Date :

# Experiment → 01

Aim → Install & configure PHP execution Environment

Theory →

What is PHP?

↳ PHP is an acronym of Hypertext preprocessor.

↳ PHP is widely used, open source scripting language.

↳ PHP scripts are executed on server.

↳ PHP is free to download & use.

What PHP can do?

↳ PHP can generate dynamic page content.

↳ PHP can collect form data.

↳ PHP can send/Receive cookies.

↳ PHP can manage DBs.

↳ PHP can encrypt data.

↳ PHP can control user access.

## Why PHP?

↳ It is platform independent.
↳ PHP is compatible with almost all servers.
↳ It supports wide range of DBs.
↳ Ease of Use.

## Installation

To start using PHP, you can:
1. Find a web host with PHP and MySQL support.
2. Install a Web server on your PC.

↳ If your server has activated support for PHP, you do not need to do anything.
Just create some .php files & place them in a web directory & the server will parse it.

↳ We do not need to compile anything or install any extra tools.

↳ However, If your server does not support PHP, then you must;

1. Install a Web server.
2. Install PHP.
3. Install a DB

Date :

Conclusion → Thus We have setup an environment to run PHP programs using XAMPP.

G.B. Shirsulare
12/03/20

Date :

# Experiment → 02

Aim → Perform different file manipulating operations on files & directories.

Theory →

↳ File handling is an important part of any web application. You often need to open and process a file for different tasks.

↳ PHP has several functions for creating, Reading, uploading & editing files.

File Handling Functions:

1. fopen( ) → Used to open the specified files.

Ex. $fp = fopen("a.txt", "r");

2. fread( ) → Used to read the content from specified file.

Ex. fread($fp, "a.txt");

3. fclose( ) → Used to close an open file.

Ex. fclose($fp);

4. fgets( ) → Used to read a single line from a file.

Ex. echo fgets($fp);

5. feof( ) → It checks whether the end-of-file has been reached.

Ex. while(! feof($fp)) {
    echo fgets($fp);
}

6. fgetc( ) → This function is used to read a single function from a file.

Date :

7. fputc() → Used to write the given content to a file.

Eg. fputc(ch, fp);

8. unlink() → Used to delete a file.

Eg. unlink("a.txt");

Date :

Conclusion →

Thus we have performed various file operations on a file using various PHP functions.

Date :

# Experiment → 03

Aim → Perform different time, date and string functions.

Theory →

↳ PHP string functions are part of the PHP core. No installation is required to use these functions.

↳ Following are some string functions;

1. strlen() → Returns length of string.
   Ex. echo strlen("abc");

2. str_word_count() → Returns No. of words in a given strings.
   Ex. str_word_count("abc");

Date :

4. strrev() → It reverses the given string.

Eg. strrev("fuki");

5. strpos() → The strpos() fn searches for a specific text within a string. If match found it returns position else False.

Eg. strpos("abc", "a");

Date and Time;

1. date() → It formats a timestamp to a more readable date and time.

Eg. date("Y/m/d", );

2. date_default_timezone_set("America/New_York");

3. mktime() → Returns a Unix timestamp for a date.

Date :

5. strtotime() → Convert a human readable string to a Unix timestamp.

Ex. strtotime("10:30pm April 15 2020");

6. time() → Returns current time as Unix timestamp.

Ex. time();

Page No. ( )

Date :

Conclusion → Thus we have executed various date, time & string functions in PHP.

12/03/20.

Date :

# Experiment → 04

Aim → Implementation of different types of arrays and its methods.

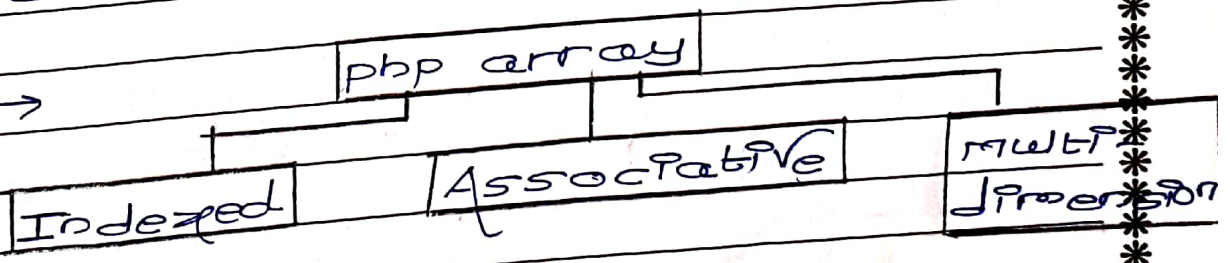
Theory →

↳ An array is a data structure that stores multiple values in one single variable.

↳ An array is a special variable which can hold more than one value at a time.

↳ array() fn is used to create arrays in php.

↳




↳ Loops can be used to handle arrays.

Date :

1. Indexed Arrays;

```
$ab = array(1, 2, 3);
for ($i=0; $i<3; $i++)
    echo $ab[$i];
```

2. Associative Arrays;

```
$ab = array("1" => 1, "2" => 2);
echo $ab["1"];
```

3. Multidimensional Arrays;

Ex.
```
$ab = array(
        array(1, 2, 3),
        array(4, 5, 6)
      );
```

FUNCTIONS;

1. array_chunk() → split array into chunks.

2. array_combine() → combines as array.

3. array_diff() → compare &

Date :

Differences between arrays.

3. array_fill() → Fill an array with values.

4. array_merge() → Merge two or more arrays to one.

5. array_pop() → Returns the last element of array.

6. array_reverse() → Reverses an array.

Date :

Conclusion →

Thus we have seen types of PHP arrays & proplemented its methods.

# Experiment → 05

**Aim →** Execute sql commands and stored procedures.

**Theory →**

↳ sql is "structured query language", which is mainly used to perform Database operations.

↳ The php allows us to manage Databases. php can be served as Backend tool that interacts with DBs.

↳ Whenever an action is performed on client site, it generates a Request to server.

↳ We can use php to handle these requests. php can receive request from client and accordingly perform actions on DB if needed.

↳ Following are some types of sql commands;

1. DML
2. DDL
3. DCL

↳ Fortunately php provides support to perform all above commands.

↳ Following are some methods are used to DB-php code;

1. connect() → create connection with DB.

2. mysqli_query() → To hit any sql query

other methods;

1. mysqli_connect();
2. mysqli_fetch_assoc();
3. mysqli_fetch_row();

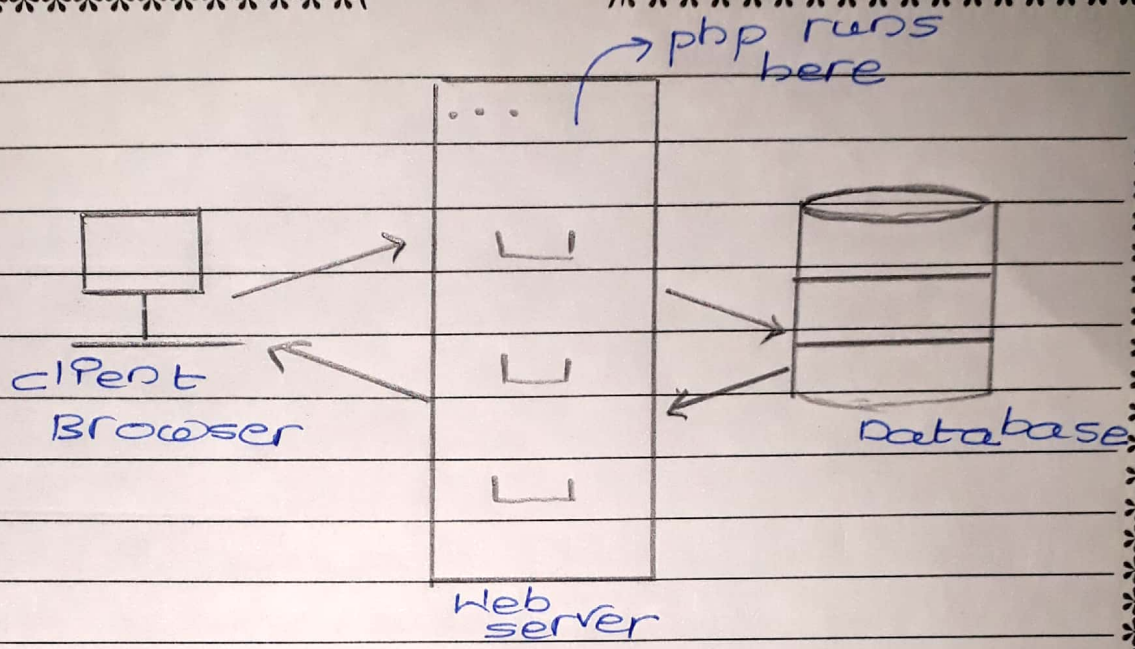



Fig. 3 tier Architecture

Conclusion → thus we have written a php code to perform sql operations.

Page No. ( )

# Experiment → 06

Aim → Implementation of cookies in php.

Theory →

↳ Cookies are one of the most important part of web development.

↳ A cookie is often used to identify a user. A cookie is a small file that the server embeds on the user's computer.

↳ Each time a same computer requests a page with browser, it will send the cookie along with page.

↳ A cookie is created with the setcookie() function.

↳ Syntax;

setcookie(name, value, ....)

↳ only the name parameter is required. All other parameters are optional.

↳ To check whether the cookie is set or not we can use isset() method.

Ex. isset($_COOKIE['name'])

↳ To modify a cookie value, just set it using setcookie().

Ex. setcookie($name);

↳ We can delete a cookie using setcookie() method with past date.

Ex. setcookie("user", "", time());

↳ cookie will expire after time() value.

Conclusion →

Thus we have implemented the cookies Login system. A system that stores cookies to Browser.

# Experiment - 07

Aim → Session management in php.

Theory →

↳ A session is a way to store information (in variables) to be used across multiple pages.

↳ When you work with an appl? you open it, do some changes, and then you close it. This is much like a session.

↳ The computer knows who you are. It knows when you start the application and when you end. But on the internet there is one problem.

↳ The Web server does not know who youare or what you do, because the HTTP address does not maintain state.

↳ session variables solve this problem by storing user information to be used across multiple pages.

↳ the session variables can last until you close browser.

↳ To start session use;
session_start()

The session is set using the $_SESSION[] global variable.

↳ To destroy a session use;
session_destroy()

↳ When you logout your system, the session variables should be deleted.

↳ Methods;
1. session_abort()
2. session_commit()
3. session_decode()
4. session_gc()

5. session_reset( )
6. session_register( ).

Conclusion →

Thus we have created a login system using cookies and sessions.

# Experiment → 08

**Aim →** Error handelling mechanism in php.

**Theory →**

↳ Error is simply a fault or the mistake in a program. It can be of several types.

↳ Error can occur due to wrong syntax or wrong logic. It is a types of mistakes caused due to improper knowledge.

↳ php error handelling ~~is~~ will give us the file name, line no. and error type.

↳ Ways to handle php errors;

1. Using die() method.
2. Custom error handeling.

↳ die() will exit php program.

1. die() →

Syntax: die($msg)

Ex. <?php
$file = fopen("g.txt","w")
?>

↳ creating a custom error handler in php is quite simple. We will create a function that can be called in php.

Syntax: error_function($params)

Ex.
<?php
function err($erno, $msg){
echo $erno, $msg;
die();
}
?>

↳ Here, err() is a custom error function which performs the specified action when the error is encountered.

Conclusion →

Thus we have created user defined errors in php.

# Experiment → 09

Aim → Exception handeling mechanism in php.

Theory →

↳ Exceptions are used to change the normal flow of a script if a specified error occurs.

↳ Exception is simply an error caused during execution of the program.

↳ With php 5 came a new object oriented way of dealing with Errors. Exception handeling is used to change the normal flow of code execution if errorneous condit° occurs.

↳ Exception should only be used with error conditions and should not be used to change control flow.

↳ This is what normally happens when an exception is triggered.

1. The code state is saved.
2. The code execution will switch to error handeler.

Ex.

```
<? php
    $number = $_POST['num'];
    if ($number < 0)
        throw new Exception ();
?>
```

↳ Try: This block is used to write a code that can cause an exception.

↳ catch: When exception is raised it will be handeled by code in catch block.

↳ throw: Throws object of Exception.

↳ We can also create user defined exception classes.

Conclusion →

thus we have seen built-in as well as user defined function exceptions in php.

# Experiment → 10

Aim → Perform AJAX to send data and retrieve from DB.

Theory →

↳ ajax is the art of exchanging data with server, update parts of web page - without reloading the whole page.

↳ Following table consist all the jQuery ajax methods;

| Method | Description. |
|---|---|
| $.ajax() | performs an Async ajax request. |
| $.ajaxprefilter() | Handle custom ajax options. |
| $.ajaxsetup() | Sets the default values for future ajax reques |

| | |
|---|---|
| $.get() | Loads the data from server get request. |
| $.getJSON() | Loads JSON encoded data from server. |
| $.parseJSON() or $.parse() | Take JSON & returns JS object. |
| $.post() | make a post request |
| ajaxComplete() | Specify success function. |
| ajaxError() | Specify error function. |
| ajaxSend() | run fn before ajax request. |
| load() | load data from the server. |
| serialize() | Encodes a set of form elements as string. |

Conclusion →

Thus we have sent an ajax request to DB & retrieved data using ajax.

# Experiment → II

App ⇒ Create email with file attachment using mail().

Theory →

↳ The mail() function allows you to send emails directly from script.

Syntax →
mail(to, subject, msg, headers, parameters)

↳ mail() function is a traditional approach to send mails. The modern PHP allows us to send mails using PHPmailer class causing reduction in code and complexity.

↳ We have to configure our localhost server inorder to send the emails.

## Parameter Values:

| Parameter | Description. |
|---|---|
| to | Required. Recepient's email address. |
| subject | Required. Subject of Email. |
| message | Required. Specifies the message to be sent. |
| headers. | Optional. Specifies cc, Bcc, CRLF, etc |
| parameters. | optional. |

Ex.
```
<? php
$to = "abc@gmail.com";
$sub = "Test mail";
mail($to, $sub, 'Hey, there!!');
?>
```

Conclusion → Thus we have sent an email with attachment using mail() function.

# Experiment → 12

Aim → Validating and sanitizing data in filters.

Theory →

↳ Sanitizing & validating the user input is one of the most common tasks in a web application. To make this task easier PHP provides a native filter extension that can you use to sanitize & filter data in php.

↳ We have some built-in filter extension you need to use ~~the~~ such as email, URLs.

↳ To validate data using filter extension you need to use; filter_var ().

↳ this fn has 3 parameters out of which the last two are optional.

↳ The 1st parameter is a value to be filtered, second parameter is the ID of the filter to apply, and 3rd parameter is the array of options related to filter.

Ex.

```
<? php
    $msg = "<h1> Hey </h1>";
    $sanitize = filter_var ($msg,
                    FILTER_SANITIZE);
    echo $ sanitize;
?>
```

We can use FILTER_VALIDATE_INT to validate integers.

FILTER_VALIDATE_Ip can be used to validate Ipv4 addresses.

FILTER_VALIDATE_Ip is used for Ipv6 addresses.

and We also have methods for Email & URL.

Conclusion →

Thus we have validat sanitized form data before form submission to php.