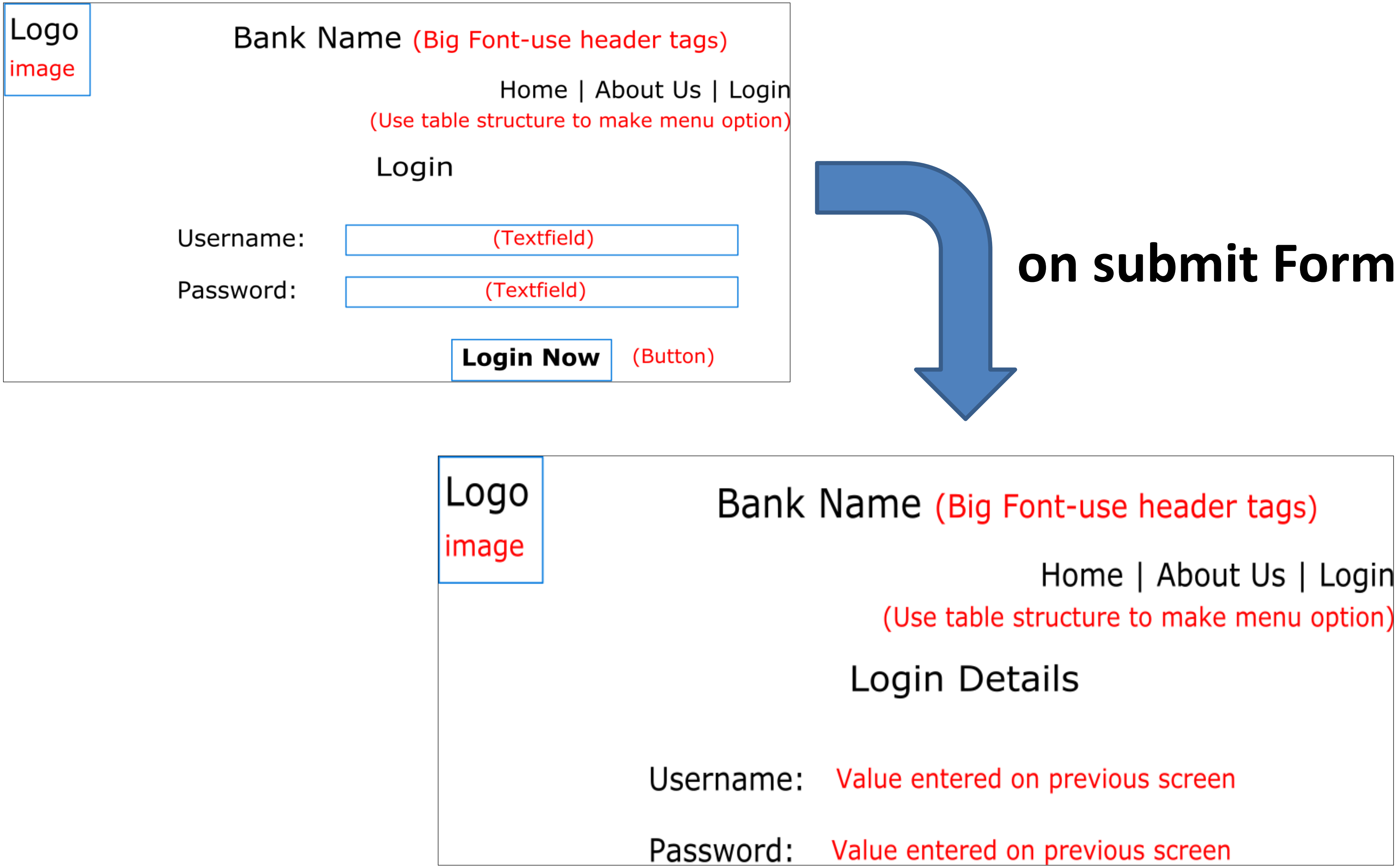
Bank Name (Big Font-use header tags)

Dashboard | Open A/C | Loan | Deposit
(Use table structure to make menu option)

Account Opening Details

Customer Name:	Values entered on previous screen
Address:	Values entered on previous screen
City:	Values entered on previous screen

Assignment 6



Logo image

Bank Name (Big Font-use header tags)

Home | About Us | Login

(Use table structure to make menu option)

Login

Username: (Textfield)

Password: (Textfield)

Login Now (Button)

on submit Form

Logo image

Bank Name (Big Font-use header tags)

Home | About Us | Login

(Use table structure to make menu option)

Login Details

Username: Value entered on previous screen

Password: Value entered on previous screen