Module-3

Bootstrap

* It is a CSS Library.

→ class="container" :-
A container represents a entire block.

→ class="row" :-
Row creates a flexbox and itself becomes a flex-container and all the cols inside the flex-container are flex-items.

→ In this library their are predefined classes for cols like col-1, col-2, col-3, ... etc. Each have different width's
Used as :-
class="col-4". (Example)
Their are 12 col classes like this.
for 6 → 50% $\left(\frac{6}{12} \times 100\right)$
for 4 → 33.33% $\left(\frac{4}{12} \times 100\right)$

* if width = auto [It takes the space req thatsit)
* if ~~width~~ we don't mention width (It takes all the remaining space

Range	Symbol	Breakpoint	Device
0-575		>=0	Mobile device
576-767	sm	>=576	Mobile landscape
768-991	md	>=768	Tablet
992-1199	lg	>=992	Desktop
1200-1399	xl	>=1200	Extra Large Desktop
1400-	xxl	>=1400	Extra Extra large desktop

> Consider, we are on class col-md-5
→ The breakpoint is always used in between 'col' and 'n'.
→ It is used to specify after which point the column will be sized accordingly.
→ The size of the column will be 5/12 for viewport >=768 pixels

Bootstrap classes :-

1) align-items :-

> Row wise adjusted
Used in row not used for flex-items

it makes when container has extra height it aligns content at the end.

2) Align-self :- (same as above but here individual flex item)

→ Values: start/ center & end
→ It defines the vertical alignment of each column inside a row when there is an extra height.
→ Applied only to flex items.

3) Justify content :-

* Values: start/center/end
* It defines the horizontal alignment of the columns inside a row when there is an extra width.
* Column wise adjusted.

(4) Offset :-

Syntax :-
Used only in flex items
Leaves gap of two units before column of the number we gave (here it is 2) (two units means $\frac{2}{12}$ portion of width)

Imp

(1) Buttons :-

Ex: <button type="button" class="btn btn-primary">Primary</button>

1. btn btn-primary → Primary
2. btn btn-secondary → Secondary
3. btn btn-success → Success
4. btn btn-danger → Danger
5. btn btn-warning → Warning
6. btn btn-info → Info
7. btn btn-light → Light
8. btn btn-dark → Dark
9. btn btn-link → Link

(2) Button:
btn btn-primary

Outline Buttons :-
btn btn-outline-primary

Sizes :-
<button type = "button" class = "btn btn-primary btn-lg">
Large button </button>
Similarly, for small button.
class = "btn btn-primary btn-sm"

Disabled state :-
class = "btn btn-lg btn-primary" disabled.
For button tag

~~class = "btn btn-lg btn-primary"~~
<a href = "#" class = "btn btn-primary btn-lg disabled"
tab index = "-1" role = "button" aria-disabled = "true">
Primary key

If you want make one box to sit on another
Ex : To make a to site on b :-

```
* {
    boxsizing : border-box;
}
.a{
    border: solid 2px black;
    display: inline-block;
    height : 100px;
    width : 1140px;
    position : absolute;
    top: 1;
    left: 1;
}
.b{
    border: solid 1px red;
    display : inline-block;
    height: 700px;
    width: 1140px;
    position: relative;
}
```

DBMS

Working with Databases :-

name	uid	Gender.
Neha	123005	female
Rohit	123001	male
Mohit	123004	male
Komal	123003	female
Nidhi	123002	female

← Attribute

→ Records

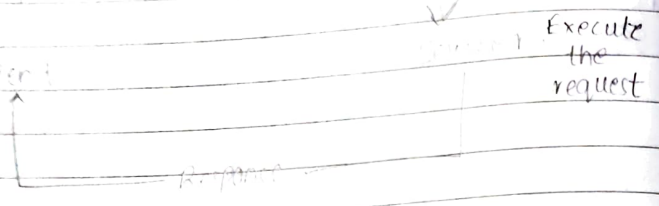
Entity :- A piece of data that is stored in the database.

Attribute :- A column in an entity table.

Conventions

1. Tablename should be in lower case
   Ex: students, faculties-cse, etc.
2. Use plural forms
   Ex: students_cse, classrooms-cse, etc.
3. Use underscore ("_") when table name contains more than one word
   Ex: students_cse, etc.
4. Attributes name should be in lower case,
   Ex: name, designation, etc.
5. Attributes name should be in singular form.
   Ex: name, designation, etc.
6. In Attributes:- More than one word should be seperated by("_") underscore
   Ex: phone_number, etc.
7. Add an additional attribute named 'id' in each table that contains auto incremented integers. This attribute is also called the primary key of the table.

- General purpose scripting language especially for web development



* PHP files extension (".php")
* Stored in c:\xampp\htdocs.
* Statements end with a semicolon.

Syntax:-

```
<?php
    echo "Hello world!";
?>
```

echo → To print any text.

<?php ?> → opening & closing tag.

Anything written out of the php code it will be considered as output printed continuously after statement.

Note:-
Html file cannot contain the php code.

php in web development:-
Easily embedded into HTML.
It is compatible with Apache web server.
It does not need to be pre-compiled.
It has built-in integration with DBMS - MySQL.

Variable :-
Starts with $ Ex: $a, $str, etc.
It consists of alphanumeric characters & underscore.
It should either start with an underscore or alphabet
It cannot start with no.
No special characters can be used.
Case sensitive

Assigning value:-

```
<?php
    $a=5;
    echo $a;
?>
```

Overwriting will change the assignment to the lastly assigned value.

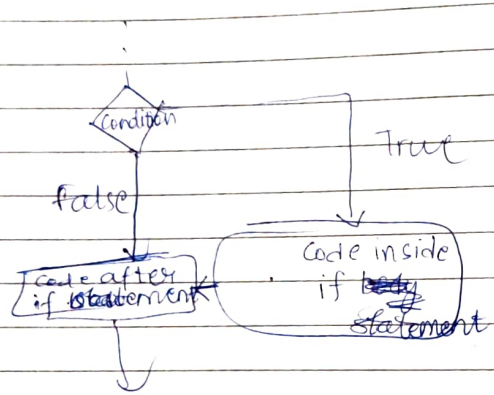
Variable can store:
Int Str float bool arr obj Null

Operators :-
1) Arithmetic operators -> +, -, *, /, %
2) Assignment operators -> =, +=, -=, *=, /=, %=
3) Increment/Decrement operators -> ++, -- (↑ or ↓ by "1")
4) Comparison operators -> ==, !==, <, <=, >, >=
5) Logical (Relational operators) -> and (&&), or (||), not

Conditional statements

```
if(boolen. expression){
    //code to execute only if boolean-expression
    evalutes to true.
    . . . .
    . . . . .
    . . . . .

}
// code that executes irrespective of the value
of the boolean expression.
. . . .
. . . .
. . . .
```

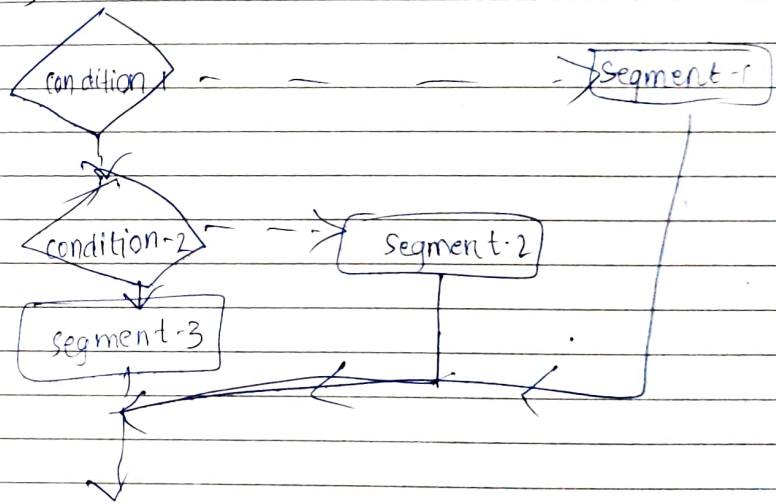


if else:

```
if (     ){
    echo"    ";
}else{
    echo"    ";
}
```

if - elseif - else:

```
if (    ){
    echo" ";
}
elseif (    ){
    echo" ";
}
.
.
.
}else{
    echo
}
```



Ex:

```
<?php
$a = 20;
$b = 10;
$operator = "*";
switch ($operator){
    case "+":
        echo $a + $b;
        break;
    case "-":
        echo $a - $b;
        break;
    case "*":
        echo $a * $b;
        break;
    case "/":
        echo $a / $b;
        break;
    default:
        echo "invalid";
}
```

Note:-

```
<?php
    $num = 7;
    echo "You choose - ". $num;
?>
```

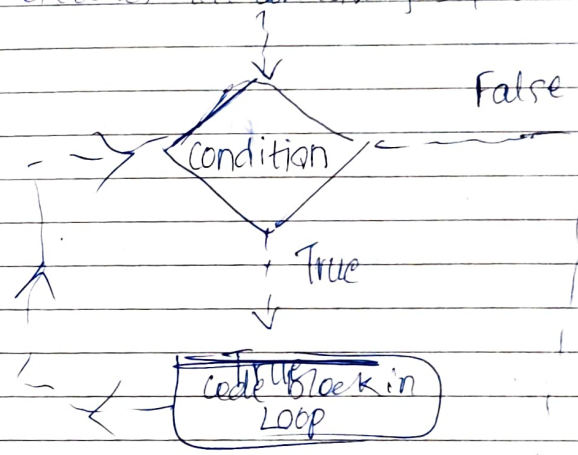
Output- You choose - 7.

Loops

1. while Loop:

```
//initialisation
while (condition){
    //code
    //operation
    //increment/derement
}
```

code executes till condition gets false.



Breaks & continue

break: To break the loop

Ex:

```
<?php
  $n=1;
  while ($n<=10){
     echo $n*2;
     if ($n==5){
       break;
     }
     $n++;
  }
?>
```

→ breaks the loop when n==5;

Note: If we use continue in place of breaks like

```
continue;
```

The iteration at n==5 continues for multiple times.

do-while loop:

```
<?php
  $n=1;
  do {
     echo $n*2;
     $n++;
  } while($n<=10);
?>
```

for loop (initialization, condition, iteration)

```
<?php
  for ($n=1; $n<=10; $n++) {
  echo $n*2;
  }
?>
```

foreach loop: Only for arrays

```
foreach ($array as $element){
  ---
  ---
  ---
}
```

Arrays

Collection of values of same datatype

1. Numeric index array
2. Associative

① Numeric index array: collection of the same kind of information starts from zero.

Ex: $presidents = array ("George", "Barack", "Donald")
(0) (1) (2)

To print: echo $presidents[i];

To update value:-

$presidents[0] = "George W";

The order in which info is present is imp

Associative Array: Collection of a pair of information

They have strs or texts as indices.

Used when the indices and their association with the values more imp than the values alone

Ex:

Key	Value
George	2001
Barak	2009
Donald	2017
Joe	2021

Association between the pair of information is imp.

Defining an associative Array:-

```
$years = array("Bill" => 1993, "George" => 2001, "Barack" => 2009);
```

(Bill → key, 1993 → value)

Printing & updating is same like Numeric inder array.

Multidimensional array:-

* Array of arrays.

Ex:-

Amit		Chirag		Vandana	
Physics	77	Physics	84	Physics	91
Chemistry	86	Chemistry	92	Chemistry	98
Maths	84	Maths	78	Maths	97

Eg

```
$a = array("Physics" => 77, "chemistry" => 86, "Maths" => 84);
$b = array("Physics" => 84, "Chemistry" => 92, "Maths" => 78);
$c = array("Physics" => 91, "Chemistry" => 98, "Maths" => 97);
```

∴ The Multidimensional array is

```
$students = array("Amit" => $a, "Chirag" => $b, "Vandana" => $
```

To print:-

```
foreach ($students as $student => $subjects) {
    foreach ($subjects as subject => $mark) {
        echo $student. "has scored ". $mark. "in". $subject
        "\n";
    }
}
```

"Strings"

* Can be expressed in both '' & "". But the only difference is ("") double quotes represents any variable with in it with the value it holds and also converts escape sequence into special characters.

Escape characters:

Denoted by a backward slash (\) and escapes the character that follows it.

Ex: \n, \t, etc

\n → newline \t → tabspace.

* Escape character also removes the special meaning for the special character and treats like normal text.

Ex:

```
<?php
    $name = "puji";
    echo "Hello $name! \n";
    echo "Hope you are having a great time.";
?>
```

If we replace third line with

```
echo "Hello \$name! \n";
```

It prints:

Hello $name! Hope you are having a great time.

Replace: $name with

Note:-

```
<?php
    echo "Hello /"Pujith/";
?>
```

Gives output as:-

Hello "Pujith"

In single quotes we can't use variable names & escape character

String concatenation operator:- (.)

```
<?php
    $name = Pujith;
    echo 'Hello' . $name . "!";
?>
```

O/P:- Hello Pujith!

Various String functions in PHP:-

Link:- php.net/manual/en.ref.strings.php

Functions:-

Ex:-

```
<?
function get_max ($a, $b){
    if ($a > $b){
        return $a;
    }else{
        return $b;
    }
}
?>
```

→ starts with this keyword

→ Variable which denotes function

→ parameters inclu... Function argume...

Def:- * It is a block of code that performs a specific task.
* It takes data called function arguments.
* It can take any number of arguments
* It returns only one value at a time.

Parts of a function

```
<?php
    function get_max ($a, $b) {        → declaring function
        if ($a > $b) {
            return $a;
        }else{
            return $b;
        }
    }
    $max = get_max (5,10);             → Function calling
    echo $max;
?>
```

Main code

O/P:- 10

Naming a function:-

- Can only contain alpha numeric characters or the underscore.
- No special characters.
- Cannot start with a no.. Can start with '_' or alphabets.
- Use "_" if the name contains two or more words.
- Self-explanatory (Dignified coding)

Note: We can also give values for variables at the declaration of function part itself. But ~~important~~ if given at declaration & calling part → calling part one is considered.

Scope of a variable:

Local variable (Locally)
Defined within function
Their scope is limited to only inside of the function

Global variable (Globally)
* Defined outside a function
* Their scope is limited to only outside the function.

Ex: <?php
$abc = "Hello World!" → Global variable
function a(){
$abc = "Hey";
echo $abc; → Local variable.
}
function b(){
echo $abc; → No scope
}
a()
b()
echo $abc;
?>

O/p :- Hey
Undefined Variable
Hello World!

Global Variables :-
We can also use global variables inside the function by using global keyword.

Ex: <?php
$abc = "Hello World!";
function a(){
global $abc;
$abc = "Hey"; → Not considered when global variable is their, new word will be assigned to the global
echo $abc;
}
a()
echo $abc;
?>

Output :-
~~Hello World!~~
Hey Hey

Note :- whenever a global variable is defined, it automatically gets added to predefined built-in associative array named $GLOBALS as a key.

$GLOBALS :-
It is a predefined built-in associative array.

<?php
$abc = "Hello World";
function a(){
$GLOBALS['abc'] = "Hey";
echo $GLOBALS['abc'];
}
a();
echo $abc;
?>

O/p :- Hey Hey

Note :- Built-in Associative Arrays / Superglobal variable
> $_POST
> $_GET
> $_FILES
> $_REQUEST
> $_SERVER
> $_COOKIE
> $_SESSION
> $_ENV

Q) What is superglobal variable?
A:- It is a variable that is always available in all scopes. This means, it can be accessed from outside of a function or from inside of a function.
> They have specific purpose.
> They are built-in and cannot be defined externally.
> They can be modified.
> They always exist as an associative array.
> They are named in capital letters.

~~Call by Reference~~ :

Call by value :

```
<?php
function add_one ($n){
    $n++
    return $n;
}
$num = 0
$num = add_one ($num);
echo $num." ";
?>
```

Op : 1

If we remove this we get o/p as 0 .so, the new return value should be stored in the variable to print

Call by reference :-

```
<?php
function add_one (& $n){
   $n++;
   return $n;
}
$num = 0;
add_one ($num);
echo $num." ";
?>
```

Call by Value	Call by refrence
• A copy of actual value is passed to the function	• A reference of the variable is passed to the function.
• Changes made to the parameter inside the function do not affect the original variable outside the function	• Changes made to the parameter inside the function affect the original variable.

This method is used in languages like Java

This is used in languages like C++

FORM Submission

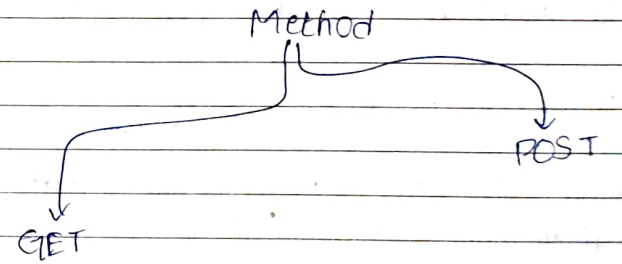
```
<form method = "post"  action = "submit.php" >
```

↓ Mention this in form tag.

Here the action = "submit.php" → As we submit the form a new http request is generated by the browser. This request is generated for the URL present in action attribute.

This http request takes all the data submitted in the form. data is went through key value pairs.

Method Attribute :- Default :- GET FORM



* POST request is preferred for form submission
Because if we use GET request after submitting the values will be shown in URL above .so, for privacy reasons we use POST request.

Q) What happens once the web server receives the HTTP Request

A:- 1) Webserver searches for the PHP file on its computer & starts executing it.

2) Data is converted into an associative array and is further stored in $_GET or $_POST superglobal variables

FOR POST REQ →
```
<?php
$email = $_POST['email'];
$password = $_POST['password'];
echo $email;
echo $password;
?>
```

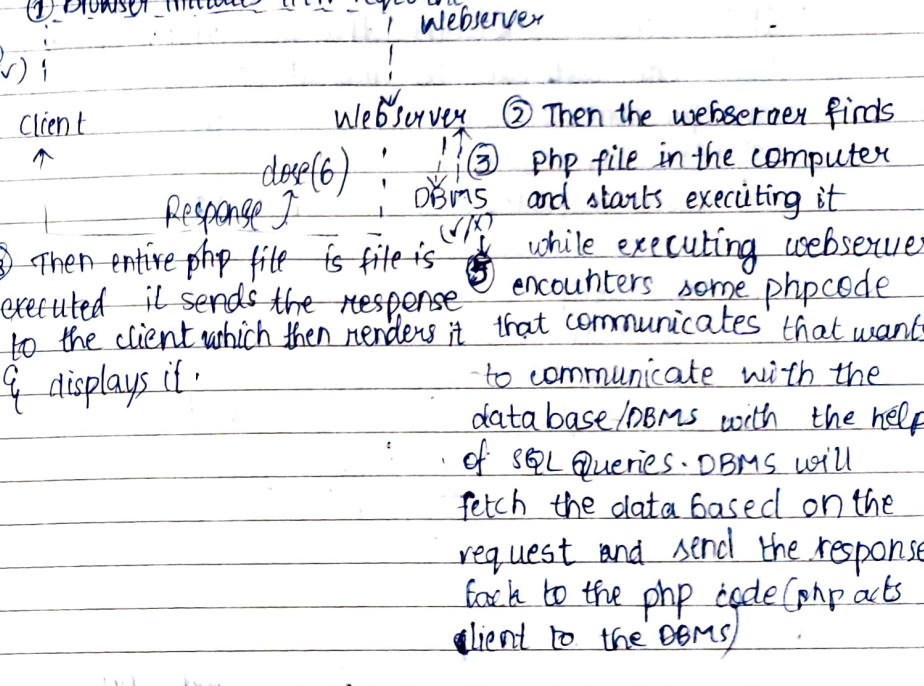
FOR GET REQ →
```
<?php
$email = $_GET['email'];
$password = $_GET['password'];
echo $email;
echo $password;
?>
```

3) Data is accessed using the superglobal variables.

4) Data is stored in the database.

Note:- Mostly commonly used request method is GET form (like opening hyperlinks & clicking on anchor tags generates in GET form & POST FORM is used in case of form submissions. So, get is used more.

Data Base Communication



Database related functions

① mysqli_connect():-

Used to establish connection with the MySQL server

Query to use to establish connection:-

```
$conn = mysqli_connect($db_hostname, $db_username, $db_password, $db_name);
```

↳ reference obj → represents connection to the mySQL server

If the info of any of these parameters is incorrect while running SQL Queries then the function returns false value

⑤ mysqli_connect_error():-
> It is used to return the last error message from the last call to mysqli_connect().
> This functions returns the message that describes the error.

```
echo mysqli_connect_error();
```

③ mysqli_query()
It is used to execute an SQL query on the selected database.

```
$result = mysql_query($conn, $sql);
```

$conn: connection reference object
$sql

④ mysqli_fetch_assoc()
It is used to fetch the data from the reference object that we get from the mysqli_query() function in case SELECT queries.

```
// It fetches data row wise
$row = mysqli_fetch_assoc($result);
```

⑤ mysqli_error():-
It returns the last error message for the last call to mysqli_query().

```
echo mysqli_error($conn);
```

⑥ mysqli_close():-
It closes the previously opened connection with the MYSQL server.

```
mysqli_close($conn);
```

CREATING PHP FILES TO FETCH & STORE DATA

Prerequisites:-
1. Registration form
2. Table
3. Database

PHP file to ~~fetch~~ store Data :-

```
<?php
$db_hostname = "127.0.0.1";
$db_username = "root";
$db_password = "";
$db_name = "test";
$conn = mysqli_connect($db_hostname, $db_username, $db_password, $db_name);
if (!$conn){
    echo "connection failed:".mysqli_connect_error();
    exit;
}

$name = $_POST['name'];
$email = $_POST['email'];
$password = $_POST['password']

$sql = "INSERT INTO users (name, email, password) VALUES ('$name', $email, '$password')"

$result = mysqli_query($conn, $sql);
if (!$result){
    echo "Error:".mysqli_error($conn);
    exit;
}
echo "Registration successful";
mysqli_close($conn); ?>
```

Fetching data using mysqli_fetch_assoc():-

* Till the completion of first if statement which lasts at exit is before everything is same.

The remaining code will be cut & pasted as

```
$sql = "SELECT * FROM users";
$result = mysqli_query($conn, $sql);
if (!$result){
    echo "Error: ", mysql_error($conn);
    exit;
}
while($row = mysqli_fetch_assoc($result)){
    echo $row['name'], "<br/>";
}
mysqli_close($conn);
?>
```

← fetches each & every row

Note:- For SELECT queries, it returns a reference object, through which we can access the fetched data.

Registration: Creating a acc process.

Identification & Authentication:-
To login into our acc with Email & phone number.

Cookies:-

Functions

① setcookie():-

setcookie("key", "value", time() + 3600);

↑ exp date time for cookie

Unix timestamp

$value = $_COOKIE['key'];

Note:- User info can be stored in cookies but critical information is not.

Problem with cookies

Important & privacy things should be only

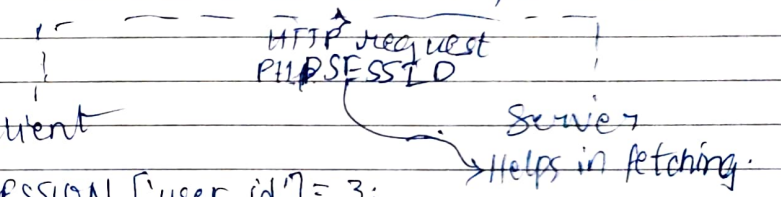
Session:-
A small piece of info. stored on the server (host computer)

Ex:- 4 users
Website creates 4 sessions to for

Sessions are more secure than cookies.

Implementing session in PHP:-

It expects the brower to send cookies:-



$_SESSION['user_id'] = 3;

or

$user_id = $_SESSION['user_id'];

Note:- When PHPSESSID is not found. For which ever mentioned reason reason it creates a new session and assign whatever it stores to the dollar session variable. for it to be accessed by php code. By default this session is empty. $_session is used to store any info in the session. And when the response goes back, the corresponding Session ID of session is stored in browser in the form of Cookies:-

(tool/system)

i) session_start() finds the session & assigns it to $_SESSION variable.
ii) Using $_SESSION variable without calling session_start() will throw an error.

isset();

session_destroy();

session_start();

For login submission page :-
php file should be designed as:-

```php
<?php
    $db_hostname = "127.0.0.1";
    $db_username = "root";
    $db_password = "";
    $db_name = "test";
    $conn = mysqli_connect ($db_hostname, $db_username, $db_password, $db_name);

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

    $email = $_POST['email'];
    $password = $_POST['password'];

    $sql = "SELECT * FROM users WHERE email = '$email' AND password = '$password' ";

    $result = mysqli_query($conn, $sql);
    if (!$result){
        echo "Error: ". mysqli_error($conn);
        exit;
    }
    $row = mysqli_fetch_assoc ($result);
    if ($row){
        echo "Hello ". $row ['name'] . "<br/>";
    }else{
        echo "Login Failed"<br/>";
?>
```