Roll.no = 02

Subject : OSWD

Page No. 1

Date. 15 | 10 | 2023

Assignment - 3.

1)# PHP Superglobal arrays.

- Superglobal arrays in PHP are predefined arrays that are accessible in all scopes throughout a script
- Super global array includes:- '$_GET',
  '$_POST',
  '$_SESSION',
  '$_COOKIE',
  '$_FILES',
  '$_SERVER',
  '$_REQUEST'.

Example:-

```
<? php

// Accessing GET parameters.
$param_value = $_GET['param_name'];

// Accessing POST parameters
$param_value = $_POST['param_name'];

// Accessing session variables
session_start();
$_SESSION['user_id'] = 123;

// Accessing server information.
echo $_SERVER['HTTP_HOST'];
```

2)

Page No. 2.
Date. 15 | 10 | 23

2 * PHP Fimction for string array files, image manip CAPTCHA.

* String
- PHP provides several Functions for string manip including 'strlen', 'substr', 'str_replace', 'strlower', 'str

Example

```
<?php
$string = "Hello, world!";

// Get string length
echo strlen($string); // o/p : 13

// Extract a substring
echo substr($string, 0, 5); // o/p: Hello.

// Replace a substring
echo str_replace("Hello", "Hi", $string); // o/p: Hi, wo

// Convert lowercase to uppercase
echo strtolower($string); // o/p: hello, world!
echo strtoupper($string); // o/p: HELLO, WORLD!
```

* Array :-
- PHP provides several Functions for array manipulati including 'array_push', 'array_pop', 'array_merge', 'array_ 'array_map', etc.

Example:-

```
<? php
$my-array = [1,2,3];

// Add an element to the end of the array.
array_push ($my_array, 4);
print_r($my_array);                    // o/p : [1,2,3,4]

// Remove and return the last element

$last-element = array_pop ($my_array);
print_r ($my-array);                   //  o/p :- [1,2,3]

// merge arrays
$ another_array = [4,5];
$ merged_array = array_merge ($my_array, $another_array)
print_r ($merged_array);               // output : [1,2,3,4,5]

?>
```

* Files :-

- PHP provides Functions for various file handling operations includes reading, writing, copying, moving, deleting etc.

Example:-

```
<? php
// Read file content
$file_content = file_get_contents ('myfile.txt');
echo $file_content;

//write to a file
file_put_contents ('myfile.txt', 'New Content');
```

// check if a file exists.

```
if (file_exists('myfile.txt')) {
    echo 'File exists';
}
```

// Delete file

```
unlink('myfile.txt');
?>
```

Image manipulation

- PHP provides functions for image manipulation using the GD extension, including creating, resizing, cropping and saving images.

```
<?php
// Load the image
$original_image = imagecreatefromjpeg('original.jpg');

// Get the dimensions of the original image
$original_width = imagesx($original_image);
$original_height = imagesy($original_image);

// New dimensions for the resized image
$new_width = 200;
$new_height = ($original_height / $original_width) * $new_width;

// Create a new image with the new dimensions
$resized_image = imagecreatetruecolor($new_width, $new_
```

02 - Assign final.

// resize the original image to the new dimensions
imagecopyresized ($resized_image, $original_image, 0, 0, 0, 0, $new_width, $new_height, $original_width, $original_height);

// save the resized image
imagejpeg ($resized_image, 'resized.jpg');

// Free up memory
imagedestroy ($original_image);
imagedestroy ($resized_image);
?>

* CAPTCHA
- CAPTCHA (Completely Automated public Turing test to tell Computers and Humans Apart) is a security measure to prevent automated interactions.
- PHP libraries and functions can generate CAPTCHA images or use third-party CAPTCHA services.

<?php
session_start();
$random_number = rand(10000, 99999);
$_SESSION['captcha'] = $random_number;

// creating an image with the CAPTCHA text
$im = imagecreatetruecolor(100, 30);
$bg_color = imagecolorallocate ($im, 255, 255, 255);
$text_color = imagecolorallocate ($im, 0, 0, 0);
imagestring ($im, 5, 10, 5, $random_number, $text_color);

header('Content-type: image/png');
imagepng($im);

02 - Avaiya Jinal

Page No. 6
Date 15 10 23

3) mysql connection libraries : mysqli pdo and oop, pdo

* mysqli pdo
- The 'mysqli' extension is commonly used for m... databases connections in PHP
- procedural approach involves using functions to ... with the database.

```
<?php

    $servername = "localhost";
    $username = "username";
    $password = "password";
    $dbname = "database";


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


// Check the connection

if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}


// Close the database
    mysqli_close($conn);
?>
```

02 - Avaiya Jinal

Page No. 7
Date 15 5 23

* OOP :-
- The 'mysqli' extension also supports an object-oriented approach for database connections and operations.

// Check connection

if ($conn->connect_error){
    die("Connection failed : " . $conn->connect_error);
}

* PDO :-
- PDO is a database access layer providing a uniform method of access to multiple database, including mysql
- It supports various database drivers and provides a consistent ~~interface~~ interface for database interactions.

<?

try{
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);

// set the PDO error mode to exception.
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
// perform operation on database
} catch(PDOException $e){
    echo "Connection failed : " . $e->getMessage();
}

// close the connection
$conn = null;
?>

Page No. 8
Date. 16 | 10 | 2023

4) SQL injection - problem, remedies.

- SQL injection is a critical security vulnerability that occurs when an attacker can manipulate an application's SQL query by injecting malicious SQL code. This can lead to unauthorized access, data leakage, and even data loss. A detailed explanation of the problem and effective remedies to prevent SQL injection

problem:-

- The problem with SQL injection lies in improper handling and validation of user input in SQL queries. Attackers exploit vulnerabilities to inject malicious code into an application's input fields, usually with forms, URL, or any other input mechanism. This injected SQL code can alter the intended SQL query's potentially leading to unauthorized access to the database or unauthorized manipulation of data.

Example:-

```
SELECT * FROM users WHERE username = '$username'
                    AND password = '$password';
```

If a malicious user enters ' OR 1=1; --' in the password field, the query would become:

```
SELECT * FROM users WHERE username = '' OR 1=1;
                    -- ' AND password = '';
```

This would return all records because '1=1' is always true, effectively bypassing the login mechanism.

Remedies :-

1. parameterized queries (prepared statements):
- use parameterized queries and prepared statement provided by the programming language and database systems. parameterized queries automatically handle escaping and quoting of parameters, preventing SQL injection attacks.

$stmt = $pdo -> prepare("SELECT * FROM users WHERE username = :username AND password :password");
$stmt -> execute(['username' => $username, 'password' => $password]);

2. Escaping and sanitizing input :-
- If you need to use user input directly in SQL queries must use of escaping functions specific to the database

Example:

$username = mysqli_real_escape_string($conn, $user_input);
$password = mysqli_real_escape_string($conn, $user_input);
$query = "SELECT * FROM users WHERE username = '$username' AND password = '$password' ";

3. Input validation
- validate user input to ensure it conforms to expected formats and types. Use regular expressions or specific validation functions.

4. Stored procedures:-
- Utilize stored procedures to encapsulate SQL logic on the database server. This minimizes the risk of SQL injection

02- Abhijeet Jamdal

Page No. 10
Date. 16 10 2023

5. Error Handling
- Implement proper error handling mechanisms prevent detailed error messages from being exp to users . log error securely.

6. white - Listing
- use a white - listing approach where you only a specific characters or patterns in user input, rath than blacklisting potentially dangerous characters

7. Database Firewalls:-
- Implement database firewalls and intrusion dete systems to monitor and block suspicious activi at the database level.

8. Education and Training:-
Educate developers and stakeholders about SQL i risks, best practices, and security measures to ensu a security - aware development environment.

Page No. 11
Date

5 oop in pHp

object - oriented programming is a programming paradigm that utilizes the concept of "objects" to organize code into reusable and modular structure. In PHP, oop provides a way to structure applications by defining classes, objects, properties, and methods. Here's a comprehensive guide to implementing oop in PHP.

Key concepts of oop in PHP:

1. classes:
- A class is a blueprint for creating objects. It defines the properties and methods that the objects of the class will have.

2. objects:-
- An object is an instance of a class. It represents a specific instance with its unique set of properties and behaviours defined by the class.

3. properties (Attributes):-
- properties are variables that hold the state or characteristics of an object. They define the objects data.

4. methods:-
- methods are functions defined within a class. They represents the behaviour or actions that objects of the class can perform.

Page No. 12
Date. 16 | 10 | 2023

5. Inheritance:-
- Inheritance allows a class to inherit properties and behaviour of a class together and controlling methods from another class. It promotes code reusability and supports the creation of specific classes based on a more general class.

6. Encapsulation
- Encapsulation involves building the data and behaviour of a class together and controlling access to it. It helps in better managing the internal state of an object.

7. polymorphism:-
- polymorphism allows objects to use methods from parent class, even if a child class has a specific implementation of the method. It provides a unified interface for multiple classes.

Example:-

```
<? php

class car {  //properties
    public $brand ;
    public $model ;

    // Constructor
    public function __construct($brand, $model) {
        $this -> brand = $brand ;
        $this -> model = $model ;
    }
```

03- Avaiya Jinal

```
// method

public Function displayInfo() {
    return "This is a {$this->brand} {$this->model}.";
}
}

// creating an object (instance of the class)
$car1 = new Car("Toyota", "Corolla");

// Accessing properties and calling methods

echo $car1->displayInfo(); //o/p: This is a Toyota Corolla.


Inheritance and polymorphism example

<?php

class Vehicle {
    public function displayInfo() {
        return "This is a generic vehicle.";
    }
}

// child class inheriting from Vehicle
class Car extends Vehicle {
    public Function displayInfo() {
        return "This is a Car.";
    }
}
```

02 - Arvaiya Jimal

Page No. 14
Date. 16 | 10 | 2024

```
class Bike extends Vehicle {
    public function displayInfo() {
        return "This is a bike.";
    }
}

// Creating objects
$car = new Car();
$bike = new Bike();

// Accessing overridden methods
echo $car->displayInfo();
echo $bike->displayInfo();
?>
```

Aaiya Jmal

5 XML in php :-

- XML (extensible markup Language) is a popular markup language used for storing and transporting data. php provides built-in functions and libraries to work with XML, allowing you to parse, create, manipulate and transform XML data. Here's a guide on how to use XML in PHP:

parsing XML :-

- PHP provides several methods to parse XML data. Two common methods are SimpleXML and DomDocument.

1. Using SimpleXML :-

```
<? php
$xml_string = '<book><title>PHP Basics</title><author>John Doe
</author></book>';
$xml = simplexml_load_string ($xml_string);

// Access elements

echo "Title" . $xml -> title . "\n";
echo "Author:" . $xml -> author . "\n";
?>
```

2. Using DomDocument

```
<? php
$xml_string = '<book><title> PHP Basics </title><author> John Doe
</author></book>

$dom = new DomDocument;
$dom -> loadXML ($xml_string);
```

@. Avaiya Jimal

// Access elements

```
$titles = $dom -> getElementsByTagName('title');
$author = $dom -> getElementsByTagName('author');

echo "Title: " . $title -> item(0) -> nodevalue. "\n";
echo "Author:" . $authors -> item(0) -> nodevalue ."\n";
```

→ Creating and modifying XML :-

1. Using SimpleXML :

```
<?php
$new_xml = new SimpleXMLElement ('<book></book>');
$new-xml -> addchild('title', 'PHP Advanced');
$new-xml -> addchild('author', 'Jue Smith');

// convert to string

$new_xml_string = $new-xml -> asXML();
echo $new-xml_string;
?>
```

2. Using DomDocument :-

```
<?php
$new-xml = new DomDocument;
$book = $new-xml -> create element ('book');
$new-xml -> appendchild ($book);
```

02 - Anaiyu Final

Page No. 17
Date 17 10 2023

```
$title = $new_xml -> createElement ('title', 'PHP Advanced');
$book -> appendchild ($title);

$author = $new_xml -> createElement ('author', 'Jane Smith');
$book -> appendchild ($author);

// Convert to string

$new_xml_string = $new_xml -> saveXML();
echo $new_xml_string;
?>
```

Accessing xml data from a file:

```
<?php

$file_path = 'path/to/xml_file.xml';
$xml = simplexml_load_file ($file_path);

// Access elements

echo "Title: ". $xml -> title . "\n";
echo "Author: ". $xml -> author . "\n";

?>
```

02 - Avijya Jindal

Handling xml Errors :

```
<?php

libxml_use_internal_errors(true);
$xml = simplexml_load_string('<invalid_xml>');

if ($xml === false) {
    echo "Failed to load xml:";
    foreach (libxml_get_errors() as $error)
    {
        echo $error->message . "\n";
    }
}

libxml_clear_errors();
?>
```

02 - Avaiya final

Page No: 14
Date: 17 | 10 | 2023

API in PHP

- Creating and Consuming APIs (Application programming interfaces) in PHP involves building endpoints to expose data or functionality and making HTTP requests to interact with external APIs. Below, we'll cover both aspects: API development and API calling.

API Development in PHP:

Building a Simple API Endpoint (HTTP GET):

```
<?php

// This is a simple API endpoint that returns a JSON response

$data = array('message' => 'Hello, this is your API response.');

header('Content-Type: application/json');
echo json_encode($data);
?>
```

Building a simple API Endpoint (HTTP POST)

```
<?php
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    $data = array('message' => 'Received POST request with data: '. json_encode($_POST));

    header('Content-Type: application/json');
    echo json_encode($data);
}
```

07 - Avaiya Jinal

API Calling in PHP

- HTTP GET

```
<?php
$api_url = 'https://jsonplaceholder.typicode.com/posts';

$ch = curl_init($api_url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

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

$data = json_decode($response, true);
print_r($data);
?>
```

- HTTP POST

```
<?php

$api_url = 'https://jsonplaceholder.typicode.com/posts';

$data = array(
    'title' => 'foo',
    'body' => 'bar',
    'userId' => 1
);
```

02- Anaiya Jmal

Page No. 91
Date 12 10 2023

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

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

$data = json_decode ($response ,true);
print_r ($data);
?>
```

Notes:-

1. cURL :- The example use a PHP library for making HTTP requests. Ensure cURL is enabled in your PHP installation.

2. JSON Handling :- In most API cases, data is exchanged in JSON format. PHP provides 'json_encode()' to convert PHP data structures to JSON and 'json_decode' to convert JSON to PHP data structures.

3. Headers :- Setting appropriate headers is crucial when interacting with APIs.

4. Error Handling :- Always include error handling and check for response status codes network issues and API - specific error responses.

02 - Avaiya Jinal

Page No. 22
Date 19 10 2023

5. API Documentation:-
- Follow the API documentation for the specific A
you're interacting with to understand the av
endpoints, request methods, and require/request pa

* Jquery:-
- Jquery is an open-source javascript library that simplifies the interactions between an HTML/CSS document, or more precisely the Document Object M and Javascript. Elaborating the term, it simplifi HTML document traversing and manipulation, browse event handling, Dom animation, Ajax interaction and cross-browser javascript development.

Example :-

```
<!DOCTYPE html>
<html>

<head>
    <script src = "http://ajax.googleapis.com/ajax/libs/j
    </script>                                13.5.1/jquery.min.js">
    <script>
        $(document).ready(function() {
            $("h1").hover(function() {
                $(this).css("colour", "green");
            }, function() {
                $(this).css("color", "aliceblue");
            });
        });
```

01 - Akshya Jimal

Page No. 23
Date. 17 10 2020

<Script>
<head>

<body>
<h1> Hello world </h1>
</body>

</html>

02 - Avaiya jinal

Page No. 24
Date. 14 | 10 | 2023

+ Composer with its commands use example

- Composer is a dependency manager for PHP that allows you to manage libraries, packages, and project dependencies in your PHP applications. It simplifies the process of integrating third-party packages and libraries into your projects. Here are the key composer commands, usage and exa

key Commands :-

1. 'composer init' : Initializes a new 'composer.json' file interactively, helping you define your project's meta and dependency.

2. 'composer install' : Install all the dependencies specified in the 'composer.json' file. This command reads 'composer.lock' file to ensure consistent installation.

3. 'composer require' :- Adds a new package to the 'composer.json' file and installs it. It automatically updates 'composer.json file and installs the specified pac

4. 'composer update' : Updates all the installed packages to the latest versions based on the version constraints specified in the 'composer.json' file.

5. 'composer remove' :
- Removes a package from the 'composer.json' file and uninstalls it.

01 Aruna Jirel

Usage and Example

Step:1 Install Composer

Step:2 Initialize a new project

Navigate to your project directory and run

composer init

Step:3 Install Dependencies.

{
"require": {
"monolog/monolog": "1.0.*"

Step:4 Update Dependencies:-

Step:5 Adding a New Dependency:-

Step:6 Removing a Dependency:-

Composer remove monolog/monolog.