MODULE-4 (ADVANCED PHP)

- **OOPS**

1) What Is Object Oriented Programming?

- **You can also write PHP code in an object-oriented style.**
- **OOP stands for Object-Oriented Programming.**
- **Procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating objects that contain both data and functions.**

- **Object-oriented programming has several advantages over procedural programming:**
- **OOP is faster and easier to execute**
- **OOP provides a clear structure for the programs**
- **OOP helps to keep the PHP code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug**
- **OOP makes it possible to create full reusable applications with less code and shorter development time**

Example :
Class === Fruits / car
Object ==== apple,banana,orange / volve audi ford

Class :

- **Define a Class**

- **A class is defined by using the class keyword, followed by the name of the class and a pair of curly braces ({}). All its properties and methods go inside the braces:**

- **Define Objects**
- **Classes are nothing without objects! We can create multiple objects from a class. Each object has all the properties and method1**
- **s defined in the class, but they will have different property values.**
- **Objects of a class are created using the new keyword.**

```
<?php
class abc
{
	public $a=10;
	public $b=30;
	function sum()
	{
			echo $sum=$this->a+$this->b;
	}
}
$obj= new abc;
$obj->sum();
?>
```

2) What Are Properties Of Object Oriented Systems?

- **Object-oriented programming (OOP) is a programming paradigm that uses objects, which are instances of classes, to organize**

code. PHP supports object-oriented programming, and here are some key properties of object-oriented systems in PHP:

1] Define a Class :-

- **A class is defined by using the class keyword, followed by the name of the class and a pair of curly braces ({}). All its properties and methods go inside the braces:**

```
<?php

class Fruit {

  // Properties

  public $name;

  public $color;

// Methods

  function set_name($name) {

    $this->name = $name;

  }

  function get_name() {

    return $this->name;

  }

}
```

?>

2] Define Objects :-

- **Classes are nothing without objects! We can create multiple objects from a class. Each object has all the properties and method1**
- **s defined in the class, but they will have different property values.**
- **Objects of a class are created using the new keyword.**

```
<?php
class abc
{
        public $a=10;
        public $b=30;
        function sum()
        {
                        echo $sum=$this->a+$this->b;
        }
}
$obj= new abc;
$obj->sum();
?>
```

3] Encapsulation:-

- **The wrapping up of data and methods into a single unit (called class) is known as encapsulation. Encapsulation is a protection mechanism for the data members and methods present inside the class. In the encapsulation technique, we are restricting the data members from access to outside world end-user.**

- **In PHP, encapsulation utilized to make the code more secure and robust. Using encapsulation, we are hiding the real implementation of data from the user and also does not allow anyone to manipulate data members except by calling the desired operation.**

4] Inheritance :-

- **Inheritance is the process of creating a new Class, called the Derived Class , from the existing class, called the Base Class . The Inheritance has many advantages, the most important of them being the reusability of code. Rather than developing new Objects from scratch, new code can be based on the work of other developers, adding only the new features that are needed. The reuse of existing classes saves time and effort.**

```
<?php
/*
Inheritance / Reusability
```

- **It is a concept of accessing the features of one class from another class.**

```
types :-

1] Single inheritance.
2] Multi-level inheritance.
3] Multiple inheritance.
4] Hierarchical Inheritance.
5] Hybrid Inheritance.

*/
// single level inheritance
class model
{
```

```
	public $a=10;
	public $b=20;
	function sum()
	{
		echo $sum=$this->a+$this->b."<br>";
	}
}

class control extends model
{
	function multi()
	{
		$this->sum();
		echo $this->a*$this->b;
	}
}
$obj=new control;
$obj->multi();
/*

1) single level
class a
{
}
class b extend a
{
}
obj: b
==================================

1) Multilevel
```

class a
{
}
class b extend a
{
}
class c extends b
{
}
obj: c

==========

3) Multiple inheritance.
class a
{
}
class b
{
}
class c extends a,b
{
}
obj: c // but Multiple Inhetitance not possible in php by extend / only one class extend

==========

4)Hierarchical Inheritance.
class a

{
}
class b extends a
{
}
class c extends a
{
}
obj : b and c

==========
====
5)Hybrid Inheritance.
Mix of any two
class a
{
}
class b extend a
{
}
class c extends b
{
}
class d extends a
{
}
***/**
5] Abstraction:-

- **An abstract class or method is defined with the abstract keyword:**

- **An abstract class is a class that contains at least one abstract method. An abstract method is a method that is declared, but not implemented in the code.**
- **When inheriting from an abstract class, the child class method mAn object of an abstract class can't be made.**
- **ust be defined with the same name, and the same or a less restricted access modifier. So, if the abstract method is defined as protected, the child class method must be defined as either protected or public, but not private.**
- **So, when a child class is inherited from an abstract class, we have the following rules:**
- **The child class method must be defined with the same name and it redeclares the parent abstract method**
- **The child class method must be defined with the same or a less restricted access modifier**
- **The number of required arguments must be the same. However, the child class may have optional arguments in addition**

```
<?php
abstract class ParentClass {
  abstract public function someMethod1();
  abstract public function someMethod2($name, $color);
  abstract public function someMethod3() : string;
}
```

6] Polymorphism:-

```
<?php
/*
Polymorphism in PHP
This word is can from Greek word poly and morphism.
Poly means "many" and morphism means property which help
us to assign more than one property.
*/
```

```
//=> Overloading Same method name with different parameter,
//since PHP doesn't support method overloading concept

/*
class abc
{
        function sum($a,$b)
        {
           echo $a+$b;
        }
        function sum($a,$b,$c)
        {
           echo $a+$b+$c;
        }
}
$obj=new xyz;
$obj->sum(5,10);
$obj->sum(5,10,5);

Note : overloading not posible in PHP
*/
//=> Overriding When same methods defined in parents and child class
//with same signature

//i.e know as method overriding
class abc
{
        function sum($a,$b)
        {
           echo $a+$b;
        }
}
class xyz extends abc
```

```
{
        function sum($a,$b)
        {
          abc::sum(5,7);
          echo $a*$b;
        }
}
$obj=new xyz;
$obj->sum(5,10);
```

3) What Is Difference Between Class And Interface?

CLASS	INTERFACE
The 'class' keyword is used to create a class.	The 'interface' keyword is used to create an interface.
An object of a class can be created.	An object of an interface cannot be created.
Class doesn't support multiple inheritance.	Interface supports multiple inheritance.
A class can inherit another class.	An Interface cannot inherit a class.
A class can be inherited by another class using the keyword 'extends'.	An Interface can be inherited by a class using the keyword 'implements' and it can be inherited by another interface using the keyword 'extends'.
A class can contain	An Interface cannot contain

constructors.	constructors.
It cannot contain abstract methods.	It consists of abstract methods only.
Variables and methods can be declared using any specifiers like public, protected, default, private.	Variables and methods are declared as public only.

4) What Is Overloading?

- Method overloading is a form of polymorphism in OOP. Polymorphism allows objects or methods to act in different ways, according to the means in which they are used. One such manner in which the methods behave according to their.
- argument types and number of arguments is method overloading.

5) What are the differences between abstract classes and interfaces?

- Interfaces are similar to abstract classes. The difference between interfaces and abstract classes are:
- Interfaces cannot have properties, while abstract classes can
- All interface methods must be public, while abstract class methods is public or protected

- **All methods in an interface are abstract, so they cannot be implemented in code and the abstract keyword is not necessary**
- **Classes can implement an interface while inheriting from another class at the same time**

- **<u>INTERFACES</u> :-**
- **Interface support multiple inheritance.**
- **Interface does'n contains data member.**
- **Interface does'n contains constructors.**
- **An interface cannot have access modifiers by default everything is assumed as public.**
- **An interface contains only incomplete member (signature of member).**
- **Member of interface can not be static.**

- **<u>ABSTRACT CLASSES</u> :-**
- **Abstract class does not support multiple inheritance.**
- **Abstract class contains data member.**
- **Abstract class contains constructors.**
- **An abstract class can contain access modifiers for the subs ,functions ,properties.**
- **An abstract class contains both incomplete (abstract) and complete member.**
- **Only complete member of abstract class can be static.**

6) Define Constructor and Destructor?

- **In object oriented programming terminology, constructor is a method defined inside a class that is called automatically at the time of creation of object. Purpose of a constructor method is to**

initialize the object. In PHP, a method of special name __construct acts as a constructor.

- **=> class name & function name are same thats called constructor**
- **In PHP, destructor method is named as __destruct. During shutdown sequence too, objects will be destroyed. Destructor method doesn't take any arguments, neither does it return any data type.**
- **Constructor :-**

```
<?php
/*
```

- **A constructor allows you to initialize an object's properties**
- **upon creation of the object.**
- **If you create a __construct() function, PHP will automatically call**
- **this function when you create an object from a class.**
- **Notice that the construct function starts with two underscores (__)!**
- **We see in the example below, that using a constructor saves us from**
- **calling the set_name() method which reduces the amount of code:**

```
__destruct()
```

```
destroy object of class

call in last

*/

class abc

{

        function simple()

        {

                echo "Simple Function <br>";

        }

        function __construct()

        {

                echo "Magic function run auto matecaly<br>";

        }

        function display()

        {

                $this->simple();

                abc::__construct();  //   __construct call by
::(scopresolution)
```

```
        }
}
$obj=new abc;
$obj->display();
?>
```

- **Destructor** :-

```
<?php
/*
```

- **A constructor allows you to initialize an object's properties**
- **upon creation of the object.**
- **If you create a __construct() function, PHP will automatically call**
- **this function when you create an object from a class.**
- **Notice that the construct function starts with two underscores (__)!**
- **We see in the example below, that using a constructor saves us from**
- **calling the set_name() method which reduces the amount of code:**

```
__destruct()
```

```
destroy object of class

call in last

*/

class abc

{

	function simple()

	{

		echo "Simple Function <br>";

	}

	function __construct()

	{

		echo "Magic function run auto matecaly<br>";

	}

	function display()

	{

		$this->simple();

		abc::__construct();  //   __construct call by
::(scopresolution)
```

```
    }
}

$obj=new abc;

$obj->display();

?>
```

7) How to Load Classes in PHP?

- **In PHP, you can load classes using the include or require statements to include the file containing the class definition.**

Include :-

```
<?php
// include work as include file code
/*
include('include1.php');
include('include1.php');
echo "Morning";
*/
// include_once same as include() but only include 1 time
// if file does not exits than iclude provide warning

include_once('include2.php');
include_once('include1.php');
include_once('include1.php');
echo "Morning";
/*
```

Differance between include & include_once
=> include_once same as include() but only include 1 time
include gives warning error when file done not exits
// Incude & require both same but if file not exits than Incude define E_warning & Require Fetel Error
***/**
?>

Require :-

```
<?php
// require work as require file code
require('Require2.php');
require('Require1.php');
echo "Morning";
// require_once same as require() but only require 1 time
// if file does not exits than iclude provide warning
/*
require_once('Require1.php');
require_once('Require1.php');
require_once('Require1.php');
echo "Morning";
*/
/*
Differance between require & include_once
=> include_once same as require() but only require 1 time
require gives Fetel error when file done not exits
// Incude & require both same but if file not exits than Incude define E_warning & Require Fetel Error
*/
?>
```

8) How to Call Parent Constructor?

- **You can call the constructor of a parent class using the parent::__construct() syntax. This is typically done in the constructor of a child class when you want to ensure that the parent class's constructor is also executed.**

9) Are Parent Constructor Called Implicitly When Create An Object Of Class?

- **When you create an object of a class, the parent constructor is not called implicitly. However, if the child class has its own constructor and it does not explicitly call the parent constructor using the parent::__construct() syntax, then the parent class's constructor is automatically called before the child class's constructor.**

10) What Happen, If Constructor Is Defined As Private Or Protected?

- **if a constructor is defined as private or protected, it affects the visibility of the constructor outside the class.**

- **<u>Private Constructor</u> :-**

- **If a constructor is declared as private, it means it can only be called from within the same class. Instances of the class cannot be created from outside the class using the 'new' keyword.**

- **This is useful in scenarios where you want to control the instantiation of objects and ensure that instances can only be created through static methods or within the class itself.**

- **Protected Constructor** :-

- **If a constructor is declared as protected, it means it can only be called from within the same class or its subclasses. Instances of the class cannot be created from outside the class or its subclasses using the 'new' keyword.**
- **This is useful when you want to allow the constructor to be used by subclasses but prevent direct instantiation from outside the class hierarchy.**

11) What are PHP Magic Methods/Functions? List them .write program for static keyword in PHP?

- **Magic methods in PHP are special methods that are aimed to perform certain tasks. These methods are named with double underscore (__) as prefix. All these function names are reserved and can't be used for any purpose other than associated magical functionality. Magical method in a class must be declared public. These methods act as interceptors that are automatically called when certain conditions are met.**
- **Following magical methods are currently available in PHP**
- **__construct(), __destruct(), __call(), __callStatic(), __get(), __set(), __isset(), __unset(), __sleep(), __wakeup(), __serialize(), __unserialize(), __toString(), __invoke(), __set_state(), __clone() and __debugInfo()**

- **Static properties can be called directly - without creating an instance of a class.**
- **Static properties are declared with the static keyword:**
- **To access a static property use the class name, double colon (::), and the property name:**

```
ClassName::staticMethod()/static_variable;
<?php
class greeting {
  public static function welcome() {
    echo "Hello World!";
  }
}
// Call static method
greeting::welcome();
?>
```

12) Create multiple Traits and use it in to a single class?

```
<?php
  /*
  PHP only supports single inheritance:

  a child class can inherit only from one single parent. So,
  what if a class needs to inherit multiple behaviors?

  OOP traits solve this problem.
```

Traits are declared with the trait keyword: as class To use a trait in a class, use the use keyword: // for inheritance Traits are used to declare methods that can be used in multiple classes.

Traits can have methods and abstract methods that can be used in multiple classes, and the methods can have any access modifier (public, private, or protected).

```
 */
trait first // use trait insted of class
{
        function method1()
{
        echo "This is method1.<br>";
}
 function method2()
{
        echo "This is method2";
}
}
class sample
{
```

```
        use first; // here use word (use) for inheritance of
        class first
}

        $obj= new sample;

        $obj->method1();

        $obj->method2();

?>
```

13) Write PHP Script of Object Iteration?

```
<?php

class MyClass

{

    public $var1 = 'value
    1'; public $var2 =
    'value 2'; public $var3
    = 'value 3';


    protected $protected = 'protected
    var'; private $private =
    'private var';

    function iterateVisible() {

       echo "MyClass::iterateVisible:\n";
       foreach ($this as $key => $value) {

           print "$key => $value\n";
       }
```

```
    }

}
$class = new MyClass();

foreach($class as $key => $value) {print
    "$key => $value\n";
}

echo "\n";
$class->iterateVisible();

?>
```

Output:-

```
var1 => value 1var2

=> value 2var3 =>
value 3

MyClass::iterateVisible:
var1 => value 1
var2 => value 2var3

=> value 3


protected => protected var
private => private var
```

14) Use of The $this keyword.

- **$this is a keyword that refers to the current object of the class. It allows you to access the properties and methods of the current object within the class using the object operator ->. For example, $this->property or $this->method() 1.**

- **The $this keyword is only available within a class and doesn't exist outside of the class. If you attempt to use $this outside of a class, you'll get an error 1.**

- **When you access an object property using the $this keyword, you use the $ with the this keyword only. And you don't use the $ with the property name. For example, $this->balance 1.**

- **Jquery :-**

1) What is jQuery?

- **jQuery is a lightweight, "write less, do more", JavaScript library.**
- **The purpose of jQuery is to make it much easier to use JavaScript on your website.**
- **jQuery takes a lot of common tasks that require many lines of JavaScript code to accomplish, and wraps them into methods that you can call with a single line of code.**
- **jQuery also simplifies a lot of the complicated things from JavaScript, like AJAX calls and DOM manipulation.**

2) How are JavaScript and jQuery different?

- **JavaScript :-**

- **It is a major scripting programming language that is used to make websites more responsive and interactive. It is one of the pivoted parts alongside HTML and CSS which are used to create web pages. If HTML & CSS decorates and designed the web pages so, Javascript makes the web pages dynamic(we can say it gives them life). JavaScript is a major client-side language. It's not only confined to website development but also used in many desktop and server programs (Node.js is the best-known example) and Some databases, like MongoDB and CouchDB, also use JavaScript. Whenever your browser parses a web page, its responsibility is to create a tree-structure presentation in memory.**

- **<u>jQuery</u> :-**
- **JQuery is a framework for JavaScript developed from JavaScript. It is the most popular JavaScript library invented by John Resign and was released in January 2006 at BarCamp NYC. It is a free, open-source library and It's a fast, concise, and rich-featured JavaScript library and also has cross-browser compatibility. The purpose of jQuery is to make life easier for the masses so that they can easily develop websites and browser-based applications using JavaScript. In a concise manner, we can say that "JQuery is a library to provide better client-side web page development" environment to the developer with the help of its feature-rich library.**

3) Which is the starting point of code execution in jQuery?

- **The jQuery starts its code execution from the $(document).ready() function which is executed whenever the**

whole HTML DOM is loaded and is totally rendered by the browser, so that the event handlers work correctly without any errors. This $(document).ready() function load the script only after the whole DOM is loaded by the browser.

- **It takes time for the browser to get the document ready when we don't use $(document).ready() function in the script tag. The jQuery in the script may get executed before some content or element on which an event handler or some other function is acting hence this may cause some problems in the webpage, so it is always necessary to start execution whenever the whole DOM is ready. So we use $(document).ready() function.**

4) Document Ready Vs Window. Load() jQuery.

- **document.ready is a jQuery event, it runs when the DOM is ready, e.g. all elements are there to be found/used, but not necessarily all content.**
- **window.onload fires later (or at the same time in the worst/failing cases) when images and such are loaded, so if you're using image dimensions for example, you often want to use this instead.**

5) What is the difference between prop and attr?

Prop() Method :-

- **This method returns the current value.**
- **This method is mainly used when the user wants to change the value for an HTML tag's attribute. This method is mainly used to set the default value for an HTML tag's attribute.**
- **It changes properties for that HTML tag as per the DOM tree.**
- **Its syntax is:**

- **$(selector).prop(property)**
- **It takes three parameters Property , value and a function.**

Attr() Method :-

- **This method returns the default value.**
- **This method is mainly used to set the default value for an HTML tag's attribute.**
- **It changes attributes for that HTML tag.**
- **Its syntax is:**
- **$(selector).attr(attribute)**
- **It takes three parameters an attribute, value, and a function.**

6) Explain Difference Between JQuery And JavaScript?

- **JavaScript :-**
- **It is a major scripting programming language that is used to make websites more responsive and interactive. It is one of the pivoted parts alongside HTML and CSS which are used to create web pages. If HTML & CSS decorates and designed the web pages so, Javascript makes the web pages dynamic(we can say it gives them life). JavaScript is a major client-side language. It's not only confined to website development but also used in many desktop and server programs (Node.js is the best-known example) and Some databases, like MongoDB and CouchDB, also use JavaScript. Whenever your browser parses a web page, its responsibility is to create a tree-structure presentation in memory.**

- **jQuery :-**
- **JQuery is a framework for JavaScript developed from JavaScript. It is the most popular JavaScript library invented by John Resign**

and was released in January 2006 at BarCamp NYC. It is a free, open-source library and It's a fast, concise, and rich-featured JavaScript library and also has cross-browser compatibility. The purpose of jQuery is to make life easier for the masses so that they can easily develop websites and browser-based applications using JavaScript. In a concise manner, we can say that "JQuery is a library to provide better client-side web page development" environment to the developer with the help of its feature-rich library.

7) How We Can Select The Specified <li> Element From The ListOf <li> Elements In ?

- **To select a specific <li> element from a list of <li> elements within a <ul> (unordered list) using HTML and JavaScript, you can use various methods. Here are a couple of common approaches:**

1. **Using JavaScript and DOM Manipulation:**
   Assuming you have an HTML structure like this:

- **html :-**

```
<ul id="myList">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
```

- **You can use JavaScript to select a specific <li> element by its index:**

- **javascript :-**

```
// Select the <ul> element
var myList = document.getElementById("myList");
// Select the first <li> element (index 0)
var specificLi = myList.getElementsByTagName("li")[0];
// Now, you can manipulate the specific <li> element as needed
specificLi.style.color = "red";
```

2. Using CSS Selectors:

You can use CSS selectors to directly target a specific <li> element:

- **javascript :-**

```
// Select the first <li> element inside the <ul> with id "myList"
var specificLi = document.querySelector("#myList li:first-child");
// Now, you can manipulate the specific <li> element as needed
specificLi.style.color = "red";
```

These examples assume that you want to select the first <li> element. If you want to select a different element, change the index or adjust the CSS selector accordingly.

Remember to execute your JavaScript code after the HTML document has been fully loaded. You can either place your script at the end of the <body> tag or use the DOMContentLoaded event.

- **javascript :-**

```
document.addEventListener("DOMContentLoaded", function() {
// Your code here
});
```

8) In <table> Design Change The Color Of Even <tr> Elements To “green”And Change The Color Of Odd <tr> Elements To “blue” Color? Give An Example Code?

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <style>
    table {
      border-collapse: collapse;
      width: 100%;
    }
    table, th, td {
      border: 1px solid black;
    }
    tr:nth-child(even) {
      background-color: green;
    }
    tr:nth-child(odd) {
      background-color: blue;
    }
  </style>
  <title>Table </title>
</head>
<body>
  <table>
    <thead>
      <tr>
        <th>Header 1</th>
        <th>Header 2</th>
        <th>Header 3</th>
      </tr>
    </thead>
    <tbody>
```

```
        <tr>
          <td>Data 1</td>
          <td>Data 2</td>
          <td>Data 3</td>
        </tr>
        <tr>
          <td>Data 4</td>
          <td>Data 5</td>
          <td>Data 6</td>
        </tr>
        <tr>
          <td>Data 7</td>
          <td>Data 8</td>
          <td>Data 9</td>
        </tr>
      </tbody>
    </table>
</body>
</html>
```

9) How We Can Implement Animation Effects In Jquery?

- **jQuery Animations - The animate() Method :-**
- **The jQuery animate() method is used to create custom animations.**
- **Syntax:**
  $(selector).animate({params},speed,callback);
- **The required params parameter defines the CSS properties to be animated.**
- **The optional speed parameter specifies the duration of the effect. It can take the following values: "slow", "fast", or milliseconds.**

- **The optional callback parameter is a function to be executed after the animation completes.**
- **You can also manipulate multiple properties at the same time, define relative values, and use pre-defined values such as "show", "hide", or "toggle".**

10) Apply jQuery validation using library.

- **Form validation is a process of confirming the relevant information entered by the user in the input field. In this article, we will be validating a simple form that consists of a username, password, and a confirmed password using jQuery, along with understanding their basic implementation through the examples.**

- **Prerequisites:**
- **HTML**
- **CSS**
- **JavaScript**
- **jQuery**

- **These are the following methods for validation of form using jQuery:**

- **Table of Content** :-

- **Using plain jQuery** :-

- **First, you need to create an index.html file consisting of an HTML form with username, email, password, and confirm password as input fields.**
- **We will use Bootstrap 4 to use Bootstrap's form controls. At the bottom of the <body> tag, include the "app.js" file having jQuery code for form validation.**
- **Create an app.js file that validates the whole form as given below in the code.**
- **In the app.js file, when the document is ready, hide all the error messages.**
- **Perform the validation task for all the input fields such as username, email, password, and confirm password.**

- **Validation using an inbuilt jQuery validation plugin :-**
- **In this part, you get to know about the inbuilt validation method of jQuery. Create an index.html file and copy the following code.**

- **Using jQuery plugins :-**
- **The jQuery plugins are used to make simpler the code of validation for the clientside. The plugin comes bundled with a useful set of validation methods, including URL and email validation while providing an API to write your own methods. Follow the below example which makes more clarifications regarding it.**

11) Create custom dynamic function for require field validator.

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
<title>Untitled Document</title>
<!-- 3 file call -->
<script src="jquery-2.1.3.min.js" type="text/javascript"></script>
<script src="jquery.bvalidator.js" type="text/javascript"></script>
<link href="bvalidator.css" type="text/css" rel="stylesheet" />
<!-- add script & call FORM id -->
<script type="text/javascript">
   $(document).ready(function () {
      $('#form1').bValidator();
   });
        </script>
<script>
// now fore direct validation from above file u take data-
bvalidator="" from the file "jquery.bvalidator.js"
// allways take <form id=""  in jquery with #name
</script>
</head>
<body>
<BR>
<BR>
<BR>
<BR>
<BR>
<form id="form1" action="" method="" >
                        <table  align="center">
            <tr>
                        <th>User Fist Name:-</th>
               <td><input type="text" name="ufn" data-
bvalidator="required,alpha" ></td>
```

```
		</tr>

		<tr>
			<th>User Name:-</th>
		  <td><input type="text" name="un" data-
bvalidator="required,rangelength[3:8]" ></td>
		</tr>
		<tr>
			<th>Password:-</th>
		  <td><input type="password" id="pass" name="pass" data-
bvalidator="required,minlength[4],maxlength[8]"></td>
		</tr>
		<tr>
			<th>Confirm Password:-</th>
		  <td><input type="password" name="cpass" data-
bvalidator="required,equalto[pass]" ></td>
		</tr>
		<tr>
			<th>Email Address:-</th>
		  <td><input type="text" name="email" data-
bvalidator="required,email" ></td>
		</tr>
		<tr>
			<th>Phone No:-</th>
		  <td><input type="text" name="pno" data-
bvalidator="required,number" > </td>
		</tr>
				<tr>
			<th>Pincode No:-</th>
		  <td><input type="text" name="pno" data-
bvalidator="required,number"  ></td>
		</tr>
		<tr>
```

```
            <td><input type="submit" name="submit" class="btn
btn-primary"/></td>
          </tr>
        </table>
      </form>
</body>
</html>
```

12) Get state data by country selection (Ajax).

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Tra
      nsitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"
/>
<title>Untitled Document</title>
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.j
s"></script>
<script>
function getState(cid)
{
      if(window.XMLHttpRequest)
      {
            xmlhttp= new XMLHttpRequest();
      }
      else
      {
            xmlhttp= new ActiveXObject("Microsoft.XMLHTTP");
      }
```

```
        xmlhttp.onreadystatechange=function()
        {
                if(xmlhttp.readyState==4 && xmlhttp.status==200)
                {
        document.getElementById("sid").innerHTML=xmlhttp.response
Text;
                }
        }
xmlhttp.open("GET","statedata?btn=" + cid,true);
xmlhttp.send();
}
function getCity(sid)
{
        $.ajax
        ({
                type: "POST",
                url: "citydata",
                data:"btn=" + sid,
                success: function(data)
                {
                        $("#city_id").html(data) ;
                }
        });
}
</script>
</head>
<body>
<a href="show">Show add data</a>
<form action="" method="post">
<table border="4" align="center">
<caption>Reg Form</caption>
      <tr>
            <td>User Name</td>
```

```
            <td><input type="text" name="name" required></td>
        </tr>
        <tr>
              <td>Country</td>
              <td>
    <select name="cid" onchange="getState(this.value)" required>
<option>----Select Country----</option>
          <?php
               foreach($country_arr as $f)
                              {
           ?>
<option value="<?php echo $f->cid;?>">
       <?php echo $f->cnm; ?>
       </option>
          <?php
                }
          ?>
          </select>
          </td>
        </tr>
        <tr>
            <td>State</td>
          <td>
          <select id="sid" name="sid" onchange="getCity(this.value)"
required>
 <option>----Select State----</option>
          </select>
          </td>
        </tr>
        <tr>
            <td>City</td>
          <td>
          <select id="city_id" name="city_id" required>
```

```
            <option>----Select city----</option>
          </select>
          </td>
        </tr>
         <tr>
        <td><input type="submit" name="submit" value="Insert"></td>
        </tr>
    </table>
</form>
</body>
</html>
```

13) Image uploading with preview.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Image Upload with Preview</title>
  <style>
    #preview-container {
      display: flex;
      justify-content: center;
      align-items: center;
      height: 200px;
      border: 2px dashed #ccc;
      overflow: hidden;
    }
    #preview {
      max-width: 100%;
```

```
    max-height: 100%;
   }
  </style>
</head>
<body>
  <h1>Image Upload with Preview</h1>
  <input type="file" id="imageInput" accept="image/*">
  <div id="preview-container">
   <img id="preview" alt="Image Preview">
  </div>
  <script>
   // Function to handle file input change
   function handleFileSelect() {
    const fileInput = document.getElementById('imageInput');
    const previewContainer = document.getElementById('preview-
container');
    const previewImage = document.getElementById('preview');
    // Check if a file is selected
    if (fileInput.files && fileInput.files[0]) {
     const reader = new FileReader();
     // Set up the reader to load the image
     reader.onload = function (e) {
      previewImage.src = e.target.result;
     };
     // Read the selected file as a data URL
     reader.readAsDataURL(fileInput.files[0]);
     // Show the preview container
     previewContainer.style.display = 'flex';
    } else {
     // If no file is selected, hide the preview container
     previewContainer.style.display = 'none';
    }
   }
```

```
    // Attach the event listener to the file input
document.getElementById('imageInput').addEventListener('change',
handleFileSelect);
  </script>
</body>
</html>
```