Program	Master of Computer Applications (MCA)	**Semester - 3**
Type of Course	Elective Courses	
Prerequisite	Basic Knowledge of Web Designing	
Course Objective	PHP is a powerful tool for making dynamic and interactive database driven web pages. Students will acquire the knowledge of the core concepts of web programming and server-side scripting using PHP. After mastering this course they may work as self employed web page developer.	
Effective From A.Y.	2023-24	

Teaching Scheme (Contact Hours)				**Examination Scheme**				
Lecture	**Tutorial**	**Lab**	**Credit**	**Theory Marks**		**Practical Marks**		**Total Marks**
				T	**T**	**P**	**P**	
4	0	4	6	40	30	20	10	100

***SEE** - Semester End Examination, **T** - Internal Theory, **P** - Internal Practical*

Course Content **T** - Teaching Hours | **W** - Weightage

Sr.	**Topics**	**T**	**W**
1	**Introduction to Server Side Scripting with PHP** Introduction to PHP, client side scripting vs server side scripting, applications of PHP, Structure of PHP page, PHP Syntax: variables decision and looping with examples, PHP and HTML, array, types of array, array functions, string functions, user defined functions, recursion, include, include_once, require, require_once functions.	**13**	**20**
2	**Form Processing and Session management** Form attributes, receiving data submitted using get method, receiving data submitted using post method, $_REQUEST, file handling, uploading a file, query string, cleaning the received data, checking correctness of data using Regular Expression, Session management using session and cookie.	**11**	**20**
3	**Database programming using PHP** Basic MySQL commands, PHP functions for database connectivity, Implementation of CRUD operations using PHP, Prepared Statement and stored procedure execution in PHP.	**15**	**20**
4	**API development and Asynchronous programming:** Introduction to API, API development for CRUD operations, Introduction to asynchronous programming, AJAX.	**8**	**20**
5	**Introduction to Laravel** Introduction to Laravel, Installation and Configuration, Routing, Views, Migrations, Models, Controllers, CRUD operations using MVC architecture.	**13**	**20**
	Total	**60**	**100**

Suggested Distribution Of Theory Marks Using Bloom's Taxonomy

Level	Remembrance	Understanding	Application	Analyze	Evaluate	Create
Weightage	10	30	60	0	0	0

NOTE : This specification table shall be treated as a general guideline for the students and the teachers. The actual distribution of marks in the question paper may vary slightly from above table.

Course Outcomes

At the end of this course, students will be able to:	
CO1	**practice** basic concepts of PHP.
CO2	**use** form processing and session management concepts in PHP web page.
CO3	**demonstrate** database operations in MySQL using PHP.
CO4	**implement** asynchronous programming in web page.
CO5	**perform** basic operations using Laravel.

CO PO Mapping

CO	CO - 1	CO - 2	CO - 3	CO - 4	CO - 5
PO - 1					
PO - 2					
PO - 3					
PO - 4					
PO - 5					
PO - 6					
PO - 7					

Reference Books

1.	**Developing Web Application** By Ralph Moseley, M.T. Savaliya | Wiley India | Latest Edition
2.	**Web Technologies, Black Book** Dreamtech Press
3.	**Head First PHP & MySQL** By Lynn Beighley, Michael Morrison | o'reilly
4.	**Developing Web Applications in PHP and AJAX** By Harwani | McGrawHill
5.	**Learning PHP, MySQL, JavaScript, CSS & HTML5** By Robin Nixon | O'Reilly
6.	**Php: A Beginner's Guide** By Vaswani Vikram | McGraw Hill Education India
7.	**Building Dynamic Web Experiences with PHP** By Dr. Surabhi Shanker | BPB Publications
8.	**Ultimate Laravel for Modern Web Development** By Drishti Jain | Orange Education Pvt Ltd

List of Practical

1.	**Structuring Content with Table, Lists, and Div Tags** 1. Create a weekly class timetable using HTML. Use the <table> tag to show the days and time slots clearly. Add some basic CSS to make the timetable look neat and easy to read. [A] 2. Create the HTML table given below that demonstrates the use of bold, italic, and underline text, along with rowspan and colspan. Apply basic CSS styling to enhance the table's appearance. [A] 3. Create an HTML document with an ordered list <ol>, unordered list <ul> and description list <dl> .[B]

	4. Create an HTML document using the <div> tag to organize content into sections. Apply basic CSS to style the sections. [C]
2.	**Design and Implementation of a Static Website Interface Using HTML and CSS** 1. Design an HTML-based login interface that includes input fields for both username and password. [A] 2. Create an HTML student registration page that includes following input fields such as name, email, phone number, gender, semester, branch, address and hobbies. [A] 3. Create a basic layout for a static webpage using HTML5 tags including, <section>, <main>, <header>, <footer>, <aside>,and <nav> . Apply CSS Flexbox for the layout. [B] 4. Implement the static webpage from Lab 2(3) with actual content, including a professional logo, images, and detailed information. Apply the necessary CSS to create an appealing webpage. [C]
3.	**Demonstration of Basics of PHP Programming** 1. WAP to display "Hello World". [A] 2. WAP to display a message using variable. [A] 3. Create a PHP script - Declare a variable $value and assign it an integer value (e.g., $value = 10;). - Use the gettype() function to display the type of $value. - Use settype($value, 'string') to change the type of $value to a string. - Use gettype() again to display the new type of $value. - Use var_dump($value) to display the value and type of $value after the conversion. [A] 4. WAP to demonstrate the use of const and define function. [B] 5. Write a PHP program to demonstrate the behavior of variables in PHP as a loosely typed language. [B] 6. Write a PHP program to demonstrate variable scopes: local, global, and static. [C] 7. WAP to swap values of two variables with the help of 3rd variable. [C] 8. WAP to swap values of two variables without using 3rd variable. [C]
4.	**Implementation of Decision-Making Statements (Part-I)** 1. WAP to check whether the given number is odd or even. [A] 2. WAP to check a person is eligible to vote. [A] 3. WAP to find greatest number from 2 numbers. [A] 4. WAP to find greatest number from 3 numbers. [A] 5. WAP that check whether a letter is a vowel or consonants. [A] 6. WAP to check whether the given number is positive, negative or zero. [A] 7. WAP to print name of month based on given number of month (i.e. 1 -> January, 2 -> February...). [B] 8. WAP to print class of result based on percentage. (i.e. less than 40% -> Fail, 40% to 50% -> Pass Class, 50% to 60% -> Second Class, 60% to 70% -> First Class, above 70% -> Distinction). [B] 9. Write a simple calculator program in PHP. [C] 10.Write a program to calculate Electricity bill in PHP. [C]
5.	**Implementation of Decision-Making Statements (Part-II)** 1. WAP to convert temperature from Fahrenheit to Celsius. [A] 2. WAP to find a diameter from given area of circle. [A] 3. WAP to make a Simple Calculator using switch case. [A] 4. WAP to print class of result based on percentage using switch case (i.e. less than 40% -> Fail, 40% to 50% -> Pass Class, 50% to 60% -> Second Class, 60% to 70% -> First Class, above 70% -> Distinction). [B] 5. WAP to check to given year is a leap or not. [B] 6. WAP that reads a number in meters, converts it to feet, and displays the result. [B] 7. Input an integer number and check the last digit of number is even or odd. [C] 8. WAP to check whether the three slide of triangle is isosceles, equilateral, scalene or right-angled triangle. [C]
6.	**Implementation of Various Loops**

1. WAP to print first n numbers using for, while and do while loop. [A]
2. WAP to print first n odd numbers using for, while and do while loop. [A]
3. WAP to demonstrate foreach loop. [A]
4. WAP to find sum of all even numbers between 1 to n. [B]
5. WAP to calculate and display sum and product of first N number. [B]
6. WAP to generate Fibonacci series of N number. [B]
7. WAP to check whether the given number is prime or not. [C]
8. WAP to find sum of first and last digit of a number. [C]
9. WAP to find given number is palindrome or not. [C]
10. WAP to check given number is Armstrong or not. [C]

7. **Implementation of patterns**

1. WAP to print following patterns: [A]

```
*
* *
* * *
* * * *
* * * * *
   (a)
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
   (b)
5
5 4
5 4 3
5 4 3 2
5 4 3 2 1
  (c)
1
2 2
3 3 3
4 4 4 4
5 5 5 5 5
   (d)
```

2. WAP to print following patterns: [B]

```
    *
   **
  ***
 ****
*****
   (a)
* * * * *
* * * *
* * *
* *
*
    (b)
*****
```

```
 ****
  ***
   **
    *
   (c)
    *
   * *
  * * *
 * * * *
* * * * *
    (d)
```

3. WAP to print following patterns: [C]

```
    *
   *  *
  *     *
 *         *
* * * *   *

    (a)
  1
 1 1
 1 2 1
1 3 3 1
1 4 6 4 1
   (b)
1 2 3 4 5
 2 3 4 5
  3 4 5
   4 5
    5
   4 5
  3 4 5
 2 3 4 5
1 2 3 4 5
   (c)
```

8. **Implementation of Array**

1. WAP to create numeric array and print it. [A]
2. WAP to create associative array and print it. [A]
3. WAP to create multidimensional array and print it. [A]
4. WAP to find the average of n number using array. [A]
5. WAP to accept n numbers in an array. Display the sum of all the numbers which are divisible by either 3 or 5. [B]
6. WAP to print the elements of an array using foreach loop. [B]
7. WAP to accept n numbers in an array. Now, enter a number and search whether the number is present or not in the list of array elements by using linear search. [B]
8. WAP to sort given array in descending order without using inbuilt function. [B]
9. Find the Kth largest and Kth smallest number in an array. [C]
10. Find first Subarray with given sum. [C]

9.	**Use of Array Function, String Function and Number Function**

1. Create a PHP script that performs the following tasks using a numeric array of student scores (derived from the previous associative array):
$scores = [85, 78, 92, 67, 90];
- Randomly shuffle the array and display the shuffled scores.
- Add a new score of 88 to the end of the array.
- Remove the last score from the array.
- Remove the first score from the array.
- Add a new score of 95 to the beginning of the array.
- Display the total number of scores in the current array.
- Check and display whether the score 90 exist in the array.
- Calculate and display the sum and product of all scores.
- Find and display the minimum and maximum scores.
- Sort the array in ascending order and display it.
- Sort the array in descending order and display it.
- Shuffle the array again and display it.
- Add two new scores (95 and 67) to the array.
- Remove any duplicate scores and display the unique scores.
- Display the array in reverse order.
- Merge the array with another array [73, 88] and display the merged array.
- Find and display the scores that exist in the merged array but not in the original array. [A]

2. Write a PHP script that performs the following tasks using PHP built-in number functions:
$numbers = [-3.7, 2.4, -1.2, 5.75];
- Get the absolute value of each number in $numbers.
- Round each number in $numbers to the nearest integer.
- Find the ceiling and floor values of each number in $numbers.
- Calculate the square root of 4 and 4 raised to the power of 3.
- Find the maximum and minimum values in the array [5, 10, -2, 8].
- Generate 5 random integers between 1 and 100.
- Calculate the floating-point remainder of 5.75 divided by 1.2.
- Format the number 1234.5678 with 2 decimals and a comma as the thousand separators.
- Convert the string "123.45abc" to float and integer types.
- Check which of these values are int, float, numeric, or NaN: 123, 12.3, "abc", "123", NAN.
- Check if values true, INF, 0.5, and null are bool, infinite, finite, or null.
- Convert decimal 255 to hexadecimal, octal, and binary, then convert "ff" (hex) and "377" (octal) back to decimal.
- Check if a variable $var = 0 is set, empty, unset it, and then check if it is set again. [A]

3. Create a PHP script that performs the following tasks using the associative array of student names and their scores:
```
$students = [
   "Alice" => 85,
   "Bob" => 78,
   "Charlie" => 92,
   "David" => 67,
   "Eve" => 90
];
```
- Display the total number of students in the array.
- Print all student names separately.
- Print all scores separately.
- Calculate and display the sum and product of all student scores.
- Find and display the lowest and highest scores in the array.
- Check if any student has scored exactly 90, and display the result.
- Find and display the name of the student who scored 92.

	- Sort and display the students by their scores in ascending order. - Sort and display the students by their scores in descending order. - Sort and display the students by their names in alphabetical order (A-Z). - Sort and display the students by their names in reverse alphabetical order (Z-A). - Merge the $students array with another associative array of students: $newStudents = ["Frank" => 75, "Grace" => 88]; - Display the merged result using both array_merge() and array_merge_recursive() and explain the difference. - Extract and display the first 3 students from the original $students array using array_slice(). [B] 4. Write a PHP script that performs the following tasks using PHP built-in string functions: $text = " Hello, World! Welcome to PHP programming. "; - Display the length of the string. - Convert the entire string to uppercase and lowercase. - Capitalize only the first character of the string. - Capitalize the first character of each word in the string. - Find and display the position of the first occurrence of the substring "World". - Replace the word "World" with "Universe" and display the result. - Extract and display the substring "Welcome to PHP" from $text. $htmlString = "Hello World"; - Convert new lines "\n" in a multiline string into HTML tags. - Remove any backslashes from a string with escape characters, for example: $escapedString = "Hello\\ World\\!"; - Display the character corresponding to ASCII value 65 using chr(). - Display the ASCII value of the character 'A' using ord(). - Compare two strings "apple" and "Apple" both case-sensitively and case-insensitively, displaying the results of the comparisons. [B 5. Create a PHP script that performs the following operations on an array of names (strings). Use array_map() function and string functions to manipulate the array values: $names = [" Alice ", "JessiCA", " charlie", "DAVID", "eVa", "AleXa", "Olivia", " Levi "]; - Trim leading and trailing whitespace from each name. - Convert all names to lowercase. - Capitalize the first letter of each name. - Find and display the length of each name. - Reverse each name and display the reversed names. - Check if any name contains the substring "vi" (case-insensitive). Print those names. - Join all names into a single string separated by a comma and a space. [C]
10.	**Implementation of User Define Function (UDF)** 1. WAP to create user define function to print your name (without argument) and call it. [A] 2. WAP to create user define function to print name (with argument) and call it 5 times with different names as an argument. [A] 3. WAP to create user define function for adding two numbers and return answer and use it to find addition of two numbers and print answer. [A] 4. WAP to calculate simple interest using method. [B] 5. WAP to generate Fibonacci series of N given number using method. [B] 6. WAP for simple calculator with the use of simple calculator. [C] 7.WAP to find sum of all odd array elements by passing array as an argument using user define functions. [C]
11.	**Implementation of Recursion** 1. WAP to print numbers from N to 1 using recursion. [A] 2. WAP to calculate sum of first n numbers using recursion. [A]

	3. WAP to calculate factorial of a number using recursion. [B] 4. WAP to generate Fibonacci series of N number using recursion. [B] 5. WAP to check whether the number is prime or not using recursion. [C] 6. WAP to find the LCM of two numbers using recursion. [C]
12.	**Demonstration of Web Page Partition and File Attributes** 1. WAP to demonstrate the use of include, require, include_once and require_once. [A] 2. Demonstrate the use of PHP's include, require, include_once, and require_once functions to partition a webpage into different sections in Lab 2(4) and Lab 3. [B] 3. Write a PHP program that creates an HTML form to do the following: - Place a text input field outside the tag and associate it with the form using the input's form attribute and the form's id. - Use the required attribute on one or more input fields to ensure the user must fill them before submitting. - Add the novalidate attribute to the tag while keeping the required attributes on inputs to observe the behaviour. - Use the autocomplete attribute on the tag. - Add the accept-charset attribute to the tag. - Utilize the target attribute with values like _self, _blank, _parent, _top, and a custom iframe name. - Set action="" or action="#" to submit the form to the same page. - Set action to a different PHP page to handle submission. - Test the form with different target values for each action. [C]
13.	**Implement of Form Processing** 1. Design a student registration form and retrieve data in controller page using following Method: GET, POST, and REQUEST. [A] 2. Design an employee registration form and retrieve data in controller page using following Method: GET, POST, and REQUEST. [B] 3. Create form with all input types and retrieve data in controller page using following Method: GET, POST, and REQUEST. [C]
14.	**Implementation of File Handling** 1. WAP to create a PHP function that takes a file path as input and checks if the file exists. [A] 2. Create a webpage which reads data from a text file and print it. [A] 3. Create a webpage which write some data into a text file and print it. [B] 4. Create a webpage which delete a file from server. [B] 5. WAP to create a PHP script to count the number of lines in a text file. [C] 6. WAP to create a PHP program that reads a CSV file and displays the data in a tabular format. [C]
15.	**Implementation of File Upload** 1. Create a webpage which accepts a file and upload it in specified folder on server. [A] 2. Design a profile page which allows changing profile picture dynamically. [B] 3. Create a webpage which accepts an image file in .jpg or .jpeg or .png format only and that to maximum of 1MB. [C] 4. Create a webpage which accepts multiple files and upload them in specified folder on server. [C]
16.	**Implementation of Regular Expression and Server-Side Validation** 1. Implement server-side validation on student registration form using PHP. [A] 2. Implement server-side validation on employee registration form using PHP. [B] 3. Implement server-side validation on form with all input types using PHP. [C]
17.	**Implementation of Session Management Using COOKIES And SESSION** 1. Create static login application using COOKIE in PHP. [A] 2. Create static login application using SESSION in PHP. [B]
18.	**Implementation OOP Concept (Part-I)** 1. WAP to create a PHP class called 'Vehicle' with properties like 'brand', 'model', and 'year'. Implement a method to display the vehicle details. [A] 2. Design a class named Book with the following attributes: title, author, and price. Include methods to input and display the book details. Instantiate two Book objects and demonstrate their usage. [A] 3. Create a class named Rectangle with attributes length and width. Implement methods to calculate and return the area and perimeter of the rectangle. [B] 4. Implement a class named Student with attributes name and rollNumber. Use a constructor to initialize these attributes and a

	destructor to display a message when an object is destroyed. Demonstrate the creation and destruction of a Student object. [B] 5. Write a class named Counter with an attribute count. Use a constructor to initialize the count and a method to increment it. Demonstrate the functionality by creating multiple instances of the Counter class and displaying their individual counts. [C] 6. Write a PHP class named BankAccount with properties such as accountNumber and balance. Implement methods to retrieve the account number and balance, and to deposit and withdraw money. [C]
19.	**Implementation OOP Concept (Part-II)** 1. Create a base class named Employee with attributes name and salary. Derive a class Manager that adds an attribute department. Implement methods to input and display details for both classes. Demonstrate inheritance by creating an object of the Manager class. [A] 2. Design a class hierarchy where Vehicle is the base class with attributes such as make and model. Derive classes Car and Motorcycle, each adding attributes specific to their type. Implement methods to display details for each class and demonstrate polymorphism by invoking the display method through base class pointers. [A] 3. Create a base class Product with properties name and price, and a method getProductInfo(). Extend it with subclasses Book and Electronics that add properties like author for books and warrantyPeriod for electronics. [A] 4. WAP to create a PHP class hierarchy for a library system, including classes like 'LibraryItem', 'Book', 'DVD', etc. Implement appropriate properties and methods for each class. [B] 5. Create a PHP class named Circle with a private property radius. Implement a public method setRadius($r) to set the value of radius, and a public method getArea() to calculate and return the area of the circle. [B] 6. Create a PHP class named Student with private attributes name and grades (an array of numbers). Implement public methods: setName($name) to assign the student's name, addGrade($grade) to add a grade to the grades array, and getAverage() to calculate and return the average grade. [B] 7. Create a PHP class hierarchy where Person is the base class with protected attributes firstName and lastName, and public methods setFirstName(), setLastName(), and getFullName(). Derive an Employee class that adds a private attribute employeeId with public methods to set and get the employeeId. Instantiate an Employee object and demonstrate setting and retrieving all details. [C] 8. Create a PHP class named Employee with private properties name and basicSalary. Include public methods to set the name and salary (with basic validation), and a method to calculate net salary after deducting 12% tax. Ensure the tax logic is encapsulated using a private method, and demonstrate the class by creating an object and displaying the employee’s name and net salary. [C]
20.	**Implementations of CRUD Database Operation** 1. Implement CRUD database operations on students table from webpage using PHP. [A] 2. Implement CRUD database operations on Student Registration page from webpage using PHP. [A] 3. Implement CRUD database operations on categories table from webpage using PHP. [B] 4. Implement CRUD database operations on products table from webpage which contains foreign key relations with categories using PHP. [C]
21.	**Implementations of CRUD Database Operation Using Prepared Statement** 1. Implement CRUD database operations on students table using prepared statement in PHP. [A] 2. Implement CRUD database operations on categories table using prepared statement in PHP. [B] 3. Implement CRUD database operations on products table from webpage which contains foreign key relations with categories using prepared statement in PHP. [C]
22.	**Implementations of CRUD Database Operation Using Stored Procedure** 1. Implement CRUD database operations on students table using stored procedure in PHP. [A] 2. Implement CRUD database operations on categories table using stored procedure in PHP. [B] 3. Implement CRUD database operations on products table from webpage which contains foreign key relations with categories using stored procedure in PHP. [C]
23.	**Implementations of Fully Functional Login and Registration Application with Remember me Functionality** 1. Implement registration application using PHP. [A] 2. Implement complete login application using PHP, also use cookie to provide remember me functionality. [B]
24.	**Implementation of REST API** 1. Implement REST API for students table in PHP. [A]

	2. Implement REST API for categories table in PHP. [B] 3. Implement REST API for products table which contains foreign key relations with categories table in PHP. [C]
25.	**Implementation of AJAX** 1. Create web page which reads cricket score from database and update in every second without reloading webpage. (Note: Create corresponding API) [A] 2. Create a webpage which have three dropdowns Country, State and City in which when page loads first time only country dropdown should be filled and other two should be disable when country is selected corresponding states should be filled in state dropdown and enable it and when state is selected corresponding cities should be filled in City dropdown and enable it and should changes available options according to selection changes in any dropdown. [A]
26.	**Demonstration of Laravel Installation, Use Routes and Views** 1. Install and configure Laravel application and create first project. [A] 2. Use Laravel routing and create static pages which can be visited from URL. [A] 3. Create menu in above application. [A] 4. Create Laravel application which contains home page, about page, contact page, and product page also create menu on top. [B]
27.	**Implementation of Database Connection and Migrations in Laravel** 1. Change settings for database connection to connect with your database in ". env" file. [A] 2. Use artisan command to create migration file and create database table for student with following fields: name, semester, branch, enrollment number, password etc. [A] 3. Practice Laravel seeding for populate database by programming for testing purpose. [B]
28.	**Implementation of Read Operation Using Model and Various Artisan Commands in Laravel** 1. Use artisan command to create model file for student and use query builder to retrieve all data from student table using DB facades. [A] 2. Use artisan command to create model file for student and write various relationship available in Laravel and use it to retrieve data from database. [B] 3. Create Login Registration application using artisan command in Laravel. [C]
29.	**Implementations of Various Calculators Using Controllers in Laravel** 1. Use artisan command to create controller class and create simple calculator application. Write business logic in controller. [A] 2. Create area calculator application for various graphics primitives like square, rectangle, circle, etc. [B]
30.	**Implementation of Insert, Update and Delete Operation in Laravel** 1. Implement Insert, update and delete operation on student table ((select appropriate fields) using Laravel. [A] 2. Implement Insert, update and delete operation on categories table (select appropriate fields) using Laravel. [B] 3. Implement Insert, update and delete operation on products table (select appropriate fields) having foreign key relationship with categories table using Laravel. [C]

Useful Links
1. https://www.php.net/ 2. https://laravel.com/