PHP

Module - 1

Why PHP?

https://kinsta.com › php-market-share

PHP Market Share in 2023 - Kinsta®

After all, 79.2% of all websites in the world can't all be wrong, and most importantly, **PHP's market share** has remained relatively steady throughout the last ...

https://w3techs.com › technologies › details › pl-php

Usage statistics of PHP for websites - W3Techs

PHP is used by 77.7% of all the websites whose server-side programming language we know. Versions of **PHP**. This diagram shows the percentages of websites using ...

Ranking · Version History · Web Servers · CSS Frameworks

Introduction

- Officially, PHP stands for "PHP: Hypertext Preprocessor" (recursive acronym) but it is also still known around the world by its original name, Personal Home Page.
- PHP is an open source, interpreted, object oriented and server side scripting language which helps us to build web applications.
- Created by Rasmus Lerdof in 1994
- Recent Version 8.2

Features

- Simple and Easy to use
- Open source Software
- Loosely typed Language
- Cross-platform Compatible
- Flexibility
- Fast and Effective Performance

Installation

- You Need
  - A Web Server
  - A Database
  - A PHP Parser

Server	Stands for	Platform
WAMP	Windows, Apache, MySQL, PHP	Windows
LAMP	Linux, Apache, MySQL, PHP	Linux
MAMP	MAC, Apache, MySQL, PHP	MAC
SAMP	Solaris, Apache, MySQL, PHP	Solaris
XAMP	Cross, Apache, MySQL, PHP	Cross Platform

Basic Syntax

- PHP Tag: <?php ?>
- Semicolon: Every PHP statements end with semicolon (;).
- Comment Line:
  - // - For single line comment
  - # - Can also be used for single line comment
  - /* ..... */ - For multiline comment
- The $ symbol: In PHP $ symbol is used in front of all variable.
- File Extension: PHP file must be saved with the extension ".php". Older PHP file extension includes - .php3, php4, .php5 etc

HELLO WORLD

```
<!DOCTYPE html>
<html>
<head>
        <title>PHP Ex</title>
</head>
<body>
<h1>This is a PHP Script</h1>

<?php
        echo “Hello World!”;
?>

</body>
</html>
```

echo & print

echo	print
Language construct	Language construct
void echo(string $arg1,string $arg2,……)	int print(string $arg)
Faster	Slower
Can’t be a part of an expression. Gives Parse Error.	Can be part of an expression <?php $a = 10; $b = 20; $a > $b ? print "A is greater" : print "B is greater"; ?>

VARIABLE

- A variable is PHP is a name of memory location that holds data temporarily.
- In PHP, as in other languages, we can assign the data to variable.
- PHP is a loosely typed language and automatically converts the variable to its correct data type.
- Syntax: $var_name = value
- Ex:

```
<?php
    $var = 123;
    echo gettype($var); //integer
    $var = "php";
    echo gettype($var); //string
?>
```

Variable Naming Rules

- Variable names must start with a letter of the alphabet or the _ (underscore) character provided it is not a PHP reserved word.
- Variable names can contain only the characters a-z, A-Z, 0-9, and _ (underscore).
- Variable names may not contain spaces. If a variable must comprise more than one word, it should be separated with the _ (underscore) character (e.g., $user_email).
- Variable names are case-sensitive. The variable $High_Score is not the same as the variable $high_score.

Reserved Keywords

__halt_compiler()	abstract	and	array()	as
break	callable	case	catch	class
clone	const	continue	declare	default
die()	do	echo	else	elseif
empty()	enddeclare	endfor	endforeach	endif
endswitch	endwhile	eval()	exit()	extends
final	finally	fn (as of PHP 7.4)	for	foreach
function	global	goto	if	implements
include	include_once	instanceof	insteadof	interface
isset()	list()	match (as of PHP 8.0)	namespace	new
or	print	private	protected	public
readonly (as of PHP 8.1.0) *	require	require_once	return	static
switch	throw	trait	try	unset()
use	var	while	xor	yield
yield from				

VARIABLE SCOPE

- Local Variable: Created with in a function and can be accessed inside the function only.
- Static Variable: A special local variable, that does not looses the value even after execution leaves the scope.
- Global Variable: Declared outside the function and can be accessed anywhere in the script. Use global keyword to declare.
- Super Global Variable: Special global variable that can be accessed outside the script. Ex: $_GET, $_POST, $_REQUEST, $_FILES, $_SESSION

CONSTANT

- A constant is a variable whose value cannot be changed at runtime.
- PHP constants are generally defined by using define() function.
- PHP constant follow the same PHP variable rules.
- Conventionally, PHP constants should be defined in uppercase letters.
- Ex:

```
<?php
    define("PI",3.14159);
    echo PI;
?>
```

MAGIC CONSTANT

- *Magic constants are the predefined constants in PHP which get changed on the basis of their use. They start with double underscore (__) and ends with double underscore.*
- *They are similar to other predefined constants but as they change their values with the context, they are called magic constants.*

Ex:

- __LINE__ : Represent the current line number of the file
- __FILE__: Represent the full path and file name of the file.
- __FUNCTION__: Represent the function name where it is used.
- __CLASS__: Represent the class name where it is used.
- __METHOD__: Represents the name of the method where it is used.

```
<?php
echo “File Path is “.__FILE__;
echo "<br><br>";
echo "Current line number is “.__LINE__;
?>
```

DATA TYPES

- PHP store the data in its proper format without letting the user out specify the data's type. PHP supports 8 primitive data types which can be further classified into 3 types.

Scalar Type

- integer : Holds number like -1, 1 ,0
- float: Holds floating point number
- boolean: Holds True / False value
- string: Holds text

Compound Type

- Array: named and indexed collection of other variable
- Object: are instances of user define classes

Special Type

- Resource: are special variable that holds reference to resources external to the PHP
- NULL: Holds NULL value

OPERATOR

Operator Type	Operators
Arithmetic Operator	+ - * / % **
Array Operators	+ (Union) == (Equality) !=(inequality)
Assignment Operator	= += -= *= /=
Bitwise Operator	&(AND) |(OR) ^(XOR) ~(NOT)
Comparison Operator	== ===(Identical) != > < >= <=
Conditional Operator	?:
Incrementing / Decrementing Operator	$X++ ++$X $X-- --$X
Logical Operator	&& || ! and or xor
String Operator	. (concatenation) .=(Concatenation assignment)

Decision Making & Loops

FUNCTION

- A definition starts with the word function.
- A name follows, which must start with a letter or underscore, followed by any number of letters, numbers, or underscores.
- The parentheses are required.
- One or more parameters, separated by commas, are optional.
- Function names are case-insensitive

Syntax:

```
function function_name(parameters){
        // Block of code
}
```

```
<?php
//Defining a function
function welcome(){
        echo "Welcome to the world of PHP";
}

//call a function
welcome();
?>
```

FUNCTION

- In PHP we can pass the function argument in 4 ways -
  - ✓Call by value
  - ✓Call by reference
  - ✓Default argument
  - ✓Variable length argument

- Call By Value:

```
<?php
function sum($x, $y){
   return $x +$y;
}

$a = 10;
$b = 20;
echo "Sum = ".sum($a, $b);

?>
```

FUNCTION

Call By Reference:

- *It is possible to pass arguments to functions by reference.*
- *A reference to the variable is manipulated by the function rather than a copy of the variable's value.*
- *We can pass an argument by reference by adding an ampersand to the variable name in either the function call or the function definition.*
- *Ex:*

```
<?php
function add(&$x, &$y){
   $x = $x + $y;
}

$a = 10;
$b = 20;
add($a, $b);
echo "Sum = ".$a;

?>
```

FUNCTION

Default Argument:

- *PHP support default argument as other language like C++.*
- *Parameters with default arguments must be the trailing parameters in the function declaration parameter list.*
- *Ex:*

```
<?php
function add($a,$b,$c=0){
      return $a+$b+$c;
}

echo "Sum=".add(1,2);
echo "<br>Sum=".add(1,2,3);
```

FUNCTION

Variable Length Argument:

- *PHP 5.6 introduced variable-length argument lists (a.k.a. varargs, variadic arguments), using the … token before the argument name to indicate that the parameter is variadic, i.e. it is an array including all supplied parameters from that one onward.*

- *Ex:*

```
function add(...$nums){
    $sum=0;
    for($i=0; $i<count($nums); $i++){
        $sum+=$nums[$i];
    }
    return $sum;
}

echo "Sum=".add(1,2);
echo "<br>Sum=".add(1,2,3);
```

STATIC VARIABLE

Ex:

```
<?php
    function counter(){
        static $count=0; //static variable
        $count++;
        echo "<br>Counter value ".$count;
    }
    counter();
    counter();
    counter();
?>
```

ARRAY

- Arrays are collections of data items stored under a single name.
- PHP array can store heterogeneous data.
- There are three types of array in PHP -
  - ✓ Indexed or Numerical array
  - ✓ Associative array
  - ✓ Multidimensional Array

NUMERICAL ARRAY

- An array with a numeric index.
- These arrays can store numbers, strings and any object but their index will be represented by numbers.
- By default array index starts from zero.
- Indexed array can be defined in two ways.
  - ✓ By using the array function- $arr=array(1,2,3,4,5);
  - ✓ By manually accessing the array index - $arr[0]=1; $arr[1]=2; $arr[2]=3;
- An array element can be accessed or modified by using the array index.
- Ex:

```
$arr=array("BCA", "BBA", "MCA", "MBA");
echo  $arr[1]; //BBA
$arr[1]= "B.Tech";
echo  $arr[1]; //B.Tech
```

NUMERICAL ARRAY

Handling array using for Loop:

```
<?php
$arr=array(10,20,30,40,50);
for($i=0;$i<count($arr);$i++)
        echo $arr[$i]," ";

?>
```

O/P: 10 20 30 40 50

Handling array using foreach loop:

```
<?php
$arr=array(10,20,30,40,50);
foreach($arr as $a)
        echo $a." ";
?>
```

O/P: 10 20 30 40 50

ASSOCIATIVE ARRAY

- The associative arrays are very similar to numeric arrays in term of functionality but they are different in terms of their index.
- Associative array will have their custom index (as string mostly) so that you can establish a strong association between key and values.
- The => operator is used to create key/value pairs in array.
- The item on the left of the => is key and the item on the right is the value.
- Create Associative Array:
  - $arr=array($key1=>$value1,$key2=>$value2, $key3 => $value3)
  - $arr[key1] = value1; $arr[key2]=value2; $arr[key3]=value3;
- Ex:
  - $arr=array("Anil"=>20,"Sunita"=>21,"Bimal"=>19,"Sam"=>21,"Tom"=>20);

ASSOCIATIVE ARRAY

Accessing associative array using foreach loop:

```
$arr=array("Anil"=>20,"Sunita"=>21,"Bimal"=>19,"Sam"=>21,"Tom"=>20);
foreach($arr as $name => $age)
        echo "Age of $name is $age<br>";
```

O/P:

Age of Anil is 20

Age of Sunita is 21

Age of Bimal is 19

Age of Sam is 21

Age of Tom is 20

MULTIDIMENSIONAL ARRAY

- PHP multidimensional array is also known as array of arrays.
- It allows you to store tabular data in an array.
- PHP multidimensional array can be represented in the form of matrix which is represented by row x column.
- Ex:

```
<?php
$arr  = array(
    array(1,2,3),
    array(4,5,6),
    array(7,8,9)
);
print_r($arr);
echo $arr[0][2];
echo $arr[2][1];
```

ARRAY FUNCTIONS

Function	Use
array array(value1, value 2··)	Create and returns an indexed array
array array(k1=>v1,k2=>v2...)	Create and returns an associative array
int count(array)	Count the number of items in the array
min(array)	Returns the minimum value of an array
max(array)	Returns the highest value in an array
array_sum(array)	Returns the sum of the values in an array
bool sort(array)	Sort an indexed array in ascending order
bool rsort(array)	Sorts an indexed array in descending order·
bool asort(array)	Sorts an associative array in ascending order, according to the value
bool arsort(array)	Sorts an associative array in descending order, according to value

ARRAY FUNCTIONS

Function	Use
bool ksort(array)	Sorts an associative array in ascending order according to the key
bool krsort(array)	Sorts an associative array in descending order according to key
array array_reverse(array)	Returns an array in reverse order
array_search($item, $array)	Searches an array for a given value and returns the key or false if not found
bool in_array($item,array)	Checks if a specified value exists in an array
array array_keys()	Returns all the keys of an array
array_values()	Returns an array containing all the values of an array
array_rand(array, [no_of_elemen])	Returns one or more random keys from an array
array array_flip(array)	Exchange all keys with their corresponding values
array array_marge(arr1, arr2,...)	Marge one or more array into one array

ARRAY FUNCTIONS

Function	Use
int array_push(array, v1, v2, ...)	Insert one or more element to the end of an array and returns the new number of elements
array_pop(array)	Delete and return the last element of an array
array_shift(array)	Removes the first element from an array, and returns the value of the removed element.
array_unshift(array, value1, value2, value3, ...)	Inserts new elements at the beginning of an array.
array array_slice(array, start, [length], [preserve])	Returns selected parts of an array.
array_splice(array, start, [length], [array2])	Removes selected elements from an array and replaces it with new elements. If length=0, the replaced array will be inserted at the starting position.

DATE TIME

- The PHP date() function is used to format a date and/or a time.

  date(string $format, ?int $timestamp = null): string

- format: is the general format which we want our output
- timestamp: Default value: time(). The optional timestamp parameter is an integer Unix timestamp in seconds between the current time and value as at 1st January, 1970 00:00:00 Greenwich Mean Time (GMT).
- Ex:

```
<?php
echo date("Y-m-d");
?>
```

DATE TIME

Parameter	Meaning
r	Returns the full date and time
d	Returns the day of the month with leading zeroes [01 to 31]
D	Returns the first 3 letters of the day name [Sun to Sat]
l(small L)	Returns day name of the week [Sunday to Saturday]
m	Returns the month number with leading zeroes [01 to 12]
M	Returns the first 3 letters of the month name [Jan to Dec]
F	Returns the month name [January to December]
y	Returns two (2) digits year format (00 to 99)
Y	Returns four digit year format
a, A	Returns whether the current time is am or pm, AM or PM respectively
h, H	Returns the hour with leading zeros [01 to 12],[00 to 23] respectively
i	Returns the minutes with leading zeroes [00 to 59]
s	Returns the seconds with leading zeroes [00 to 59]

DATE TIME

Example:

date_default_timezone_set('Asia/Kolkata');	Set time zone for India
echo date("r");	Fri, 03 Feb 2023 09:40:51 +0530
echo date("Y-m-d");	2023-02-03
echo date("H:i:s");	09:40:51
echo date("h-m-d A");	09-02-03 AM
echo date("F j, Y, h:i a");	February 3, 2023, 09:40 am

STRING

- A string is series of characters, where a character is the same as a byte.
- As of PHP 7.0.0, there are no particular restrictions regarding the length of a string on 64-bit builds.
- On 32-bit builds and in earlier versions, a string can be as large as up to 2GB (2147483647 bytes maximum)
- Single quoted String: The simplest way to specify a string is enclosed it in single quotes (' '). To specify a literal single quote, escape it with a backslash.
  - ✓ Ex: $str= 'String defined using single quote'
- Double quoted string: The double quotes are used to create relatively complex strings compared to single quotes. Variable names can be used inside double quotes and their values will be displayed.
- Ex:

```
$version=5;
echo "Current PHP version is: $version";
```

STRING FUNCTIONS

Function	Application
strlen($str)	Returns the length of the string
str_word_count($str)	Counts the number of words in the string
strrev($str)	Reverse a string
strpos($str,$search_key)	Search a specific text. If match found, it will return the 1st matching index otherwise return false.
str_replace($char,$replacing_char,$str)	Replace some character with some other character.
substr($string,$start,[$length])	Returns an extracted part of the string on successful executing otherwise return false. Length is optional. If start value is -ve, parsing will start form the end

STRING FUNCTIONS

Function	Application
ltrim($str,[$char])	Strip white space or newline character form left of the string
rtrim($str,[$char])	Strip white space or newline character form right of the string
trim($str,[$char])	Strip white space or newline character form both left and right of the string
join($separator,$array)	Returns string from the elements of an array.
implode($separator,$array)	Same as join but binary safe
explode($separator,$str)	Breaks a string into array based on the separator value
str_split($str,[$length])	Breaks a string into array based on length variable. If length is not specified, default value is 1.

STRING FUNCTIONS

Function	Application
crypt($str,[$salt])	Return hashed string based on DES based algorithm.
md5($str,[$raw])	Return the hashed string as 32 character hexadecimal number. $raw is either true or false. If true md5() will return 16 character binary format.
strtoupper($str)	Converts the string to upper case
strtolower($str)	Converts the string to lower case
lcfirst($str)	Converts the first character of a string to lower case
ucfirst($str)	Converts the first character of a string to upper case
ucwords($str)	Converts the first character of each word in a string into upper case.

STRING FUNCTIONS

Function	Application
str_repeat(string, no_of_repetation)	Repeat the string for multiple times.
strcmp(str1, str2)	Perform case sensitive comparison of two string. Returns, < 0 if str1 is less than str2; > 0 if str1 is greater than str2, and 0 if they are equal.
strcasecmp(str1, str2)	Perform case in-sensitive comparison of two string. Returns, < 0 if str1 is less than str2; > 0 if str1 is greater than str2, and 0 if they are equal.

HTML FORM

HTML Form Tags	
Tag	**Description**
<form>	It defines an HTML form to enter inputs by the user side.
<input>	It defines an input control.
<textarea>	It defines a multi-line input control.
<label>	It defines a label for an input element.
<select>.	It defines a drop-down list
<option>	It defines an option in a drop-down list.
<button>	It defines a clickable button
HTML Input Tag Attributes	
<input type="text">	Defines a one line text input field
<input type="radio">	Defines a radio button(For selecting one from multiple options)
<input type="checkbox">	Defines a checkbox(For selecting multiple options)
<input type="submit">	Defines a submit button.(For submitting the form)

HTML FORM Attributes

Attribute	Description
name	Specifies a name used to identify the form.
action	Specifies an address (url) where to submit the form (default: the submitting page). Ex: <form action="action_page.php">
method	Specifies the HTTP method used when submitting the form (default: GET). Ex: <form action="action_page.php" method="get"> or <form action="action_page.php" method="post">
target	Specifies the target of the address in the action attribute (default: _self).
enctype	Specifies the encoding of the submitted data (default: is url-encoded).

GET vs POST

GET	POST
Parameter are passed to the target page by appending them with the URI.	Parameter are passed to the target page by encoding them with the request object.
Parameters remain in browser history as they are part of URI.	Parameters are not saved in browser history.
As parameters are part of URL, they are visible to the external user.	Parameters are never visible to the external user.
As parameters are visible and are stored in browser log, GET is less secure.	POST is secure than GET.
Performance is little bit faster than POST.	Take a little bit more time than GET to encapsulate the parameters with the request object.

Capturing Form Data

- To access the value of a particular form field, you can use the following super global variables·
  - ✓$_GET - Contains a list of all the field names and values sent by a form using the get method
  - ✓$_POST - Contains a list of all the field names and values sent by a form using the post method
  - ✓$_REQUEST - Contains the values of both the $_GET and $_POST variables

Capturing Form Data

HTML Form:

```
<form action="action.php" method="post">
 <p>Your name: <input type="text" name="name" /></p>
 <p>Your age: <input type="text" name="age" /></p>
 <p><input type="submit" /></p>
</form>
```

action.php Page

```
<?php
    $name = $_POST['name'];
    $age = $_POST['age'];
    echo "Hi $name, You are $age years old"
?>
```

PHP isset()

- The isset() function checks whether a variable is set, which means that it has to be declared and is not NULL.
- This function returns true if the variable exists and is not NULL, otherwise it returns false.

```
<?php
$a = 0;
if (isset($a)) {
  echo "Variable 'a' is set";
}
$b = null;
if (isset($b)) {
  echo "Variable 'b' is set.";
} else {
   echo "Variable 'b' is not set.";
}
?>
```

```
O/P:
Variable 'a' is set
Variable 'b' is not set.
```

Fetching Data From Multivalued Field

- A multi-select list box or a group of check box allows user to select multiple valus
- The trick is to add square brackets ([]) after the field name in HTML form to fetch all the data to PHP page.
- When PHP engine sees a submitted form field name with square brackets at the end, it creates a nested array of values within the $_GET or $_POST and $_REQUEST super global array, rather than a single value.

```
<form action="action.php" method="post">
 <p>
   Language Known
   <input type="checkbox" name="lang[]" value="English">English
   <input type="checkbox" name="lang[]" value="Hindi">Hindi
   <input type="checkbox" name="lang[]" value="Spanish">Spanish
 </p>
 <p><input type="submit" /></p>
</form>
```

```
<?php
  $languages  = $_POST['lang'];
  foreach($languages as $l){
      echo "$l ";
   }
 ?>
```

Page Redirection

- In PHP, redirection is done by using header() function as it is considered to be the fastest method to redirect traffic from one web page to another.
- The main advantage of this method is that it can navigate from one location to another without the user having to click on a link or button.

```
<html>
<body>
<form action="controller.php" method="post">
Name: <input type="text" name="uname"> <br><br>
<input type="submit" name="submit">
</form>
</body>
</html>
```

```
<?php
if(isset($_POST['submit'])){
        $name=$_POST['uname'];
        if($name==""){
                header("location:info.html");
        }else{

        header("location:thankyou.html");
        }
}
```

PHP OOP

As object oriented programming is faster and easy to execute, PHP introduced object oriented programing in PHP 5.

Class:

A class is defined by using the 'class' keyword, followed by the name of the class and a pair of curly braces ({}).

Ex:

```
class Student{
        // Code to be executed
}
```

Object:

We can create multiple objects from a class. Each object has all the properties and methods defined in the class, but they will have different property values. Objects of a class is created using the new keyword.

Ex:

```
class Student{
        // Code to be executed
}
$std=new Student();
```

Data Member and Member Function:

Variable declaration is almost same as procedural PHP. But here with the variable name we need to specify the access modifier.

In PHP 3 access modifiers available.

- public - the property or method can be accessed from everywhere. This is default. Public data member can also be defined with 'var' keyword.
- protected - the property or method can be accessed within the class and by classes derived from that class
- private - the property or method can ONLY be accessed within the class

Ex: private $var1; public $var2;

Function definitions look much like standalone PHP functions but are local to the class. Functions will be used to set and access object data member.

Once the data and member function is declared they can be accessed by the following syntax.

obj_name -> data_member;

obj_name -> member_function();

In PHP, **$this** keyword references the current object of the class. The $this keyword allows you to access the properties and methods of the current object within the class using the object operator (->)

Ex 1:

```
<?php
class Fruit {
  public $name;
  protected $color;
  private $weight;
}

$mango = new Fruit();
$mango->name = 'Mango'; // OK
$mango->color = 'Yellow'; // ERROR
$mango->weight = '300'; // ERROR
?>
```

Ex 2:

```
class Student{
        private $name;
        private $roll;
        function set_name($name){
            $this->name = $name;
        }
        function set_roll($roll){
            $this->roll=$roll;
        }
function get_info(){
            return $this->name." ".$this->roll;
        }
}
$s = new Student();
$s->set_info("Will", 2);
echo $s->get_info();
```

Constructor:

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.

Destructor:

A destructor is called when the object is destructed or the script is stopped or exited.

If you create a __destruct() function, PHP will automatically call this function at the end of the script.

Note:

- constructor and destruct function starts with two underscores (__)
- Before PHP 5 Constructor have the same name as class name
- __construct and __destruct is called magic methods

Ex:

```
<?php
   class Student{
      private $name, $roll;

      function __construct($name, $roll){
         $this->name = $name;
         $this->roll = $roll;
         echo "Student Object is Created";
      }

      function __destruct()
      {
         echo "<br>Obj Destroyed";
      }
   }
   $s = new Student("Jhon", 1);
   unset($s);
   var_dump($s);
?>
```

The static Keyword:

The static keyword is used to declare properties and methods of a class as static. Static properties and methods can be used without creating an instance of the class.

These data member and member functions can also be accessed statically within an instantiated class object.

A class can have both static and non-static methods. A static method can be accessed from a method in the same class using the self keyword and double colon (::)

Ex:

```
<?php
class Employee{
   public static $eid;
   public static $name;

   static function get_info(){
      echo self::$name." ".self::$eid;
   }
}
Employee::$name = "James";
```

```
Employee::$eid = 101;
Employee::get_info();
?>
```

Inheritance:

Inheritance is a mechanism of extending an existing class by inheriting a class. We create a new class with all functionality of that existing class, and we can add new members to the new class.

When we inherit one class from another we say that inherited class is a subclass and the class who has inherit is called parent class.

We declare a new class with additional keyword extends.

Ex: class B extends A

PHP classes supports Single, Multilevel and Hierarchical inheritance but does not support Multiple inheritance. Multiple inheritance can be achieved by using interfaces.

Single Inheritance Example:

```
<?php
   class Student{
      protected $roll;
      protected $name;
      function set_roll_name($r, $n){
         $this->name = $n;
         $this->roll = $r;
      }
   }

   class MCA extends Student{
      protected $branch;
      function __construct($roll, $name){
         $this->set_roll_name($roll, $name);
         $this->branch = "MCA";
      }
      function display(){
         echo $this->name." ".$this->roll." ".$this->branch;
      }
   }
   $s1 = new MCA(1, "Lily");
   $s1->display();
?>
```

How to Call the Parent Constructor?

If the child class does not have its own constructor, then the base class constructor will automatically be called. But if both the child and parent class have constructors, the constructor of the child class doesn't automatically call the constructor of its parent class. Use

parent::__construct(arguments) to call the parent constructor from the constructor in the child class.

```
<?php
        class Student{
        protected $roll;
        protected $name;

        function __construct($r, $n){
            $this->name = $n;
            $this->roll = $r;
        }
    }

    class MCA extends Student{
        protected $branch;
        function __construct($roll, $name){
            $this->branch = "MCA";
            parent::__construct($roll, $name);
        }

        function display(){
            echo $this->name." ".$this->roll." ".$this->branch;
        }
    }

    $s1 = new MCA(1, "Lily");
    $s1->display();
?>
```

Multilevel Inheritance Example:

```
 <?php
    class Student{
        protected $name;
        protected $roll;

        function __construct($n, $r){
            $this->name = $n;
            $this->roll = $r;
        }
    }

    class MCA extends Student{
        protected $branch;

        function __construct(){
            $this->branch = "MCA";
        }
    }
```

```
   Class FirstMCA extends MCA{
      protected $year;

      function __construct($n, $r){
         $this->year= "1st";
         parent::__construct();
         Student::__construct($n, $r);
      }
      function get_info(){
         echo $this->name." ".$this->roll." ".$this->branch." ".$this->year;
      }
   }

   $sam = new FirstMCA("Sam", 1);
   $sam->get_info();
?>
```

Abstract Class and Methods:

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. An abstract class or method is defined with the abstract keyword. An abstract class can have properties and methods as a regular class. But it cannot be instantiated.

In most cases, an abstract class will contain at least one abstract method though it is not required. If a class contains one or more abstract methods, it must be an abstract class. If a class extends an abstract class, it must implement all abstract methods or itself be declared abstract.

Ex: Abstract class and Hierarchical Inheritance

```
<?php
   abstract class Human{
      protected $name;
      function set_name($n){
         $this->name = $n;
      }
      abstract function get_info();
   }

   class Male extends Human{
      protected $gender;
      function __construct(){
         $this->gender = "Male";
      }

      function get_info(){
         echo $this->name." ".$this->gender;
      }
   }
```

```
   class Female extends Human{
      protected $gender;
      function __construct()
      {
         $this->gender = "Female";
      }
      function get_info(){
         echo $this->name." ".$this->gender;
      }
   }

   $jhon = new Male();
   $jhon->set_name("Jhon");
   $jhon->get_info();

   $jane = new Female();
   $jane->set_name("Jane");
   $jane->get_info();
?>
```

Interface:

An interface is similar to a class except that it cannot contain code. An interface can define method names and arguments, but not the contents of the methods. Any classes implementing an interface must implement all methods defined by the interface. A class can implement multiple interfaces. An interface is declared using the "interface" keyword. An interface consists of methods that contain no implementation. In other words, all methods of the interface are abstract methods. An interface can also include constants.

Ex: Interface and Multiple Inheritance

```
<?php
   interface Calcualtor{
      function add();
      function sub();
   }

   interface TestInterface{

   }

   class TestClass{
      function test_func(){
         echo "Test Function is called";
      }
   }

   class TestCalc extends TestClass implements Calcualtor, TestInterface {
      protected $x;
```

```
    protected $y;
    function __construct($x, $y)
    {
       $this->x = $x;
       $this->y = $y;
    }
    function add(){
       return $this->x + $this->y;
    }

    function sub(){
       return $this->x - $this->y;
    }
  }

  $c = new TestCalc(10, 5);
  echo $c->add();
  echo "<br>";
  echo $c->sub();
  echo "<br>";
  $c->test_func();
?>
```

Abstract Class vs Interface:

- Abstract class can have abstract methods and non-abstract methods, but interface can only have abstract mentors,
- 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.
- A class inherits from an abstract class by using extends keyword, but inherits from interface using implements keyword.
- Interface supports multiple inheritance, but abstract does not.

Exception

An error is an unexpected program result that cannot be handled by the program itself. An exception is unexpected program result that can be handled by the program itself. Examples of exception include trying to open a file that does not exist. This exception can be handled by either creating the file or presenting the user with an option of searching for the file.

Why handle exception?

- Avoid unexpected results on our pages which can be very annoying or irritating to our end users

- Improve the security of our applications by not exposing information which malicious users may use to attack our applications
- PHP Exceptions are used to change the normal flow of a program if any predictable error occurs.

Exception handling is almost similar in all programming languages. It changes the normal flow of the program when a specified error condition occurs, and this condition is known as exception. PHP offers the following keywords for this purpose:

try -The try block contains the code that may have an exception or where an exception can arise. When an exception occurs inside the try block during runtime of code, it is caught and resolved in catch block. The try block must be followed by catch or finally block. A try block can be followed by minimum one and maximum any number of catch blocks.

catch -The catch block contains the code that executes when a specified exception is thrown. It is always used with a try block, not alone. When an exception occurs, PHP finds the matching catch block.

throw - It is a keyword used to throw an exception. It also helps to list all the exceptions that a function throws but does not handle itself.

finally - The finally block contains a code, which is used for clean-up activity in PHP. Basically, it executes the essential code of the program.

Example:

```
<?php
function div($a, $b){
   try{
      if($b==0){
         throw new Exception("Divisor Can not be zero");
      }
   } catch(Exception  $e){
      echo $e;
      return "";
   } finally {
      echo "Finally Called";
   }
   return $a/$b;
}
$a = 10;
$b = 0;

echo div($a,$b);
?>
```

Creating a Custom Exception Class

To create a custom exception handler you must create a special class with functions that can be called when an exception occurs in PHP. The class must be an extension of the exception class.

The custom exception class inherits the properties from PHP's exception class and you can add custom functions to it.

Example:

```
<?php
class CustomException extends Exception{
   function getError(){
      echo $this->getMessage()." at line number ".$this->getLine();
   }
}

$a=3;
$b=0;
try{
   if($b==0){
      throw new CustomException("Divisor Can not be zero");
   }
   if($b==1){
      throw new Exception("Divisor Can not be 1");
   }
} catch(CustomException  $ce){
   echo $ce->getError();
} catch(Exception  $e){
   echo $e;
}
?>
```

Example 2:

```
<?php
class OddNumberException extends Exception{
   function getMyMessage(){
      echo "This is an Odd number";
   }
}

try{
   $num = 41;
   if($num%2==0){
      echo "This is an even number";
   } else {
      throw new OddNumberException("Odd Number");
   }
} catch(OddNumberException $oe){
   $oe->getMyMessage();
}
?>
```

PHP

SESSION AND COOKIE

What is Session Tracking ?

- HTTP is a stateless and volatile protocol
- When there is a need to maintain the conversational state, session tracking is needed.
- Session tracking is a mechanism that servers use to maintain state about a series of requests from the same user (that is, requests originating from the same browser) across some period of time.

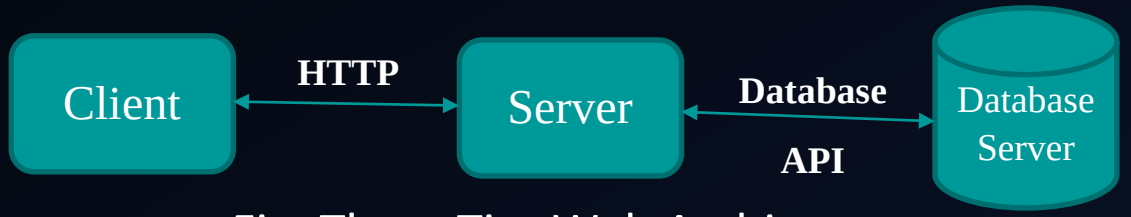
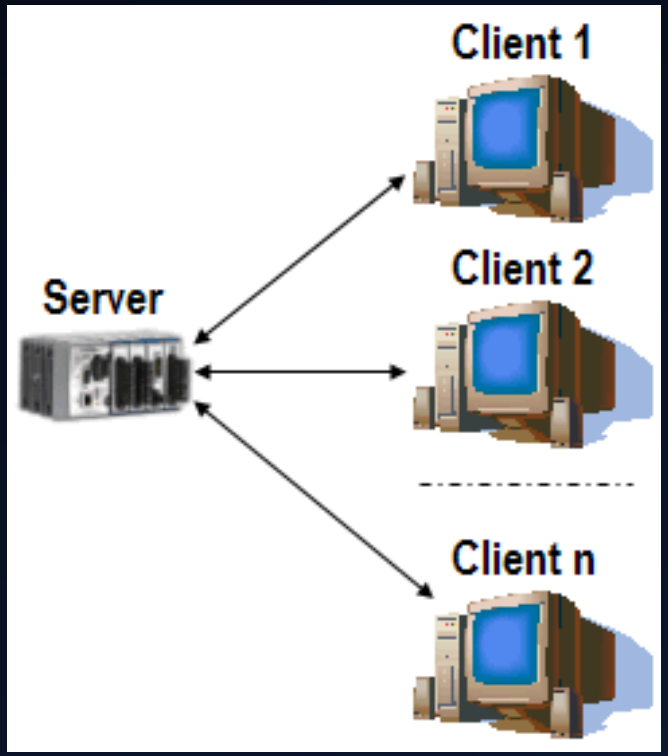
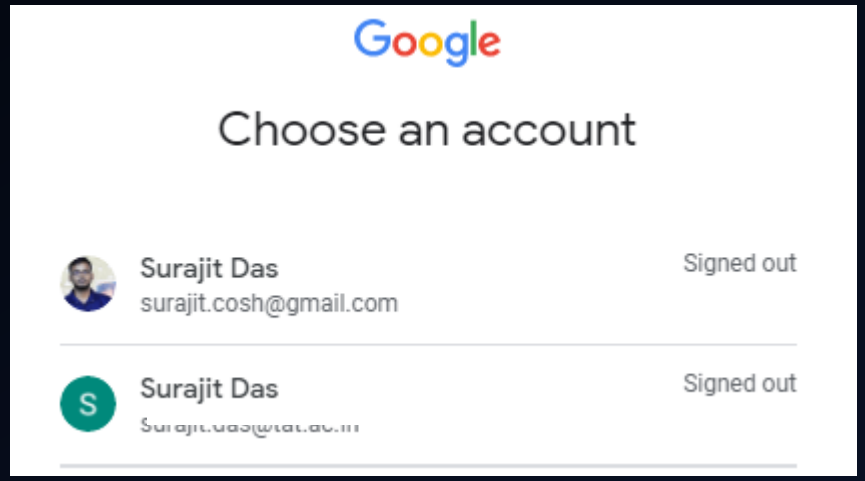


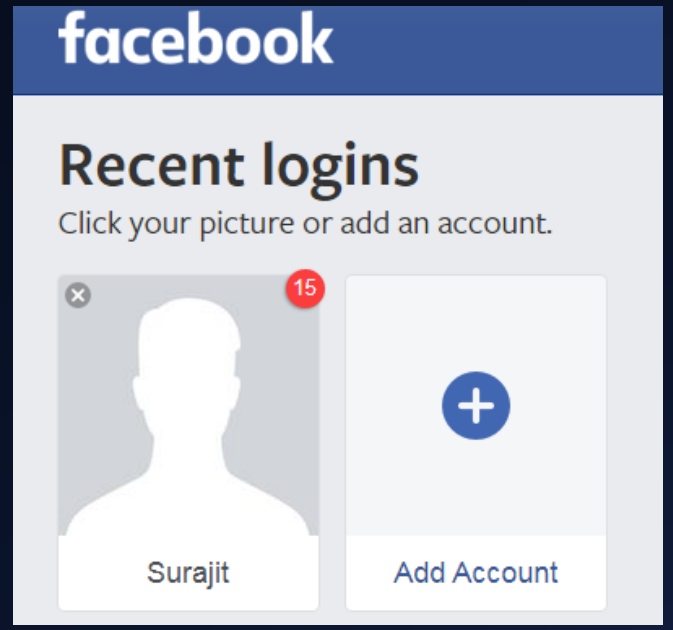
Fig: Three Tire Web Architecture



COOKIE:

- A cookie is a small text segment that the web server stores on the client computer.
- Usually cookies are stored in browser's catch memory
- When the same browser sends any request to the same server then it sends those cookie information to the server and server uses that information to identify the user.





Create Cookie with PHP:

- A cookie in PHP can be created by using setcookie() function.

  setcookie(name, value,[expire, path, domain, secure])

- Name: The name of the cookie, used as cookie variable name.
- Value: Sets the value of the named variable. The value must be a string type data.
- Expire: Specify a future time in seconds since 00:00:00 GMT on 1st Jan, 1970. After this time cookie will become inaccessible. If the parameter is not set, then cookie will automatically expire when the web browser is closed.
- Path: The path on the server in which the cookie will be available on.
- Domain: The domain for which the cookie is available.
- Secure: Indicates that the cookie should only be transmitted over a secure HTTPs connection. When set to 1, cookie will be sent over HTTPs otherwise set to 0, means cookie can be sent by regular HTTP.

- Ex: setcookie("branch", "MCA", time()+120);

Types of Cookie:

- Persistence Cookie
  - Cookie with expire time
  - Stays on clients computer even after closing the browser
  - setcookie("branch","MCA",time()+120);
- Non-Persistence Cookie
  - Cookie without expire time
  - Automatically removed from the client's computer when browser is closed
  - Ex: setcookie("branch","MCA“);

- time():built-in function in PHP which returns the current time measured in the number of seconds since 00:00:00 GMT on 1st Jan, 1970.

Retrieve a Cookie with PHP:

- Once the cookie is set, they can only be accessed on the next page load. PHP cookies can be accessed by either using the super global variable $_COOKIE[“cookie_name”] or by using $HTTP_COOKIE_VARS variable.

Ex:

```
<?php
setcookie("branch","MCA",time()+120);
echo "Cookie value is: ".$_COOKIE["branch"];
?>
```

Output:

On First load:

Notice: Undefined index: branch in **C:\xampp\htdocs\testproject\cookie1.php** on line **8**

On 2nd Load onwards:

Cookie value is: MCA

To overcome the error on first load, we can rewrite the code in the following way–

Ex:

```
<?php
setcookie("branch","MCA",time()+120);

if(isset($_COOKIE["branch"])){
        echo "Cookie value is: ".$_COOKIE["branch"];
}else{
        echo "No Cookie Value is set";
}
?>
```

Output:

On First load:

No Cookie Value is set

On 2nd Load onwards:

Cookie value is: MCA

Modify Cookie:

- we can use the same setcookie() function with the same name and the modified values.

Ex:

```
<?php
setcookie("branch","MBA",time()+(365*86400));
if(isset($_COOKIE["branch"])){
        echo "Cookie value is: ".$_COOKIE["branch"];
}else{
        echo "No Cookie Value is set";
}
?>
```

Output:

Cookie value is: MBA

Delete Cookie:

- To destroy a cookie before its expiry time, we need to set the expiry time to a time that has already passed.

Ex:

```
<?php
setcookie("branch","MBA",time()-3600);
?>
```

Ex: Last visit date time

```
<?php
 date_default_timezone_set("Asia/kolkata");
if(!isset($_COOKIE['last_visit'])){
    echo "Welcome to our Website";
}else{
   echo "Welcome back to our page";
   echo "<br>You have last visited us on: ".$_COOKIE['last_visit'];
}
setcookie('last_visit',date('d-m-Y h:i:s a'),time()+360*86400);
?>
```

O/P:

On 1st load:

Welcome to our Website

On 2nd load onwards:

Welcome back to our page

You have last visited us on: 09-02-2020 07:33:15 am

Disadvantages of Cookie:

- Clients can disable cookies on their browsers.
- cookie can contain a very limited amount of information (not more than 4 kb).
- Most browsers restrict the number of cookies that can be set by a single domain to not more than 20 cookies.
- Less secure

SESSION:

- Session is a global variable, stored at server, which help us to access data across the various pages of an entire website.
- A session creates a file in a temporary directory on the **server**.
- When a session is started following things happen –
  - PHP first creates a unique identifier for that particular session which is a random string of 32 hexadecimal numbers.
  - A cookie called PHPSESSID is automatically sent to the user's computer to store unique session identification string.
  - A file is automatically created on the server in the designated temporary directory and bears the name of the unique identifier prefixed by sess_ i.e, sess_3c7foj34c3jj973hjkop2fc937e3443.

Create Session:

- A PHP session is easily started by making a call to the session_start() function. This function first checks if a session is already started and if none is started then it starts one.
- Session variables are stored in associative array called $_SESSION[ ].
- Syntax: $_SESSION[‘variable_name’] = ’value’;

Ex:

```
<?php
session_start();
$_SESSION['user_name']='Ram Kumar';
echo $_SESSION['user_name'];
?>
```

O/P:

Ram Kumar

Ex: Sharing session in multiple pages

session_first.php

```
<?php
session_start();
$_SESSION['name']="Ram Kumar";
?>
<a href="session_second.php">Next</a>
```

session_second.php

```
<?php
session_start();
$_SESSION['roll']=123456;
?>
<a href="session_third.php">Next</a>
```

session_third.php

```
<?php
session_start();
echo "Name: ".$_SESSION['name'];
echo "<br>Roll: ".$_SESSION['roll'];
?>
```

Destroying Session:

- session_destroy() - a single call can destroy all the session variables
- unset(‘session_name’) - to unset a particular session variable.

Ex:

```
<?php
session_start();
unset($_SESSION[‘name’]);  //Delete a single session variable
session_destroy();  //Destroy the session(Delete all session variables)
?>
```

Ex: Page Visit Count Using Session

```
<?php
session_start();
if(isset($_SESSION['count'])){
        $c=$_SESSION['count'];
        $c++;
        $_SESSION['count']=$c;
}else{
        $_SESSION['count']=1;
}
echo "Number of visit: ".$_SESSION['count'] ;
?>
```

O/P:
Number of visit: 5

Difference between Cookie & Session:

Cookie	Session
Cookies are stored as a text file at the client side.	Sessions are stored at the server side.
Cookie is limited by the size and can only store 4KB of data.	Session can store unlimited amount of data.
Storing Multiple value is difficult	Storing multiple value by using session become easier.
As cookie stored at the client side, cookie is less secure.	As session stored at the server end, session is more secure than cookie.

FILE HANDLING WITH PHP

OPENING A FILE

- Since PHP is a server side programming language, it allows to create, access, and manipulate files and directories on the web server using the PHP file system functions
- The PHP fopen() function is used to open a file.
- resource fopen (string $fname , string $mode)

Modes	Description
r	Open a file for read only.
w	Open a file for write only. Erases the contents of the file or creates a new file if it doesn't exist.
a	Open a file for write only. The existing data in file is preserved. File pointer starts at the end of the file. Creates a new file if the file doesn't exist
x	Creates a new file for write only. Returns FALSE and an error if file already exists
r+, w+, a+, x+	Open a file for read/write.

USEFUL FUNCTIONS

Functions	Description
filesize(filename)	Returns the size of a file.
boolean file_exists(path)	checks whether a file or directory exists.
string fread(resource, length)	Used to read data of the file. the second parameter specifies the maximum number of bytes to read.
string fgets(resource)	Used to read single line from the file.
string fgetc(resource)	Used to read single character from the file.
int readfile(file_name)	Reads a file and writes it into the output buffer. Returns the number of bytes read from the file on success or false on failure. This function is useful if all you want do to is open up a file and read it's contain.
int fwrite(resource, string)	Writes the contents of string to the file system pointed by the resource. Returns the number of bytes written or false on error.

USEFUL FUNCTIONS

Functions	Description
boolean fclose(resource $handle)	The file pointed to by handle is closed. Returns true on success or false on failure.
copy(source_file, destination_file)	Used to copy file. Returns true on success or false on failure.
unlink(file_name)	The unlink function is used to delete the file. Returns true on success or false on failure.
rename(old_name, new_name)	Rename a file with a new name. Returns true on success or false on failure.
mkdir(path)	Attempts to create a directory with the name provided with the path. Returns true on success or false on failure.
rmdis(directory_name)	Attempts to remove the directory with the given name. Returns true on success or false on failure.

EXAMPLES

Writing into file	Reading from the file	Append Data to the file

$fp=fopen("info.txt",'w');
$str="Hello User !! Welcome to the world of PHP.";
fwrite($fp,$str);
fclose($fp);
	$fp=fopen("info.txt",'r');
	$content = fread($fp,filesize("info.txt"));
	echo $content;
	fclose($fp);
		$fp=fopen("info.txt",'a');
		$str="In this course you will learn PHP";
		fwrite($fp,$str);
		readfile("info.txt");
		fclose($fp);

Copy a file	**Delete a file**	**Create Directory**

if(copy("info.txt","php.txt")){
echo "File Copied Successfully";
}else{
echo "File not copied. Something wrong";
}
	if(unlink("php.txt"))
	echo "File deleted";
	else
	echo "Error while deleting the file";
		if(mkdir("Temp"))
		echo "Directory created";
		else
		echo "Error while creating the directory";

PHP INCLUDE

- PHP include helps us to take all the text/code/markup that exists in the specified file and copies it into the file that uses the include statement.
- Including files is very useful when you want to include the same PHP, HTML, or text on multiple pages of a website.
- PHP supports two ways for content inclusion-
  - include() - upon failure produce a warning (E_WARNING) and the script will continue
  - require() - upon failure produce a fatal error (E_COMPILE_ERROR) and stop the script
- Syntax - include '*filename*'; OR require '*filename*';

PHP INCLUDE

- The include_once() function can be used to include a PHP file in another one, when you may need to include the called file more than once. If it is found that the file has already been included, calling script is going to ignore further inclusions.
- require_once() function can be used to include a PHP file in another one, when you may need to include the called file more than once. If it is found that the file has already been included, calling script is going to ignore further inclusions.

PHP FILE UPLOAD

- File upload in PHP allows you to upload files with different extensions to the server.
- We can use HTML forms and enable the users to upload files to the server.
- These files are stored in a temporary directory unless moved to a target location for permanent storage.
- Some important checks before uploading a file –
  - The form need to have method attribute set to **post** and enctype attribute is set to **multipart/form-data**
  - In your "php.ini" file, search for the file_uploads directive, and set it to On

PHP FILE UPLOAD

- The PHP global $_FILES contains all the information of file. By the help of $_FILES global, we can get file name, file type, file size, temp file name and errors associated with file.
  - $_FILES['filename']['name]: returns file name.
  - $_FILES['filename']['type']: returns MIME type of the file.
  - $_FILES['filename']['size']: returns the size of the file in bytes.
  - $_FILES['filename']['tmp_name']: returns temporary file name of the file which was stored on the server.
  - $_FILES['filename']['error']: returns error code associated with the file.

- boolean move_uploaded_file(filename, destination): moves the uploaded file to a new location. The move_uploaded_file() function checks internally if the file is uploaded thorough the POST request. It moves the file if it is uploaded through the POST request.

PHP FILE UPLOAD EXAMPLE

HTML Form:

```
<html>
<body>
<form action="file_upload.php" method="post" enctype="multipart/form-data">
<input type="file" name="my_file">
<input type="submit" name="submit" value="Upload File">
</form>
</body>
</html>
```

PHP Code

```
<?php
if(isset($_POST['submit'])){
        $file=$_FILES['my_file'];
        $file_name=$file['name'];
        $path="Images/".$file_name;
        echo "File Name: ".$file_name;
        echo "<br>File Type: ".$file['type'];
        echo "<br>File Size: ".$file['size'];
        echo "<br>Temporary file name: ".$file['tmp_name'];
        echo "<br>";
        if(move_uploaded_file($file['tmp_name'],$path)){
                echo "File uploaded successfully";
        } else { echo "Error Code: ".$file['error']; }
}
?>
```

PHP FILE DOWNLOAD

- The readfile() function reads a file and writes it to the output buffer and helps us to perform the download operation.
- Ex:

```
<?php
$file = 'monkey.gif';

if (file_exists($file)) {
    header('Content-Description: File Transfer'); // Optional
    header('Content-Type: application/octet-stream');
    header('Content-Disposition: attachment; filename="'.basename($file).'"');
    header('Content-Length: ' . filesize($file)); // Optional
    readfile($file);
    exit;
}
?>
```

PHP HEADER FUNCTION

- An HTTP header is a field of an HTTP request or response that passes additional context and metadata about the request or response.
- The header() function is an predefined PHP native function. With header() HTTP functions we can control data sent to the client or browser by the Web server before some other output has been sent.
- Some of the important uses of the header() in PHP are listed below:
- Redirect page: It is used to redirect a from one web page to another web page in PHP.
  - Ex: header('Location:give your url here');
- Set Content-Type: We can change the content type using header()
  - Ex: header("Content-type:application/pdf");
- Cache-control: HTTP headers to prevent page caching
  - header("Cache-Control: no-cache");

THANK YOU!

MySQL

Introduction

- ✓ MySQL is an open-source relational database management system (RDBMS).
- ✓ MySQL is written in C and C++.
- ✓ MySQL was created by a Swedish company, MySQL AB and was first released in 1995
- ✓ Sun Microsystems acquired MySQL AB in 2008.
- ✓ On April 20, 2009, it was announced that Oracle Corporation would acquire Sun
- ✓ MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses.

Use MySQL in XAMPP

- ✓ Locate XAMPP installation directory and go inside mysql/bin folder.
- ✓ Open command prompt at the location
- ✓ Type mysql –u username -p and hit enter
  - In common cases username is root

- ✓ Type the password at the prompt if no password for the database just hit enter
- ✓ You are ready to use MySQL / MariaDB

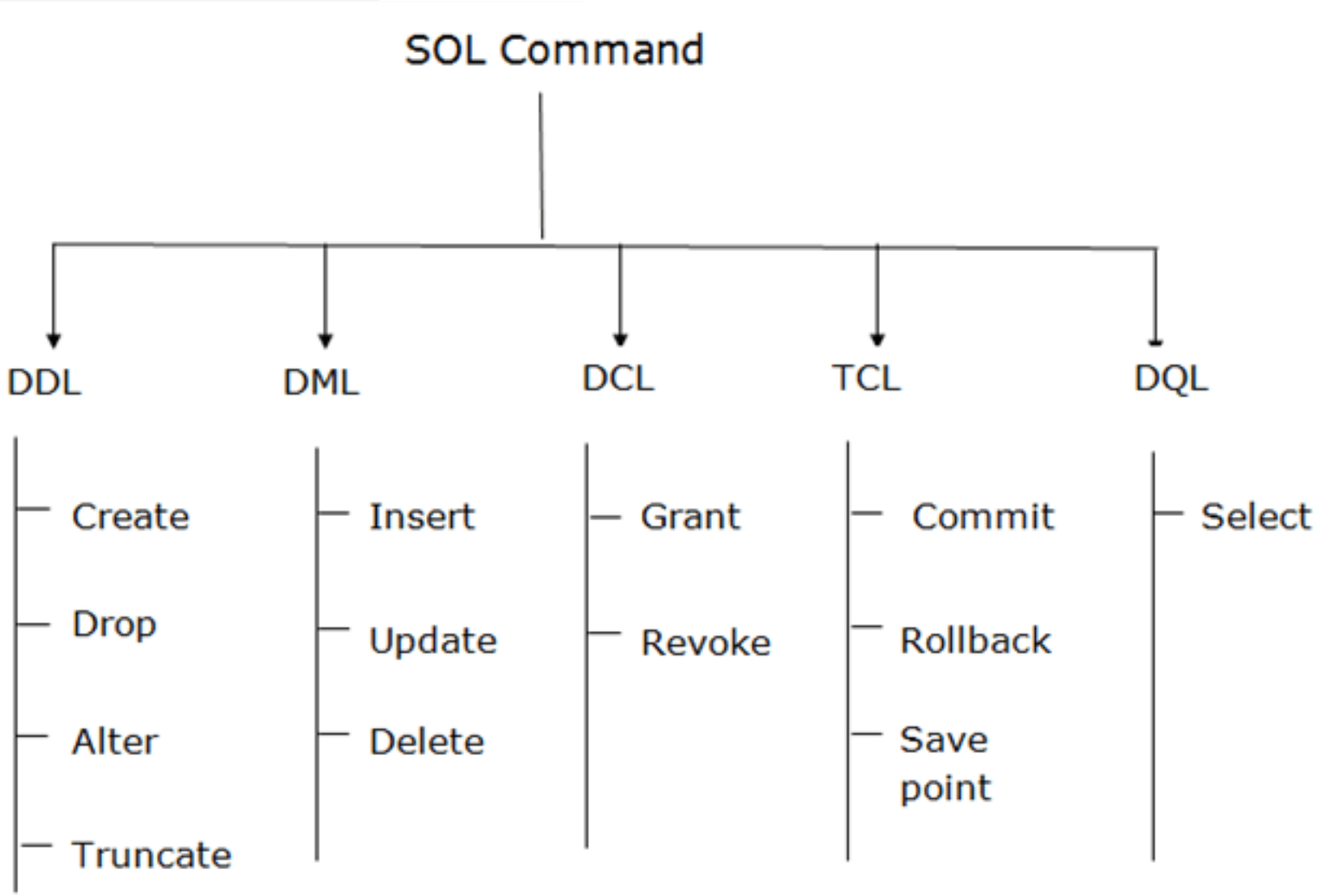
Get Started with MySQL

- ✓ Display available databases by using show databases;
- ✓ Either use one of the existing databases or you can create your own database.
- ✓ To create your own database use - create database database_name;
  - Ex: create database sit_mca;

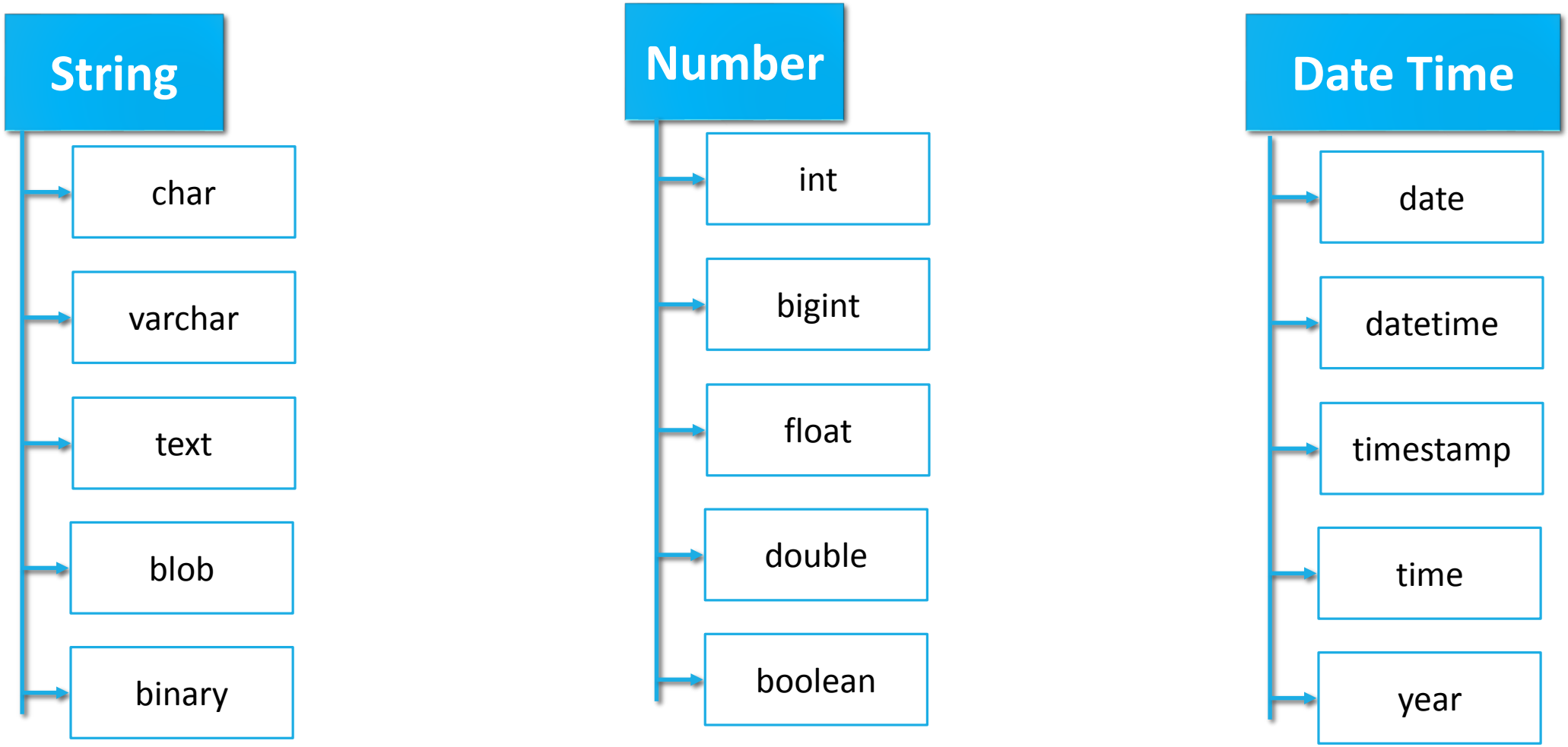
- ✓ To select any database use – use <database_name>;
  - Ex: use sit_mca;

- ✓ To display the available tables in the database use – show tables;

SQL Commands



MySQL Data Types



Common Queries

- ✓ Create table: CREATE TABLE table_name (
  column1 datatype, column2 datatype, column3 datatype, ...);
- ✓ Insert Data: INSERT INTO table_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);
- ✓ Display data: SELECT column1, column2, ... FROM table_name;
- ✓ Update Record: UPDATE table_name SET column1 = value1, column2 = value2, ... WHERE condition;
- ✓ Delete Record: DELETE FROM table_name WHERE condition;

PHP + MySQL

PHP with MySQL:

- PHP can work with a MySQL database using –
  - ~~✓MySQL~~
  - ✓MySQLi Procedural (‘i’ in MYSQLI stands for improved.)
  - ✓MySQLi Object Oriented
  - ✓PDO (PHP Data Objects)
- Earlier version of PHP used the MySQL extension. However this extension was deprecated in 2012. PDO was introduced in PHP 5.1. In PHP 5 and later versions developer can go with MySQLi or PDO.

MySQLi Functions	Description
mysqli_connect(host, username, password, [dbname, port, socket])	Opens a new connection to the MySQL server.
mysqli_select_db(connection, name)	mysqli_select_db() function is used to change the default database for the connection.
mysqli_error(connection)	returns the last error description for the most recent function call, if any.
mysqli_close(connection)	closes a previously opened database connection.
mysqli_query(connection, query)	mysqli_query() function performs a query against a database. For successful SELECT, SHOW, DESCRIBE, or EXPLAIN queries it will return a mysqli_result object. For other successful queries it will return TRUE. FALSE on failure
mysqli_num_rows(mysqli_result)	Returns an integer representing the number of rows in result set.
mysqli_fetch_all(result, [resulttype])	mysqli_fetch_all() function fetches all result rows and returns the result-set as an associative array, a numeric array, or both. resulttype can be MYSQLI_ASSOC MYSQLI_NUM (this is default) MYSQLI_BOTH
mysqli_fetch_assoc(result)	Returns an associative array of strings representing the fetched row. NULL if there are no more rows in result-set

PHP with MySQL:

Connect to MySQL Database using MySQLi procedure:

```
<?php
$con=mysqli_connect("localhost","root","12345","mydatabase");
if($con){
        echo "Connected";
}else{
        echo "Error: ".mysqli_error($con);
}
?>
```

CRUD Operation with PHP and MySQL

Add Student

Roll No: 4

Name: Bimal Kar

Mobile: 7894587569

Address: CTC

Branch MCA

ADD

Availabale Student Records

Roll	Name	Mobile	Address	Branch	Action
1	Ram Kumar	9865326589	Bhubaneswar	MCA	Update Delete
3	Riya Barma	7896532658	Kolkata	MBA	Update Delete
4	Bimal Kar	7894587569	CTC	MCA	Update Delete

Update Contact Information

Roll: 3

Mobile:

Address:

Back Update

Create Require Database & Table:

1. Connect to MySQL prompt.
2. Create Database by using

   create database studentinformationsystem;

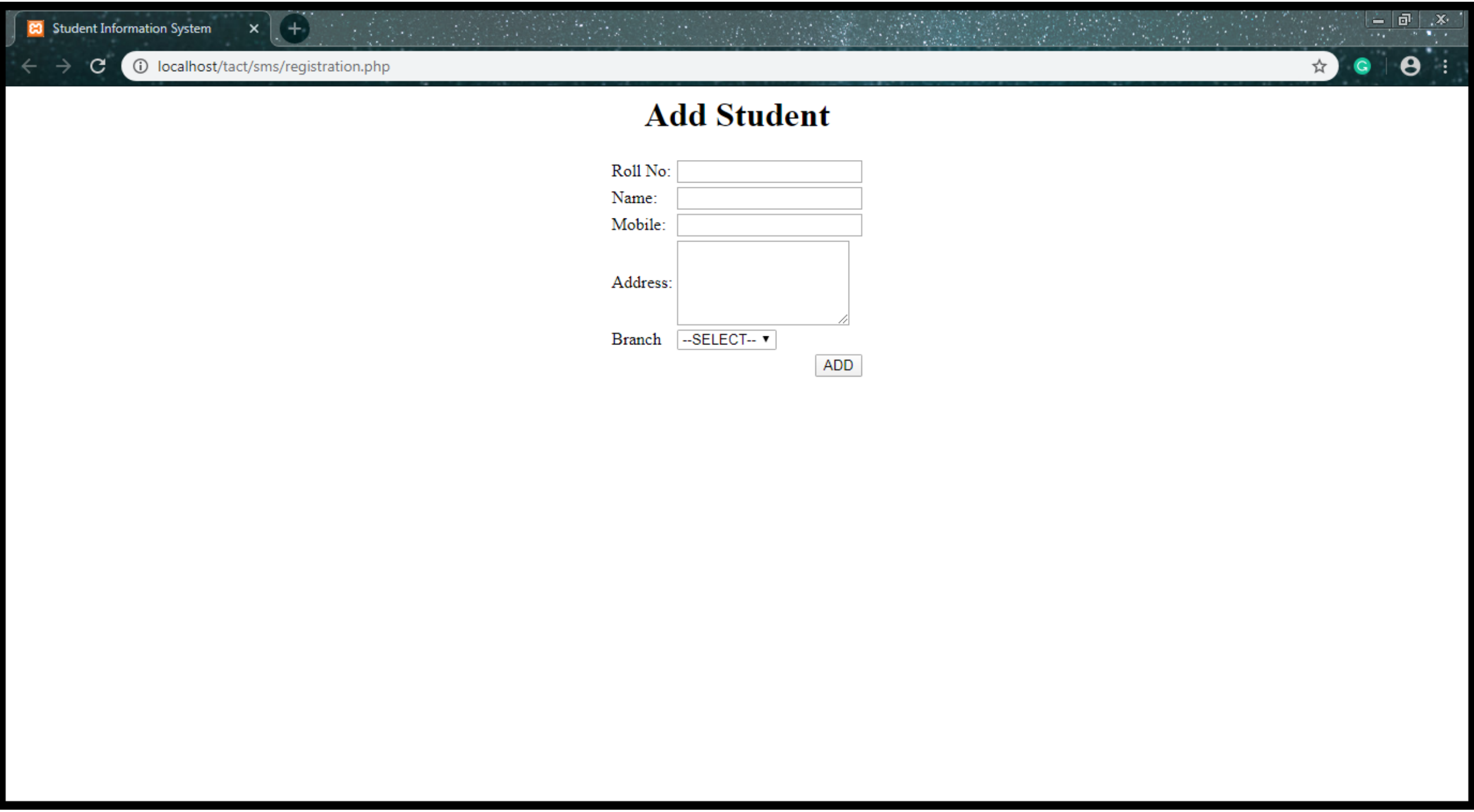
3. Create the database table:

   create table student(roll int(5) primary key,name varchar(20),mobile bigint(10), address varchar(20), branch varchar(5));

HTML form to add student:

registration.php (.html)

```
<html>
<head>
<title>Student Information System</title>
</head>
<body>
<center>
        <h1>Add Student</h1>
</center>
<form action="insert.php" method="post">
<table align="center">
<tr><td>Roll No:</td>
<td><input type="text" name="roll"></td></tr>
<tr><td>Name:</td>
<td><input type="text" name="name"></td></tr>
<tr><td>Mobile:</td><td><input type="text" name="mobile"></td> </tr>
<tr>     <td>Address:</td>
<td><textarea rows="5" cols="20" name="address"></textarea></td></tr>
<tr><td>Branch</td>
        <td><select name="branch">
                <option value="">--SELECT--</option>
                <option value="BCA">BCA</option>
                <option value="BBA">BBA</option>
                <option value="MCA">MCA</option>
                <option value="MBA">MBA</option>
                </select>
        </td>
        </tr>
        <tr><td colspan="2" align="right">
        <input type="submit" value="ADD" name="submit">
        </td></tr>
</table>
</form>
</body>
</html>
```

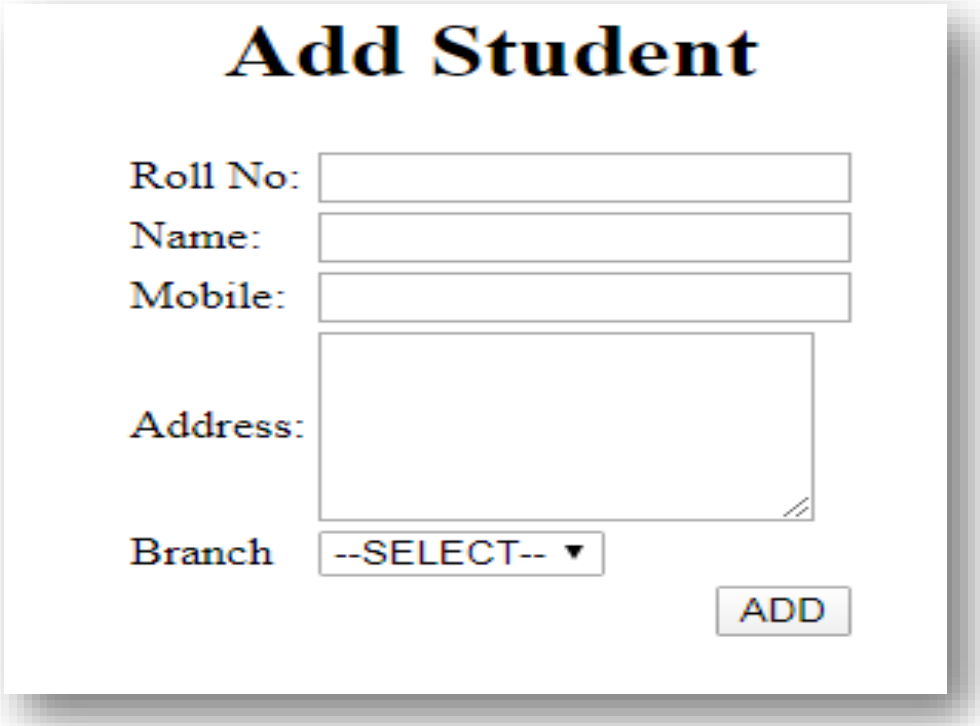

Student Information System
localhost/tact/sms/registration.php
Add Student
Roll No:
Name:
Mobile:
Address:
Branch
--SELECT--
ADD

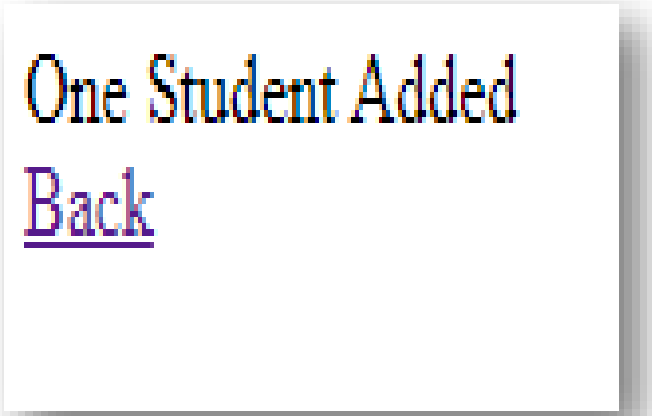
insert.php

```
<?php
if(isset($_POST['submit'])){
		$roll=$_POST['roll'];
		$name=$_POST['name'];
		$mobile=$_POST['mobile'];
		$address=$_POST['address'];
		$branch=$_POST['branch'];
		$con=mysqli_connect("localhost","root","12345","studentinformationsystem") or die(mysqli_error($con));

		$qry="insert into student values($roll,'$name',$mobile,'$address','$branch')";
		if(mysqli_query($con,$qry)){
			echo "One Student Added";
		}else{
			echo "Error while inserting:".mysqli_error($con);
		}
		mysqli_close($con);
}
?>
<br>
<a href='registration.php'>Back</a>
```

Insert Operation:





display.php

```
<?php
$con=mysqli_connect("localhost","root","12345","studentinformationsystem") or die(mysqli_error($con));
$qry="select * from student";
$result=mysqli_query($con,$qry);
if(mysqli_num_rows($result)>0){
        echo "<table border='1' align='center' cellpadding='3'>";
        echo "<caption>Available Student Records</caption>";
        echo
"<tr><th>Roll</th><th>Name</th><th>Mobile</th><th>Address</th><th>Branch</th><th>Action</th></tr>";
        while($student=mysqli_fetch_assoc($result)){
                echo "<tr><td>$student[roll]</td>
                        <td>$student[name]</td>
                        <td>$student[mobile]</td>
                        <td>$student[address]</td>
                        <td>$student[branch]</td>
                        <td>
                                <a href='update.php?roll=$student[roll]'>Update</a>
                                <a href='delete.php?roll=$student[roll]'>Delete</a>
                        </td> </tr>";
        }
        echo "</table>";
} ?>
```

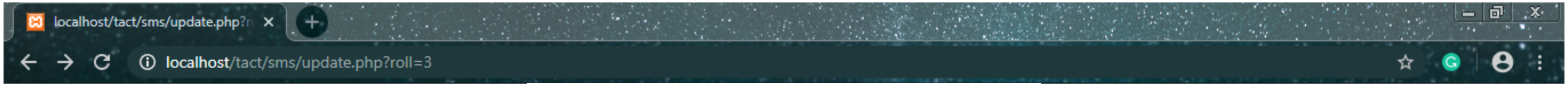
Availabale Student Records

Roll	Name	Mobile	Address	Branch	Action
1	Ram Kumar	9865326589	Bhubaneswar	MCA	Update Delete
3	Riya Barma	8956235869	CTC	MBA	Update Delete
4	Bimal Kar	7894587569	CTC	MCA	Update Delete
5	Amit Sah	7854125897	Banglore	MBA	Update Delete

update.php

```
<html>
<body>
<form action="" method="post">
<table align="center">
<caption>Update Contact Information</caption>
<tr><td>Roll:</td>
		<td><?php
			if(isset($_GET['roll']))
				echo $_GET['roll'];
			?> </td></tr>
<tr>	<td>Mobile:</td>
		<td><input type="text" name="mobile"></td></tr>
<tr>	<td>Address:</td>
		<td><textarea rows="5" cols="20"
name="address"></textarea></td></tr>
<tr><td colspan="2">		<a href="display.php">Back</a>
		<input type="submit" name="update"
value="Update">
</td></tr>
</table>
</form>
</body>
```

```
<?php
if(isset($_GET['roll']))
		$roll=$_GET['roll'];
if(isset($_POST['update'])){
		$con=mysqli_connect("localhost","root","12345
","studentinformationsystem") or die(mysqli_error($con));
		$mobile=$_POST['mobile'];
		$address=$_POST['address'];
		$qry="update student set
mobile=$mobile,address='$address' where roll=$roll";
		mysqli_query($con,$qry);
		if(mysqli_affected_rows($con)>0){
			echo "Student Info Updated";
		}else{
			echo "Error ".mysqli_error($con);
		}
		mysqli_close($con);
}
?>
```



Student Info Updated

Update Contact Information

Roll: 3

Mobile:

Address:

Back Update

Availabale Student Records

Roll	Name	Mobile	Address	Branch	Action
1	Ram Kumar	9865326589	Bhubaneswar	MCA	Update Delete
3	Riya Barma	7854897525	Bhubaneswar	MBA	Update Delete
4	Bimal Kar	7894587569	CTC	MCA	Update Delete
5	Amit Sah	7854125897	Banglore	MBA	Update Delete

delete.php

```
<?php
$con=mysqli_connect("localhost","root","12345","studentinformationsystem") or die(mysqli_error($con));
$roll=$_GET['roll'];
$qry="delete from student where roll=$roll";
if(mysqli_query($con,$qry)){
        header("location:display.php");
}else{
        echo "Error";
}
?>
```

Availabale Student Records

Roll	Name	Mobile	Address	Branch	Action
1	Ram Kumar	9865326589	Bhubaneswar	MCA	Update Delete
3	Riya Barma	7854897525	Bhubaneswar	MBA	Update Delete
4	Bimal Kar	7894587569	CTC	MCA	Update Delete

Module 5

Design Pattern

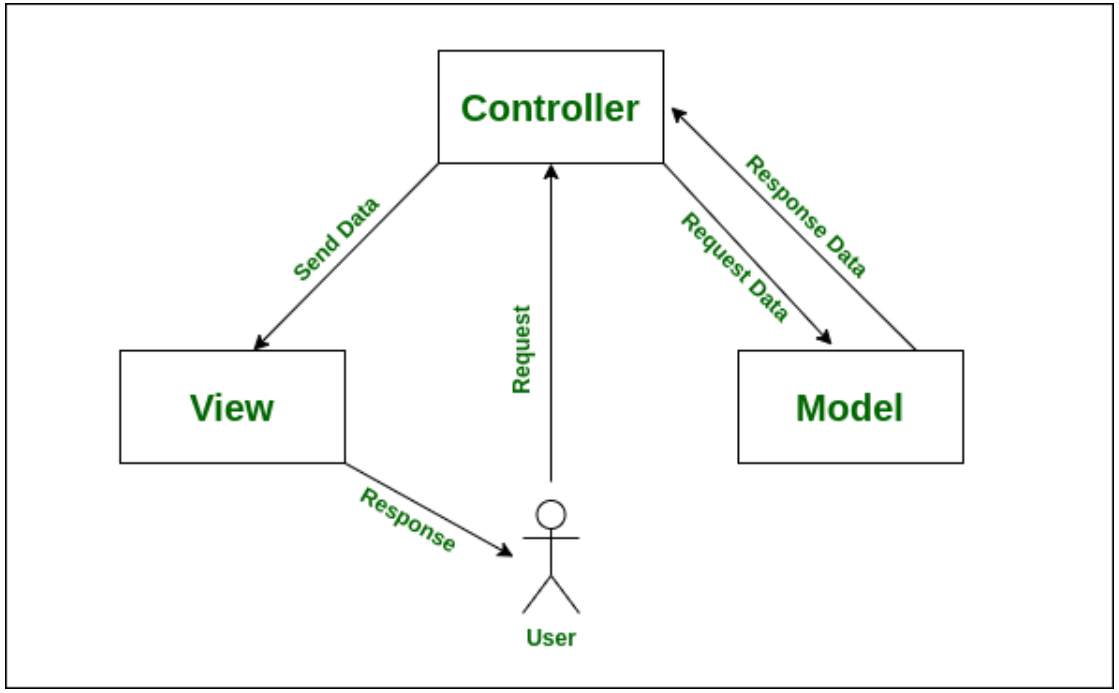
- In software engineering, a Design Pattern is a general repeatable solution to commonly occurring problem in software Design.
- Good Object-oriented designs should be **reusable**, **maintainable** and **extensible**
- Some of the most significant benefits of implementing design patterns in PHP are:
  - PHP Design patterns help solve repetitive problems faced during development
  - Using design patterns in PHP makes communication between designers and developers more efficient
  - You can be confident that other developers will understand your code since it follows design patterns
  - Following best practices helps build more robust applications
  - It helps make development faster and easier

MVC Design Pattern

- MVC stands for Model, View & Controller. It is a software design pattern that divides the application into the interconnection of three main components.
- PHP MVC is an application design pattern that separates the application data and business logic (model) from the presentation (view).
- The main advantage of Architecture is Reusability, Security and Increasing the performance of Application.

MVC Design Pattern

- Model – this part is concerned with the business logic and the application data. It can be used to perform data validations, process data and store it.
- Views – this part deals with presenting the data to the user. View is responsible for sending requests to the controller and receives a response from it. It is a place where all the templates are stored which contains HTML, CSS, and JS files.



- Controller - The controller acts as an interface between view and model. Its responsibility(mostly) is to get the request from view and send it to model to perform a necessary operation. This means it separates user interface from business logic and handles how the application will respond to user interaction in the view.

MVC Design Pattern : Advantages

The MVC design pattern offers advantages in terms of code organization, maintainability, testability, extensibility, and collaboration, making it a widely used pattern in software development.

- **Separation of concerns**: MVC promotes a clear separation of concerns between the model, view, and controller components. This separation allows developers to focus on specific aspects of the application without tightly coupling them together. It improves code organization and maintainability.
- **Modularity and reusability:** With the separation of concerns, each component of MVC can be developed and tested independently. This modularity enhances code reusability since the components can be used in different contexts or combined with other components to build new features or applications.

MVC Design Pattern: Advantages

- **Code maintainability:** By separating the application logic from the user interface, MVC makes it easier to maintain and update the code. Changes in one component, such as the view or the model, can be made without affecting the other components as long as the interfaces are preserved.
- **Testability:** The separation of concerns in MVC facilitates unit testing. The model, view, and controller can be tested individually with different test cases, ensuring that each component functions correctly. This promotes better test coverage and more robust software.
- **Flexibility and extensibility:** MVC allows for flexibility in evolving and extending the application. New views can be added to present the same data in different ways, and new controllers can be introduced to handle additional user interactions or business logic. The modularity of MVC makes it easier to introduce changes and adapt to evolving requirements.

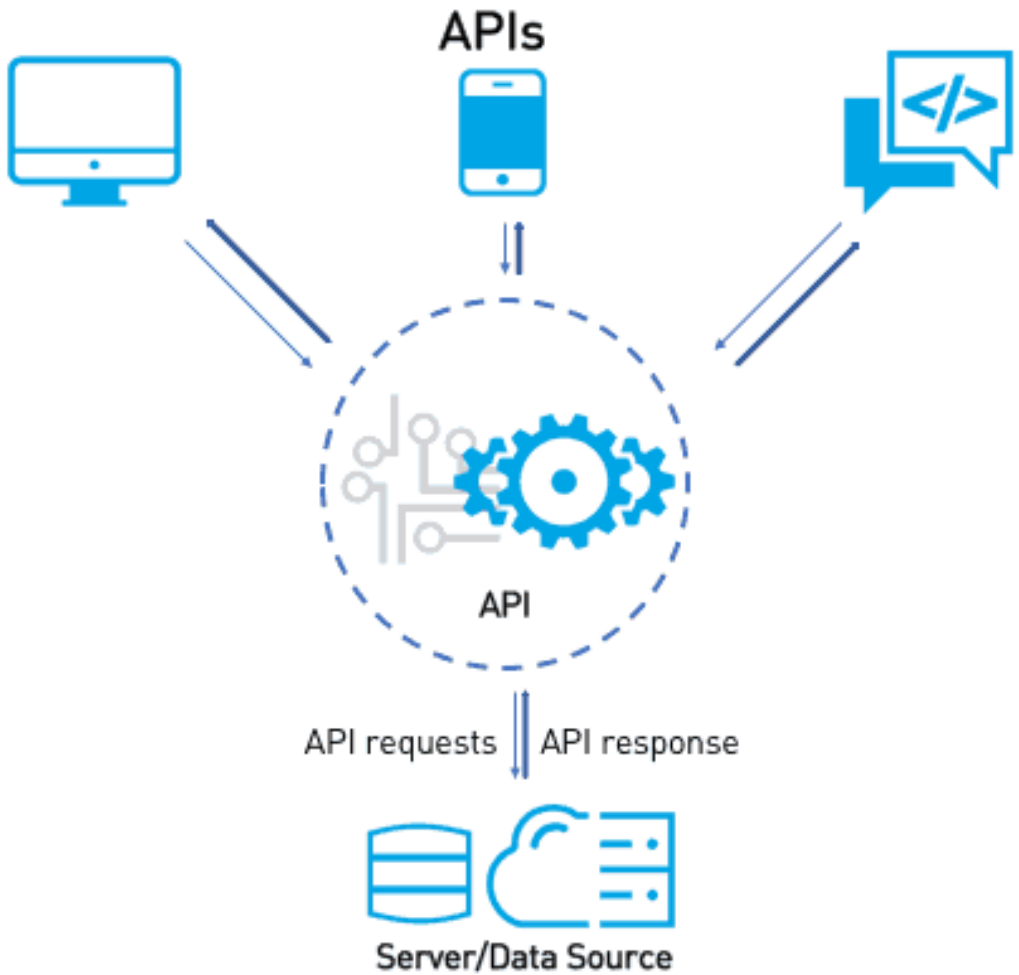
MVC Design Pattern: Advantages

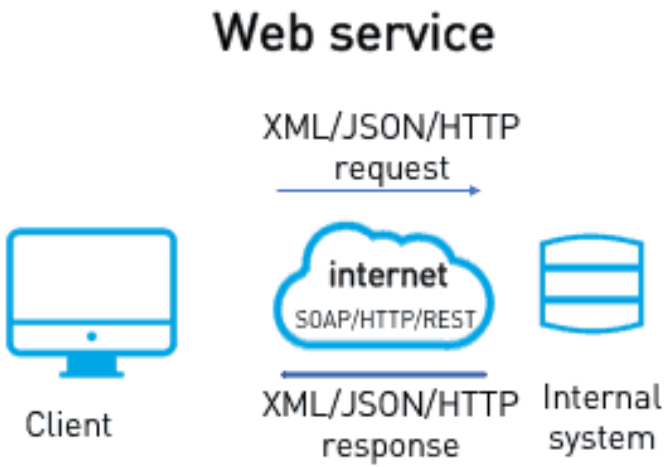
- **Collaboration:** The separation of responsibilities in MVC enables collaboration among developers working on different components. Multiple developers can work on the model, view, and controller simultaneously without interfering with each other's work. This promotes parallel development and faster iterations.
- **User interface consistency:** MVC helps maintain consistency in the user interface. Since the view component is responsible for rendering the user interface, any changes or updates to the UI can be easily applied across the application by modifying the view component.

Web Service

- A **web service** (**WS**) is server running on a computer device, listening for requests at a particular port over a network, serving web documents (HTTP, JSON, XML, images).
- Web Services do not use the World Wide Web (WWW), a human user interface running on the Internet, but rather a machine-to-machine service running on the Internet using the WWW protocols.
- The method of communication between two devices over the network.
- It is a collection of standards or protocols for exchanging information between two devices or application.

Web Service


APIs
API
API requests
API response
Server/Data Source


Web service
XML/JSON/HTTP
request
internet
SOAP/HTTP/REST
XML/JSON/HTTP
response
Client
Internal
system

Web Service Characteristics

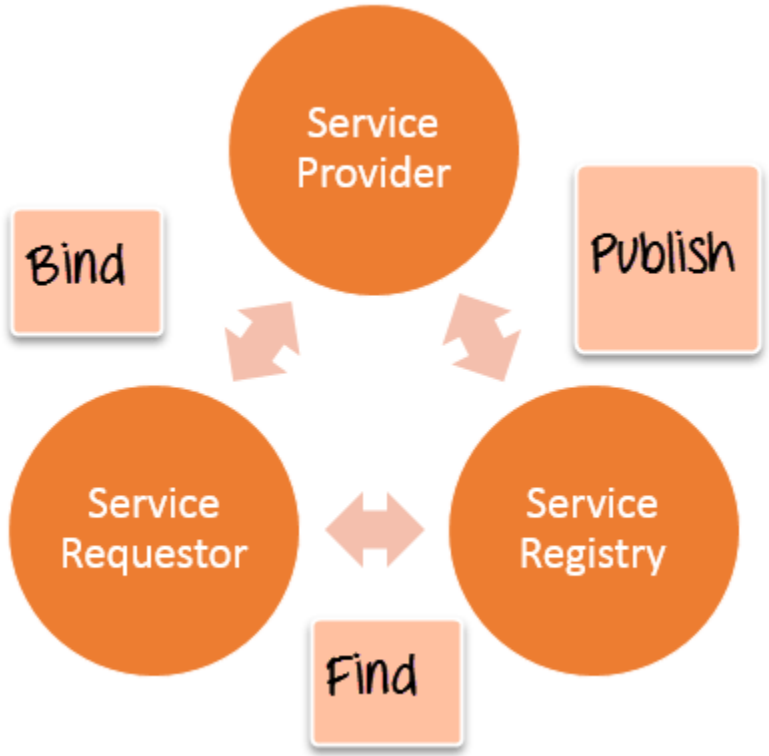
- **Interchangeable Data Representation:** Web services use XML / JSON at data representation and data transportation layers. Using XML / JSON eliminates any networking, operating system, or platform binding. Web services based applications are highly interoperable at their core level.
- **Loosely Coupled:** Loosely coupled means that the client and the web service are not bound to each other, which means that even if the web service changes over time, it should not change the way the client calls the web service.
- **Supports Remote Procedure Calls(RPCs):** Web services allow clients to invoke procedures, functions, and methods on remote objects. Remote procedures expose input and output parameters that a web service must support.

Web Service Characteristics

- **Synchronous or Asynchronous functionality** –In synchronous operations, the client will actually wait for the web service to complete an operation. Asynchronous operations allow a client to invoke a service and then execute other functions in parallel.
- **Supports Document Exchange** – One of the key benefits of XML is its generic way of representing not only data but also complex documents. These documents can be as simple as representing a current address, or they can be as complex as representing an entire book.

Web Service Architecture

- **Service Provider**: This is the provider of the web service. The service provider implements the service and makes it available on the Internet.
- **Service Requestor**: This is any consumer of the web service. The requestor utilizes an existing web service by opening a network connection and sending an XML request.
- **Service Registry**: This is a logically centralized directory of services. The registry provides a central place where developers can publish new services or find existing ones. It therefore serves as a centralized clearing house for companies and their services.



Web Service Architecture

- **Service Transport Layer:** This layer is responsible for transporting messages between applications. Currently, this layer includes Hyper Text Transport Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), and newer protocols such as Blocks Extensible Exchange Protocol (BEEP).
- **Messaging Layer:** This layer is responsible for encoding messages in a common XML/JSON format so that messages can be understood at either end.
- **Service Description:** This layer is responsible for describing the public interface to a specific web service. Currently, service description is handled via the Web Service Description Language (WSDL).
- **Service Discovery Layer**: This layer is responsible for centralizing services into a common registry and providing easy publish/find functionality.

Web Service Types

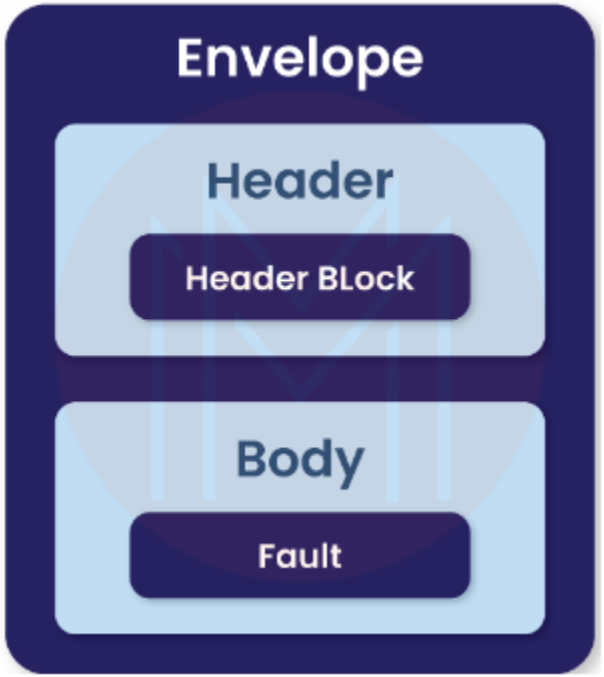
- There are mainly two types of web services.
    1. SOAP web services.
    2. RESTful web services.

SOAP Web Service

- SOAP is an XML-based protocol for accessing web services over HTTP.
- It has some specification which could be used across all applications.
- It is platform independent and language independent. By using SOAP, you will be able to interact with other programming language applications.
- At first, the SOAP client creates a service request, and it is sent to the ‘SOAP request handler’ of a web server through HTTP or HTTPS transport protocols. Now, the web server receives the request and then processes it to the web service provider.
- The service provider responds to the 'SOAP request handler' in terms of requested parameters or data or return values. Eventually, the response is sent to the service requestor. Note that XML format is used for sending a request and receiving a response in SOAP protocol.

SOAP Message Format

- The SOAP specification defines something known as a "**SOAP message**" which is what is sent to the web service and the client application.
- An **Envelope element** that identifies the XML document as a SOAP message – This is the containing part of the SOAP message and is used to encapsulate all the details in the SOAP message. This is the root element in the SOAP message.
- A **Header element** that contains header information – The header element can contain information such as authentication credentials which can be used by the calling application
- A **Body element** that contains call and response information – This element is what contains the actual data which needs to be sent between the web service and the calling application.
- **Fault:** It is the sub-element and the child element of the SOAP body. It is used to report errors and error status information.



SOAP Message Format

SOAP Request

```
POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml;
charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>

<soap:Envelope
xmlns:soap="http://www.w3.org/2003/05/s
oap-envelope/"
soap:encodingStyle="http://www.w3.org/20
03/05/soap-encoding">

<soap:Body xmlns:m="http://www.example.
org/stock">
  <m:GetStockPrice>
    <m:StockName>IBM</m:StockName>
  </m:GetStockPrice>
</soap:Body>

</soap:Envelope>
```

SOAP Response

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn
<?xml version="1.0"?>

<soap:Envelope
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">
<SOAP-ENV:Fault>
       <faultcode xsi:type="xsd:string">SOAP-ENV:Client</faultcode>
      <faultstring xsi:type="xsd:string">
         Failed to locate method (GetTutorialID) in class (GetTutorial)
       </faultstring>
   </SOAP-ENV:Fault>
<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPriceResponse>
    <m:Price>34.5</m:Price>
  </m:GetStockPriceResponse>
</soap:Body>
</soap:Envelope>
```

SOAP Web Service Advantages

- **Neutrality:** SOAP supports any programming language and works in a distributed enterprise environment
- **Independence:** SOAP messages can be processed on any platform. In other terms, it can work on both Windows and Linux.
- **Scalability:** As SOAP uses HTTPS transport protocol, it can be scaled easily. With HTTPS, SOAP overcomes firewall problems.
- **Security:** SOAP comes with its own security standard - WS security.

SOAP Web Service Disadvantages

- **Slower:** The use of XML formats in SOAP decreases the speed of requests and responses significantly.
- **Resource-consuming:** SOAP requires long-length Payloads for transferring even simple string messages. So, it needs larger bandwidth for exchanging information.
- **Firewall Latency:** Since SOAP uses HTTPS protocol, firewall latency may occur during data exchange. This is because firewalls analyze HTTPS protocols usually.
- **Hard Learning Curve:** As SOAP works based on many protocols and standards, developers must thoroughly understand these matters. For this, they need to dive deep into these protocols and standards to know better.
- **Tight Coupling:** SOAP requires a tight coupling between the client and the server. If any fault occurs on either side, it will affect both the client and the server.

REST Web Service

- REST stands for REpresentational State Transfer.
- REST is web standards based architecture and uses HTTP Protocol.
- It revolves around resource where every component is a resource and a resource is accessed by a common interface using HTTP standard methods.
- REST was first introduced by Roy Fielding in 2000.
- Properties of REST, such as performance, scalability, and modifiability, that enable services to work best on the Web.
- In the REST architectural style, data and functionality are considered resources and are accessed using **Uniform Resource Identifiers (URIs)**, typically links on the Web.

Principles of REST Web Service

- **URI Resource Identification:** A RESTful web service exposes a set of resources that identify the targets of the interaction with its clients. Resources are identified by URIs, which provide a global addressing space for resource and service discovery.
- **Uniform Interface:** Resources are manipulated using a fixed set of four create, read, update, delete operations: PUT, GET, POST, and DELETE. PUT creates a new resource, which can be then deleted by using DELETE. GET retrieves the current state of a resource in some representation. POST transfers a new state onto a resource.
- **Self-Descriptive Messages:** As resources are decoupled from their representation, content can be accessed through a large number of formats like HTML, PDF, JPEG, XML, plain text, JSON, etc.
- **Use of Hyperlinks for State Interactions:** Every interaction with a resource is stateless; that is, request messages are self-contained. Stateful interactions are based on the concept of explicit state transfer. Several techniques exist to exchange state, such as URI rewriting, cookies, and hidden form fields.

RESTful Methods

Method	Description
GET	Retrieve information about the REST API resource
POST	Create a REST API resource
PUT	The PUT method replaces all current representations of the target resource with the request payload.
PATCH	The PATCH method applies partial modifications to a resource. Unlike PUT Request, PATCH does partial update e.g. Fields that need to be updated by the client, only that field is updated without modifying the other field. But PATCH is not safe as it perform a non-read-only operation.
DELETE	Delete a REST API resource or related component

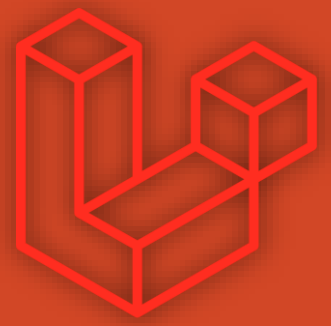
REST Web Service Advantages

- **Fast**: RESTful Web Services are fast because there is no strict specification like SOAP. It consumes less bandwidth and resource.
- **Language and Platform independent**: RESTful web services can be written in any programming language and executed in any platform.
- **Can use SOAP**: RESTful web services can use SOAP web services as the implementation.
- **Permits different data format**: RESTful web service permits different data format such as Plain Text, HTML, XML and JSON.

SOAP vs REST

SOAP	REST
SOAP stands for Simple Object Access Protocol.	REST stands for REpresentational State Transfer.
It has both state and stateless design	It has only a stateless design
SOAP is independent of transport protocols. And it uses HTTP, SMTP, TCP, FTP, etc.	Only HTTP or HTTPS protocol can be used in REST API
SOAP is a communication protocol	REST API has a client-server architecture style
It uses only XML language with schemas, so efficiency is not good as REST API.	Efficiency is good since it uses JSON, HTML , XML, formats
Works slower	Works faster
SOAP requires more bandwidth and resource than REST.	REST requires less bandwidth and resource than SOAP.
SOAP defines its own security.	RESTful web services inherits security measures from the underlying transport.
Need tight coupling between client and server, which in turn increases complexity	REST API is usually separated from data storage and the back end. That is why it is independent and flexible.

Laravel



Introduction

- Laravel is a free and open-source PHP web framework created by Taylor Otwell.
- Laravel following the model–view–controller (MVC) architectural pattern and based on Symfony.
- Taylor Otwell created Laravel and first beta release was made available on June 9, 2011

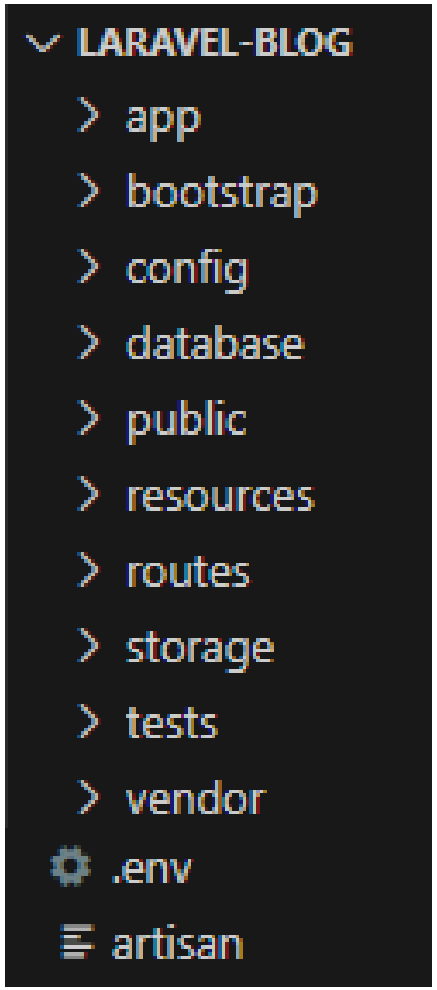
Features

- **MVC Architecture Support**: Laravel supports MVC architecture. It provides faster development process as in MVC; one programmer can work on the view while other is working on the controller to create the business logic for the web application.
- **Libraries and Modularity**: Laravel is very popular as some Object-oriented libraries, and pre-installed libraries are added in this framework which includes – authentication module, bcrypt password hashing, CSRF protection, Faker etc.
- **ORM:** Laravel incorporates a query builder which helps in querying databases using various simple chain methods. It provides ORM (Object Relational Mapper) called Eloquent.
- **E-mail:** Laravel includes a mail class which helps in sending mail with rich content and attachments from the web application.
- **Authentication:** User authentication is a common feature in web applications. Laravel eases designing authentication as it includes features such as register, forgot password and send password reminders.

Features

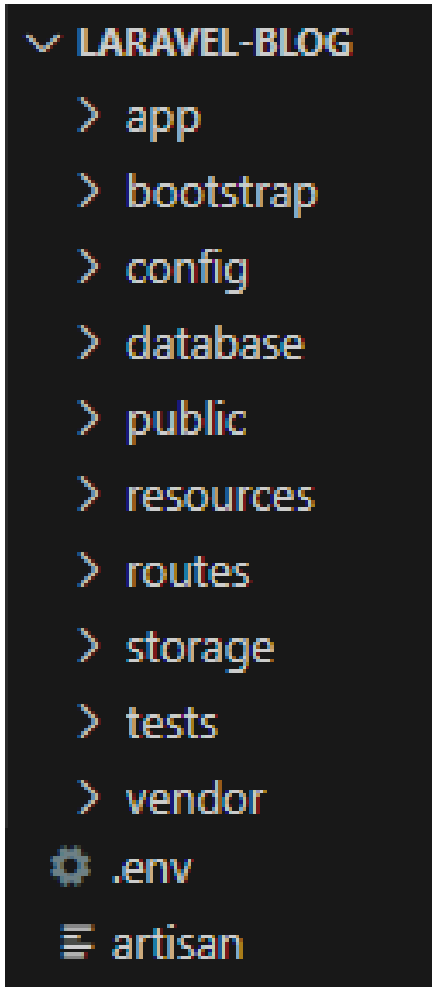
- **Template Engine:** Laravel uses the Blade Template engine, a lightweight template language used to design hierarchical blocks and layouts with predefined blocks that include dynamic content.
- **Testability:** Laravel includes features and helpers which helps in testing through various test cases. This feature helps in maintaining the code as per the requirements.
- **Secure Migration System:** Laravel framework can expand the database without allowing the developers to put much effort every time to make changes, and the migration process of Laravel is very secure and full-proof. In the whole process, php code is used rather than SQL code.

Project Structure



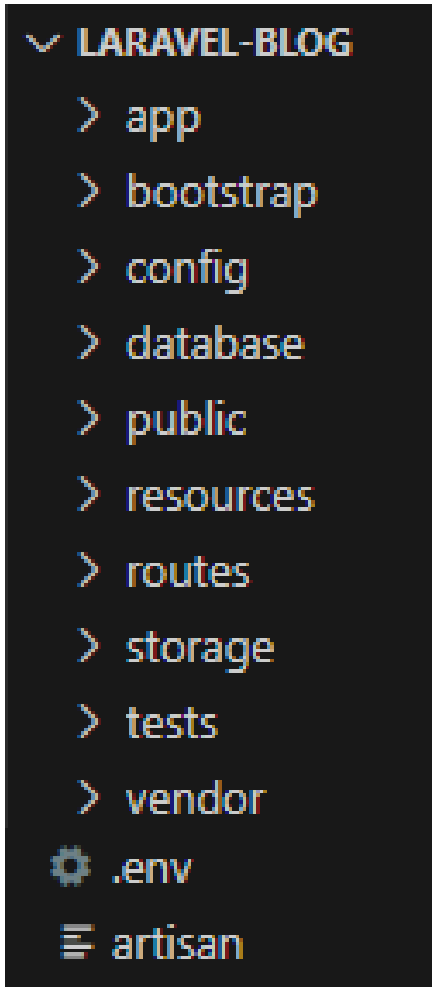
- **app:** This directory is the heart of the framework and contains backend code of our web application like Controllers, Broadcasts, Providers, Custom Artisan Commands, Middleware, etc. This directory further contains many sub-directories like Console, Exception, Http, Provider etc.
- **bootstrap:** The bootstrap directory contains the app.php file which bootstraps the framework. Bootstraps typically refers to the process of initializing or starting up the framework or software system. This directory also houses a cache directory which contains framework generated files for performance optimization such as the route and services cache files.
- **config:** The config directory, as the name implies, contains all of your application's configuration files.

Project Structure



- **database:** The database directory contains your database migrations, model factories, and seeds.
- **public:** The public directory contains the index.php file, which is the entry point for all requests entering your application and configures autoloading. This directory also houses your assets such as images, JavaScript, and CSS.
- **resources:** The resources directory contains your views as well as your raw, un-compiled assets such as CSS or JavaScript.
- **routes:** The routes directory contains all of the route definitions for your application. By default, several route files are included with Laravel: web.php, api.php, console.php, and channels.php.

Project Structure



- **storage:** The storage directory contains your logs, compiled Blade templates, file based sessions, file caches, and other files generated by the framework..
- **test:** The tests directory contains your automated tests. Example PHPUnit unit tests and feature tests are provided out of the box. Each test class should be suffixed with the word Test.
- **vendor:** The vendor directory contains your Composer dependencies..
- **.env:** Laravel's default .env file contains some common configuration values that may differ based on whether your application is running locally or on a production web server.
- **artisan:** Artisan is the name of the command-line interface included with Laravel. It provides a number of helpful commands for your use while developing your application. It is driven by the powerful Symfony Console component.

Important Commands

Command	Purpose
composer global require laravel/installer	Install laravel installer globally
laravel new PROJECT_NAME	Create a new laravel project
php artisan serve OR php -S localhost:8000 -t public/	Run Lravel project
php artisan make:controller <name>	Create a new Controller class
php artisan make:migration <name>	Create a new migration file
php artisan make:model <name>	Create a new Eloquent model class