1. PHP का परिचय

PHP क्या है?

PHP (Hypertext Preprocessor) एक सर्वर-साइड स्क्रिप्टिंग लैंग्वेज है जिसका उपयोग वेब डेवलपमेंट में किया जाता है। यह एक ओपन-सोर्स भाषा है जिसे वेब पेजों को डायनामिक और इंटरैक्टिव बनाने के लिए उपयोग किया जाता है। PHP को HTML के साथ मिलाकर इस्तेमाल किया जाता है और यह बैकएंड प्रोग्रामिंग के लिए लोकप्रिय विकल्प है।

PHP की इतिहास और विकास यात्रा

PHP की शुरुआत 1994 में Rasmus Lerdorf ने की थी। यह शुरुआत में 'Personal Home Page' के रूप में विकसित किया गया था, लेकिन बाद में इसका नाम बदलकर PHP: Hypertext Preprocessor कर दिया गया।

PHP के प्रमुख संस्करण:

- PHP 3 (1998): पहला व्यापक रूप से अपनाया गया संस्करण।
- PHP 4 (2000): बेहतर परफॉरमेंस और फीचर्स के साथ।
- PHP 5 (2004): ऑब्जेक्ट-ओरिएंटेड प्रोग्रामिंग (OOP) की सुविधा जोड़ी गई।
- PHP 7 (2015): तेज़ गति और कम मेमोरी उपयोग।
- PHP 8 (2020): JIT (Just-In-Time) कंपाइलर और नए फीचर्स।

PHP का उपयोग क्यों करें?

PHP को वेब डेवलपमेंट के लिए कई कारणों से पसंद किया जाता है:

1. **ओपन-सोर्स और फ्री:** इसे मुफ्त में डाउनलोड और उपयोग किया जा सकता है।
2. **डायनामिक कंटेंट:** PHP के माध्यम से डायनामिक वेबसाइट्स बनाई जा सकती हैं।
3. **डेटाबेस सपोर्ट:** यह MySQL, PostgreSQL, SQLite, और अन्य डेटाबेस को सपोर्ट करता है।
4. **इंटीग्रेशन:** इसे HTML, CSS, JavaScript, और अन्य तकनीकों के साथ आसानी से इंटीग्रेट किया जा सकता है।
5. **सर्वर-साइड स्क्रिप्टिंग:** PHP का कोड क्लाइंट-साइड पर नहीं दिखता, जिससे सिक्योरिटी बनी रहती है।
6. **प्लेटफ़ॉर्म स्वतंत्र:** Windows, Linux, और MacOS पर चलता है।
7. **तेज़ और कुशल:** अन्य भाषाओं की तुलना में तेज़ एक्सीक्यूशन स्पीड होती है।

Static और Dynamic Websites का अंतर

विशेषता	Static Website	Dynamic Website
कंटेंट बदलता है?	नहीं	हां
डेटा बेस उपयोग	नहीं	हां
उपयोग में आसान	हां	थोड़ा जटिल

टेक्नोलॉजी	HTML, CSS	PHP, MySQL, JavaScript
उदाहरण	Portfolio, Blog	E-commerce, Social Media

Static वेबसाइट्स केवल HTML और CSS पर आधारित होती हैं और कंटेंट को बार-बार मैन्युअली अपडेट करना पड़ता है। Dynamic वेबसाइट्स में PHP का उपयोग होता है जिससे डेटा ऑटोमेटिकली अपडेट हो सकता है।

2. PHP इंस्टालेशन और सेटअप

XAMPP/WAMP इंस्टॉल करना

PHP को अपने कंप्यूटर पर चलाने के लिए हमें एक लोकल सर्वर चाहिए होता है। इसके लिए सबसे ज्यादा उपयोग होने वाले सॉफ्टवेयर XAMPP और WAMP हैं।

XAMPP इंस्टॉल करने के स्टेप्स:

1. Apache Friends **वेबसाइट पर जाएं।**
2. **अपनी ऑपरेटिंग सिस्टम (Windows/Linux/Mac) के अनुसार XAMPP डाउनलोड करें।**
3. **इंस्टॉल करने के बाद XAMPP कंट्रोल पैनल खोलें।**
4. **Apache और MySQL को स्टार्ट करें।**
5. **अब आपका लोकल सर्वर तैयार है।**

WAMP इंस्टॉल करने के स्टेप्स:

1. WAMP Server की आधिकारिक वेबसाइट पर जाएं।
2. अपने सिस्टम के अनुसार 32-bit या 64-bit वर्जन डाउनलोड करें।
3. इंस्टॉल करने के बाद WAMP कंट्रोल पैनल से **Apache और MySQL** स्टार्ट करें।
4. ब्राउज़र में `localhost` टाइप करें, अगर WAMP का डैशबोर्ड खुले तो इंस्टॉलेशन सफल रहा।

पहला PHP स्क्रिप्ट लिखना और चलाना

PHP फाइल बनाना

1. अपने कंप्यूटर के **C:/xampp/htdocs/** फोल्डर में जाएं।
2. एक नया फोल्डर बनाएं, उदाहरण: `myphp`।
3. उसके अंदर `index.php` नाम से एक फाइल बनाएं।
4. इसमें नीचे दिया गया कोड लिखें:

```
<?php
  echo "Hello, PHP World!";
?>
```

PHP फाइल को रन करना

1. ब्राउज़र खोलें और URL बार में टाइप करें:
   `http://localhost/myphp/index.php`
2. अगर स्क्रीन पर **Hello, PHP World!** दिख रहा है तो PHP सही तरीके से काम कर रहा है।

PHP फाइल का स्ट्रक्चर

PHP फाइल्स .php एक्सटेंशन के साथ सेव की जाती हैं। एक सामान्य PHP फाइल इस प्रकार होती है:

```
<?php
 // PHP कोड यहाँ लिखा जाता है
 echo "यह एक PHP स्क्रिप्ट है!";
?>
```

- PHP स्क्रिप्ट को हमेशा <?php . . . ?> टैग के अंदर लिखा जाता है।
- echo का उपयोग स्क्रीन पर आउटपुट दिखाने के लिए किया जाता है।
- हर स्टेटमेंट के अंत में ; (सेमीकोलन) लगाना जरूरी होता है।

PHP में कमेंट्स का उपयोग

PHP में कोड को समझाने के लिए **कमेंट्स** का उपयोग किया जाता है। ये कोड को प्रभावित नहीं करते।

कमेंट्स के प्रकार:

1. **सिंगल लाइन कमेंट:**

```
// यह एक सिंगल लाइन कमेंट है
```

2. **मल्टी-लाइन कमेंट:**

```
/*
```

यह एक मल्टी-लाइन कमेंट है

जो कई लाइनों में हो सकता है।

*/

3. PHP के मूलभूत तत्व

3.1 वेरिएबल्स और कॉन्स्टेंट्स

वेरिएबल्स क्या होते हैं?

वेरिएबल्स ऐसे स्टोरेज कंटेनर होते हैं जिनमें डेटा को अस्थायी रूप से रखा जाता है। PHP में वेरिएबल्स $ (डॉलर साइन) से शुरू होते हैं।

उदाहरण:

```
<?php
  $name = "Rahul";
  $age = 25;
  echo "नाम: $name, उम्र: $age";
?>
```

PHP में कॉन्स्टेंट्स

कॉन्स्टेंट्स वे होते हैं जिनकी वैल्यू स्क्रिप्ट के रनटाइम के दौरान बदली नहीं जा सकती।

```
<?php
 define("SITE_NAME", "MyWebsite");
 echo SITE_NAME;
?>
```

3.2 डेटा टाइप्स

PHP में निम्नलिखित डेटा टाइप्स होते हैं:

1. **String: टेक्स्ट डेटा (`"Hello"`)**
2. **Integer: संख्यात्मक मान (`25`)**
3. **Float: दशमलव संख्या (`3.14`)**
4. Boolean: `true` **या** `false`
5. **Array: एक से अधिक मानों को स्टोर करने के लिए**
6. **Object: क्लास बेस्ड डेटा**

उदाहरण:

```
<?php
 $x = "Hello"; // String
 $y = 25; // Integer
 $z = 3.14; // Float
 $is_valid = true; // Boolean
```

```
?>
```

3.3 ऑपरेटर्स

PHP में ऑपरेटर्स निम्नलिखित प्रकार के होते हैं:

- **गणितीय ऑपरेटर्स:** `+`, `-`, `*`, `/`, `%`
- **तुलनात्मक ऑपरेटर्स:** `==`, `!=`, `>`, `<`, `>=`, `<=`
- **लॉजिकल ऑपरेटर्स:** `&&`, `||`, `!`

उदाहरण:

```
<?php
  $a = 10;
  $b = 5;
  echo $a + $b; // 15
?>
```

3.4 कंट्रोल स्ट्रक्चर (If-Else, Switch, Loops)

If-Else स्टेटमेंट

```
<?php
  $age = 18;
  if ($age >= 18) {
    echo "आप वोट डाल सकते हैं";
```

```
 } else {
  echo "आप वोट नहीं डाल सकते";
 }
?>
```

Switch स्टेटमेंट

```
<?php
 $day = "Monday";
 switch ($day) {
  case "Monday":
   echo "आज सोमवार है";
    break;
  case "Tuesday":
   echo "आज मंगलवार है";
    break;
  default:
   echo "कोई मान्यता प्राप्त दिन नहीं";
 }
?>
```

लूप्स (Loops)

PHP में तीन मुख्य प्रकार के लूप होते हैं:

1. **For Loop:**

```
<?php
 for ($i = 1; $i <= 5; $i++) {
  echo "संख्या: $i <br>";
 }
?>
```

2. **While Loop:**

```
<?php
 $x = 1;
 while ($x <= 5) {
  echo "संख्या: $x <br>";
  $x++;
 }
?>
```

3. **Do-While Loop:**

```
<?php
 $y = 1;
 do {
  echo "संख्या: $y <br>";
  $y++;
```

```
} while ($y <= 5);
?>
```

4. PHP में फंक्शन्स

4.1 बिल्ट-इन फंक्शन्स

PHP में पहले से मौजूद कई बिल्ट-इन फंक्शन्स होते हैं, जो विभिन्न कार्यों के लिए उपयोग किए जाते हैं। कुछ महत्वपूर्ण फंक्शन्स निम्नलिखित हैं:

String Functions

```
<?php
 $text = "Hello PHP";
 echo strlen($text); // String की लंबाई बताता है
 echo strrev($text); // String को उल्टा करता है
 echo strtoupper($text); // Uppercase में बदलता है
 echo strtolower($text); // Lowercase में बदलता है
?>
```

Mathematical Functions

```
<?php
```

```
echo max(10, 20, 30); // सबसे बड़ा नंबर देता है

echo min(10, 20, 30); // सबसे छोटा नंबर देता है

echo round(3.6); // संख्या को राउंड करता है (4)

echo sqrt(16); // Square root निकालता है (4)

?>
```

Date & Time Functions

```
<?php

echo date("Y-m-d H:i:s"); // वर्तमान दिनांक और समय देता है

echo time(); // वर्तमान टाइमस्टैम्प देता है

?>
```

4.2 यूजर-डिफाइंड फंक्शन्स

यूजर-डिफाइंड फंक्शन्स वे होते हैं जो हम खुद बनाते हैं।

सिंपल फंक्शन:

```
<?php

 function greet() {

   echo "Hello, Welcome to PHP!";

 }
```

```
greet(); // फंक्शन कॉल करना
?>
```

फंक्शन विद पैरामीटर्स:

```
<?php
 function addNumbers($a, $b) {
   return $a + $b;
 }
 echo addNumbers(5, 10); // आउटपुट: 15
?>
```

4.3 फंक्शन आर्गुमेंट्स और रिटर्न वैल्यू

PHP में फंक्शन्स को पैरामीटर्स दिए जा सकते हैं और वे एक वैल्यू रिटर्न कर सकते हैं।

डिफॉल्ट पैरामीटर:

```
<?php
 function greetUser($name = "Guest") {
   return "Hello, $name!";
 }
 echo greetUser(); // आउटपुट: Hello, Guest!
 echo greetUser("Rahul"); // आउटपुट: Hello, Rahul!
```

```
?>
```

4.4 वेरिएबल स्कोप (Local, Global, Static)

लोकल वेरिएबल:

```
<?php
 function testFunction() {
  $x = 10; // लोकल वेरिएबल
   echo $x;
 }
 testFunction(); // आउटपुट: 10
?>
```

ग्लोबल वेरिएबल:

```
<?php
 $y = 20; // ग्लोबल वेरिएबल
 function display() {
  global $y;
  echo $y;
 }
 display(); // आउटपुट: 20
```

```
?>
```

स्टैटिक वेरिएबल:

```
<?php
 function counter() {
  static $count = 0;
  $count++;
  echo $count;
 }
 counter(); // आउटपुट: 1
 counter(); // आउटपुट: 2
?>
```

---

5. PHP में ऐरे (Arrays)

5.1 Indexed Arrays

Indexed Arrays वे Arrays होते हैं जिनमें values को numerical index (0, 1, 2...) के जरिए access किया जाता है।

Indexed Array Example:

```
<?php
 $fruits = array("Apple", "Banana", "Mango");
```

```
echo $fruits[0]; // Output: Apple
?>
```

Indexed Array with Loop:

```
<?php
 $cars = ["BMW", "Audi", "Mercedes"];
 foreach ($cars as $car) {
  echo $car . "<br>";
 }
?>
```

5.2 Associative Arrays

Associative Arrays में keys को numbers की बजाय strings के रूप में define किया जाता है।

Associative Array Example:

```
<?php
 $person = array("name" => "Rahul", "age" => 25, "city" => "Delhi");
 echo $person["name"]; // Output: Rahul
?>
```

Associative Array with Loop:

```
<?php
 $marks = ["Math" => 90, "Science" => 85, "English" => 88];
```

```
 foreach ($marks as $subject => $score) {
   echo "$subject : $score <br>";
 }
?>
```

5.3 Multidimensional Arrays

Multidimensional Arrays वे होते हैं जिनमें एक array के अंदर multiple arrays होते हैं।

Multidimensional Array Example:

```
<?php
 $students = array(
   array("Rahul", 25, "Delhi"),
   array("Amit", 22, "Mumbai"),
   array("Sneha", 24, "Kolkata")
 );
 echo $students[0][0]; // Output: Rahul
?>
```

Multidimensional Array with Loop:

```
<?php
 $employees = [
   ["Name" => "John", "Age" => 30, "Department" => "HR"],
```

```
  ["Name" => "Sara", "Age" => 28, "Department" => "Finance"],
  ["Name" => "Mike", "Age" => 35, "Department" => "IT"]
];

foreach ($employees as $employee) {
  foreach ($employee as $key => $value) {
    echo "$key: $value | ";
  }
  echo "<br>";
}
?>
```

5.4 Array Functions

PHP में कई useful array functions होते हैं जो arrays को manipulate करने में मदद करते हैं।

Commonly Used Array Functions:

```
<?php
 $numbers = [10, 20, 30, 40, 50];
 echo count($numbers); // Output: 5 (Total elements in array)

 array_push($numbers, 60); // Adds 60 to the end of array
```

```
print_r($numbers);

array_pop($numbers); // Removes last element (60)
print_r($numbers);

sort($numbers); // Sorts array in ascending order
print_r($numbers);
?>
```

---

6. फॉर्म्स के साथ काम करना

6.1 HTML फॉर्म्स से डाटा प्राप्त करना

PHP dynamic websites में form data collect करने के लिए GET और POST methods का उपयोग किया जाता है।

सिंपल HTML फॉर्म:

```
<form action="process.php" method="post">
  Name: <input type="text" name="name">
  Email: <input type="email" name="email">
  <input type="submit" value="Submit">
</form>
```

यह form data को process.php पेज पर भेजेगा।

6.2 GET और POST मेथड का अंतर

GET Method	POST Method
URL में data show होता है	Data hidden रहता है
Small data transfer के लिए अच्छा	Large data भेजने के लिए उपयोगी
Security कम होती है	ज्यादा secure होता है
Bookmark और cache किया जा सकता है	नहीं किया जा सकता

GET Method Example:

```
<?php
 if (isset($_GET['name'])) {
   echo "Hello, " . $_GET['name'];
 }
?>
```

URL: `http://example.com/process.php?name=Rahul`

POST Method Example:

```
<?php
 if ($_SERVER["REQUEST_METHOD"] == "POST") {
  echo "Hello, " . $_POST['name'];
 }
?>
```

POST method data को URL में नहीं दिखाता, इसलिए यह secure होता है।

6.3 फॉर्म वेलिडेशन (Form Validation)

User द्वारा enter किए गए data को validate करना ज़रूरी होता है।

Basic Form Validation Example:

```
<?php
 if ($_SERVER["REQUEST_METHOD"] == "POST") {
  $name = trim($_POST['name']);
  if (empty($name)) {
   echo "Name is required";
  } else {
   echo "Hello, $name";
  }
```

```
}
?>
```

इसमें `trim()` function spaces हटाने के लिए और `empty()` function खाली check करने के लिए use होता है।

6.4 PHP से फॉर्म हैंडलिंग

PHP का उपयोग करके form data को database में store किया जा सकता है।

Form Handling with MySQL:

```
<?php
  $conn = new mysqli("localhost", "root", "", "mydatabase");
  if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
  }

  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST['name'];
    $email = $_POST['email'];

    $sql = "INSERT INTO users (name, email) VALUES ('$name', '$email')";
    if ($conn->query($sql) === TRUE) {
```

```
    echo "New record created successfully";
  } else {
    echo "Error: " . $sql . "<br>" . $conn->error;
  }
}
$conn->close();
?>
```

यह code user data को MySQL database में insert करेगा।

---

7. PHP सुपरग्लोबल्स

PHP में सुपरग्लोबल्स predefined variables होते हैं, जो script के किसी भी भाग में access किए जा सकते हैं।

7.1 $_GET, $_POST, और $_REQUEST

ये तीनों arrays यूज़र द्वारा भेजे गए data को store करने के लिए उपयोग किए जाते हैं।

$_GET Example:

```
<?php
```

```
  echo "Hello, " . $_GET['name'];
?>
```

URL: `http://example.com/index.php?name=Rahul`

$_POST Example:

```
<?php
  echo "Hello, " . $_POST['name'];
?>
```

POST method data को URL में नहीं दिखाता, इसलिए secure होता है।

$_REQUEST Example:

```
<?php
  echo "Hello, " . $_REQUEST['name'];
?>
```

$_REQUEST, GET और POST दोनों से data fetch कर सकता है।

7.2 $_SESSION और $_COOKIE

Sessions:

Sessions का उपयोग user data को temporary रूप से store करने के लिए किया जाता है।

```
<?php
  session_start();
  $_SESSION['username'] = "Rahul";
  echo "Session set for " . $_SESSION['username'];
?>
```

Cookies:

Cookies client-side data store करने के लिए उपयोग होते हैं।

```
<?php
  setcookie("user", "Rahul", time() + (86400 * 30), "/");
  echo "Cookie set for user: " . $_COOKIE['user'];
?>
```

7.3 $_FILES (File Uploading)

PHP में file upload करने के लिए $_FILES सुपरग्लोबल का उपयोग किया जाता है।

File Upload Example:

```
<form action="upload.php" method="post" enctype="multipart/form-data">
  Select file: <input type="file" name="fileToUpload">
  <input type="submit" value="Upload">
</form>
<?php
  if ($_FILES["fileToUpload"]["error"] == 0) {
    move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], "uploads/" .
$_FILES["fileToUpload"]["name"]);
    echo "File uploaded successfully!";
  } else {
    echo "Error in uploading file.";
  }
?>
```

---

8. फाइल हैंडलिंग

PHP में file handling का उपयोग files को create, read, write, delete, और modify करने के लिए किया जाता है।

8.1 File Open, Read, Write

PHP में `fopen()`, `fread()`, `fwrite()`, और `fclose()` functions का उपयोग करके files को handle किया जाता है।

File Create & Write Example:

```
<?php
  $file = fopen("sample.txt", "w"); // Open file in write mode
  fwrite($file, "Hello, this is a sample text file.");
  fclose($file);
?>
```

File Read Example:

```
<?php
  $file = fopen("sample.txt", "r"); // Open file in read mode
  echo fread($file, filesize("sample.txt"));
  fclose($file);
?>
```

8.2 File Uploading and Downloading

PHP के `$_FILES` सुपरग्लोबल का उपयोग file upload के लिए किया जाता है।

File Upload Example:

```
<form action="upload.php" method="post" enctype="multipart/form-data">
  Select file: <input type="file" name="fileToUpload">
  <input type="submit" value="Upload">
</form>
<?php
  if ($_FILES["fileToUpload"]["error"] == 0) {
    move_uploaded_file($_FILES["fileToUpload"]["tmp_name"], "uploads/" .
$_FILES["fileToUpload"]["name"]);
    echo "File uploaded successfully!";
  } else {
```

```
  echo "Error in uploading file.";
 }
?>
```

File Download Example:

```
<?php
 $file = "uploads/sample.pdf";
 if (file_exists($file)) {
  header('Content-Description: File Transfer');
  header('Content-Type: application/octet-stream');
  header('Content-Disposition: attachment; filename="' . basename($file) . '"');
  readfile($file);
  exit;
 } else {
  echo "File does not exist.";
 }
?>
```

8.3 File Deletion

PHP में `unlink()` function का उपयोग किसी file को delete करने के लिए किया जाता है।

File Delete Example:

```
<?php
 $file = "sample.txt";
 if (file_exists($file)) {
  unlink($file);
  echo "File deleted successfully!";
 } else {
  echo "File not found.";
 }
?>
```

---

9. PHP और डेटाबेस (MySQL)

9.1 MySQL Database का Introduction

MySQL एक open-source relational database management system (RDBMS) है, जिसका उपयोग data store और manage करने के लिए किया जाता है। PHP और MySQL को एक साथ use करके dynamic websites बनाई जा सकती हैं।

9.2 PHP से Database Connection

PHP में MySQL database से connect करने के लिए `mysqli_connect()` function का उपयोग किया जाता है।

Database Connection Example:

```
<?php
  $servername = "localhost";
  $username = "root";
  $password = "";
  $dbname = "mydatabase";

  // Create connection
  $conn = new mysqli($servername, $username, $password, $dbname);

  // Check connection
  if ($conn->connect_error) {
```

```
    die("Connection failed: " . $conn->connect_error);
  }
  echo "Connected successfully";
?>
```

9.3 CRUD Operations (Create, Read, Update, Delete)

Database में data insert, retrieve, update और delete करने के लिए निम्नलिखित queries का उपयोग किया जाता है।

Create (Insert Data):

```
<?php
  $sql = "INSERT INTO users (name, email) VALUES ('Rahul',
'rahul@example.com')";
  if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
  } else {
    echo "Error: " . $sql . "<br>" . $conn->error;
  }
?>
```

Read (Fetch Data):

```
<?php
  $sql = "SELECT id, name, email FROM users";
  $result = $conn->query($sql);
  if ($result->num_rows > 0) {
    while($row = $result->fetch_assoc()) {
      echo "ID: " . $row["id"]. " - Name: " . $row["name"]. " - Email: " . $row["email"].
"<br>";
    }
  } else {
    echo "No records found";
  }
```

```
?>
```

Update (Modify Data):

```
<?php
  $sql = "UPDATE users SET email='newemail@example.com' WHERE
name='Rahul'";
  if ($conn->query($sql) === TRUE) {
    echo "Record updated successfully";
  } else {
    echo "Error updating record: " . $conn->error;
  }
?>
```

Delete (Remove Data):

```
<?php
  $sql = "DELETE FROM users WHERE name='Rahul'";
  if ($conn->query($sql) === TRUE) {
    echo "Record deleted successfully";
  } else {
    echo "Error deleting record: " . $conn->error;
  }
?>
```

9.4 Prepared Statements और Security

SQL Injection से बचने के लिए prepared statements का उपयोग किया जाता है।

Prepared Statement Example:

```
<?php
  $stmt = $conn->prepare("INSERT INTO users (name, email) VALUES (?, ?)");
  $stmt->bind_param("ss", $name, $email);

  // Set parameters and execute
  $name = "Amit";
```

```
  $email = "amit@example.com";
  $stmt->execute();

  echo "New record inserted successfully";
?>
```

Prepared statements SQL queries को dynamic और secure बनाते हैं।

---

10. PHP सेशन्स और कुकीज़

10.1 Cookies का उपयोग और Security

Cookies client-side data store करने के लिए उपयोग किए जाते हैं और user-specific जानकारी save करने में मदद करते हैं।

Set a Cookie:

```
<?php
  setcookie("user", "Rahul", time() + (86400 * 30), "/");
  echo "Cookie set for user: " . $_COOKIE['user'];
?>
```

Retrieve a Cookie:

```
<?php
  if(isset($_COOKIE["user"])) {
    echo "Welcome back, " . $_COOKIE["user"];
  } else {
    echo "Cookie not found.";
```

```
 }
?>
```

Delete a Cookie:

```
<?php
 setcookie("user", "", time() - 3600, "/");
 echo "Cookie deleted.";
?>
```

10.2 Sessions का Concept और Implementation

Sessions server-side data store करने के लिए उपयोग किए जाते हैं और sensitive data को secure रखने में मदद करते हैं।

Start a Session:

```
<?php
 session_start();
 $_SESSION["username"] = "Rahul";
 echo "Session set for: " . $_SESSION["username"];
?>
```

Retrieve a Session:

```
<?php
 session_start();
 echo "Welcome, " . $_SESSION["username"];
?>
```

Destroy a Session:

```
<?php
 session_start();
 session_unset();
 session_destroy();
 echo "Session destroyed.";
```

```
?>
```

10.3 Login System using Sessions

Sessions का उपयोग user authentication system में किया जाता है।

Login Page (login.php)

```
<?php
  session_start();
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = $_POST['password'];

    if ($username == "admin" && $password == "12345") {
      $_SESSION["username"] = $username;
      header("Location: dashboard.php");
    } else {
      echo "Invalid credentials!";
    }
  }
?>

<form method="post">
  Username: <input type="text" name="username">
  Password: <input type="password" name="password">
  <input type="submit" value="Login">
</form>
```

Dashboard Page (dashboard.php)

```
<?php
  session_start();
  if (!isset($_SESSION["username"])) {
    header("Location: login.php");
    exit();
  }
```

```
  echo "Welcome, " . $_SESSION["username"];
?>
<a href="logout.php">Logout</a>
```

Logout Page (logout.php)

```
<?php
  session_start();
  session_unset();
  session_destroy();
  header("Location: login.php");
?>
```

---

11. PHP में ऑब्जेक्ट ओरिएंटेड प्रोग्रामिंग (OOP)

11.1 Classes and Objects

PHP में OOP का मुख्य आधार Classes और Objects होते हैं।

Class और Object का Example:

```
<?php
  class Car {
    public $brand;
    public function setBrand($name) {
      $this->brand = $name;
    }
    public function getBrand() {
      return $this->brand;
    }
  }
```

```
  $myCar = new Car();
  $myCar->setBrand("BMW");
  echo $myCar->getBrand(); // Output: BMW
?>
```

11.2 Constructor & Destructor

Constructor function class का object create होने पर automatically execute होता है, और Destructor function object destroy होने पर run होता है।

Constructor & Destructor Example:

```
<?php
  class Person {
    public $name;

    public function __construct($name) {
      $this->name = $name;
      echo "Object created for $name";
    }

    public function __destruct() {
      echo "Object destroyed for $this->name";
    }
  }

  $p1 = new Person("Rahul");
?>
```

11.3 Inheritance

Inheritance का उपयोग एक class की properties और methods को दूसरी class में reuse करने के लिए किया जाता है।

Inheritance Example:

```
<?php
  class Animal {
    public function sound() {
      echo "Animals make sound";
    }
  }

  class Dog extends Animal {
    public function bark() {
      echo "Dog barks";
    }
  }

  $d = new Dog();
  $d->sound(); // Output: Animals make sound
  $d->bark(); // Output: Dog barks
?>
```

11.4 Polymorphism

Polymorphism का मतलब है कि एक ही method अलग-अलग classes में अलग behavior दिखा सकता है।

Polymorphism Example:

```
<?php
  class Shape {
    public function draw() {
      echo "Drawing a shape";
    }
  }

  class Circle extends Shape {
    public function draw() {
      echo "Drawing a circle";
```

```
  }
 }

 $s = new Circle();
 $s->draw(); // Output: Drawing a circle
?>
```

11.5 Encapsulation

Encapsulation का उपयोग data hiding और security के लिए किया जाता है।

Encapsulation Example:

```
<?php
 class BankAccount {
  private $balance = 1000;

  public function getBalance() {
   return $this->balance;
  }

  public function deposit($amount) {
   $this->balance += $amount;
  }
 }

 $account = new BankAccount();
 $account->deposit(500);
 echo $account->getBalance(); // Output: 1500
?>
```

---

12. एरर और एक्सेप्शन हैंडलिंग

PHP में errors और exceptions को handle करने के लिए विभिन्न techniques उपलब्ध हैं।

12.1 PHP Errors और Warnings

PHP में तीन मुख्य प्रकार के errors होते हैं:

1. **Notice: छोटे errors, जो script को execute होने से नहीं रोकते।**
2. **Warning: गंभीर errors, लेकिन script execute होती रहती है।**
3. **Fatal Error: गंभीर errors, जिससे script execution रुक जाती है।**

Error Example:

```
<?php
  echo $undefined_variable; // Notice: Undefined variable
  include("non_existing_file.php"); // Warning: File not found
  $result = 10 / 0; // Fatal Error: Division by zero
?>
```

12.2 Try-Catch Exception Handling

Exceptions error handling का एक बेहतर तरीका है।

Try-Catch Example:

```
<?php
  function divide($num1, $num2) {
    if ($num2 == 0) {
      throw new Exception("Cannot divide by zero!");
    }
    return $num1 / $num2;
  }
```

```
  try {
    echo divide(10, 0);
  } catch (Exception $e) {
    echo "Error: " . $e->getMessage();
  }
?>
```

12.3 Custom Exception Handling

हम custom exceptions भी बना सकते हैं।

Custom Exception Example:

```
<?php
  class CustomException extends Exception {}

  function checkNumber($num) {
    if ($num < 0) {
      throw new CustomException("Negative numbers not allowed!");
    }
    return "Number is valid";
  }

  try {
    echo checkNumber(-5);
  } catch (CustomException $e) {
    echo "Error: " . $e->getMessage();
  }
?>
```

---

13. वास्तविक जीवन के PHP प्रोजेक्ट्स

इस सेक्शन में कुछ रियल-वर्ल्ड PHP प्रोजेक्ट्स दिए गए हैं, जो आपको PHP के प्रैक्टिकल इस्तेमाल को समझने में मदद करेंगे।

13.1 User Authentication System (Login/Signup)

यह प्रोजेक्ट एक बेसिक लॉगिन और साइनअप सिस्टम बनाता है।

Database Structure:

```
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    username VARCHAR(50) NOT NULL,
    password VARCHAR(255) NOT NULL
);
```

Signup Page (signup.php)

```
<?php
  $conn = new mysqli("localhost", "root", "", "mydatabase");
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = password_hash($_POST['password'], PASSWORD_BCRYPT);
    $sql = "INSERT INTO users (username, password) VALUES ('$username',
'$password')";
    $conn->query($sql);
    echo "Signup Successful!";
  }
?>
```

Login Page (login.php)

```
<?php
  session_start();
  $conn = new mysqli("localhost", "root", "", "mydatabase");
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $username = $_POST['username'];
    $password = $_POST['password'];
```

```
    $result = $conn->query("SELECT * FROM users WHERE
username='$username'");
    $user = $result->fetch_assoc();

    if (password_verify($password, $user['password'])) {
      $_SESSION['username'] = $username;
      header("Location: dashboard.php");
    } else {
      echo "Invalid credentials!";
    }
  }
?>
```

13.2 Blog Website (Post, Comment System)

इस प्रोजेक्ट में users को ब्लॉग पोस्ट बनाने और उन पर कमेंट करने की सुविधा मिलती है।

Database Structure:

```
CREATE TABLE posts (
    id INT AUTO_INCREMENT PRIMARY KEY,
    title VARCHAR(255),
    content TEXT,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Post Creation (add_post.php)

```
<?php
  $conn = new mysqli("localhost", "root", "", "blog");
  if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $title = $_POST['title'];
    $content = $_POST['content'];
    $sql = "INSERT INTO posts (title, content) VALUES ('$title', '$content')";
    $conn->query($sql);
    echo "Post added successfully!";
  }
```

```
?>
```

13.3 E-commerce Cart System (Add to Cart, Checkout)

इस प्रोजेक्ट में एक सिंपल ई-कॉमर्स सिस्टम होगा, जिसमें यूज़र items को cart में add, remove, और checkout कर सकते हैं।

Cart Page (cart.php)

```
<?php
 session_start();
 if (!isset($_SESSION['cart'])) {
  $_SESSION['cart'] = [];
 }

 if (isset($_POST['add_to_cart'])) {
  $_SESSION['cart'][] = $_POST['item_name'];
 }
?>

<form method="post">
 <input type="hidden" name="item_name" value="Laptop">
 <input type="submit" name="add_to_cart" value="Add to Cart">
</form>

<h3>Your Cart:</h3>
<ul>
 <?php foreach ($_SESSION['cart'] as $item) {
  echo "<li>$item</li>";
 } ?>
</ul>
```

---

14. PHP फ्रेमवर्क्स का परिचय

PHP frameworks development को तेज और structured बनाते हैं। कुछ लोकप्रिय frameworks निम्नलिखित हैं:

14.1 Laravel Framework

Laravel एक modern, expressive, और सुरक्षित PHP framework है। यह MVC (Model-View-Controller) pattern पर आधारित होता है।

Laravel Installation:

```
composer create-project --prefer-dist laravel/laravel myLaravelApp
```

Laravel Routes Example:

```
Route::get('/welcome', function () {
    return view('welcome');
});
```

14.2 CodeIgniter Framework

CodeIgniter एक lightweight और high-performance PHP framework है।

CodeIgniter Installation:

```
composer create-project codeigniter4/appstarter myCodeIgniterApp
```

CodeIgniter Controller Example:

```
class Home extends CI_Controller {
    public function index() {
        echo "Welcome to CodeIgniter!";
    }
}
```

14.3 Symfony Framework

Symfony एक robust और scalable PHP framework है, जो enterprise applications के लिए उपयोग किया जाता है।

Symfony Installation:

composer create-project symfony/skeleton mySymfonyApp

Symfony Controller Example:

```
use Symfony\Component\HttpFoundation\Response;
class DefaultController {
    public function index() {
        return new Response('Hello Symfony!');
    }
}
```

---

15. PHP में सुरक्षा के बेहतरीन उपाय

PHP में security बहुत महत्वपूर्ण होती है ताकि website को hacking और data theft से बचाया जा सके।

15.1 SQL Injection से बचाव

SQL Injection से बचने के लिए Prepared Statements और Parameterized Queries का उपयोग करें।

Unsafe Query (SQL Injection Possible):

```php
<?php
  $username = $_GET['username'];
  $sql = "SELECT * FROM users WHERE username = '$username'";
  $result = $conn->query($sql);
?>
```

यह query unsafe है क्योंकि कोई malicious user इसमें SQL injection कर सकता है।

Secure Query (Using Prepared Statements):

```php
<?php
  $stmt = $conn->prepare("SELECT * FROM users WHERE username = ?");
  $stmt->bind_param("s", $username);
  $stmt->execute();
?>
```

15.2 Cross-Site Scripting (XSS) से सुरक्षा

XSS से बचने के लिए htmlspecialchars() function का उपयोग करें।

Unsafe Code (XSS Vulnerability):

```php
<?php
  echo "Welcome " . $_GET['name'];
?>
```

अगर user `<script>alert('Hacked');</script>` input देता है, तो script execute हो जाएगी।

Secure Code (Using htmlspecialchars()):

```php
<?php
  echo "Welcome " . htmlspecialchars($_GET['name']);
?>
```

यह script को plain text बना देगा और execute नहीं होने देगा।

15.3 पासवर्ड हैशिंग और एन्क्रिप्शन

Plain text passwords को store करना unsafe होता है, इसलिए password_hash() function का उपयोग करें।

Password Hashing Example:

```
<?php
  $password = "mypassword";
  $hashedPassword = password_hash($password, PASSWORD_BCRYPT);
  echo $hashedPassword;
?>
```

Verify Password:

```
<?php
  if (password_verify("mypassword", $hashedPassword)) {
    echo "Password is valid!";
  } else {
    echo "Invalid password!";
  }
?>
```

15.4 सुरक्षित फाइल अपलोडिंग

Users द्वारा अपलोड की गई फाइल्स को validate करना ज़रूरी है ताकि malicious scripts अपलोड न हों।

Unsafe File Upload (Risk of Malicious Scripts):

```
<?php
  move_uploaded_file($_FILES["file"]["tmp_name"], "uploads/" .
$_FILES["file"]["name"]);
```

```
?>
```

Secure File Upload (Validating File Type & Size):

```
<?php
  $allowedTypes = ['image/jpeg', 'image/png'];
  if (in_array($_FILES["file"]["type"], $allowedTypes) && $_FILES["file"]["size"] < 500000) {
    move_uploaded_file($_FILES["file"]["tmp_name"], "uploads/" . $_FILES["file"]["name"]);
    echo "File uploaded successfully!";
  } else {
    echo "Invalid file type or size exceeded!";
  }
?>
```

---

16. API और वेब सर्विसेस

16.1 REST API क्या है?

REST (Representational State Transfer) API एक architecture style है जिसका उपयोग Web Services बनाने के लिए किया जाता है। यह HTTP प्रोटोकॉल का उपयोग करके client-server communication को आसान बनाता है।

REST API के प्रमुख सिद्धांत:

- **Stateless: Server को client की पिछली requests याद नहीं रहतीं।**
- **Client-Server Architecture: Client और Server अलग-अलग काम करते हैं।**
- **JSON/XML Data Format: REST API ज्यादातर JSON data भेजता और प्राप्त करता है।**

- **HTTP Methods:**
    - **GET (डेटा प्राप्त करने के लिए)**
    - **POST (नया डेटा जोड़ने के लिए)**
    - **PUT (डेटा अपडेट करने के लिए)**
    - **DELETE (डेटा हटाने के लिए)**

16.2 PHP से API कॉल करना

PHP का उपयोग करके REST API से डेटा प्राप्त किया जा सकता है।

GET Request Example:

```
<?php
  $url = "https://jsonplaceholder.typicode.com/posts/1";
  $response = file_get_contents($url);
  $data = json_decode($response, true);
  echo "Title: " . $data['title'];
?>
```

POST Request Example (cURL Method):

```
<?php
  $url = "https://jsonplaceholder.typicode.com/posts";
  $data = array("title" => "New Post", "body" => "This is a test post", "userId" =>
1);

  $ch = curl_init($url);
  curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
  curl_setopt($ch, CURLOPT_POST, true);
  curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($data));
  curl_setopt($ch, CURLOPT_HTTPHEADER, array('Content-Type:
application/json'));
```

```
  $response = curl_exec($ch);
  curl_close($ch);

  echo $response;
?>
```

16.3 JSON Handling

PHP में JSON डेटा को encode और decode करने के लिए `json_encode()` और `json_decode()` functions का उपयोग किया जाता है।

JSON Encode Example:

```
<?php
  $arr = array("name" => "Rahul", "age" => 25, "city" => "Delhi");
  $jsonData = json_encode($arr);
  echo $jsonData; // Output: {"name":"Rahul","age":25,"city":"Delhi"}
?>
```

JSON Decode Example:

```
<?php
  $json = '{"name":"Rahul","age":25,"city":"Delhi"}';
  $data = json_decode($json, true);
  echo "Name: " . $data['name'];
?>
```

---

17. PHP डिप्लॉयमेंट और ऑप्टिमाइज़ेशन

17.1 Hosting & Domain Setup

PHP वेबसाइट को live करने के लिए एक domain name और hosting server की आवश्यकता होती है।

Steps to Host a PHP Website:

1. **Domain Name Register करें** (जैसे कि GoDaddy, Namecheap से)।
2. **Web Hosting खरीदें** (Shared, VPS या Cloud Hosting चुनें)।
3. **cPanel या FTP के माध्यम से Files Upload करें।**
4. **Database (MySQL) को Import करें।**
5. **Domain को Hosting से Connect करें।**
6. **Website को Browser में Test करें।**

17.2 PHP Website को Live कैसे करें?

PHP वेबसाइट को deploy करने के लिए हम **Apache Server, Nginx, या Cloud Platforms** (जैसे AWS, DigitalOcean) का उपयोग कर सकते हैं।

Deploying via cPanel:

1. **File Manager में जाएं और public_html फोल्डर में PHP Files Upload करें।**
2. **MySQL Database Create करें और Database Credentials को Update करें।**
3. **Website को Browser में खोलकर Test करें।**

Deploying via FTP (File Transfer Protocol):

1. **FTP Client (जैसे FileZilla) डाउनलोड करें।**
2. **FTP Credentials से Server Connect करें।**

3. **Website की सभी Files को Server पर Upload करें।**

17.3 Performance Optimization Techniques

PHP वेबसाइट की स्पीड और परफॉर्मेंस बढ़ाने के लिए निम्नलिखित techniques का उपयोग किया जाता है:

1. Caching Enable करें

- **Opcode Cache: APCu या OPcache का उपयोग करें।**
- **Page Cache: WordPress जैसी websites में plugins जैसे कि WP Super Cache का उपयोग करें।**

2. Database Optimization करें

- **Indexing का उपयोग करें।**
- **Unused data को remove करें।**
- **Query Execution Time को Monitor करें।**

3. Image & Asset Optimization करें

- **Images को Compress करें (जैसे TinyPNG का उपयोग करें)।**
- **CSS & JavaScript Minify करें।**
- **CDN (Content Delivery Network) का उपयोग करें।**

4. PHP Code Optimization करें

- **Unnecessary loops और complex queries को optimize करें।**
- **Prepared Statements का उपयोग करें।**
- **Autoloading classes को implement करें।**

5. Security & Speed Boosters

- **SSL Certificate Install करें।**
- **Gzip Compression Enable करें।**
- **Unused PHP Modules को Disable करें।**

18. PHP Unit Testing (Code Testing)

18.1 PHPUnit Introduction

PHPUnit एक popular testing framework है जो PHP applications के लिए automated testing प्रदान करता है।

PHPUnit Installation:

```
composer require --dev phpunit/phpunit
```

Basic Test Case:

```
<?php
use PHPUnit\Framework\TestCase;

class SampleTest extends TestCase {
    public function testAddition() {
        $this->assertEquals(4, 2 + 2);
    }
}
?>
```

---

19. WebSockets और Real-Time Applications

19.1 WebSockets क्या होते हैं?

WebSockets एक protocol है जो real-time bidirectional communication की अनुमति देता है।

WebSocket Server (Ratchet Library)

```
composer require cboden/ratchet
<?php
use Ratchet\MessageComponentInterface;
use Ratchet\ConnectionInterface;

class ChatServer implements MessageComponentInterface {
    public function onOpen(ConnectionInterface $conn) {
        echo "New connection established!";
    }
    public function onMessage(ConnectionInterface $from, $msg) {
        echo "Message received: $msg";
    }
    public function onClose(ConnectionInterface $conn) {
        echo "Connection closed";
    }
    public function onError(ConnectionInterface $conn, \Exception $e) {
        echo "Error: " . $e->getMessage();
    }
}
?>
```

---

20. PHP & Cloud Integration

20.1 AWS S3 में PHP से फाइल अपलोड करना

```
composer require aws/aws-sdk-php
<?php
use Aws\S3\S3Client;

$s3 = new S3Client([
    'region'  => 'us-west-2',
    'version' => 'latest',
    'credentials' => [
        'key'    => 'YOUR_AWS_ACCESS_KEY',
        'secret' => 'YOUR_AWS_SECRET_KEY',
    ],
]);

$result = $s3->putObject([
    'Bucket' => 'your-bucket-name',
    'Key'    => 'uploads/sample.txt',
    'Body'   => 'Hello, World!'
]);
?>
```

---

21. Advanced PHP Security

21.1 CSRF (Cross-Site Request Forgery) Prevention

```
<?php
session_start();
$_SESSION['csrf_token'] = bin2hex(random_bytes(32));
?>
<form method="post">
    <input type="hidden" name="csrf_token" value="<?php echo
$_SESSION['csrf_token']; ?>">
```

```
    <input type="submit" value="Submit">
</form>
```

---

22. PHP & Machine Learning

22.1 PHP-ML Library का उपयोग

```
composer require php-ai/php-ml
<?php
use Phpml\Classification\KNearestNeighbors;

$samples = [[1, 2], [3, 4], [5, 6]];
$labels = ['A', 'B', 'C'];

$classifier = new KNearestNeighbors();
$classifier->train($samples, $labels);

echo $classifier->predict([3, 4]); // Output: B
?>
```

---

23. PHP में Multi-threading और Parallel Processing

23.1 Multi-threading क्या है?

Multi-threading का उपयोग एक साथ multiple tasks को execute करने के लिए किया जाता है। PHP में multi-threading को pthreads extension के माध्यम से implement किया जा सकता है।

Multi-threading Installation (pthreads)

pecl install pthreads

Basic Multi-threading Example:

```
<?php
class MyThread extends Thread {
    public function run() {
        echo "Thread ID: " . $this->getThreadId() . "\n";
    }
}

$thread1 = new MyThread();
$thread2 = new MyThread();

$thread1->start();
$thread2->start();

$thread1->join();
$thread2->join();
?>
```

---

24. PHP & Microservices Architecture

24.1 Microservices क्या हैं?

Microservices Architecture एक software development approach है, जिसमें application को छोटे-छोटे independent services में divide किया जाता है।

Microservices के फायदे:

- **Scalable और Maintainable**
- Independent Deployment
- Fault Isolation

PHP में Microservice API बनाना (Slim Framework)

```
composer require slim/slim "^4.0"
<?php
require 'vendor/autoload.php';
use Slim\Factory\AppFactory;

$app = AppFactory::create();

$app->get('/hello/{name}', function ($request, $response, $args) {
    $response->getBody()->write("Hello, " . $args['name']);
    return $response;
});

$app->run();
?>
```

---

25. PHP में GraphQL Integration

25.1 GraphQL क्या है?

GraphQL एक API query language है, जो REST API का एक बेहतर विकल्प प्रदान करता है। यह clients को आवश्यक data को precisely fetch करने की सुविधा देता है।

GraphQL के फायदे:

- **Over-fetching और Under-fetching से बचाव।**
- **एक ही request में multiple resources को fetch करना।**
- **API versioning की आवश्यकता नहीं होती।**

PHP में GraphQL Server Setup (Webonyx Library)

composer require webonyx/graphql-php

GraphQL Schema Example:

```
<?php
use GraphQL\GraphQL;
use GraphQL\Type\Schema;
use GraphQL\Type\Definition\ObjectType;
use GraphQL\Type\Definition\Type;

$QueryType = new ObjectType([
    'name' => 'Query',
    'fields' => [
        'hello' => [
            'type' => Type::string(),
            'resolve' => function() {
                return 'Hello, GraphQL!';
            }
```

```
        ]
    ]
]);

$schema = new Schema([
    'query' => $QueryType
]);

$query = '{ hello }';
$result = GraphQL::executeQuery($schema, $query);
echo json_encode($result->toArray());
?>
```
