Date:_____

Practical No. 1: Environment Setup.

1. Install and configure PHP, Web Server and MySQL database using XAMPP/WAMP/LAMP/MAMP.
2. Create a web page that displays “Hello World.”

A. Objectives:

A development environment is required to write, compile, run, and debug any application. This practical will help student to set up PHP environment for executing PHP program using different server like XAMPP or WAMP server.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.
3. **Engineering practices for society, sustainability and environment(PO5):** Apply appropriate technology in context of society, sustainability, environment and ethical practices.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency ‘**Develop Interactive Web application using PHP and MySQL**’:

1. Installing and configuring softwares as per the requirements.
2. Programming skills.
3. Debugging skills.

D. RelevantCourse Outcomes (COs):

1. Develop PHP scripts using variables, operators and control structures.

E. PracticalOutcomes:

1. Install and configure web application developmentenvironment for PHP and MySQL.

F. RelevantAffective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices

G. Prerequisite Theory:

XAMPP is one of the most popular software pack to set up web application development environment for PHP with all required software components. XAMPP is an Open Source AMP stack which stands for***Cross platform, Apache, MariaDB, PHP*** and ***Perl***. ***Apache*** is cross platform web server, ***MariaDB*** is the most widely used database developed by MySQL, ***PHP*** is a backend scripting language and ***Perl*** is a programming used for web development. ***X*** denotes Cross-platform, which means that it can work on different platforms such as Windows, Linux, and macOS.

XAMPP allows a local host or server to test its website and clients on computers and laptops before releasing them to the main server. It provides a suitable environment for testing and verifying the functioning working of projects based on Apache, Perl, MySQL database, and PHP on the host's system. It also includes administrative tools such as phpMyAdmin, Filezilla FTP Server, Mercury mail server and JSP Tomcat server.

H. Resources Required:

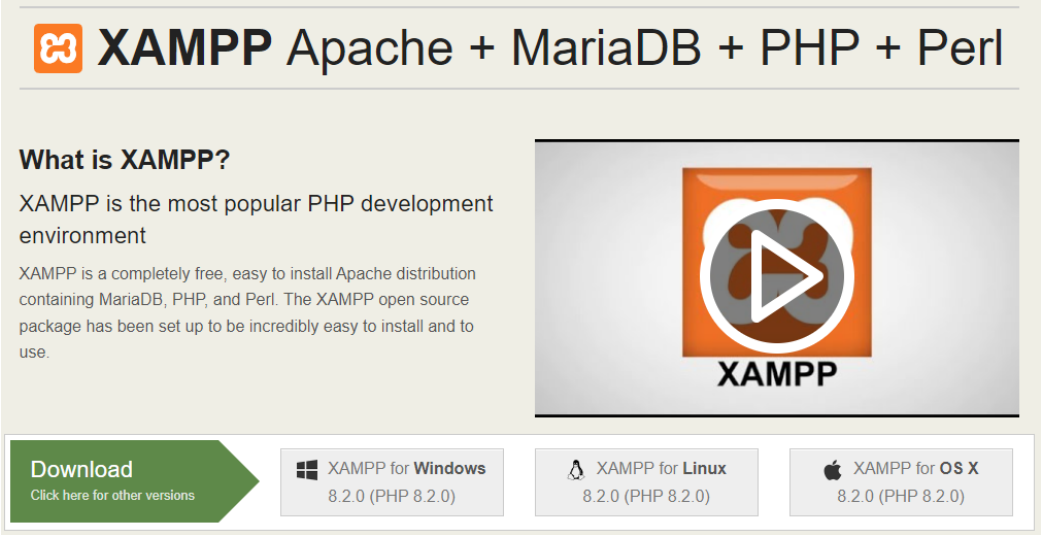
Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:
5.	Internet Connection	-

I. Procedure:

Below steps describes installation of XAMPP on Windows operating system. Steps are similar for Linux and Mac operating systems.

i. Install XAMPP

1. Open the XAMPP website. Go to https://www.apachefriends.org/index.html in your computer's web browser.

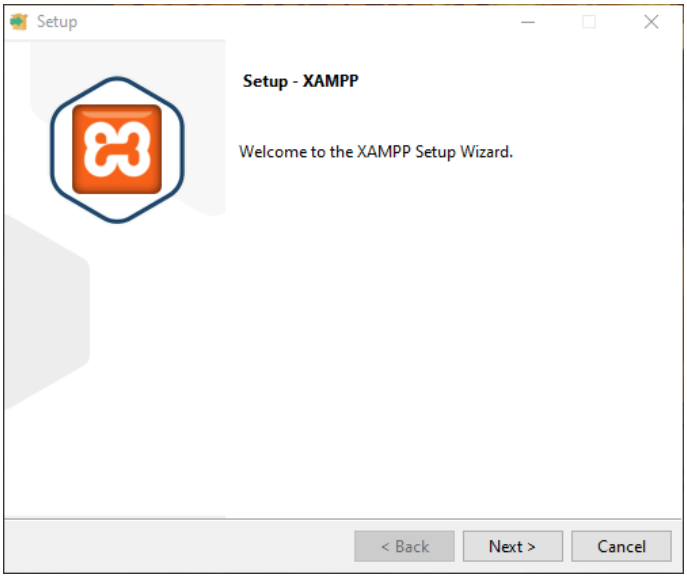


Download latest XAMPP for Windows operating system.

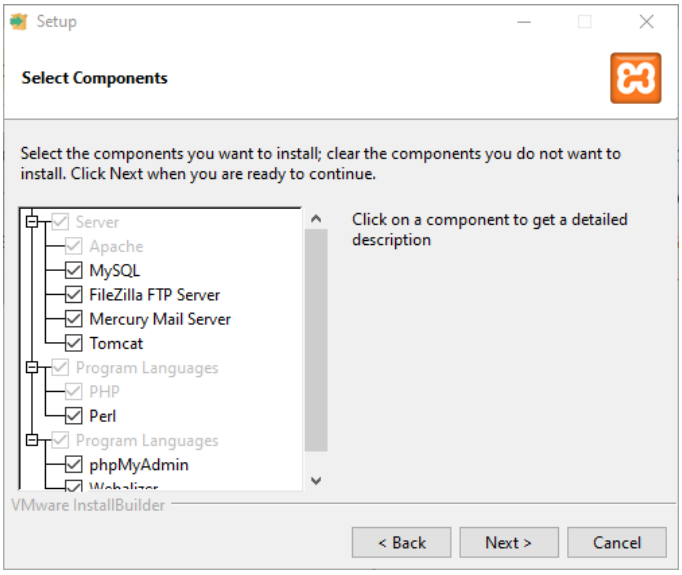
2. Once the XAMPP setup has been downloaded, you can start the installation by double clicking on the .exe file.
3. An active antivirus program can interfere with the installation process, so it is best to temporarily disable any antivirus software until all XAMPP components have been successfully installed.

4. Because User Account Control (UAC) restricts writing access to the C: drive and can interfere with the XAMPP installation, it is recommended that this be disabled for the duration of the installation.

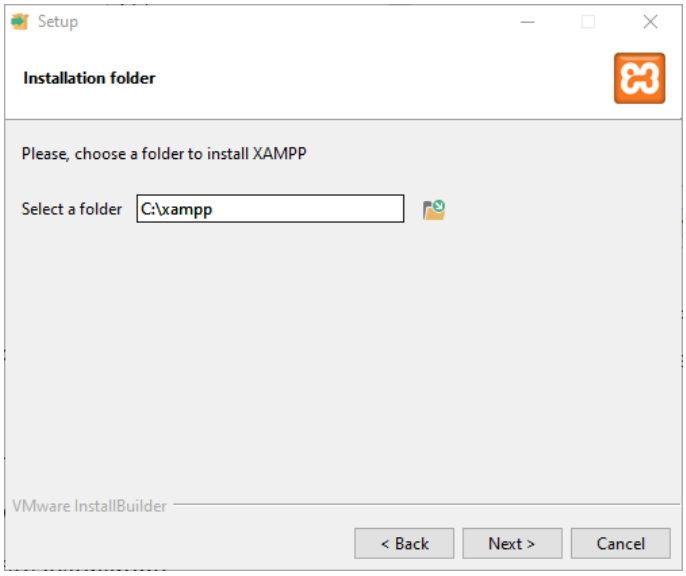
5. After that the start screen of the XAMPP setup wizard should appear automatically. Click on 'Next' to configure the installation settings.



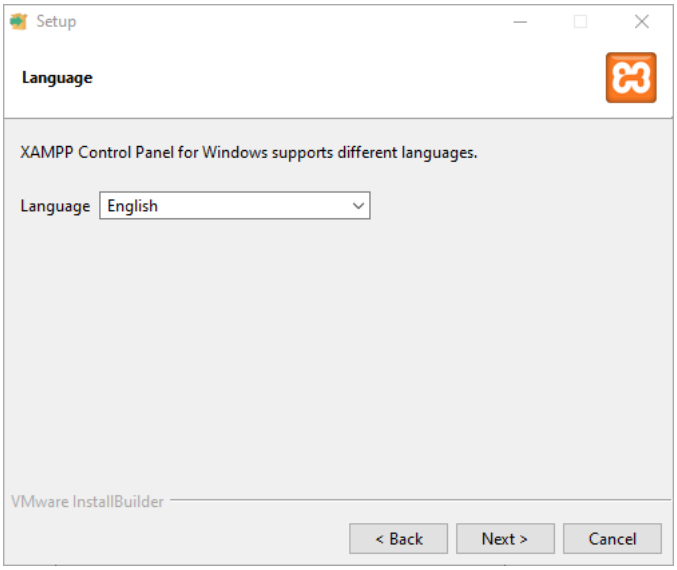
6. Under 'Select Components', you have the option to exclude individual components of the XAMPP software bundle from the installation. But for a full local test server, we recommend you install using the standard setup and all available components. After making your choice, click 'Next'.



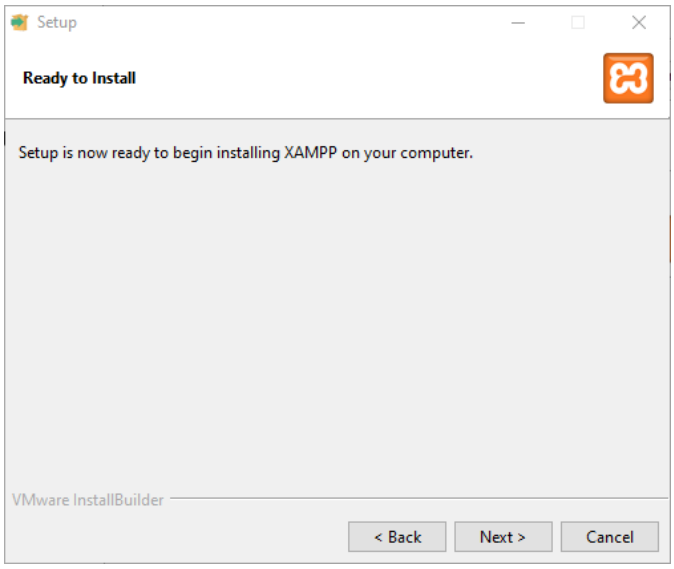
7. In this next step, you have the chance to choose where you'd like the XAMPP software packet to be installed. If you opt for the standard setup, then a folder with the name *xampp* will be created under *C:* for you. After you've chosen a location, click 'Next'.



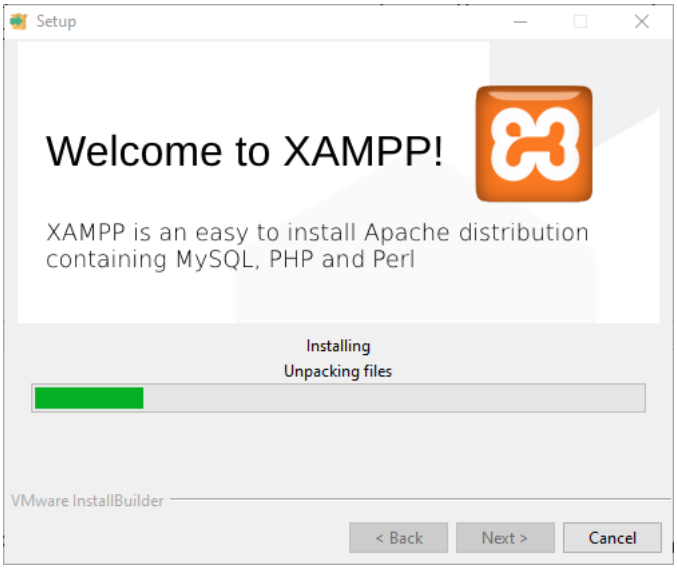
8. Select the language in the next dialog box.



9. On *Ready to Install* screen click on "Next".

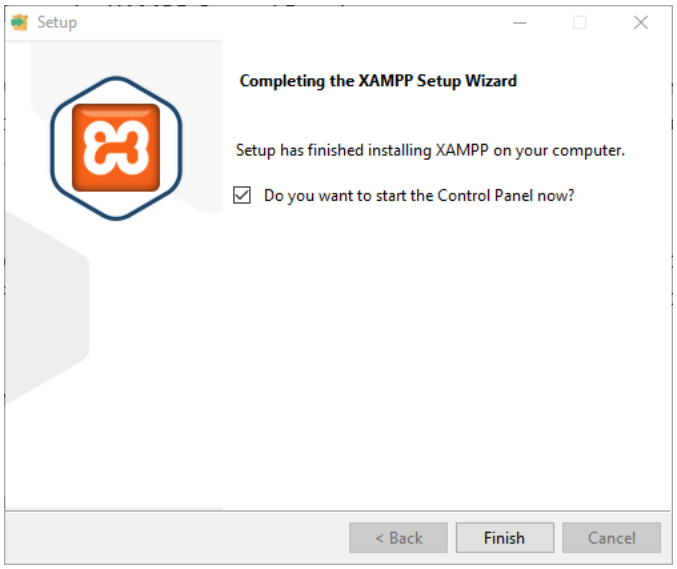


10. Once all the preferences have been decided, click to start the installation. The setup wizard will unpack and install the selected components and save them to the designated directory. This process may take few minutes.



11. Your Firewall may interrupt the installation process to block the some components of the XAMPP. Use the corresponding check box to enable communication between the Apache server and your private network or work network. Remember that making your XAMPP server available for public networks isn't recommended.

12. Once all the components are unpacked and installed, you can close the setup wizard by clicking on 'Finish'. Click to tick the corresponding check box and open the XAMPP Control Panel once the installation process is finished.



13. XAMPP Control Panel provides controls for the individual components of your xampp test server. The control panel user interface allows you to start or stop individual modules: Apache, MySQL, FileZilla, Mercury and Tomcat. The XAMPP Control Panel also offers you various other buttons, including:

Config: allows you to configure the XAMPP as well as the individual components

Netstat: shows all running processes on the local computer

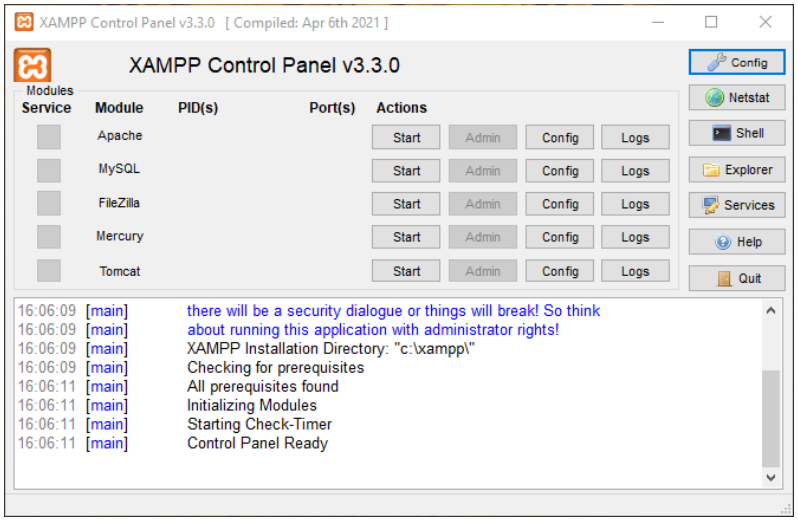
Shell: opens a UNIX shell

Explorer: opens the XAMPP folder in Windows Explorer

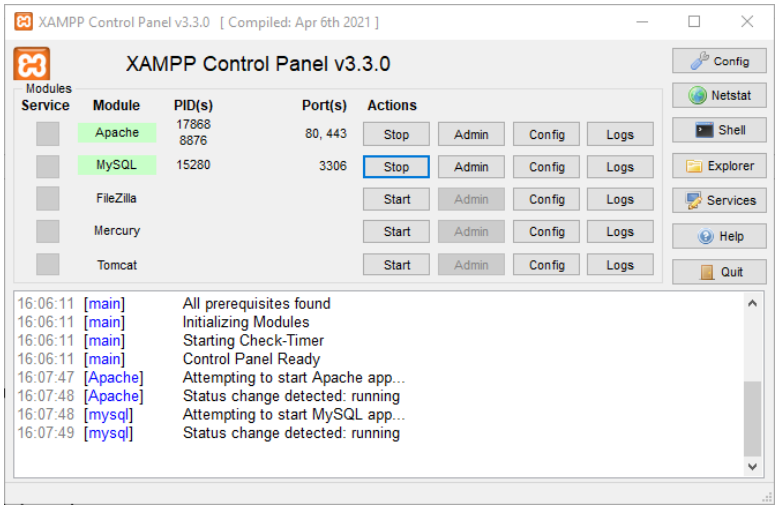
Services: shows all services currently running in the background

Help: offers links to user forums

Quit: closes the XAMPP Control Panel



14. Individual modules can be started or stopped on the XAMPP Control Panel through the corresponding buttons under 'Actions'. You can see which modules have been started because their names are highlighted green under the 'Module' title.

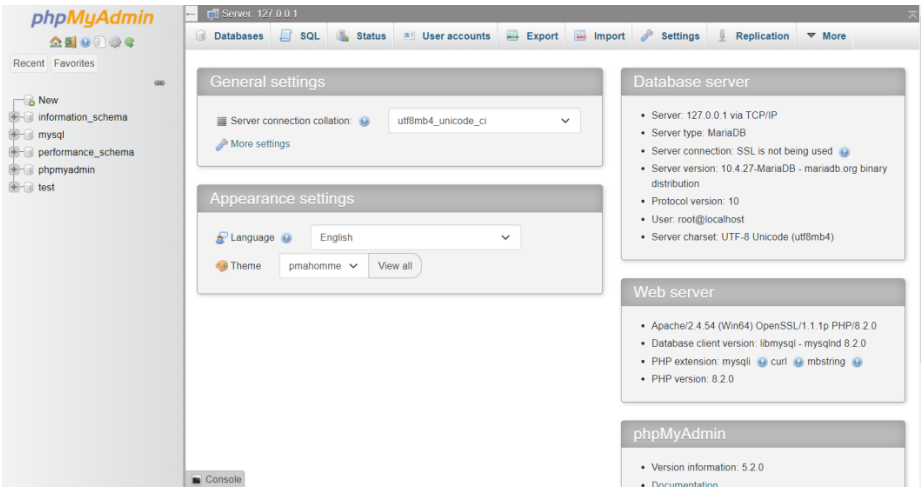


If a module can't be started as a result of an error, you'll be informed of this straight away in red font. A detailed error report can help you identify the cause of the issue.

15. You have an 'Admin' option located on the Control Panel for every module in your XAMPP.

16. Click on the Admin button of your Apache server to go to the web address of your web server. The Control Panel will now start in your standard browser, and you'll be led to the dashboard of your XAMPP's local host. The dashboard features numerous links to websites for useful information as well as the open source project BitNami, which offers you many different applications for your XAMPP, like WordPress or other content management systems. Alternatively, you can reach the dashboard through localhost/dashboard/.

17. You can use the Admin button of your database module to open phpMyAdmin. Here, you can manage the databases of your web projects that you're testing on your XAMPP. Alternatively, you can reach the administration section of your MySQL database via localhost/phpmyadmin/



ii. Testing XAMPP installation

To check whether your test server is installed and configured correctly, you have the option to create a PHP test page, store them on your XAMPP's local host, and retrieve them via the web browser.

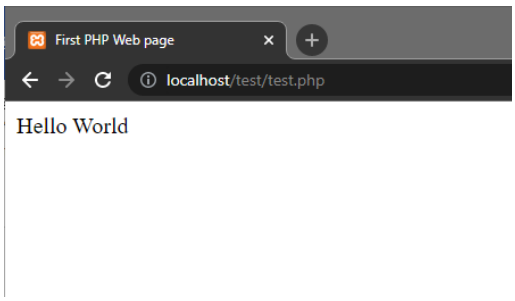
18. Open the XAMPP directory through the 'Explorer' button in the Control Panel and choose the folder *htdocs* (*C:\xampp\htdocs* for standard installations). This directory should store all the web pages that you want to test on your XAMPP

server. The *htdocs* folder should already contain data to help configuration of the web server. But you should store your own projects in a new folder (for example '*test'* folder).

19. You can create a new PHP file with below code in your editor and storing it as *test.php* in your '*test'* folder (*C:\xampp\htdocs\test*):

```
test.php
1  <html>
2      <head>
3          <title>First PHP Web page</title>
4      </head>
5      <body>
6          <?php
7              echo '<p>Hello World</p>';
8          ?>
9      </body>
10 </html>
11
```

20. Now open a web browser and load your PHP page via localhost/test/test.php. If your browser window displays the words 'Hello World', then you've successfully installed and configured your XAMPP.



Output:

Snapshot of XAMPP Control Panel after installation.

Snapshot of “Hello World” script code.

Output:

Snapshot of output of “Hello World” script in web browser.

Output:

J. References:

1. https://www.apachefriends.org
2. https://phpandmysql.com/extras/installing-xampp
3. https://www.youtube.com/watch?v=at19OmH2Bg4
4. https://www.youtube.com/watch?v=PaDgry5QAt4
5. https://www.w3schools.com/php

Sign

Date:_____

Practical No. 2: Form Introduction.

1]. Create a web page that collects user information using a form and displays it when the user clicks the submit button.

A. Objectives:

Forms are important component of the web application that allows it to collect information from the users. Most websites use different forms for various tasks such as log in, registration, contact us, and application specific information collection. Now a days you rarely see any website without a form. This practical will help students to design a form to collect user data using PHP.

B. RelevantProgram Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop web pages using form controls with validation to collect user inputs in PHP.

E. PracticalOutcomes:

1. Develop web pages using Form controls such as text box, button, check box, radio button, text area etc.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.

2. Follow Coding standards and practices.
3. Follow ethical practices

G. Prerequisite Theory:

Web forms are one of the most common ways for a user to interact with a web application. Forms allow users to enter data, which is typically sent to a web server for processing and storage or used on the client-side to update the interface in some way immediately.

The HTML of a web form is made up of one or more form controls (also known as widgets) and some additional elements to help structure the overall form - these are commonly referred to as HTML forms. Most common controls are single or multi-line text boxes, dropdown boxes, buttons, checkboxes and radio buttons. There are some other elements such as date, time, day color, file etc. Form controls can also be programmed to enforce specific formats or values to be entered (form validation).

Define HTML Form:

The HTML <form> element is used to create an HTML form for user input:

```
<form>
...
form elements
...
</form>
```

Define HTML Form Element:

HTML form elements are mostly created using the <input> element.

```
<formaction="/action.php" method="get">
  <label for="uname">User Name:</label><br>
  <input type="text" id="uname" name="uname"><br>
  <label for="password">Password:</label><br>
  <input type="password" id="password" name="password">
<input type="submit" value="Submit">
</form>
```

- The *<label>* tag defines a label for many form elements. The *<label>* element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focus on the input element.
- The <input type="*submit*"> defines a button for submitting the form data to a form-handler. The form-handler is typically a file on the server with a script for processing input data.
- The action attribute defines the action to be performed when the form is submitted. Usually, the form data is sent to a file on the server when the user clicks on the submit button.

- The method attribute specifies the HTTP method to be used when submitting the form data. The form-data can be sent as URL variables (with method="*get*") or as HTTP post transaction (with method="*post*").
- The HTML id attribute is used to specify a unique id for an HTML element. You cannot have more than one element with the same id in an HTML document.
- Each input field must have a *name* attribute to be submitted. If the *name* attribute is omitted, the value of the input field will not be sent at all. It is used by PHP script to read form data from that input element.
- The *type* attribute defines type of input element. By default value of the type attribute is "text". Different types of input elements are as follow:

Element Type	Description
<input type="text">	Displays a single-line text input field.
<input type="radio">	Displays a radio button (for selecting one of many choices)
<input type="checkbox">	Displays a checkbox (for selecting choices)
<input type="submit">	Displays a submit button (for submitting the form)
<input type="button">	Displays a clickable button
<input type="reset">	Displays a reset button (for resetting the form)
<input type="search">	Displays a text input field for search field.
<input type="password">	Displays a text input field for password.
<input type="number">	Displays a text input field for numbers.
<input type="tel">	Displays a text input field for telephone.
<input type="url">	Displays a text input field for URL.
<input type="email">	Displays a text input field for email.
<input type="range">	Displays a range slider.
<input type="file">	Displays a file-select field and a "Browse" button for file uploads
<input type="color">	Displays an input field for color picker.
<input type="datetime-local">	Displays a date and time input field, with no time zone
<input type="date">	Displays an input fields for date
<input type="time">	Displays an input fields for time.
<input type="month">	Displays an input fields for month.
<input type="week">	Displays an input fields for week.
<textarea> … <textarea>	Displays a multi-line plain-text input field.

TextBox:

The <input type="text">provides a single-line input field to input text.

```
<form action="/action.php" method="get">
  <label for="fname">First name:</label><br>
<input type="text" id="fname" name="fname"><br>
<label for="lname">Last name:</label><br>
<input type="text" id="lname" name="lname">
<input type="submit" value="Submit">
</form>
```

Radio Button:

Radio Button allows user to select only one choice from a limited number of choices.

```
<form action="/action.php" method="get">
  <input type="radio" id="male" name="gender"value="male">
<label for="male">Male</label><br>
<input type="radio" id="female" name="gender"
                                  value="female">
<label for="female">Female</label><br>
<input type="radio" id="other" name="gender" value="other">
<label for="other">Other</label>
<input type="submit" value="Submit">
</form>
```

Check Box:

Check Box allows a user to select zero or more choices from a limited number of choices.

```
<form action="/action.php" method="get">
<input type="checkbox" id="fruit1" name="friut1"
                                  value="Mango">
<label for=" fruit1">Mango</label><br>

<input type="checkbox" id="fruit2" name="friut2"
                                  value="Banana">
<label for=" fruit2">Banana</label><br>

<input type="checkbox" id="fruit3" name="friut3"
                                  value="Grapes">
<label for=" fruit3">Grapes </label><br>
<input type="submit" value="Submit">
</form>
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1]. Create a web page that collects user information using a form and displays it when the user clicks the submit button.

J. References:

1. https://www.w3schools.com/html/html_forms.asp
2. https://developer.mozilla.org/en-US/docs/Web/HTML/Element/form
3. https://www.quackit.com/html/codes/html_form_code.cfm
4. https://www.javatpoint.com/html-form
5. https://www.geeksforgeeks.org/html-forms
6. https://www.tutorialspoint.com/html/html_forms.htm

Sign

Practical No. 3: Variables, Operators and Expression

1]. Write a script to implement a simple calculator for mathematical operations.

2]. A company has following payment scheme for their staff:

- **Net Salary = Gross Salary – Deduction**
- **Gross Salary = Basic pay + DA + HRA + Medical**
- **Deduction = Insurance + PF**

where, DA (Dearness Allowance) = 50% of Basic pay

HRA (House Rent Allowance) = 10% of Basic pay

Medical = 4% of Basic pay

Insurance = 7% of Gross salary

PF (Provident Fund) = 5% of Gross salary

3]. Write a script to take the basic salary of an employee as input and calculate the net payment to any employee.

A. Objectives:

Variables, Operators and Expressions are core part of any programming language.

- Variables are used to store data.
- Operators are used to perform various types of operations on data.
- Anything that you write in PHP script is an expression.

This practical will allow students to practise writing PHP scripts that use variables, operators, and expressions to solve simple problems.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop PHP scripts using variables, operators and control structures.

E. Practical Outcomes:

1. Use PHP variables to store data in PHP scripts.
2. Perform operation on data using operator in PHP scripts.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

Variables:

A variable is a named area of storage, where you can store a value.

- In PHP, variables are represented by a dollar sign (*$*) followed by the name of the variable.
- Variable namesare case-sensitive. For example,*$var* and *$Var* are two different variables.
- A valid variable name starts with a letter or underscore, followed by any number of letters, numbers, or underscores.
- A PHP variable name cannot contain spaces.
- For example,*$enrollmentno, $subject_name, $_itemid* are valid variable names and *$12var, $student name, part_id* are invalid variable names.
- PHP is loosely-typed language, so it doesn't need to specify data type of variables. It automatically analyse assigned value and defined data type of variable.
- Assignment Operator (=) is used to assign the value to a variable.For example,

```
$subject_name = "Introduction to Web Development";
```

PHP has total eight different data types which can be used to define variables:

- Integers − whole numbers, without a decimal point. E.g. – 415, 8341.
- Doubles − floating-point numbers. E.g. – 12.5, 3.14.
- Booleans − two possible values either TRUE or FALSE.
- NULL − special type that only has one value: NULL.

- Strings − sequences of characters. E.g. - 'Introduction to Web Development'.
- Arrays − named and indexed collections of other values.
- Objects − instances of programmer-defined classes
- Resources − special variables that hold references to resources external to PHP. (such as database connections).

Operators:

Operator is a symbol used to perform operations on operands (variables or values).For example:

```
$a = $b + 10;
```

Above code uses arithmetic operator (+) to add 10 to variable *$b* and assign it to variable *$a*.

PHP operators can be categorized into following types:

Arithmetic Operators:

Operator	Name	Example	Description
+	Addition	`$a + $b`	Sum of two operands
-	Subtraction	`$a - $b`	Difference of two operands
*	Multiplication	`$a * $b`	Multiply tow operands
/	Division	`$a / $b`	Quotient of operands
%	Modulo	`$a % $b`	Reminder of operands
++	Increment	`$a++`	Same as `$a = $a + 1`
--	Decrement	`$a--`	Same as `$a = $a - 1`

Assignment Operators:

Operator	Name	Example	Description
=	Assign	`$a = $b`	Value of right operand is assigned to left operand
+=	Add then assign	`$a += $b`	Same as `$a = $a + $b`
-=	Subtract then assign	`$a -= $b`	Same as `$a = $a - $b`
*=	Multiply then assign	`$a *= $b`	Same as `$a = $a * $b`
/=	Divide then assign (Quotient)	`$a /= $b`	Same as `$a = $a / $b`
%=	Divide then assign (Reminder)	`$a %= $b`	Same as `$a = $a % $b`

Bitwise Operators:

Operator	Name	Example	Description
&	Bitwise AND	`$a & $b`	Bitwise AND operation between $a and $b
|	Bitwise OR	`$a | $b`	Bitwise OR operation between $a and $b
^	Bitwise XOR	`$a ^ $b`	Bitwise XOR operation between $a and $b
~	Bitwise NOT	`~ $a`	Bitwise NOT operation on $a
<<	Left shift	`$a << $b`	Left shift bits of $a by $b steps
>>	Right shift	`$a >> $b`	Right shift bits of $a by $b steps

Comparison Operators:

Operator	Name	Example	Description
==	Equal	`$a == $b`	Returns TRUE if $a is equal to $b
!=	Not equal	`$a != $b`	Returns TRUE if $a is not equal to $b
<>	Not equal	`$a <> $b`	Returns TRUE if $a is not equal to $b
===	Identical	`$a === $b`	Returns TRUE if $a and $b are equal and of same data type
!==	Not identical	`$a !== $b`	Returns TRUE if $a and $b are not equal or of different data type
<	Less than	`$a < $b`	Returns TRUE if $a is less than $b
>	Greater than	`$a > $b`	Returns TRUE if $a is greater than $b
<=	Less than or equal to	`$a <= $b`	Returns TRUE if $a is less than or equal to $b
>=	Greater than or equal to	`$a >= $b`	Returns TRUE if $a is greater than or equal to $b
<=>	Spaceship	`$a <=> $b`	Return -1 if $a is less than $b Return 0 if $a is equal $b Return 1 if $a is greater than $b

Logical Operators:

Operator	Name	Example	Description
and	Logical AND	`$a and $b`	Returns TRUE if both $a and $b are true
or	Logical OR	`$a or $b`	Returns TRUE if either aorbis true
xor	Logical XOR	`$a xor $b`	Returns TRUE if either aorbis true,

			but not both are TRUE
!	Logical NOT	!$a	Returns TRUE if $a is FALSE
&&	Logical AND	$a && $b	Returns TRUE if both $a and $b are true
||	Logical OR	$a || $b	Returns TRUE if either aorbis true

String Operators:

Operator	Name	Example	Description
.	Concatenation	$a . $b	Concatenate both $a and $b
.=	Concatenation and assign	$a .= $b	Same as $a = $a . $b

Expressions

In PHP, anything that has a value is an expression, so most of the statements you write in PHP scrips are expression. For Example:

```
$a = $b;              // $b is an expression
$a = $b * $c;         // $b * $c is an expression
$a = sum($b, $c);     // sum($b, $c) is an expression
$b++;                 // $b++ is an expression
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Write a script to implement a simple calculator for mathematical operations.

2. A company has following payment scheme for their staff:

- Net Salary = Gross Salary – Deduction
- Gross Salary = Basic pay + DA + HRA + Medical
- Deduction = Insurance + PF

where, DA (Dearness Allowance) = 50% of Basic pay

HRA (House Rent Allowance) = 10% of Basic pay

Medical = 4% of Basic pay

Insurance = 7% of Gross salary

PF (Provident Fund) = 5% of Gross salary

Write a script to take the basic salary of an employee as input and calculate the net payment to any employee.

J. References:

1. https://www.php.net/manual/en/language.variables.php
2. https://www.php.net/manual/en/language.expressions.php
3. https://www.w3schools.com/php/php_variables.asp
4. https://www.w3schools.com/php/php_operators.asp
5. https://www.geeksforgeeks.org/php-variables

Sign

Practical No. 4: Decision making statements and Loops

1]. Write a script that reads the name of the car and displays the name of the company the car belongs to as per the below table:

Car	Company
Safari, Nexon, Tigor, Tiago	Tata
XUV700, XUV300, Bolero	Mahindra
i20, Verna, Venue, Creta	Hyundai
Swift, Alto, Baleno, Brezza	Suzuki

2]. Write a script to read the marks of 4 subjects and display the result as per the below instructions:

GTU GRADE	Mark-Range
AA	85 - 100
AB	75 - 84
BB	65 - 74
BC	55 - 64
CC	45 - 54
CD	40 - 44
DD	35 - 39
FF	< 35 (FAIL)

- Each of the four subjects is worth 100 marks.
- If a student gets less than 35 marks in any subject, then he/she will be marked as FAIL, otherwise he/she will be marked as PASS.

The result contains the grade of each individual subject in tabular format as per the above table.

3]. Write a script to display Fibonacci numbers up to a given term.

4]. Write a script to display a multiplication table for the given number.

A. Objectives:

This practical will help student to practice writing PHP scripts using Decision making structure and Loops.

- Conditional statements are used to perform different actions based on different conditions.
- Loops are used to run same block of code again and again certain number of times.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.

2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop PHP scripts using variables, operators and control structures.

E. Practical Outcomes:

1. Use decision making statements in PHP scripts.
2. Use loops in PHP scripts.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

Decision Making Statements:

Controls statements are used to control are used to control the flow of execution of program based on certain conditions. In PHP, there are following decision making statements:

- `if` statement
- `if...else` statement
- `if...elseif...else` statement
- `switch` statement

`if` statement:

`if` statement allow us to run a block of code if certain condition is true. If condition is false it will not execute block of code.

```
if(condition) {
      Block of code     // statements to execute if condition is
                        // true
}
```

`if...else`statement:

`if...else`executes a block of code if certain condition is true and another block of code if condition is false.

```
if(condition) {
      Block of code     // statements to execute if
                        // condition is true
}
else {
      Block of code     // statements to execute if
                        // condition is false
}
```

`if...elseif...else` statement:

It is similar to multiple `if...else` statements. It executes different blocks of code based on different conditions.

```
if(condition) {
      Block of code     // statements to execute if this
                        // condition is true
}
elseif(condition) {
      Block of code     // statements to execute if this
                        // condition is true
} elseif(condition) {
      Block of code     // statements to execute if this
                        // condition is true
} else {
      Block of code     // statements to execute if all
                        // conditions are false
}
```

`switch` statement:

The switch statement is used to perform different actions based on different conditions. Use the switch statement to select one of many blocks of code to be executed.

```
switch(n) {
case value1:
code to be executed if n== value1;
break;
case value2:
code to be executed if n== value2;
break;
case value3:
code to be executed if n== value3;
break;
case value4:
code to be executed if n== value4;
break;
    ......
default:
code to be executed if n != any case;
}
```

Loops:

Loops in PHP are used to execute the same block of code a specified number of times. PHP supports below loop statements:

- `while` loop
- `do...while` loop
- `for` loop
- `foreach` loop

`while` loop:

`while` loop executes a block of code as long as the specified condition is true.

```
while(condition) {
      Block of code          // statements to execute
                             // till condition is true
}
```

do...while loop:

do...while loop executes a block of code once, and then repeats the loop as long as the specified condition is true

```
do {
      Block of code          // statements to execute till
                             // condition is true. Executed once
                             // before checking condition.
} while(condition);
```

for loop:

for loop executes a block of code a specified number of times.

```
for(init counter; condition; increment/decrement counter)   {
      Block of code           // statements to execute
                              // till condition is true
}
```

foreach loop:

foreach loop executes a block of code for each element in an array.

```
foreach ($array as $val)  {
      Block of code           // statements to execute
                              // for each element in an array
}
```

break statement:

break statement is used to terminate the execution of a loop prematurely.

```
while(condition1) {
      Block of code           // statements to execute
                              // till condition1 is true
      if(condition2) {
            break;            // exit while loop is condition2
                              // is true
      }
}
```

continue statement:

continue statement is used to halt the current iteration of a loop and start next iteration of loop. It does not terminate the loop.

```
while(condition1) {

      if(condition2) {
            continue;         // if condition2 is true then
                              // remaining part of the loop will
                              //  not be executed in this
                              // iteration
      }
      Block of code           // statements to execute
                              // till condition1 is true
}
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Write a script that reads the name of the car and displays the name of the company the car belongs to as per the below table:

Car	Company
Safari, Nexon, Tigor, Tiago	Tata
XUV700, XUV300, Bolero	Mahindra
i20, Verna, Venue, Creta	Hyundai
Swift, Alto, Baleno, Brezza	Suzuki

2. Write a script to read the marks of 4 subjects and display the result as per the below instructions:

GTU GRADE	Mark-Range
AA	85 - 100
AB	75 - 84
BB	65 - 74
BC	55 - 64
CC	45 - 54
CD	40 - 44
DD	35 - 39
FF	< 35 (FAIL)

- Each of the four subjects is worth 100 marks.
- If a student gets less than 35 marks in any subject, then he/she will be marked as FAIL, otherwise he/she will be marked as PASS.

The result contains the grade of each individual subject in tabular format as per the above table.

3. Write a script to display Fibonacci numbers up to a given term.

4. Write a script to display a multiplication table for the given number.

J. References:

1. https://www.w3schools.com/php/php_if_else.asp
2. https://www.w3schools.com/php/php_looping.asp
3. https://www.geeksforgeeks.org/php-decision-making
4. https://www.geeksforgeeks.org/php-loops
5. https://www.tutorialspoint.com/php/php_decision_making.htm
6. https://www.tutorialspoint.com/php/php_loop_types.htm

Sign

Practical No. 5: Arrays

1]. Write a script to calculate the length of a string and count the number of words in the given string without using string functions.

2]. Write a script to sort a given indexed array.

3]. Write a script to perform 3 x 3 matrix Multiplication.

4]. Write a script to encode a given message into equivalent Morse code.

A. Objectives:

In PHP, array data structure allows user to store multiple elements of similar data type under a single variable. Array provide below advantages:

- No need to use multiple variables to store different data.
- Easy to traverse data in array using loops.
- Easy to sort data stored in array.

This practical will help student to practice writing PHP scripts using arrays.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop PHP scripts using arrays and functions.

E. Practical Outcomes:

1. Develop PHP scripts using one-dimensional, multi-dimensional and associative arrays.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

PHP array is a collection of similar data times stored in a single variable.It is basically an ordered map, which contains values on the basis of keys/indexes.PHP arrays allows traversing and processing of data items using a single loop. There are three types of arrays in PHP.

1. Indexed arrays
2. Associative arrays
3. Multi-dimensional arrays

Indexed/Numeric arrays

These type of arrays can store data of any type. They have integer indexes that start at zero by default. The Indexed array can be created as follow:

```
$subject_codes = array(4330701,4330702, 4330703, 4330704);
```

Or it can be assigned manually as below:

```
$subject_codes[0] = 4330701;
$subject_codes[1] =4330702;
$subject_codes[2] =4330703;
$subject_codes[3] =4330704;
```

Below example shows accessing data stored in the array:

```
$subject_codes = array(4330701, 4330702, 4330703, 4330704);
echo "Subject codes: <br>";
echo $subject_codes[0] . "<br>";
echo $subject_codes[1] . "<br>";
echo $subject_codes[2] . "<br>";
echo $subject_codes[3] . "<br>";
```

Another way to access all elements in the array using*for* loops is as below:

```
$subject_codes = array(4330701, 4330702, 4330703, 4330704);
echo "Subject codes: <br>";
for($i = 0; $i< count($subject_codes); $i++) {
      echo $subject_codes[$i] . "<br>";
}
```

We can access array data using *foreach* loop as below:

```
$subject_codes = array(4330701, 4330702, 4330703, 4330704);
echo "Subject codes: <br>";
foreach ($subject_codes as $code) {
      echo $code . "<br>";
}
```

Associative arrays

Associative arrays are similar to indexed arrays but instead of integer indexes values are assigned to user-defined keys of string type.Below example shows how to assign associative array:

```
$subjects = array("4330701" => "SLP",
                  "4330702" => "RDBMS",
                  "4330703" => "BOS",
                  "4330704" => "DSA");
```

Or it can be assigned manually as below:

```
$subjects["4330701"] = "SLP";
$subjects["4330702"] = "RDBMS";
$subjects["4330703"] = "BOS";
$subjects["4330704"] = "DSA";
```

Below example shows accessing data stored in the associative array:

```
$subjects = array("4330701" => "SLP",
                  "4330702" => "RDBMS",
                  "4330703" => "BOS",
                  "4330704" => "DSA");
echo "Subjects: <br>";
echo $subjects["4330701"] . "<br>";
echo $subjects["4330702"] . "<br>";
echo $subjects["4330703"] . "<br>";
echo $subjects["4330704"] . "<br>";
```

We can access associative array data using *foreach* loop as below:

```
$subjects = array("4330701" => "SLP",
                  "4330702" => "RDBMS",
                  "4330703" => "BOS",
                  "4330704" => "DSA");
echo "Subjects: <br>";
foreach ($subjects as $code => $sub) {
      echo $code . ": " . $sub . "<br>";
}
```

Multidimensional arrays

A multidimensional array is an array containing one or more arrays. PHP supports multidimensional array that are two or more levels deep. However, arrays more than three levels are hard to manage. Below is example of two dimensional array:

```
$subjects = array (
array("4330701","SLP",150),
array("4330702","RDBMS",150),
array("4330703","BOS",150),
array("4330704","DSA",150)
);
```

Below example shows a way to access all elements in the multidimensional array using *for* loops is as below:

```
$subjects = array (
array("4330701","SLP",150),
array("4330702","RDBMS",150),
array("4330703","BOS",150),
array("4330704","DSA",150)
);

for ($row = 0; $row < count($subjects); $row++) {
echo "Subject " . $row . ": <br>";
for ($col = 0; $col < count($subjects[0]); $col++) {
echo $subjects[$row][$col] . "<br>";
  }
}
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Write a script to calculate the length of a string and count the number of words in the given string without using string functions.

2. Write a script to sort a given indexed array.

3. Write a script to perform 3 x 3 matrix Multiplication.

4. Write a script to encode a given message into equivalent Morse code.

J. References:

1. https://www.php.net/manual/en/language.types.array.php
2. https://www.w3schools.com/php/php_arrays.asp
3. https://www.geeksforgeeks.org/php-arrays
4. https://www.tutorialspoint.com/php/php_arrays.htm
5. https://www.freecodecamp.org/news/how-to-use-arrays-in-php/

Sign

Practical No. 6: Functions

1]. Consider a currency system in which there are notes of 7 denominations, namely Rs. 1, Rs. 2, Rs. 5, Rs. 10, Rs. 20, Rs. 50 and Rs. 100. Write a function that computes the smallest number of notes that will combine for a given amount of money.

2]. Write scripts using string functions:

- **to check if the given string is lowercase or not.**
- **to reverse the given string.**
- **to remove white spaces from the given string.**
- **to replace the given word from the given string.**

3]. Write scripts using math functions:

- **to generate a random number between the given range.**
- **to display the binary, octal and hexadecimal of a given decimal number.**
- **to display the sin, cos and tan of the given angle.**

4]. Write a script to display the current date and time in different formats.

A. Objectives:

A function is a block of reusable code that is used to perform a specific action. Functions provide below advantages:

- Reduce duplication of the code.
- Modularisation of the code.
- Improve clarity of the code.
- Information hiding.

This practical will help student to practice writing PHP scripts using user defined functions and in-built functions.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop PHP scripts using arrays and functions.

E. Practical Outcomes:

1. Develop PHP scripts using in-built and user defined functions.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

A function is a block of code written in a program to perform some specific task. Functions take inputs as parameters, executes a block of statements or perform operations on these parameters and returns the result. There are two types of functions in PHP:

- **Built-in functions:**PHP provides large collectionof built-in library functions (more than 1000 functions). Whenever we need, we can just call these built-in functions as per our requirements.
- **User defined functions:** PHP allows us to create our own customised functions called user defined functions. We can create our own packages of code and use them whenever required.

User defined functions:

User defined functions are defined as below:

```
functionfunctionname() {
      Block of code;
}
```

Note:A function name must start with a letter or an underscore. Function names are NOT case-sensitive.

Below code is example of PHP function which prints a message on the browser.

```
<?php
functionwriteMessage() {
echo "Welcome to the PHP course..!!";
}

writeMessage (); // function call
?>
```

Function with arguments:

Data can be passed to functions through arguments. Arguments are specified inside the parentheses after the function name. There can be any number of arguments separated by comma.

```
<?php
functionprintSubjects($code, $subject) {
echo "Subject name for code $code is $subject <br>";
}

printSubjects("4330701","Scripting Language -Python");
printSubjects("4330702","Relational Database Management
System");
printSubjects("4330703","Basics of Operating System");
printSubjects("4330704","Data Structures and Algorithms");
?>
```

Arguments are by default passed by value, which means that a copy of the value is passed to the function, so the original variable that was passed into the function is not changed when we modify argument in the function. We can pass argument by reference, where changes made to the argument also change the original variable that was passed in. The & operator is used to pass variable by reference in argument. Below example show how to pass argument by reference:

```
<?php
functionincrementVar(&$var) {
  $var += 1;
return $var;
}
$a = 5;
$res = incrementVar($a);
echo "Value after increment is $res <br>";
?>
```

Below example shows how to use default argument:

```
<?php
functionareaofCircle($radius = 10) {
  $area = 2 * 3.14 * $radius;
echo "Area of Circle is : $area <br>";
}

areaofCircle();
areaofCircle(20);
?>
```

Function with returning value:

Function can return a value using *return* statements:

```
<?php
functionsumArray($arr) {
  $sum = 0;
for ($i = 0; $i< count($arr); $i++) {
    $sum += $arr[$i];
  }
return $sum;
}
$a = array(4, 9, 11, 25, 17);
$s = sumArray($a);

echo "Sum of array is: $s <br>";
?>
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Consider a currency system in which there are notes of 7 denominations, namely Rs. 1, Rs. 2, Rs. 5, Rs. 10, Rs. 20, Rs. 50 and Rs. 100. Write a function that computes the smallest number of notes that will combine for a given amount of money.

2. Write scripts using string functions:

- to check if the given string is lowercase or not.
- to reverse the given string.
- to remove white spaces from the given string.
- to replace the given word from the given string.

3. Write scripts using math functions:

- to generate a random number between the given range.
- to display the binary, octal and hexadecimal of a given decimal number.
- to display the sin, cos and tan of the given angle.

4. Write a script to display the current date and time in different formats.

J. References:

1. https://www.php.net/manual/en/language.functions.php
2. https://www.w3schools.com/php/php_functions.asp
3. https://www.geeksforgeeks.org/php-functions
4. https://www.tutorialspoint.com/php/php_functions.htm
5. https://zetcode.com/php/function

Sign

Practical No. 7: OOP Concepts

1]. Write a script to define a class with constructor and destructor.

2]. Create an object of a class and access its public properties and methods.

3]. Write a script that uses the set attribute and get attribute methods to access a class's private attributes of a class.

4]. Write a script to demonstrate single inheritance.

5]. Write a script to demonstrate multiple inheritance.

6]. Write a script to demonstrate multilevel inheritance.

7]. Write a script to demonstrate method overriding.

8]. Write a script to demonstrate method overloading based on the number of arguments.

9]. Write a script to demonstrate a simple interface.

10]. Write a script to demonstrate a simple abstract class.

11]. Write a script to demonstrate cloning of objects.

A. Objectives:

From PHP5, you can also write PHP code in an object-oriented style. Object-Oriented Programming (OOP) is a programming paradigm that emphasizes the use of objects and classes to structure code and data in a way that promotes code reuse, modularity, and maintainability. In PHP, OOP has several practical significances, including:

- Code organization
- Encapsulation
- Inheritance
- Polymorphism
- Modularity

Overall, OOP in PHP can make it easier to write more maintainable, extensible, and reliable code, which can ultimately lead to faster development times and fewer bugs.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
6. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop PHP scripts by applying object-oriented concepts.

E. Practical Outcomes:

1. Develop PHP scripts with the help of various object-oriented concepts like class, object, constructor, inheritance, interface, overloading and overriding.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

Before diving into implementing OOP concepts in PHP, it's important to have a solid understanding of the following concepts:

- **Classes and Objects:** A class is a blueprint for creating objects, while an object is an instance of a class. Classes define the properties and methods that objects will have.
- **Inheritance:** Inheritance is a mechanism that allows a new class to be based on an existing class, inheriting its properties and methods. This allows for code reuse and can make it easier to organize your code.
- **Encapsulation:** Encapsulation is the practice of hiding the implementation details of a class from the outside world, so that the class can only be accessed through its public interface. This helps to prevent unintended changes to the state of an object.

- **Polymorphism:** Polymorphism is the ability for objects of different classes to be used interchangeably. This is often achieved through the use of interfaces or abstract classes.
- **Abstraction:** Abstraction is the process of identifying the essential features of a concept, and ignoring the details that are not relevant. This is often achieved through the use of abstract classes or interfaces.
- **Access Modifiers:** Access modifiers are keywords that determine the visibility of properties and methods in a class. The three access modifiers in PHP are public, protected, and private.
- **Static Methods and Properties:** Static methods and properties belong to the class itself, rather than to a specific instance of the class. They can be accessed without creating an object.

By understanding these concepts, you'll have a solid foundation for implementing OOP in PHP. You can start by defining classes and creating objects, and then work your way up to using inheritance, encapsulation, polymorphism, and other advanced OOP concepts.

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code:

1. Write a script to define a class with constructor and destructor.

2. Create an object of a class and access its public properties and methods.

3. Write a script that uses the set attribute and get attribute methods to access a class's private attributes of a class.

4. Write a script to demonstrate single inheritance.

5. Write a script to demonstrate multiple inheritance.

6. Write a script to demonstrate multilevel inheritance.

7. Write a script to demonstrate method overriding.

8. Write a script to demonstrate method overloading based on the number of arguments.
9. Write a script to demonstrate a simple interface.
10. Write a script to demonstrate a simple abstract class.
11. Write a script to demonstrate cloning of objects.

J. References:

1. https://www.w3schools.com/php/php_oop_intro.asp
2. https://code.tutsplus.com/tutorials/object-oriented-php-for-beginners--net-12762
3. https://www.sitepoint.com/object-oriented-php-basics/
4. https://www.geeksforgeeks.org/object-oriented-programming-concepts-in-php/
5. https://www.php.net/manual/en/language.oop5.php
6. https://www.tutorialspoint.com/php/php_object_oriented.htm

Sign

Practical No. 8: Forms

1]. Create a web page using a form to collect employee information.

2]. Extend practical - 8(i) to validate user information using regular expressions.

3]. Create two distinct web pages to demonstrate information passing between them using URL - Get method.

4]. Create two different web pages to demonstrate information passing between web pages using Hidden variables - Post method.

A. Objectives:

In PHP, forms are a powerful tool for collecting data from users and processing that data on the server side. Here are some practical significances of using forms in PHP:

1. **User Input:** Forms allow users to enter information or data that can be processed by PHP scripts on the server. This can include simple text input, file uploads, or even complex data such as dates or email addresses.
2. **Data Validation:** Forms can be used to ensure that the data entered by users is valid and meets certain criteria. For example, a form can check that an email address is formatted correctly or that a password meets certain complexity requirements.
3. **Security:** Forms can be used to improve the security of PHP applications. For example, forms can be used to implement measures such as CAPTCHA to prevent automated attacks or CSRF tokens to prevent cross-site request forgery attacks.
4. **User Experience:** Forms can help to improve the user experience of PHP applications by allowing users to interact with the application in a more intuitive way. For example, forms can be used to allow users to search for specific content or to create new accounts.
5. **Data Processing:** Forms allow PHP scripts to process user data on the server side. This can include saving data to a database, sending email notifications, or performing complex calculations.

Overall, forms are a fundamental part of PHP development and are an essential tool for building interactive, user-friendly, and secure web applications.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.

4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
6. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop web pages using form controls with validation to collect user inputs in PHP.

E. Practical Outcomes:

1. Create PHP scripts with the use of various form elements, perform validation and implement form processing.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

Forms handling in PHP involves collecting data submitted by users through HTML forms, processing that data using PHP scripts, and then taking actions based on that data. Here are the steps involved in forms handling in PHP:

1. **Creating an HTML form:** First, create an HTML form that users can fill out. This form should include input fields like text boxes, radio buttons, and checkboxes, and it should have a "submit" button that sends the data to a PHP script.
2. **Setting the "action" attribute:** In the form tag, you should set the "action" attribute to the filename of the PHP script that will process the data. When the user submits the form, the data will be sent to this script for processing.
3. **Retrieving form data:** In the PHP script that will process the form data, you should use the $_POST superglobal to retrieve the data submitted by the user. The $_POST array contains key-value pairs, where the key is the name of the form field and the value is the data entered by the user.

4. **Validating form data:** Before processing the form data, you should validate it to ensure that it is in the correct format and that it contains the required data. You can use PHP's built-in functions for data validation, or you can create custom validation functions.
5. **Sanitizing form data:** After validating the form data, you should sanitize it to prevent malicious input, such as SQL injection attacks. You can use PHP's built-in functions for data sanitization, such as htmlspecialchars() or filter_var().
6. **Processing form data:** Once the form data has been validated and sanitized, you can process it according to your application's requirements. For example, you might insert the data into a database, send an email, or redirect the user to a different page.
7. **Displaying feedback:** After the form data has been processed, you should provide feedback to the user. This might include a message confirming that their data was successfully submitted or an error message if the data could not be processed.

In summary, forms handling in PHP involves creating an HTML form, processing the data submitted through the form using a PHP script, validating and sanitizing the data, and then taking actions based on that data. By following these steps, you can create robust and secure web applications that can handle a wide range of user inputs.

ExampleHTML Form:

```
<!DOCTYPE html>
<html>
<head>
	<title>Registration Form</title>
	</head>
<body>
<h2>Registration Form</h2>
<form action="register.php" method="post">
	<label for="username">Username:</label>
	<input type="text" name="username" id="username"><br>

	<label for="email">Email:</label>
	<input type="email" name="email" id="email"><br>

	<label for="password">Password: </label>
	<input type="password" name="password" id="password"><br>

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

The form has three input fields for the user to enter their desired username, email, and password. The form's action attribute is set to register.php, which is the PHP script that will handle the form data. The method attribute is set to post, indicating that the form data will be sent as a POST request.

PHP Script (register.php):

```
<?php
	// Retrieve form data
	$username = $_POST['username'];
	$email = $_POST['email'];
	$password = $_POST['password'];

	// Validate form data
	if (empty($username) || empty($email) || empty($password))
	{
		echo "Please fill in all fields.";
		exit;
	}

	// Sanitize form data
	$username = htmlspecialchars($username);
	$email = filter_var($email, FILTER_SANITIZE_EMAIL);
	$password = htmlspecialchars($password);

	// Process form data (in this example, just display it)
	echo "Thank you for registering!<br>";
	echo "Your username is: $username<br>";
	echo "Your email is: $email<br>";
	echo "Your password is: $password<br>";
?>
```

The PHP script retrieves the form data using the $_POST superglobal and then validates and sanitizes the data to prevent malicious input. In this example, the script checks that all fields are filled in and uses the htmlspecialchars() and filter_var() functions to sanitize the data.

Finally, the script processes the form data by displaying it back to the user. In a real-world scenario, you might store the form data in a database or send it in an email.

That's it! This is a simple example of a registration page in PHP.

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Create a web page using a form to collect employee information.
2. Extend practical - 8(i) to validate user information using regular expressions.
3. Create two distinct web pages to demonstrate information passing between them using URL - Get method.
4. Create two different web pages to demonstrate information passing between web pages using Hidden variables - Post method.

J. References:

1. https://www.w3schools.com/php/php_forms.asp
2. https://www.geeksforgeeks.org/php-form-handling/
3. https://www.tutorialspoint.com/php/php_forms.htm
4. https://www.php.net/manual/en/tutorial.forms.php
5. https://www.tutorialrepublic.com/php-tutorial/php-form-validation.php

Sign

Practical No. 9: Session, Cookies

1]. Create web pages to demonstrate passing information using Session.

2]. Write a script to demonstrate storing and retrieving information from cookies.

A. Objectives:

Sessions and cookies are important concepts in web development, and they have practical significance in PHP in several ways:

1. **User authentication:** Sessions and cookies can be used to authenticate users on a website. When a user logs in, their session ID can be stored in a cookie, and the server can verify the ID to ensure that the user is authorized to access certain pages or features.
2. **Personalization:** Sessions and cookies can be used to personalize the user experience on a website. For example, a session can be used to store the user's preferred language or theme, and a cookie can be used to remember the user's login credentials for future visits.
3. **Shopping carts:** Sessions and cookies can be used to create and manage shopping carts on an e-commerce website. The contents of the user's cart can be stored in a session, and a cookie can be used to remember the user's cart across multiple visits.
4. **Tracking user behaviour:** Cookies can be used to track user behaviour on a website, such as which pages they visit and which links they click. This information can be used to improve the website's usability and performance.
5. **Security:** Sessions and cookies can be used to enhance the security of a website. For example, a session can be used to prevent CSRF (cross-site request forgery) attacks, while cookies can be used to prevent XSS (cross-site scripting) attacks.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

6. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop web pages using form controls with validation to collect user inputs in PHP.

E. Practical Outcomes:

1. Implement session and cookie to store and manage user data.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

In PHP, sessions and cookies are two mechanisms used to store and manage data related to a user's interaction with a website or web application.

SESSION

Sessions are a way of storing information on the server about a user's activity on a website. When a user visits a website, the server assigns them a unique session ID, which is stored in a cookie on the user's computer. The server then uses this session ID to keep track of the user's activity as they move around the website. The session data can include things like user preferences, shopping cart items, or authentication status. Sessions can be started using the session_start() function and the session data can be accessed through the $_SESSION superglobal array.

COOKIE

Cookies are small text files that are stored on a user's computer by their web browser. Cookies are often used to remember user preferences or login information, or to track user behavior across different pages or sessions. In PHP, cookies can be set using the setcookie() function. Cookies can have an expiration time, after which they are automatically deleted, or they can be set to expire when the user closes their browser. Cookie data can be accessed using the $_COOKIE superglobal array.

COMPARISION

Both sessions and cookies have their advantages and disadvantages. Sessions are generally more secure because the data is stored on the server, but they can be slower and less flexible. Cookies are faster and more flexible, but they can be less secure because the data is stored on the user's computer. When using cookies, it's important to make sure that sensitive information, such as passwords or credit card numbers, is not stored in the cookie.

In summary, sessions and cookies are two important mechanisms used to store and manage user-related data in PHP. By understanding how to use these mechanisms correctly, developers can create more secure and user-friendly web applications.

Examples

Using Session

1. Start the session using the session_start() function.

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

2. Set a session variable.

```
<?php
$_SESSION[‘username’] = ‘Ashish’;
>
```

3. Access the session variable on another page.

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

Using Cookie

1. Set a cookie using the setcookie() function.

```
<?php
$cookie_name "username";
$cookie_value = "Ashish";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/");
?>
```

2. Access the cookie value.

```
<?php
if(isset($_COOKIE [$cookie_name])) {
      echo "Welcome". $_COOKIE [$cookie_name];
}
?>
```

Note that when using sessions and cookies, it's important to be careful about what data is stored and how it's accessed, to prevent security issues. For example, sensitive information like passwords or credit card numbers should never be stored in a cookie or session.

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Create web pages to demonstrate passing information using Session.
2. Write a script to demonstrate storing and retrieving information from cookies.

J. References:

1. https://www.php.net/manual/en/features.sessions.php
2. https://www.php.net/manual/en/function.setcookie.php
3. https://www.tutorialspoint.com/php/php_sessions.htm
4. https://www.tutorialspoint.com/php/php_cookies.htm
5. https://www.w3schools.com/php/php_sessions.asp
6. https://www.w3schools.com/php/php_cookies.asp
7. https://www.geeksforgeeks.org/php-sessions/
8. https://www.geeksforgeeks.org/php-cookies/

Sign

Practical No. 10: Database

1]. Create a web page that reads employee information using a form and stores it in the database.

2]. Create a web page for employee log-in.

3]. Write a script to upload an image to the server.

4]. After an employee log in, create a home web page that displays basic employee information.

5]. Create a web page to delete employee profiles from the database.

6]. Create a web page that allows employees to change their password.

A. Objectives:

Databases are an essential part of many web applications developed with PHP. Here are some practical reasons why databases are important in PHP development:

1. **Data storage:** Databases provide a structured way to store and organize data for PHP applications. This makes it easy to manage and retrieve data using SQL queries.
2. **Scalability:** Databases can handle large amounts of data and high traffic volumes, making it a great choice for PHP applications that need to grow and scale over time.
3. **Security:** Databases provide strong security features to protect data and prevent unauthorized access. This includes user authentication, data encryption, and role-based access control.
4. **Data analysis:** Databases can be used to perform complex data analysis and generate reports. This is useful for PHP applications that need to process large amounts of data and extract meaningful insights.
5. **Data synchronization:** Databases can be used to synchronize data between different systems and applications. This is useful for PHP applications that need to integrate with other systems and exchange data in real-time.

Overall, databases are an essential tool for PHP developers, providing reliable data storage, scalability, security, data analysis, and data synchronization. They play a crucial role in many web applications, and their importance cannot be overstated.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.

3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
6. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop and host interactive websites using PHP and MySQL database.

E. Practical Outcomes:

1. Implement various database operations using PHP script.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

MySQL is a popular relational database management system that is commonly used in web development with PHP. Here are some MySQL functions that can be used in PHP to interact with MySQL databases:

1. **mysqli_connect()** - This function is used to establish a connection to a MySQL database server.
2. **mysqli_query()** - This function is used to execute a MySQL query on a connected database.
3. **mysqli_fetch_array()** - This function is used to fetch the result of a MySQL query as an array.
4. **mysqli_insert_id()** - This function is used to get the auto-generated ID of the last inserted record in a table.

5. **mysqli_real_escape_string()** - This function is used to escape special characters in a string to prevent SQL injection attacks.
6. **mysqli_num_rows()** - This function is used to get the number of rows returned by a MySQL query.
7. **mysqli_error()** - This function is used to get the error message associated with the last MySQL operation.
8. **mysqli_close()** - This function is used to close the connection to a MySQL database server.

These are just a few examples of the MySQL functions that can be used in PHP. There are many other functions available for working with MySQL databases in PHP, and their usage depends on the specific needs of your application.

Example

Assuming we have a MySQL database with a table named "users" that has columns "id", "username", and "password", we can use PHP to perform various operations on this table.

1. Connect to the MySQL database:

```
<?php
$host = "localhost"; $username = "db_user";
$password = "db_password";
$dbname = "my_database";

// Create connection
$conn = mysqli_connect($host, $username, $password, $dbname);

// Check connection
if (!$conn) {
      die("Connection failed: mysqli_connect_error());
}
echo "Connected successfully";
?>
```

2. Insert a new user into the "users" table:

```
<?php
$username = "john_doe";
$password = "my_password";

// Escape special characters in the username and password to
// prevent SQL injection
$username = mysqli_real_escape_string($conn, $username);
$password mysqli_real_escape_string($conn, $password); =

// Create and execute the SQL query to insert a new user
$sql = "INSERT INTO users (username, password) VALUES
('$username', '$password')";
if (mysqli_query($conn, $sql)) {
      echo "New record created successfully";
}
else {
      echo "Error: " . $sql . "<br>" . mysqli_error($conn);
}
?>
```

3. Retrieve all users from the "users" table:

```
<?php
// Create and execute the SQL query to retrieve all users
$sql = "SELECT * FROM users";
$result = mysqli_query($conn, $sql);

// Check if any rows were returned
if (mysqli_num_rows($result) > 0) {
      // Output data of each row
      while($row = mysqli_fetch_assoc($result)) {
            echo "id: ". $row["id"]. " - Username:" .
                              $row["username"]. "<br>";
      }
}
else {
      echo "0 results";
}
?>
```

4. Update the password for a specific user in the "users" table:

```
<?php
$username "ashish";
$new_password = "new_password";

// Escape special characters in the username and new password to
// prevent SQL injec
$username = mysqli_real_escape_string($conn, $username);
$new_password = mysqli_real_escape_string($conn, $new_password);

// Create and execute the SQL query to update the password for a
// specific user
$sql = "UPDATE users SET password='$new_password' WHERE
username='$username'
if (mysqli_query($conn, $sql)) {
      echo "Record updated successfully";
}
else {
      echo "Error updating record: mysqli_error($conn);
}
?>
```

5. Delete a specific user from the "users" table:

```
<?php
$username "ashish";

// Escape special characters in the username to prevent SQL
// injection
$username = mysqli_real_escape_string($conn, $username);

// Create and execute the SQL query to delete a specific user
$sql = "DELETE FROM users WHERE username='$username'
if (mysqli_query($conn, $sql)) {
      echo "Record deleted successfully";
}
else {
      echo "Error deleting record: mysqli_error($conn);
}
?>
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Create a web page that reads employee information using a form and stores it in the database.
2. Create a web page for employee log-in.
3. Write a script to upload an image to the server.
4. After an employee log in, create a home web page that displays basic employee information.
5. Create a web page to delete employee profiles from the database.
6. Create a web page that allows employees to change their password.

J. References:

1. https://www.php.net/manual/en/book.mysql.php
2. https://www.w3schools.com/php/php_mysql_intro.asp
3. https://www.tutorialspoint.com/php/php_mysql.htm
4. https://phpdelusions.net/mysqli
5. https://www.geeksforgeeks.org/mysqli-procedural-functions
6. https://www.guru99.com/mysql-php-and-other-database-access-methods.html

Sign

Practical No. 11: Email, PDF, JSON

1]. Write a script to generate a salary slip for an employee in PDF format.

2]. Write a script to send an email.

3]. Write a script to convert an associative array into JSON string format and vice versa.

A. Objectives:

Email, PDF, and JSON are three distinct types of data formats that can be used in PHP for various practical purposes.

1. **Email:** PHP provides built-in email functions that allow developers to send and receive emails using different protocols such as SMTP, IMAP, and POP3. Email is widely used for communication purposes in various web applications, such as sending verification emails, password reset links, newsletters, and order confirmations.
2. **PDF:** PDF is a popular file format used for creating and sharing documents that can be viewed on different devices and platforms. PHP provides various libraries and tools for generating and manipulating PDF documents, such as FPDF, TCPDF, and mPDF. PDFs are often used for generating invoices, receipts, reports, and other types of documents in web applications.
3. **JSON:** JSON (JavaScript Object Notation) is a lightweight data interchange format that is easy to read and write for humans and machines. JSON is often used for exchanging data between client-side and server-side applications using AJAX (Asynchronous JavaScript and XML) and RESTful (Representational State Transfer) APIs. PHP provides built-in functions for encoding and decoding JSON data, such as json_encode() and json_decode().

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.

6. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.

D. Relevant Course Outcomes (COs):

1. Develop and host interactive websites using PHP and MySQL database.

E. Practical Outcomes:

1. Implement Email, PDF and JSON facility for data exchange and document generation using PHP script.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Prerequisite Theory:

Email

Email is a common communication tool used in web development, and PHP provides built-in functions and libraries to send and receive emails. Here are the basic steps to send an email in PHP.

- Configure the SMTP server settings
- Set up the email headers
- Compose the email message

Example

```
$to = 'recipient@example.com';
$subject = 'Test email';
$message = 'Hello, this is a test email!';

$headers = 'From: sender@example.com' . "\r\n" .
      'Reply-To: sender@example.com' ."\r\n".
      'MIME-Version: 1.0' . "\r\n" .
      'Content-type: text/html; charset=iso-8859-1' . "\r\n".
      'X-Mailer: PHP/'. phpversion();

mail($to, $subject, $message, $headers);
```

PDF

PDF (Portable Document Format) is a commonly used document format that allows for easy sharing and printing of documents. PHP provides several libraries and functions that allow for the creation and manipulation of PDF documents. Here are the basic steps to create a PDF document in PHP:

- Install a PDF library: There are several PDF libraries available for PHP, such as FPDF, TCPDF, and DOMPDF. You need to install and configure a PDF library to generate the PDF document.
- Create a PDF object: Once you have installed a PDF library, you can create a PDF object to represent the document. This object contains methods and properties for adding content to the PDF document, such as text, images, tables, and other elements.
- Add content to the PDF document: You can use the methods provided by the PDF library to add content to the PDF document. For example, you can use the SetFont() method to set the font style, Cell() method to add text to the document, and Image() method to add images to the document.
- Output the PDF document: Once you have added all the content to the PDF document, you can output the document to the browser or save it to a file. The PDF library provides methods to output the PDF document in various formats, such as PDF, HTML, or image.

Here is an example code snippet to create a PDF document using the FPDF library in PHP:

```
<?php
require('fpdf/fpdf.php');

$pdf new FPDF(); =
$pdf->AddPage();
$pdf->SetFont('Arial','B',16);
$pdf->Cell(40,10, 'Hello World!');
$pdf->Output();
?>
```

JSON

JSON (JavaScript Object Notation) is a lightweight data interchange format that is commonly used for exchanging data between web applications. PHP provides several built-in functions and libraries to encode and decode JSON data. Here are the basic steps to work with JSON data in PHP:

Encode PHP data into JSON format: To encode PHP data into JSON format, you can use the json_encode() function. This function converts a PHP object or array into a JSON string.

```
$person = array(
      "name" => "John",
      "age" => 30,
      "city" => "New York"
);

$json = json_encode($person);
echo $json;
```

In this example, we have defined a PHP array person and encoded it into a JSON string using the json_encode() function. The output of the code will be:

```
{"name":"John","age":30,"city":"New York"}
```

H. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

I. Source code and Output:

1. Write a script to generate a salary slip for an employee in PDF format.
2. Write a script to send an email.
3. Write a script to convert an associative array into JSON string format and vice versa.

J. References:

1. https://www.php.net/manual/en/function.mail.php
2. https://www.tutorialspoint.com/php/php_sending_emails.htm
3. https://www.sitepoint.com/generate-pdfs-php/
4. https://www.geeksforgeeks.org/how-to-generate-pdf-file-using-php/
5. https://www.php.net/manual/en/function.json-encode.php
6. https://www.w3schools.com/js/js_json_php.asp

Sign

Practical No. 12: Simple Web Application

1]. Create a simple web application for Employee Management with 3-4 web pages and host it using cPanel and Filezilla.

A. Objectives:

Development of simple web application as mini-project will give an experience of problem-solving along with group members, by using knowledge and under the guidance of a faculty. Mini-project will help students to develop different skills such as: team work, researching on a topic, problem solving, time management, planning, code development, testing and documentation.

B. Relevant Program Outcomes (POs):

1. **Basic and Discipline specific knowledge (PO1):** Apply knowledge of basic mathematics, science and engineering fundamentals and engineering specialization to solve the Computer Engineering problems.
2. **Problem analysis (PO2):** Identify and analyse well-defined Computer Engineering problems using codified standard methods.
3. **Design/development of solutions (PO3):** Design solutions for Computer Engineering well-defined technical problems and assist with the design of systems components or processes to meet specified needs.
4. **Engineering Tools, Experimentation and Testing (PO4):** Apply modern Computer Engineering tools and appropriate technique to conduct standard tests and measurements.
5. **Project Management (PO6):** Use engineering management principles individually, as a team member or a leader to manage projects and effectively communicate about well-defined engineering activities.
6. **Life-long learning (PO7):** Ability to analyse individual needs and engage in updating in the context of technological changes in field of engineering.

C. Competency and Practical Skills:

This practical is expected to develop the following skills for the industry-identified competency '**Develop Interactive Web application using PHP and MySQL**':

1. Programming skills.
2. Debugging skills.
3. Project management skills.

D. Relevant Course Outcomes (COs):

1. Develop PHP scripts using variables, operators and control structures.
2. Develop PHP scripts using arrays and functions.
3. Develop PHP scripts by applying object oriented concepts.

4. Develop web pages using form controls with validation to collect user inputs in PHP.
5. Develop and host interactive websites using PHP and MySQL database.

E. Practical Outcomes:

1. Develop simple application using PHP and MySQL.

F. Relevant Affective domain Outcomes (ADOs):

1. Maintain tools and equipments.
2. Follow Coding standards and practices.
3. Follow ethical practices.

G. Resources Required:

Sr. No	Instrument /Components	Configuration/Specification
1.	Computer System	Processor: RAM: Operating System:
2.	XAMPP server	XAMPP Version:
3.	Text Editor	Editor:
4.	Web Browser	Browser:

H. Source code and Output: