**PHP mysqli\_connect() Function:**

* The mysqli\_connect() function in PHP is used to connect you to the database.
* In the previous version of the connection mysql\_connect() was used for connection and then there comes mysqli\_connect() where i means improved version of connection and is more secure than mysql\_connect().
* The mysqli extension provides enhanced features, such as support for prepared statements and transaction management, and is recommended for new projects.
* After calling mysqli\_connect(), if the connection is successful, a mysqli object representing the connection is returned. This object can be assigned to a variable and then used to execute queries and interact with the database.

**Syntax:**

Object oriented style:

$mysqli -> new mysqli(host, username, password, dbname)

Procedural style:

mysqli\_connect(host, username, password, dbname)

**Parameters used:**

host: It is optional and it specify the host name or IP address. In case of local server localhost is used as a general keyword to connect local server and run the program.

username: It is optional and it specify mysql username. In local server username is root.

Password: It is optional and it specify mysql password.

database\_name: It is database name where operation perform on data. It also optional.

**Return values:**

It returns an object which represent MySql connection. If connection failed then it return FALSE.

**PHP \_\_call() Function:**

* In PHP, the `\_\_call` method is a magic method that allows you to catch and handle calls to inaccessible or undefined methods within an object.
* This method is automatically called whenever an object is treated as a function and the requested method is not accessible or does not exist.
* The `\_\_call` method provides a way to dynamically handle method calls and can be useful for implementing functionality such as method overloading or delegation.

**Syntax:**

class MyClass {

public function \_\_call($name, $arguments) {

// Code to handle the method call

}

}

**parameters:**

1. `$name` is a string that represents the name of the method being called.

2. `$arguments` is an array that contains the arguments passed to the method.

* Within the `\_\_call` method, you can implement custom logic to handle the method call.
* You can inspect the `$name` parameter to determine which method was called and perform the necessary actions based on that information.
* The `$arguments` parameter allows you to access the arguments passed to the method.

**Example**:

class MyClass {

public function \_\_call($name, $arguments) {

if ($name === 'hello') {

echo 'Hello, ';

echo implode(' ', $arguments);

} else {

echo 'Method not found: ' . $name;

}}}

$obj = new MyClass();

$obj->hello('John', 'Doe'); // Output: Hello, John Doe

$obj->unknownMethod(); // Output: Method not found: unknownMethod

* In the example above, when the `hello` method is called on the `$obj` object, it prints "Hello, " followed by the names passed as arguments. If an unknown method is called, it outputs a message indicating that the method was not found.