SWAMI SACCHIDANAND POLYTECHNIC COLLEGE, VISNAGAR (384315)

BRANCH: IT ENGINEERING **SEMESTER: 5th SEM**

SUBJECT NAME: PHP **SUBJECT CODE: 2ET4040504**

1. **What is Open Source?**

- **Open Source refers to software whose source code is made freely available to the public, allowing anyone to view, use, modify, and distribute the code. The term is most commonly used in the context of software development, but it can also apply to other types of creative and collaborative work.**

Examples of Open Source Software:

- **Linux** (Operating system)
- **Android** (Mobile OS)
- **Mozilla Firefox** (Web browser)
- **LibreOffice** (Office suite)
- **Apache** (Web server)
- **Python** (Programming language)

2. **Explain Following Terms:**

(A) PHP =

Definition:
PHP is a server-side scripting language designed for web development but also used as a general-purpose programming language.

Key Features:

- **Runs on the server before the webpage is sent to the browser.**
- **Embeds easily within HTML.**

- **Used to create dynamic web pages, handle forms, manage sessions, interact with databases, etc.**

(B) Apache =

Definition:
Apache is a free, open-source web server software that delivers web content (like HTML, PHP pages) to users via the internet.

Key Features:

- **One of the most widely used web servers globally.**
- **Supports multiple programming languages (including PHP via modules).**
- **Highly customizable with modules (e.g., mod_php, mod_ssl).**

(C) MySQL =

Definition:
MySQL is an open-source relational database management system (RDBMS) that uses SQL (Structured Query Language) to manage data.

- **Key Features:**
- **Stores and retrieves data for websites and applications.**
- **Works well with PHP to handle user data, login systems, etc.**
- **Fast, reliable, and used in many large-scale websites (e.g., Facebook).**

3. Relationship Between PHP, MySQL, and Apache

These three components work together to create dynamic, database-driven websites and are often bundled in what's called the LAMP stack:

1. PHP (Hypertext Preprocessor) – *The Server-Side Script*

- **A programming language used to create dynamic web pages.**

- **Executes on the server and can interact with databases, send emails, and generate HTML output.**
- **Example: Validating a form, processing login, or fetching data from MySQL.**

2. MySQL – *The Database*

- **A relational database management system (RDBMS) used to store and manage data.**
- **PHP scripts send SQL queries to MySQL to retrieve, insert, update, or delete data.**
- **Example: Storing user data, blog posts, products, etc.**

3. Apache – *The Web Server*

- **A web server software that handles HTTP requests from clients (browsers).**
- **Receives requests, processes them (often through PHP), and serves the result (HTML) to the browser.**
- **Acts as a bridge between PHP and the web.**

Visual Summary:

[Browser]

↓

[Apache Web Server]

↓

[PHP Processor]

↔

[MySQL Database]

↓

[HTML Output sent to Browser]

4. Explain PHP structure and syntax.

- **A PHP script can be placed anywhere in the document.**
- **A PHP script starts with <?php and ends with ?>:**

```
<?php

 // PHP code goes here

?>
```

- **The default file extension for PHP files is ".php".**
- **A PHP file normally contains HTML tags, and some PHP scripting code.**

```
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

5. **Explain how to create the PHP pages.**

Step 1: Set Up a Local Server (AMP Stack)
PHP runs on a server. For development, use a local server like:

- **XAMPP (Windows/Linux/macOS)**
- **WAMP (Windows only)**
- **MAMP (macOS)**

Step 2: Locate the Web Directory
Once XAMPP is installed:

- **Go to: C:\xampp\htdocs\ (this is the web root folder)**
- **Create a folder here for your project, e.g., myproject**

Step 3: Create a PHP File

- **Open a code editor like Notepad, VS Code, or Sublime Text.**
- **Write your PHP code.**

Example (p1.php)

<?php

echo "Hello, Rudraksh! This is your first PHP page.";

?>

- **Save the file as index.php in: C:\xampp\htdocs\myproject**

Step 4: Run Apache Server

- **Open XAMPP Control Panel**
- **Start the Apache module (turn it green)**

Step 5: Open the PHP Page in Browser

- **Open a browser and type:**

http://localhost/myproject/index.php

6. Write rules of PHP syntax.

- **PHP code must be written inside <?php ... ?> tags** within the HTML file.
- **HTML can be written outside or inside PHP tags**, but to output HTML from PHP, it must be enclosed in a string and properly echoed or printed.
- **File must be saved with a .php extension** for the PHP code to execute on the server.
- **PHP code is executed on the server side**, and the resulting output (usually HTML) is sent to the client/browser.
- **Variables defined in PHP can be embedded in HTML using echo/print**, ensuring correct string handling and escaping.
- **Forms written in HTML can send data to PHP scripts** using the GET or POST method with the action attribute pointing to a .php file.
- **PHP code can be conditionally used to control which HTML elements are shown**, using control statements like if, else, switch, and loops.
- **Quotes and HTML tags must be properly opened and closed** when outputting HTML from PHP to avoid syntax errors.
- **Avoid writing large blocks of HTML inside PHP**, as it reduces code readability; instead, switch between PHP and HTML when needed.

- **Superglobal variables** like `$_GET`, `$_POST`, `$_SESSION`, etc., are used in PHP to receive and process data from HTML forms.

7. Explain Constant with example.

➢ **A constant is a name or an identifier for a fixed value. Constant are like variables except that one they are defined, they cannot be undefined or changed (except magic constants).**

- **Constants are very useful for storing data that doesn't change while the script is running. Common examples of such data include configuration settings such as database username and password, website's base URL, company name, etc.**
- **Constants are defined using PHP's define() function, which accepts two arguments: the name of the constant, and its value. Once defined the constant value can be accessed at any time just by referring to its name.**

8. **Explain Variable with example. Write rules for naming variable.**
    - **Variables are used to store data, like string of text, numbers, etc. Variable values can change over the course of a script. Here're some important things to know about variables:**
    - **In PHP, a variable does not need to be declared before adding a value to it. PHP automatically converts the variable to the correct data type, depending on its value.**
    - **After declaring a variable it can be reused throughout the code.**
    - **The assignment operator (=) used to assign value to a variable.**
    - **In PHP variable can be declared as: $var_name = value;**

9. **Explain static variable and global variable with example.**

    Static Variable =

    => A static variable in PHP retains its value even after the function exits. It is initialized only once and does not reset each time the function is called.

```
<?php
function myCounter() {
   static $count = 0;  // static variable
   $count++;
   echo $count . "<br>";
}
myCounter();
```

myCounter();

myCounter();

?>

Output = 1

2

3

Global Variable =

=>A global variable is defined outside any function and can be accessed inside a function by using the global keyword.

```
<?php
$a = 5;  // global variable
function displayValue() {
   global $a;
   echo $a;
}
displayValue();
?>
```

Output =

5

10. **What is Data type? Why PHP is called a loosely typed language?**
    =>A data type defines the type of data a variable can hold.
    In PHP, data types determine what kind of value is stored in a variable (e.g., number, text, true/false, etc.).

=>PHP is called a *loosely typed* language because:

1. **No need to declare data types explicitly.**
   You do not have to specify whether a variable is an integer, string, etc.

```
$x = 10;       // integer
$x = "Hello";  // now it's a string
```

2. **Automatic type conversion**
   PHP automatically converts data types as needed during execution.

```
$a = "10";
$b = 5;
echo $a + $b;   // Output: 15 (PHP converts "10" to integer)
```