**Software Requirements:**

Operating System Front end **:**  PHP

Data Base **:** My sql

Web Browser **:** WampServer

**Hardware Requirements:**

## Intel Celeron/ P3/p4 : 600 MHz or above.

## RAM (SD/DDR) : 1 GB

Hard Disc **:** 500 GB

**SOFTWRAE DESCRIPTION**

# PHP PROGRAMMING

PHP is a [scripting](http://en.wikipedia.org/wiki/Scripting_programming_language) language designed to fill the gap between [SSI](http://en.wikipedia.org/wiki/Server_Side_Includes) (Server Side Includes) and [Perl](http://en.wikibooks.org/wiki/Perl), intended for the Web environment. Its principal application is the implementation of Web pages having dynamic content. PHP has gained quite a following in recent times, and it is one of the frontrunners in the Open Source software movement. Its popularity derives from its C-like syntax, and its simplicity. The newest version of PHP is 5.5 and it is heavily recommended to always use the newest version for better security, performance and of course features.

If you've been to a website that prompts you to login, you've probably encountered a server-side scripting language. Due to its market saturation, this means you've probably come across PHP. [PHP](http://en.wikipedia.org/wiki/PHP) was designed by [Rasmus Lerdorf](http://en.wikipedia.org/wiki/Rasmus_Lerdorf) to display his resume online and to collect data from his visitors.

Basically, PHP allows a static webpage to become dynamic. "PHP" is an acronym that stands for "PHP: Hypertext Preprocessor". The word "Preprocessor" means that PHP makes changes before the HTML page is created. This enables developers to create powerful applications that can publish a blog, remotely control hardware, or run a powerful website such as Wikipedia or Wikibooks. Of course, to accomplish something such as this, you need a database application such as MySQL.

Before you embark on the wonderful journey of Server Side Processing, it is recommended that you have a basic understanding of the [HyperText Markup Language (HTML)](http://en.wikibooks.org/wiki/HTML). But [PHP](http://en.wikipedia.org/wiki/PHP) can also be used to build [GUI](http://en.wikipedia.org/wiki/GUI)-driven applications for example by using [PHP-GTK](http://en.wikipedia.org/wiki/PHP-GTK).

PHP is a [server-side scripting](http://en.wikipedia.org/wiki/Server-side_scripting) language designed for [web development](http://en.wikipedia.org/wiki/Web_development) but also used as a [general-purpose programming language](http://en.wikipedia.org/wiki/General-purpose_programming_language). As of January 2013, PHP was installed on more than 240 million [websites](http://en.wikipedia.org/wiki/Website) (39% of those sampled) and 2.1 million [web servers](http://en.wikipedia.org/wiki/Web_server). Originally created by [Rasmus Lerdorf](http://en.wikipedia.org/wiki/Rasmus_Lerdorf) in 1994, the [reference implementation](http://en.wikipedia.org/wiki/Reference_implementation) of PHP (powered by the [Zend Engine](http://en.wikipedia.org/wiki/Zend_Engine)) is now produced by The PHP Group. While PHP originally stood for *Personal Home Page*, it now stands for *PHP: Hypertext Preprocessor*, which is a [recursive](http://en.wikipedia.org/wiki/Recursive_acronym) [backronym](http://en.wikipedia.org/wiki/Backronym).

PHP code can be simply mixed with [HTML](http://en.wikipedia.org/wiki/HTML) code, or it can be used in combination with various [templating engines](http://en.wikipedia.org/wiki/Web_template_system) and [web frameworks](http://en.wikipedia.org/wiki/Web_framework). PHP code is usually processed by a PHP [interpreter](http://en.wikipedia.org/wiki/Interpreter_%28computing%29), which is usually implemented as a web server's native [module](http://en.wikipedia.org/wiki/Plugin_%28computing%29) or a [Common Gateway Interface](http://en.wikipedia.org/wiki/Common_Gateway_Interface) (CGI) executable. After the PHP code is interpreted and executed, the web server sends resulting output to its client, usually in form of a part of the generated web page; for example, PHP code can generate a web page's HTML code, an image, or some other data. PHP has also evolved to include a [command-line interface](http://en.wikipedia.org/wiki/Command-line_interface) (CLI) capability and can be used in [standalone](http://en.wikipedia.org/wiki/Computer_software) [graphical applications](http://en.wikipedia.org/wiki/Graphical_user_interface).

The canonical PHP interpreter, powered by the Zend Engine, is [free software](http://en.wikipedia.org/wiki/Free_software) released under the [PHP License](http://en.wikipedia.org/wiki/PHP_License). PHP has been widely ported and can be deployed on most web servers on almost every [operating system](http://en.wikipedia.org/wiki/Operating_system) and [platform](http://en.wikipedia.org/wiki/Computing_platform), free of charge.

Despite its popularity, no written [specification](http://en.wikipedia.org/wiki/Formal_specification) or standard existed for the PHP language until 2014, leaving the canonical PHP interpreter as a [*de facto*](http://en.wikipedia.org/wiki/De_facto) standard. Since 2014, there is ongoing work on creating a formal PHP specification.

PHP development began in 1994 when [Rasmus Lerdorf](http://en.wikipedia.org/wiki/Rasmus_Lerdorf) wrote a series of [Common Gateway Interface](http://en.wikipedia.org/wiki/Common_Gateway_Interface) (CGI) binaries in C, which he used to maintain his [personal homepage](http://en.wikipedia.org/wiki/Personal_homepage). He extended them to add the ability to work with [web forms](http://en.wikipedia.org/wiki/Web_form) and to communicate with [databases](http://en.wikipedia.org/wiki/Database), and called this implementation "Personal Home Page/Forms Interpreter" or PHP/FI.

PHP/FI could be used to build simple, dynamic [web applications](http://en.wikipedia.org/wiki/Web_application). Lerdorf initially announced the release of PHP/FI as "Personal Home Page Tools (PHP Tools) version 1.0" publicly to accelerate [bug](http://en.wikipedia.org/wiki/Software_bug) location and improve the code, on the [Usenet](http://en.wikipedia.org/wiki/Usenet) discussion group *comp.infosystems.www.authoring.cgi* on June 8, 1995. This release already had the basic functionality that PHP has as of 2013. This included Perl-like variables, form handling, and the ability to embed HTML. The [syntax](http://en.wikipedia.org/wiki/Syntax) resembled that of Perl but was simpler, more limited and less consistent.

Early PHP was not intended to be a new programming language, and grew organically, with Lerdorf noting in retrospect: "I don’t know how to stop it, there was never any intent to write a programming language […] I have absolutely no idea how to write a programming language, I just kept adding the next logical step on the way." A development team began to form and, after months of work and [beta](http://en.wikipedia.org/wiki/Beta_development_stage) testing, officially released PHP/FI 2 in November 1997.

One criticism of PHP is that it was not originally designed, but instead it was developed organically; among other things, this has led to inconsistent naming of functions and inconsistent ordering of their parameters. In some cases, the function names were chosen to match the lower-level libraries which PHP was "wrapping", while in some very early versions of PHP the length of the function names was used internally as a [hash function](http://en.wikipedia.org/wiki/Hash_function), so names were chosen to improve the distribution of hash values.

[Zeev Suraski](http://en.wikipedia.org/wiki/Zeev_Suraski) and [Andi Gutmans](http://en.wikipedia.org/wiki/Andi_Gutmans) rewrote the [parser](http://en.wikipedia.org/wiki/Parser) in 1997 and formed the base of PHP 3, changing the language's name to the [recursive acronym](http://en.wikipedia.org/wiki/Recursive_acronym) *PHP: Hypertext Preprocessor*. Afterwards, public testing of PHP 3 began, and the official launch came in June 1998. Suraski and Gutmans then started a new [rewrite](http://en.wikipedia.org/wiki/Rewrite_%28programming%29) of PHP's core, producing the [Zend Engine](http://en.wikipedia.org/wiki/Zend_Engine) in 1999. They also founded [Zend Technologies](http://en.wikipedia.org/wiki/Zend_Technologies) in [Ramat Gan](http://en.wikipedia.org/wiki/Ramat_Gan), Israel.

On May 22, 2000, PHP 4, powered by the Zend Engine 1.0, was released. As of August 2008 this branch reached version 4.4.9. PHP 4 is no longer under development nor will any security updates be released.

On July 13, 2004, PHP 5 was released, powered by the new Zend Engine II. PHP 5 included new features such as improved support for [object-oriented programming](http://en.wikipedia.org/wiki/Object-oriented_programming), the PHP Data Objects (PDO) extension (which defines a lightweight and consistent interface for accessing databases), and numerous performance enhancements. In 2008 PHP 5 became the only stable version under development. [Late static binding](http://en.wikipedia.org/wiki/Late_static_binding) had been missing from PHP and was added in version 5.3.

Many high-profile open-source projects ceased to support PHP 4 in new code as of February 5, 2008, because of the GoPHP5 initiative, provided by a consortium of PHP developers promoting the transition from PHP 4 to PHP 5.

Over time, PHP interpreters became available on most existing [32-bit](http://en.wikipedia.org/wiki/32-bit) and [64-bit](http://en.wikipedia.org/wiki/64-bit) operating systems, either by building them from the PHP source code, or by using pre-built binaries. For the PHP versions 5.3 and 5.4, the only available [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows) binary distributions were 32-bit [x86](http://en.wikipedia.org/wiki/X86) builds, requiring Windows 32-bit compatibility mode while using [Internet Information Services](http://en.wikipedia.org/wiki/Internet_Information_Services) (IIS) on a 64-bit Windows platform. PHP version 5.5 made the 64-bit [x86-64](http://en.wikipedia.org/wiki/X86-64) builds available for Microsoft Windows.

PHP 6 and Unicode

PHP received mixed reviews due to lacking native [Unicode](http://en.wikipedia.org/wiki/Unicode) support at the core language level. In 2005, a project headed by Andrei Zmievski was initiated to bring native Unicode support throughout PHP, by embedding the [International Components for Unicode](http://en.wikipedia.org/wiki/International_Components_for_Unicode) (ICU) library, and representing text strings as [UTF-16](http://en.wikipedia.org/wiki/UTF-16) internally. Since this would cause major changes both to the internals of the language and to user code, it was planned to release this as version 6.0 of the language, along with other major features then in development.

However, a shortage of developers who understood the necessary changes, and performance problems arising from conversion to and from UTF-16, which is rarely used in a web context, led to delays in the project. As a result, a PHP 5.3 release was created in 2009, with many non-Unicode features back-ported from PHP 6, notably namespaces. In March 2010, the project in its current form was officially abandoned, and a PHP 5.4 release was prepared containing most remaining non-Unicode features from PHP 6, such as traits and closure re-binding. Initial hopes were that a new plan would be formed for Unicode integration, but as of 2014 none has been adopted.

PHP 7

As of 2014, work is underway on a new major PHP version named PHP 7. There was some dispute as to whether the next major version of PHP was to be called PHP 6 or PHP 7. While the PHP 6 unicode experiment had never been released, a number of articles and book titles referenced the old PHP 6 name, which might have caused confusion if a new release were to reuse the PHP 6 name. After a vote, the name PHP 7 was chosen.

PHP 7 gets its foundations from an experimental PHP [branch](http://en.wikipedia.org/wiki/Branching_%28revision_control%29) that was originally named *phpng* (*PHP next generation*), which aims at optimizing PHP performance by refactoring the Zend Engine while retaining near-complete language compatibility. As of 14 July 2014, [WordPress](http://en.wikipedia.org/wiki/WordPress)-based benchmarks, which serve as the main benchmark suite for phpng project, show an almost 100% increase in performance. Changes from phpng are also expected to make it easier to improve performance in the future, as more compact data structures and other changes are seen as better suited for a successful migration to a [just-in-time](http://en.wikipedia.org/wiki/Just-in-time_compilation) (JIT) compiler. Because of the significant changes, this reworked Zend Engine will be called *Zend Engine 3*, succeeding the Zend Engine 2 used in PHP 5.

In terms of new language features, PHP 7 will add features such as return type declarations, which will complement its existing parameter type declarations. PHP 7 will also contain an improved variable syntax which is internally consistent and complete, resolving a long-standing issue in PHP, what will allow use of ->, [], (), {}, and :: operators with arbitrary meaningful left-hand-side expressions.

## Syntax

The following [Hello world program](http://en.wikipedia.org/wiki/Hello_world_program) is written in PHP code embedded in an [HTML](http://en.wikipedia.org/wiki/HTML) document:

<!DOCTYPE html>

<html>

<head>

<title>PHP Test</title>

</head>

<body>

<?php echo '<p>Hello World</p>'; ?>

</body>

</html>

However, as PHP does not need to be embedded in HTML or used with a web server, the simplest version of a Hello World program can be written like this, with the closing tag omitted as preferred in files containing pure PHP code (prior to PHP 5.4.0, this short syntax for echo() only works with the short\_open\_tag configuration setting enabled, while for PHP 5.4.0 and later it is always available):

<?= 'Hello world';

The PHP interpreter only executes PHP code within its [delimiters](http://en.wikipedia.org/wiki/Delimiter). Anything outside its delimiters is not processed by PHP (although non-PHP text is still subject to [control structures](http://en.wikipedia.org/wiki/Control_structure) described in PHP code). The most common delimiters are <?php to open and ?> to close PHP sections. <script language="php"> and </script> delimiters are also available, as are the shortened forms <? or <?= (which is used to echo back a [string](http://en.wikipedia.org/wiki/String_%28computer_science%29) or [variable](http://en.wikipedia.org/wiki/Variable_%28programming%29)) and ?> as well as [ASP](http://en.wikipedia.org/wiki/Active_Server_Pages)-style short forms <% or <%= and %>. Short delimiters make script files less portable, since support for them can be disabled in the local PHP configuration, and they are therefore discouraged. The purpose of all these delimiters is to separate PHP code from non-PHP code, including HTML.

The first form of delimiters, <?php and ?>, in [XHTML](http://en.wikipedia.org/wiki/XHTML) and other [XML](http://en.wikipedia.org/wiki/XML) documents, creates correctly formed XML "processing instructions". This means that the resulting mixture of PHP code and other markup in the server-side file is itself well-formed XML.

Variables are prefixed with a [dollar symbol](http://en.wikipedia.org/wiki/Dollar_sign), and a [type](http://en.wikipedia.org/wiki/Primitive_type) does not need to be specified in advance. PHP 5 introduced *type hinting* that allows functions to force their parameters to be objects of a specific class, arrays, interfaces or [callback functions](http://en.wikipedia.org/wiki/Callback_function). However, type hints can not be used with scalar types such as integer or string.

Unlike function and class names, variable names are case sensitive. Both double-quoted ("") and [heredoc](http://en.wikipedia.org/wiki/Heredoc) strings provide the ability to interpolate a variable's value into the string. PHP treats [newlines](http://en.wikipedia.org/wiki/Newline) as [whitespace](http://en.wikipedia.org/wiki/Whitespace_character) in the manner of a [free-form language](http://en.wikipedia.org/wiki/Free-form_language), and statements are terminated by a semicolon. PHP has three types of [comment syntax](http://en.wikipedia.org/wiki/Comparison_of_programming_languages_%28syntax%29#Comments): /\* \*/ marks block and inline comments; // as well as # are used for one-line comments. The echo statement is one of several facilities PHP provides to output text, *e.g.*, to a web browser.

In terms of keywords and language syntax, PHP is similar to most high level languages that follow the C style syntax. if conditions, for and while loops, and function returns are similar in syntax to languages such as C, C++, C#, Java and Perl.

### Data types

PHP stores whole numbers in a platform-dependent range, either a 64-bit or 32-bit [signed](http://en.wikipedia.org/wiki/Signed_number_representations) [integer](http://en.wikipedia.org/wiki/Integer_%28computer_science%29) equivalent to the [C-language long type](http://en.wikipedia.org/wiki/C_variable_types_and_declarations). Unsigned integers are converted to signed values in certain situations; this behavior is different from other programming languages. Integer variables can be assigned using decimal (positive and negative), [octal](http://en.wikipedia.org/wiki/Octal), [hexadecimal](http://en.wikipedia.org/wiki/Hexadecimal), and [binary](http://en.wikipedia.org/wiki/Binary_code) notations.

[Floating point](http://en.wikipedia.org/wiki/Floating_point) numbers are also stored in a platform-specific range. They can be specified using floating point notation, or two forms of [scientific notation](http://en.wikipedia.org/wiki/Scientific_notation). PHP has a native [Boolean](http://en.wikipedia.org/wiki/Boolean_datatype) type that is similar to the native Boolean types in [Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29) and [C++](http://en.wikipedia.org/wiki/C%2B%2B). Using the Boolean type conversion rules, non-zero values are interpreted as true and zero as false, as in [Perl](http://en.wikipedia.org/wiki/Perl) and C++.

The null data type represents a variable that has no value; *NULL* is the only allowed value for this data type.

Variables of the "resource" type represent references to resources from external sources. These are typically created by functions from a particular extension, and can only be processed by functions from the same extension; examples include file, image, and database resources.

Arrays can contain elements of any type that PHP can handle, including resources, objects, and even other arrays. Order is preserved in lists of values and in [hashes](http://en.wikipedia.org/wiki/Hash_table) with both keys and values, and the two can be intermingled. PHP also supports [strings](http://en.wikipedia.org/wiki/String_%28computing%29), which can be used with single quotes, double quotes, nowdoc or [heredoc](http://en.wikipedia.org/wiki/Heredoc) syntax.

The Standard PHP Library (SPL) attempts to solve standard problems and implements efficient data access interfaces and classes.

### Functions

PHP has hundreds of functions provided by the core language functionality and thousands more available via various extensions; these functions are well documented in the online PHP documentation. However, the built-in library has a wide variety of naming conventions and associated inconsistencies, as described under [history](http://en.wikipedia.org/wiki/PHP#ORGANIC) above.

Additional functions can be defined by the developer:

function myAge($birthYear) // defines a function, this one is named "myAge"

{

$yearsOld = date('Y') - $birthYear; // calculates the age

return $yearsOld . ' year' . ($yearsOld != 1 ? 's' : ''); // returns the age in a descriptive form

}

echo 'I am currently ' . myAge(1981) . ' old.'; // outputs the text concatenated

// with the return value of myAge()

// As the result of this syntax, myAge() is called.

// In 2014, the output of this sample program will be 'I am currently 33 years old.'

In PHP, normal functions are not [first-class](http://en.wikipedia.org/wiki/First-class_function) and can only be referenced by their name directly, or dynamically by a variable containing the name of the function (referred to as "variable functions"). User-defined functions can be created at any time without being [prototyped](http://en.wikipedia.org/wiki/Function_prototype). Functions can be defined inside code blocks, permitting a [run-time decision](http://en.wikipedia.org/wiki/Dynamic_dispatch) as to whether or not a function should be defined. Function calls must use parentheses, with the exception of zero-argument class [constructor](http://en.wikipedia.org/wiki/Constructor_%28computer_science%29) functions called with the PHP new operator, where parentheses are optional.

Until PHP 5.3, support for true [anonymous functions](http://en.wikipedia.org/wiki/Anonymous_functions) or [closures](http://en.wikipedia.org/wiki/Closure_%28computer_science%29) did not exist in PHP. While create\_function() exists since PHP 4.0.1, it is merely a thin wrapper around eval() that allows normal PHP functions to be created during program execution.Also, support for variable functions allows normal PHP functions to be used, for example, as [callbacks](http://en.wikipedia.org/wiki/Callback_function) or within [function tables](http://en.wikipedia.org/wiki/Function_table). PHP 5.3 added support for closures, which are true anonymous, first-class functions, whose syntax can be seen in the following example:

function getAdder($x)

{

return function($y) use ($x)

{

return $x + $y;

};

}

$adder = getAdder(8);

echo $adder(2); // prints "10"

In the example above, getAdder() function creates a closure using passed argument $x (the keyword use imports a variable from the lexical context), which takes an additional argument $y, and returns the created closure to the caller. Such a function is a first-class object, meaning that it can be stored in a variable, passed as a parameter to other functions, etc.

The [goto](http://en.wikipedia.org/wiki/Goto) flow control statement is used as in the following example:

function lock()

{

$file = fopen('file.txt', 'r+');

retry:

if (!flock($file, LOCK\_EX | LOCK\_NB))

goto retry;

fwrite($file, 'Success!');

fclose($file);

}

When flock() is called, PHP opens a file and tries to lock it. The target label retry: defines the point to which execution should return if flock() is unsuccessful and goto retry; is called. The goto statement is restricted and requires that the target label be in the same file and context.

The goto statement has been supported since PHP 5.3.

### Objects

Basic [object-oriented programming](http://en.wikipedia.org/wiki/Object-oriented_programming) functionality was added in PHP 3 and improved in PHP 4. Object handling was completely rewritten for PHP 5, expanding the feature set and enhancing performance. In previous versions of PHP, objects were handled like [value types](http://en.wikipedia.org/wiki/Value_type). The drawback of this method was that the whole object was copied when a variable was assigned or passed as a parameter to a method. In the new approach, objects are referenced by [handle](http://en.wikipedia.org/wiki/Handle_%28computing%29), and not by value.

PHP 5 introduced private and protected [member variables](http://en.wikipedia.org/wiki/Member_variable) and methods, along with [abstract classes](http://en.wikipedia.org/wiki/Abstract_type), [final classes](http://en.wikipedia.org/wiki/Final_type), [abstract methods](http://en.wikipedia.org/wiki/Abstract_method), and [final methods](http://en.wikipedia.org/wiki/Final_method). It also introduced a standard way of declaring [constructors](http://en.wikipedia.org/wiki/Constructor_%28computer_science%29) and [destructors](http://en.wikipedia.org/wiki/Destructor_%28computer_science%29), similar to that of other object-oriented languages such as [C++](http://en.wikipedia.org/wiki/C%2B%2B), and a standard [exception handling](http://en.wikipedia.org/wiki/Exception_handling) model. Furthermore, PHP 5 added [interfaces](http://en.wikipedia.org/wiki/Interface_%28computing%29) and allowed for multiple interfaces to be implemented. There are special interfaces that allow objects to interact with the runtime system. [Objects](http://en.wikipedia.org/wiki/Object_%28computer_science%29) implementing ArrayAccess can be used with [array](http://en.wikipedia.org/wiki/Array_data_type) syntax and objects implementing [Iterator](http://en.wikipedia.org/wiki/Iterator) or [IteratorAggregate](http://en.wikipedia.org/wiki/IteratorAggregate) can be used with the foreach [language construct](http://en.wikipedia.org/wiki/Language_construct). There is no [virtual table](http://en.wikipedia.org/wiki/Virtual_table) feature in the engine, so [static variables](http://en.wikipedia.org/wiki/Static_variable) are bound with a name instead of a reference at compile time.

If the developer creates a copy of an object using the reserved word clone, the Zend engine will check whether a \_\_clone() method has been defined. If not, it will call a default \_\_clone() which will copy the object's properties. If a \_\_clone() method is defined, then it will be responsible for setting the necessary properties in the created object. For convenience, the engine will supply a function that imports the properties of the source object, so the programmer can start with a by-value [replica](http://en.wiktionary.org/wiki/replica) of the source object and only override properties that need to be changed.

The following is a basic example of [object-oriented programming](http://en.wikipedia.org/wiki/Object-oriented_programming) in PHP:

class Person

{

public $firstName;

public $lastName;

public function \_\_construct($firstName, $lastName = '') { // optional second argument

$this->firstName = $firstName;

$this->lastName = $lastName;

}

public function greet() {

return 'Hello, my name is ' . $this->firstName .

(($this->lastName != '') ? (' ' . $this->lastName) : '') . '.';

}

public static function staticGreet($firstName, $lastName) {

return 'Hello, my name is ' . $firstName . ' ' . $lastName . '.';

}

}

$he = new Person('John', 'Smith');

$she = new Person('Sally', 'Davis');

$other = new Person('iAmine');

echo $he->greet(); // prints "Hello, my name is John Smith."

echo '<br />';

echo $she->greet(); // prints "Hello, my name is Sally Davis."

echo '<br />';

echo $other->greet(); // prints "Hello, my name is iAmine."

echo '<br />';

echo Person::staticGreet('Jane', 'Doe'); // prints "Hello, my name is Jane Doe."

The [visibility](http://en.wikipedia.org/wiki/Visibility_%28computer_science%29) of PHP properties and methods is defined using the [keywords](http://en.wikipedia.org/wiki/Keyword_%28computer_programming%29) public, private, and protected. The default is public, if only [var](http://en.wikipedia.org/wiki/Variable_%28programming%29) is used; var is a synonym for public. Items declared public can be accessed everywhere. protected limits access to [inherited classes](http://en.wikipedia.org/wiki/Inherited_class) (and to the class that defines the item). private limits visibility only to the class that defines the item. Objects of the same type have access to each other's private and protected members even though they are not the same instance. PHP's member visibility features have sometimes been described as "highly useful." However, they have also sometimes been described as "at best irrelevant and at worst positively harmful."

## Implementations

The original, only complete and most widely used PHP implementation is powered by the [Zend Engine](http://en.wikipedia.org/wiki/Zend_Engine) and known simply as PHP. To disambiguate it from other implementations, it is sometimes unofficially referred to as "Zend PHP". The Zend Engine [compiles](http://en.wikipedia.org/wiki/Compiler) PHP [source code](http://en.wikipedia.org/wiki/Source_code) on-the-fly into an internal format that it can execute, thus it works as an [interpreter](http://en.wikipedia.org/wiki/Interpreter_%28computing%29). It is also the "reference implementation" of PHP, as PHP has no formal specification, and so the semantics of Zend PHP define the semantics of PHP itself. Due to the complex and nuanced semantics of PHP, defined by how Zend works, it is difficult for competing implementations to offer complete compatibility.

PHP's single-request-per-script-execution model, and the fact the Zend Engine is an interpreter, lead to inefficiency. As a result, various products have been developed to help improve PHP performance. In order to speed up execution time and not have to compile the PHP source code every time the web page is accessed, PHP scripts can also be deployed in the PHP engine's internal format by using an [opcode](http://en.wikipedia.org/wiki/Opcode) cache, which works by [caching](http://en.wikipedia.org/wiki/Cache_%28computing%29) the compiled form of a PHP script (opcodes) in [shared memory](http://en.wikipedia.org/wiki/Shared_memory) to avoid the overhead of [parsing](http://en.wikipedia.org/wiki/Parsing) and [compiling](http://en.wikipedia.org/wiki/Compiling) the code every time the script runs. An opcode cache, [Zend Opcache](http://en.wikipedia.org/wiki/Zend_Opcache), is built into PHP since version 5.5. Another example of a widely used opcode cache is the [Alternative PHP Cache](http://en.wikipedia.org/wiki/Alternative_PHP_Cache) (APC), which is available as a [PECL](http://en.wikipedia.org/wiki/PHP_Extension_Community_Library) extension.

While Zend PHP is still the most popular implementation, several other implementations have been developed. Some of these are [compilers](http://en.wikipedia.org/wiki/Compiler) or support [JIT compilation](http://en.wikipedia.org/wiki/JIT_compilation), and hence offer performance benefits over Zend PHP at the expense of lacking full PHP compatibility. Alternative implementations include the following:

* [HipHop Virtual Machine](http://en.wikipedia.org/wiki/HipHop_Virtual_Machine) (HHVM) – developed at Facebook and available as open source, it converts PHP code into a high-level bytecode (commonly known as an [intermediate language](http://en.wikipedia.org/wiki/Intermediate_language)), which is then translated into x86-64 machine code dynamically at runtime by a [just-in-time](http://en.wikipedia.org/wiki/Just-in-time_compiler) (JIT) compiler, resulting in up to 6× performance improvements.
* [Parrot](http://en.wikipedia.org/wiki/Parrot_virtual_machine) – a virtual machine designed to run dynamic languages efficiently; Pipp transforms the PHP source code into the [Parrot intermediate representation](http://en.wikipedia.org/wiki/Parrot_intermediate_representation), which is then translated into the Parrot's bytecode and executed by the virtual machine.
* [Phalanger](http://en.wikipedia.org/wiki/Phalanger_%28compiler%29) – compiles PHP into [Common Intermediate Language](http://en.wikipedia.org/wiki/Common_Intermediate_Language) (CIL) bytecode
* [HipHop](http://en.wikipedia.org/wiki/HipHop_for_PHP) – developed at Facebook and available as open source, it transforms the PHP scripts into [C++](http://en.wikipedia.org/wiki/C%2B%2B) code and then compiles the resulting code, reducing the server load up to 50%. In early 2013, Facebook deprecated it in favor of HHVM due to multiple reasons, including deployment difficulties and lack of support for the whole PHP language, including the create\_function() and eval() constructs.

## Licensing

PHP is [free software](http://en.wikipedia.org/wiki/Free_software) released under the [PHP License](http://en.wikipedia.org/wiki/PHP_License), which stipulates that:

Products derived from this software may not be called "PHP", nor may "PHP" appear in their name, without prior written permission from group@php.net. You may indicate that your software works in conjunction with PHP by saying "[Foo](http://en.wikipedia.org/wiki/Foo) for PHP" instead of calling it "PHP Foo" or "phpfoo".

This restriction on use of the name *PHP* makes the PHP License incompatible with the [GNU General Public License](http://en.wikipedia.org/wiki/GNU_General_Public_License) (GPL), while the Zend License is incompatible due to an advertising clause similar to that of the original license of BSD.

## Development and community

PHP includes various [free and open-source libraries](http://en.wikipedia.org/wiki/List_of_PHP_libraries) in its source distribution, or uses them in resulting PHP binary builds. PHP is fundamentally an [Internet](http://en.wikipedia.org/wiki/Internet)-aware system with built-in modules for accessing [File Transfer Protocol](http://en.wikipedia.org/wiki/File_Transfer_Protocol) (FTP) servers and many database servers, including [PostgreSQL](http://en.wikipedia.org/wiki/PostgreSQL), [MySQL](http://en.wikipedia.org/wiki/MySQL), [Microsoft SQL Server](http://en.wikipedia.org/wiki/Microsoft_SQL_Server) and [SQLite](http://en.wikipedia.org/wiki/SQLite) (which is an embedded database), [LDAP](http://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol) servers, and others. Numerous functions familiar to C programmers, such as those in the [stdio](http://en.wikipedia.org/wiki/Stdio.h) family, are available in standard PHP builds.

PHP allows developers to write [extensions](http://en.wikipedia.org/wiki/Software_extension) in [C](http://en.wikipedia.org/wiki/C_%28programming_language%29) to add functionality to the PHP language. PHP extensions can be compiled statically into PHP or loaded dynamically at runtime. Numerous extensions have been written to add support for the [Windows API](http://en.wikipedia.org/wiki/Windows_API), process management on [Unix-like](http://en.wikipedia.org/wiki/Unix-like) [operating systems](http://en.wikipedia.org/wiki/Operating_system), multibyte strings ([Unicode](http://en.wikipedia.org/wiki/Unicode)), [cURL](http://en.wikipedia.org/wiki/CURL), and several popular [compression formats](http://en.wikipedia.org/wiki/Archive_format). Other PHP features made available through extensions include integration with [IRC](http://en.wikipedia.org/wiki/Internet_Relay_Chat), dynamic generation of images and [Adobe Flash](http://en.wikipedia.org/wiki/Adobe_Flash) content, *PHP Data Objects* (PDO) as an abstraction layer used for accessing databases, and even [speech synthesis](http://en.wikipedia.org/wiki/Speech_synthesis). Some of the language's core functions, such as those dealing with strings and arrays, are also implemented as extensions. The [PHP Extension Community Library](http://en.wikipedia.org/wiki/PHP_Extension_Community_Library) (PECL) project is a repository for extensions to the PHP language.

Some other projects, such as *Zephir*, provide the ability for PHP extensions to be created in a high-level language and compiled into native PHP extensions. Such an approach, instead of writing PHP extensions directly in C, simplifies the development of extensions and reduces the time required for programming and testing.

[Zend Technologies](http://en.wikipedia.org/wiki/Zend_Technologies) provides a [certification](http://en.wikipedia.org/wiki/Zend_Certified_Engineer) exam for programmers to become certified PHP developers.

## Installation and configuration

There are two primary ways for adding support for PHP to a web server – as a native web server module, or as a CGI executable. PHP has a direct module interface called [Server Application Programming Interface](http://en.wikipedia.org/wiki/Server_Application_Programming_Interface) (SAPI), which is supported by many web servers including [Apache HTTP Server](http://en.wikipedia.org/wiki/Apache_HTTP_Server), [Microsoft IIS](http://en.wikipedia.org/wiki/Microsoft_Internet_Information_Server), [Netscape](http://en.wikipedia.org/wiki/Netscape) (now defunct) and [iPlanet](http://en.wikipedia.org/wiki/IPlanet). Some other web servers, such as OmniHTTPd, support the [Internet Server Application Programming Interface](http://en.wikipedia.org/wiki/Internet_Server_Application_Programming_Interface) (ISAPI), which is a [Microsoft](http://en.wikipedia.org/wiki/Microsoft)'s web server module interface. If PHP has no module support for a web server, it can always be used as a [Common Gateway Interface](http://en.wikipedia.org/wiki/Common_Gateway_Interface) (CGI) or [FastCGI](http://en.wikipedia.org/wiki/FastCGI) processor; in that case, the web server is configured to use PHP's CGI executable to process all requests to PHP files.

PHP-FPM (FastCGI Process Manager) is an alternative FastCGI implementation for PHP, bundled with the official PHP distribution since version 5.3.3. When compared to the older FastCGI implementation, it contains some additional features, mostly useful for heavily loaded web servers.

When using PHP for command-line scripting, a PHP [command-line interface](http://en.wikipedia.org/wiki/Command-line_interface) (CLI) executable is needed. PHP supports a CLI SAPI as of PHP 4.3.0. The main focus of this SAPI is developing [shell](http://en.wikipedia.org/wiki/Command_line_interface) applications using PHP. There are quite a few differences between the CLI SAPI and other SAPIs, although they do share many of the same behaviors.

PHP can also be used for writing desktop [graphical user interface](http://en.wikipedia.org/wiki/Graphical_user_interface) (GUI) applications, by using the [PHP-GTK](http://en.wikipedia.org/wiki/PHP-GTK) extension. PHP-GTK is not included in the official PHP distribution, and as an extension it can be used only with PHP versions 5.1.0 and newer. The most common way of installing PHP-GTK is compiling it from the source code.

When PHP is installed and used in [cloud](http://en.wikipedia.org/wiki/Cloud_computing) environments, [software development kits](http://en.wikipedia.org/wiki/Software_development_kit) (SDKs) are provided for using cloud-specific features. For example:

* [Amazon Web Services](http://en.wikipedia.org/wiki/Amazon_Web_Services) provides the AWS SDK for PHP
* [Windows Azure](http://en.wikipedia.org/wiki/Windows_Azure) can be used with the Windows Azure SDK for PHP.

Numerous configuration options are supported, affecting both core PHP features and extensions. Configuration file php.ini is searched for in different locations, depending on the way PHP is used. The configuration file is split into various sections, while some of the configuration options can be also set within the web server configuration.

## [http://upload.wikimedia.org/wikipedia/commons/thumb/8/82/LAMP_software_bundle.svg/400px-LAMP_software_bundle.svg.png](http://en.wikipedia.org/wiki/File:LAMP_software_bundle.svg)

## Use

A broad overview of the LAMP software bundle, displayed here together with [Squid](http://en.wikipedia.org/wiki/Squid_%28software%29).

PHP is a general-purpose scripting language that is especially suited to [server-side](http://en.wikipedia.org/wiki/Server-side_scripting) [web development](http://en.wikipedia.org/wiki/Web_development), in which case PHP generally runs on a [web server](http://en.wikipedia.org/wiki/Web_server). Any PHP code in a requested file is [executed](http://en.wikipedia.org/wiki/Execution_%28computing%29) by the PHP runtime, usually to create [dynamic web page](http://en.wikipedia.org/wiki/Dynamic_web_page) content or dynamic images used on websites or elsewhere. It can also be used for [command-line](http://en.wikipedia.org/wiki/Command-line) scripting and [client-side](http://en.wikipedia.org/wiki/Client-side) [graphical user interface](http://en.wikipedia.org/wiki/Graphical_user_interface) (GUI) applications. PHP can be deployed on most web servers, many [operating systems](http://en.wikipedia.org/wiki/Operating_system) and [platforms](http://en.wikipedia.org/wiki/Computing_platform), and can be used with many [relational database management systems](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS). Most [web hosting](http://en.wikipedia.org/wiki/Web_hosting) providers support PHP for use by their clients. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.

PHP acts primarily as a [filter](http://en.wikipedia.org/wiki/Filter_%28software%29), taking input from a file or stream containing text and/or PHP instructions and outputting another stream of data. Most commonly the output will be HTML, although it could be [JSON](http://en.wikipedia.org/wiki/JSON), [XML](http://en.wikipedia.org/wiki/XML) or [binary data](http://en.wikipedia.org/wiki/Binary_data) such as image or audio formats. Since PHP 4, the PHP [parser](http://en.wikipedia.org/wiki/Parser) [compiles](http://en.wikipedia.org/wiki/Compiler) input to produce [bytecode](http://en.wikipedia.org/wiki/Bytecode) for processing by the [Zend Engine](http://en.wikipedia.org/wiki/Zend_Engine), giving improved performance over its [interpreter](http://en.wikipedia.org/wiki/Interpreter_%28computing%29) predecessor.

Originally designed to create dynamic [web pages](http://en.wikipedia.org/wiki/Web_page), PHP now focuses mainly on [server-side scripting](http://en.wikipedia.org/wiki/Server-side_scripting),[[126]](http://en.wikipedia.org/wiki/PHP#cite_note-126) and it is similar to other server-side scripting languages that provide dynamic content from a web server to a [client](http://en.wikipedia.org/wiki/Client_%28computing%29), such as [Microsoft](http://en.wikipedia.org/wiki/Microsoft)'s [ASP.NET](http://en.wikipedia.org/wiki/ASP.NET), [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems)' [JavaServer Pages](http://en.wikipedia.org/wiki/JavaServer_Pages), and [mod\_perl](http://en.wikipedia.org/wiki/Mod_perl). PHP has also attracted the development of many [software frameworks](http://en.wikipedia.org/wiki/Software_framework) that provide building blocks and a design structure to promote [rapid application development](http://en.wikipedia.org/wiki/Rapid_application_development) (RAD). Some of these include [PRADO](http://en.wikipedia.org/wiki/PRADO_%28framework%29), [CakePHP](http://en.wikipedia.org/wiki/CakePHP), [Symfony](http://en.wikipedia.org/wiki/Symfony), [CodeIgniter](http://en.wikipedia.org/wiki/CodeIgniter), [Laravel](http://en.wikipedia.org/wiki/Laravel), [Yii Framework](http://en.wikipedia.org/wiki/Yii_Framework), and [Zend Framework](http://en.wikipedia.org/wiki/Zend_Framework), offering features similar to other [web application frameworks](http://en.wikipedia.org/wiki/List_of_web_application_frameworks).

MYSQL

MySQL is (as of July 2013) the world's second most widely used [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS) and most widely used open-source RDBMS. It is named after co-founder [Michael Widenius](http://en.wikipedia.org/wiki/Michael_Widenius)'s daughter, my. The [SQL](http://en.wikipedia.org/wiki/SQL) acronym stands for [Structured Query Language](http://en.wikipedia.org/wiki/Structured_Query_Language).

The MySQL development project has made its [source code](http://en.wikipedia.org/wiki/Source_code) available under the terms of the [GNU General Public License](http://en.wikipedia.org/wiki/GNU_General_Public_License), as well as under a variety of [proprietary](http://en.wikipedia.org/wiki/Proprietary_software) agreements. MySQL was owned and sponsored by a single [for-profit](http://en.wikipedia.org/wiki/Business) firm, the [Swedish](http://en.wikipedia.org/wiki/Sweden) company [MySQL AB](http://en.wikipedia.org/wiki/MySQL_AB), now owned by [Oracle Corporation](http://en.wikipedia.org/wiki/Oracle_Corporation).

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used [LAMP](http://en.wikipedia.org/wiki/LAMP_%28software_bundle%29) open source web application software stack (and other ['AMP'](http://en.wikipedia.org/wiki/List_of_AMP_packages) stacks). LAMP is an acronym for "[Linux](http://en.wikipedia.org/wiki/Linux), [Apache](http://en.wikipedia.org/wiki/Apache_HTTP_Server), MySQL, [Perl](http://en.wikipedia.org/wiki/Perl)/[PHP](http://en.wikipedia.org/wiki/PHP)/[Python](http://en.wikipedia.org/wiki/Python_%28programming_language%29)." [Free-software](http://en.wikipedia.org/wiki/Free_software)-open source projects that require a full-featured database management system often use MySQL.

For proprietary use, several paid editions are available, and offer additional functionality. Applications which use MySQL databases include: [TYPO3](http://en.wikipedia.org/wiki/TYPO3), [MODx](http://en.wikipedia.org/wiki/MODx), [Joomla](http://en.wikipedia.org/wiki/Joomla), [WordPress](http://en.wikipedia.org/wiki/WordPress), [phpBB](http://en.wikipedia.org/wiki/PhpBB), [MyBB](http://en.wikipedia.org/wiki/MyBB), [Drupal](http://en.wikipedia.org/wiki/Drupal) and other software. MySQL is also used in many high-profile, large-scale [websites](http://en.wikipedia.org/wiki/Website), including [Google](http://en.wikipedia.org/wiki/Google)[[13]](http://en.wikipedia.org/wiki/MySQL#cite_note-mysqlatgoogle-14)[[14]](http://en.wikipedia.org/wiki/MySQL#cite_note-15) (though not for searches), [Face book](http://en.wikipedia.org/wiki/Facebook), [Twitter](http://en.wikipedia.org/wiki/Twitter), [Flickr](http://en.wikipedia.org/wiki/Flickr), and [YouTube](http://en.wikipedia.org/wiki/YouTube).

## Interfaces

MySQL is a [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS), and ships with no [GUI](http://en.wikipedia.org/wiki/Graphical_user_interface) tools to administer MySQL databases or manage data contained within the databases. Users may use the included [command line](http://en.wikipedia.org/wiki/Command_line) tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, [MySQL Workbench](http://en.wikipedia.org/wiki/MySQL_Workbench) is actively developed by Oracle, and is freely available for use.

### Graphical

The official [MySQL Workbench](http://en.wikipedia.org/wiki/MySQL_Workbench) is a free integrated environment developed by MySQL AB, that enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, [MySQL GUI Tools](http://en.wikipedia.org/wiki/MySQL_GUI_Tools). Similar to other third-party packages, but still considered the authoritative MySQL front end, MySQL Workbench lets users manage database design & modeling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).

MySQL Workbench is available in two editions, the regular [free and open source](http://en.wikipedia.org/wiki/Free_and_open_source_software) *Community Edition* which may be downloaded from the MySQL website, and the proprietary *Standard Edition* which extends and improves the feature set of the Community Edition.

Third-party proprietary and free graphical administration applications (or "front ends") are available that integrate with MySQL and enable users to work with database structure and data visually. Some well-known front ends, in alphabetical order, are:

* [Adminer](http://en.wikipedia.org/wiki/Adminer) – a free MySQL front end written in one [PHP](http://en.wikipedia.org/wiki/PHP) script, capable of managing multiple databases, with many [CSS](http://en.wikipedia.org/wiki/Cascading_Style_Sheets) skins available
* [Chive](http://en.wikipedia.org/wiki/Chive) a free, open source, web-based database management tool designed as an alternative to phpMyAdmin
* [Database Workbench](http://en.wikipedia.org/wiki/Database_Workbench) – a software application for development and administration of multiple relational databases including MySQL, with interoperationality between different database systems
* [DBEdit](http://en.wikipedia.org/wiki/DBEdit) – a free front end for MySQL and other databases
* [HeidiSQL](http://en.wikipedia.org/wiki/HeidiSQL) – a full featured free front end that runs on [Windows](http://en.wikipedia.org/wiki/Windows), and can connect to local or remote MySQL servers to manage databases, tables, column structure, and individual data records. Also supports specialised GUI features for date/time fields and enumerated multiple-value fields.
* [LibreOffice Base](http://en.wikipedia.org/wiki/LibreOffice) – LibreOffice Base allows the creation and management of databases, preparation of forms and reports that provide end users easy access to data. Like [Microsoft Access](http://en.wikipedia.org/wiki/Microsoft_Access), it can be used as a front-end for various database systems, including Access databases (JET), ODBC data sources, and MySQL or [PostgreSQL](http://en.wikipedia.org/wiki/PostgreSQL).
* [Navicat](http://en.wikipedia.org/wiki/Navicat) – a series of proprietary graphical database management applications, developed for Windows, Macintosh and Linux
* [OpenOffice.org](http://en.wikipedia.org/wiki/OpenOffice.org) – freely available [OpenOffice.org Base](http://en.wikipedia.org/wiki/OpenOffice.org_Base) can manage MySQL databases if the entire suite is installed
* [phpMyAdmin](http://en.wikipedia.org/wiki/PhpMyAdmin) – a free Web-based front-end, widely installed by [web hosting services](http://en.wikipedia.org/wiki/Web_hosting_service) since it is developed in PHP and included in the LAMP stack, and [MAMP](http://en.wikipedia.org/wiki/MAMP), [XAMPP](http://en.wikipedia.org/wiki/XAMPP) and [WAMP](http://en.wikipedia.org/wiki/WAMP_%28software_bundle%29) software bundle installers
* [SQLBuddy](http://en.wikipedia.org/wiki/SQLBuddy) – a free Web-based front end, developed in PHP
* [SQLyog](http://en.wikipedia.org/wiki/SQLyog) – proprietary, but there is also a free 'community' edition available
* [Toad for MySQL](http://en.wikipedia.org/wiki/TOAD_%28software%29) – a free development and administration front end for MySQL from [Dell Software](http://en.wikipedia.org/wiki/Dell_Software)
* [Webmin](http://en.wikipedia.org/wiki/Webmin) – a free Web-based management utility and a MySQL front end, developed in Perl with some parts written in Java

Other available proprietary MySQL front ends include [dbForge Studio for MySQL](http://en.wikipedia.org/w/index.php?title=DbForge_Studio_for_MySQL&action=edit&redlink=1), DBStudio, [Epictetus](http://en.wikipedia.org/wiki/Epictetus_Database_Client), [Microsoft Access](http://en.wikipedia.org/wiki/Microsoft_Access), [Oracle SQL Developer](http://en.wikipedia.org/wiki/Oracle_SQL_Developer), SchemaBank, [SQLPro SQL Client](http://en.wikipedia.org/wiki/SQLPro_SQL_Client), [Toad Data Modeler](http://en.wikipedia.org/wiki/Toad_Data_Modeler) and [DaDaBIK](http://en.wikipedia.org/wiki/DaDaBIK).

### Command line

MySQL ships with many [command line](http://en.wikipedia.org/wiki/Command_line) tools, from which the main interface is 'mysql' client. Third parties have also developed tools to manage MySQL servers.

* MySQL Utilities – a set of utilities designed to perform common maintenance and administrative tasks. Originally included as part of the MySQL Workbench, the utilities are now a stand-alone download available from Oracle.
* Percona Toolkit – a cross-platform toolkit for MySQL, developed in [Perl](http://en.wikipedia.org/wiki/Perl).[[31]](http://en.wikipedia.org/wiki/MySQL#cite_note-32) Percona Toolkit can be used to prove replication is working correctly, fix corrupted data, automate repetitive tasks, and speed up servers. Percona Toolkit is included with several [Linux](http://en.wikipedia.org/wiki/Linux) distributions such as [CentOS](http://en.wikipedia.org/wiki/CentOS) and [Debian](http://en.wikipedia.org/wiki/Debian), and packages are available for [Fedora](http://en.wikipedia.org/wiki/Fedora_%28operating_system%29) and [Ubuntu](http://en.wikipedia.org/wiki/Ubuntu_%28operating_system%29) as well. Percona Toolkit was originally developed as Maatkit, but as of late 2011, Maatkit is no longer developed.

### Programming

MySQL works on many [system platforms](http://en.wikipedia.org/wiki/System_platform), including [AIX](http://en.wikipedia.org/wiki/AIX_operating_system), [BSDi](http://en.wikipedia.org/wiki/BSD/OS), [FreeBSD](http://en.wikipedia.org/wiki/FreeBSD), [HP-UX](http://en.wikipedia.org/wiki/HP-UX), [eComStation](http://en.wikipedia.org/wiki/EComStation), [i5/OS](http://en.wikipedia.org/wiki/IBM_i5/OS), [IRIX](http://en.wikipedia.org/wiki/IRIX), [Linux](http://en.wikipedia.org/wiki/Linux), [OS X](http://en.wikipedia.org/wiki/OS_X), [Microsoft Windows](http://en.wikipedia.org/wiki/Microsoft_Windows), [NetBSD](http://en.wikipedia.org/wiki/NetBSD), [Novell NetWare](http://en.wikipedia.org/wiki/Novell_NetWare), [OpenBSD](http://en.wikipedia.org/wiki/OpenBSD), [OpenSolaris](http://en.wikipedia.org/wiki/OpenSolaris), [OS/2](http://en.wikipedia.org/wiki/OS/2) Warp, [QNX](http://en.wikipedia.org/wiki/QNX), [Oracle Solaris](http://en.wikipedia.org/wiki/Solaris_%28operating_system%29), [Symbian](http://en.wikipedia.org/wiki/Symbian), [SunOS](http://en.wikipedia.org/wiki/SunOS), [SCO OpenServer](http://en.wikipedia.org/wiki/SCO_OpenServer), SCO [UnixWare](http://en.wikipedia.org/wiki/UnixWare), [Sanos](http://en.wikipedia.org/wiki/Sanos) and [Tru64](http://en.wikipedia.org/wiki/Tru64). A port of MySQL to [OpenVMS](http://en.wikipedia.org/wiki/OpenVMS) also exists.

MySQL is written in [C](http://en.wikipedia.org/wiki/C_%28programming_language%29) and [C++](http://en.wikipedia.org/wiki/C%2B%2B). Its SQL parser is written in [yacc](http://en.wikipedia.org/wiki/Yacc), but it uses a home-brewed [lexical analyzer](http://en.wikipedia.org/wiki/Lexical_analysis). Many [programming languages](http://en.wikipedia.org/wiki/Programming_language) with language-specific [APIs](http://en.wikipedia.org/wiki/Application_programming_interface) include [libraries](http://en.wikipedia.org/wiki/Library_%28computing%29) for accessing MySQL databases. These include MySQL Connector/Net for integration with Microsoft's [Visual Studio](http://en.wikipedia.org/wiki/Visual_Studio) (languages such as [C#](http://en.wikipedia.org/wiki/C_Sharp_%28programming_language%29) and [VB](http://en.wikipedia.org/wiki/Visual_Basic) are most commonly used) and the JDBC driver for Java. In addition, an [ODBC](http://en.wikipedia.org/wiki/ODBC) interface called [MyODBC](http://en.wikipedia.org/wiki/MyODBC) allows additional programming languages that support the ODBC interface to communicate with a MySQL database, such as [ASP](http://en.wikipedia.org/wiki/Active_Server_Pages) or [ColdFusion](http://en.wikipedia.org/wiki/Adobe_ColdFusion). The [HTSQL](http://en.wikipedia.org/wiki/HTSQL) – [URL](http://en.wikipedia.org/wiki/Uniform_resource_locator)-based query method also ships with a MySQL adapter, allowing direct interaction between a MySQL database and any web client via structured URLs.

## Features

MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary [Enterprise Server](http://en.wikipedia.org/wiki/MySQL_Enterprise). MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plug-in, but otherwise shares the version numbering system and is built from the same code base.

Major features as available in MySQL 5.6:

* A broad subset of [ANSI SQL 99](http://en.wikipedia.org/wiki/SQL:1999), as well as extensions
* Cross-platform support
* [Stored procedures](http://en.wikipedia.org/wiki/Stored_procedure), using a procedural language that closely adheres to [SQL/PSM](http://en.wikipedia.org/wiki/SQL/PSM)
* [Triggers](http://en.wikipedia.org/wiki/Database_trigger)
* [Cursors](http://en.wikipedia.org/wiki/Cursor_%28databases%29)
* Updatable [views](http://en.wikipedia.org/wiki/View_%28SQL%29)
* [Online DDL](http://en.wikipedia.org/wiki/Data_Definition_Language) when using the InnoDB Storage Engine.
* [Information schema](http://en.wikipedia.org/wiki/Information_schema)
* Performance Schema
* A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.
* [X/Open XA](http://en.wikipedia.org/wiki/X/Open_XA) [distributed transaction processing](http://en.wikipedia.org/wiki/Distributed_transaction_processing) (DTP) support; [two phase commit](http://en.wikipedia.org/wiki/Two-phase-commit_protocol) as part of this, using the default [InnoDB](http://en.wikipedia.org/wiki/InnoDB) storage engine
* Transactions with [save points](http://en.wikipedia.org/wiki/Savepoint) when using the default InnoDB Storage Engine. The NDB Cluster Storage Engine also supports transactions.
* [ACID](http://en.wikipedia.org/wiki/Atomicity,_consistency,_isolation,_durability) compliance when using InnoDB and NDB Cluster Storage Engines
* [SSL](http://en.wikipedia.org/wiki/Secure_Sockets_Layer) support
* Query [caching](http://en.wikipedia.org/wiki/Cache_%28computing%29)
* Sub-[SELECTs](http://en.wikipedia.org/wiki/Select_%28SQL%29) (i.e. nested SELECTs)
* Built-in [Replication](http://en.wikipedia.org/wiki/Database_replication) support (i.e. Master-Master Replication & Master-Slave Replication) with one master per slave, many slaves per master. [Multi-master replication](http://en.wikipedia.org/wiki/Multi-master_replication) is provided in [MySQL Cluster](http://en.wikipedia.org/wiki/MySQL_Cluster), and multi-master support can be added to unclustered configurations using Galera Cluster.
* Full-text [indexing](http://en.wikipedia.org/wiki/Index_%28database%29) and searching
* Embedded database library
* [Unicode](http://en.wikipedia.org/wiki/Unicode) support
* Partitioned tables with pruning of partitions in optimizer
* [Shared-nothing](http://en.wikipedia.org/wiki/Shared-nothing) clustering through [MySQL Cluster](http://en.wikipedia.org/wiki/MySQL_Cluster)
* Multiple storage engines, allowing one to choose the one that is most effective for each table in the application.
* Native storage engines InnoDB, MyISAM, Merge, Memory (heap), [Federated](http://en.wikipedia.org/wiki/MySQL_Federated), Archive, [CSV](http://en.wikipedia.org/wiki/Comma-separated_values), Black hole, NDB Cluster.
* Commit grouping, gathering multiple transactions from multiple connections together to increase the number of commits per second.

The developers release minor updates of the MySQL Server approximately every two months. The sources can be obtained from MySQL's website or from MySQL's [Bazaar](http://en.wikipedia.org/wiki/Bazaar_%28software%29) repository, both under the GPL license.

## History

MySQL was created by a Swedish company, MySQL AB, founded by [David Ax mark](http://en.wikipedia.org/wiki/David_Axmark), [Allan Larsson](http://en.wikipedia.org/wiki/Allan_Larsson) and [Michael "Monty" Widenius](http://en.wikipedia.org/wiki/Michael_%28Monty%29_Widenius). The first version of MySQL appeared on 23 May 1995. It was initially created for personal usage from [mySQL](http://en.wikipedia.org/wiki/MSQL) based on the low-level language [ISAM](http://en.wikipedia.org/wiki/ISAM), which the creators considered too slow and inflexible. They created a new [SQL](http://en.wikipedia.org/wiki/Structured_Query_Language) interface, while keeping the same [API](http://en.wikipedia.org/wiki/Application_programming_interface) as mySQL. By keeping the API consistent with the mySQL system, many developers were able to use MySQL instead of the (proprietarily licensed) mSQL antecedent.

### Legal and acquisition impacts

On 15 June 2001, [NuSphere](http://en.wikipedia.org/wiki/NuSphere) sued MySQL AB, TcX DataKonsult AB and its original authors Michael ("Monty") Widenius and David Ax mark in U.S District Court in Boston for "breach of contract, tortious interference with third party contracts and relationships and unfair competition".

In 2002, MySQL AB sued Progress NuSphere for [copyright](http://en.wikipedia.org/wiki/Copyright_infringement) and [trademark infringement](http://en.wikipedia.org/wiki/Trademark_infringement) in [United States district court](http://en.wikipedia.org/wiki/U.S._District_Court_for_the_District_of_Massachusetts). NuSphere had allegedly violated MySQL's copyright by linking MySQL's GPL'ed code with NuSphere Gemini table without being in compliance with the license. After a preliminary hearing before Judge Patti Saris on 27 February 2002, the parties entered settlement talks and eventually settled. After the hearing, [FSF](http://en.wikipedia.org/wiki/Free_Software_Foundation) commented that "Judge Saris made clear that she sees the GNU GPL to be an enforceable and binding license."

In October 2005, Oracle Corporation acquired [Innobase](http://en.wikipedia.org/wiki/Innobase) OY, the [Finnish](http://en.wikipedia.org/wiki/Finland) company that developed the third-party InnoDB storage engine that allows MySQL to provide such functionality as transactions and [foreign keys](http://en.wikipedia.org/wiki/Foreign_key). After the acquisition, an Oracle [press release](http://en.wikipedia.org/wiki/News_release) mentioned that the contracts that make the company's software available to [MySQL AB](http://en.wikipedia.org/wiki/MySQL_AB) would be due for renewal (and presumably renegotiation) sometime in 2006. During the MySQL Users Conference in April 2006, MySQL issued a press release that confirmed that MySQL and InfoBase OY agreed to a "multi-year" extension of their licensing agreement.

In February 2006, Oracle Corporation acquired [Sleepy cat Software](http://en.wikipedia.org/wiki/Sleepycat_Software), makers of the [Berkeley DB](http://en.wikipedia.org/wiki/Berkeley_DB), a database engine providing the basis for another MySQL storage engine. This had little effect, as Berkeley DB was not widely used, and was dropped (due to lack of use) in MySQL 5.1.12, a pre-GA release of MySQL 5.1 released in October 2006.

In April 2009, Oracle Corporation entered into an agreement to purchase Sun Microsystems, then owners of MySQL copyright and trademark. Sun's board of directors unanimously approved the deal; it was also approved by Sun's shareholders, and by the U.S. government on 20 August 2009. On 14 December 2009, Oracle pledged to continue to enhance MySQL as it had done for the previous four years.

A movement against Oracle's acquisition of MySQL, to "Save MySQL “from Oracle was started by one of the MySQL founders, [Monty Widenius](http://en.wikipedia.org/wiki/Monty_Widenius). The petition of 50,000+ developers and users called upon the European Commission to block approval of the acquisition. At the same time, several Free Software [opinion leaders](http://en.wikipedia.org/wiki/Opinion_leadership) (including [Eben Moglen](http://en.wikipedia.org/wiki/Eben_Moglen), [Pamela Jones](http://en.wikipedia.org/wiki/Pamela_Jones) of [Groklaw](http://en.wikipedia.org/wiki/Groklaw), [Jan Wildeboer](http://en.wikipedia.org/w/index.php?title=Jan_Wildeboer&action=edit&redlink=1) and [Carlo Piana](http://en.wikipedia.org/wiki/Carlo_Piana), who also acted as co-counsel in the merger regulation procedure) advocated for the unconditional approval of the merger. As part of the negotiations with the European Commission, Oracle committed that MySQL server will continue until at least 2015 to use the dual-licensing strategy long used by MySQL AB, with proprietary and GPL versions available. The antitrust of the EU had been "pressuring it to divest MySQL as a condition for approval of the merger". But, as revealed by [WikiLeaks](http://en.wikipedia.org/wiki/WikiLeaks), the US Department of Justice and Antitrust, at the request of Oracle, pressured the EU to unconditionally approve the merger. The European Commission eventually unconditionally approved Oracle's acquisition of MySQL on 21 January 2010.

In January 2009, prior to Oracle's acquisition of MySQL, Monty Widenius started a GPL-only fork, [MariaDB](http://en.wikipedia.org/wiki/MariaDB). MariaDB is based on the same code base as MySQL server 5.1 and strives to maintain compatibility with Oracle-provided versions.

INPUT DESIGN

Input design is the process of converting user-originated inputs to a computer-based format. Input design is one of the most expensive phases of the operation of computerized system and is often the major problem of a system. Input design is a part of overall design, which requires careful attribute. Inaccurate input data are the most common cause of errors in data processing. The goal of designing input data is to make data entry as easy, logical and free from errors. In the system design phase input data are collected and organized in to groups of similar data.

Input Forms Are

student Registration

company

**OUTPUT DESIGN**

Output design generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application. Computer output is the most important and direct source of information to the user. Output design is very important phase because the output will be in an interactive manner. The report of data transaction is seen as a output in this project

Output Forms are,

Email alert

Placed student

DATABASE DESIGN

The database design is a must for any application developed especially more for the data store projects. Since the android application system method involves retrieving the information in the table and produced to the users, proper handling of the table is a must. In the project, login table is designed to be unique in accepting the username and the length of the username and password should be greater than zero. The different users view the data in different format according to the privileges given.

**TESTING**

***Test Case Design***

One of the most important phases in the development of any software is testing. Testing is a dynamic technique of verification and validation, which ensures that the software meets the expectations of the user. It involves executing an implementation of the software with test data. Test data is the set of data that the system will process as a normal input. A test case is a set of sequential steps to execute a test, operating on a set of predefined inputs to produce certain expected outputs.

Testing demonstrates the software functions work according to specifications. In addition data collected from testing provides a good indication of software reliability and quality. Testing results in the deduction of number of errors. In this project critical modules are tested. The fields are tested for various validation methods using the different testing techniques.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.
* Features to be tested
* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

**Introduction**

The most important phase in system development life cycle is system testing. The number and nature of errors in a newly designed system depends on the system specifications and the time frame given for the design.

A newly designed system should have all the subsystems working together, but in reality each subsystems work independently. During this phase, all the subsystems are gathered into one pool and tested to determine whether it meets the user requirements.

Testing is done at two level -Testing of individual modules and testing the entire system. During the system testing, the system is used experimentally to ensure that the software will run according to the specifications and in the way the user expects. Each test case is designed with the intent of finding errors in the way the system will process it.

Testing plays a very critical role in determining the reliability and efficiency of software and hence is a very important stage in software development. Software testing is done at different levels. They are the unit testing and system testing which comprises of integration testing and acceptance testing.

**TYPES OF TESTING**

**Unit Testing**

This is the first level of testing. The different modules are tested against the specifications produced during the integration. This is done to test the internal logic of each module. Those resulting from the interaction between modules are initially avoided. The input received and output generated is also tested to see whether it falls in the expected range of values. Unit testing is performed from the bottom up, starting with the smallest and lowest modules and proceeding one at a time.

The units in a system are the modules and routines that are assembled and integrated to perform a specific function. The programs are tested for correctness of logic applied and detection of errors in coding. Each of the modules was tested and errors are rectified. They were then found to function properly.

**Integration Testing**

In integration testing, the tested modules are combined into sub-systems, which are then tested. The goal of integration testing to check whether the modules can be integrated properly emphasizing on the interfaces between modules. The different modules were linked together and integration testing done on them.

**Validation Testing**

The objective of the validation test is to tell the user about the validity and reliability of the system. It verifies whether the system operates as specified and the integrity of important data is maintained. User motivation is very important for the successful performance of the system.

All the modules were tested individually using both test data and live data. After each module was ascertained that it was working correctly and it had been "integrated" with the system. Again the system was tested as a whole. We hold the system tested with different types of users. The System Design, Data Flow Diagrams, procedures etc. were well documented so that the system can be easily maintained and upgraded by any computer professional at a later

**System Testing**

The integration of each module in the system is checked during this level of testing. The objective of system testing is to check if the software meets its requirements. System testing is done to uncover errors that were not found in earlier tests. This includes forced system failures and validation of total system as the user in the operational environment implements it. Under this testing, low volumes of transactions are generally based on live data. This volume is increased until the maximum level for each transactions type is reached. The total system is also tested for recovery after various major failures to ensure that no data are lost during the breakdown.

**User Acceptance Testing**

The objective of the acceptance test is to tell the user about the validity and reliability of the system. It verifies whether the system operates as specified and the integrity of important data is maintained. User motivation is very important for the successful performance of the system.

All the modules were tested individually using both test data and live data. After each module was ascertained that it was working correctly and it had been "integrated" with the system. Again the system was tested as a whole. We hold the system tested with different types of users. The System Design, Data Flow Diagrams, procedures etc. were well documented so that the system can be easily maintained and upgraded by any computer professional at a later

**SYSTEM Implementation**

Implementation is the process that actually yields the lowest-level system elements in the system hierarchy (system breakdown structure). The system elements are made, bought, or reused. Production involves the hardware fabrication processes of forming, removing, joining, and finishing; or the software realization processes of coding and testing; or the operational procedures development processes for operators' roles. If implementation involves a production process, a manufacturing system which uses the established technical and management processes may be required.

The purpose of the implementation process is to design and create (or fabricate) a system element conforming to that element’s design properties and/or requirements. The element is constructed employing appropriate technologies and industry practices. This process bridges the system definition processes and the integration process.

System Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively. The existing system was long time process.

The proposed system was developed using PHP. The existing system caused long time transmission process but the system developed now has a very good user-friendly tool, which has a menu-based interface, graphical interface for the end user. After coding and testing, the project is to be installed on the necessary system. The executable file is to be created and loaded in the system. Again the code is tested in the installed system. Installing the developed code in system in the form of executable file is implementation.

**CONCLUSION**

It is concluded that the application works well and satisfy the end users. The application is tested very well and errors are properly debugged. The application is simultaneously accessed from more than one system. Simultaneous login from more than one place is tested.

This system is user friendly so everyone can use easily. Proper documentation is provided. The end user can easily understand how the whole system is implemented by going through the documentation. The system is tested, implemented and the performance is found to be satisfactory. All necessary output is generated. Thus, the project is completed successfully.

Further enhancements can be made to the application, so that the application functions very attractive and useful manner than the present one. The speed of the transactions become more enough now.

**SCOPE FOR FUTUREWORK:-**

There is scope for future development of this project. The world of computer fields is not static; it is always subject to be dynamic. The technology which is famous today becomes outdated the very next day. To keep abstract of technical improvements, the system may be further refined. So, it is not concluded. Yet it will improve with further enhancements.

Enhancements can be done in an efficient manner. In our future work we will implement our system with three search engines ( for eg Google, bing, yahoo).So that the user can obtain very best results .

**Bibliography**

Textual Reference

* **PHP book** by Vasvani (TMH publications).
* **Beginning PHP5** by WROX.
* **Informatics practices** by Sumita Arora.
* **Head First PHP & MySQL** by Lynn Beighley and Michael Morrison(O’Reilly)
* [**PHP Objects, Patterns, and Practice**](https://www.goodreads.com/book/show/1569843.PHP_Objects_Patterns_and_Practice) by [Matt Zandstra](https://www.goodreads.com/author/show/79192.Matt_Zandstra)
* [**PHP and MySQL Web Development**](https://www.goodreads.com/book/show/40540.PHP_and_MySQL_Web_Development) by [Luke Welling](https://www.goodreads.com/author/show/22734.Luke_Welling)
* **PHP Object-Oriented Solutions** by [David Powers](https://www.goodreads.com/author/show/17995.David_Powers)
* **Modern PHP: New Features and Good Practice** by [Josh Lockhart](https://www.goodreads.com/author/show/7819917.Josh_Lockhart)
* **Learning PHP Design Patterns** by [William Sanders](https://www.goodreads.com/author/show/8086513.William_Sanders)
* [**PHP 5 Objects, Patterns, and Practice**](https://www.goodreads.com/book/show/136867.PHP_5_Objects_Patterns_and_Practice) by [Matt Zandstra](https://www.goodreads.com/author/show/79192.Matt_Zandstra)
* [**PHP for the Web: Visual QuickStart Guide**](https://www.goodreads.com/book/show/9683472-php-for-the-web) by [Larry Ullman](https://www.goodreads.com/author/show/22735.Larry_Ullman)

Online Reference:

* [www.wikipedia.com](http://www.wikipedia.com)
* [www.w3schools.com](http://www.w3schools.com)
* <http://www.phpreferencebook.com/>
* <http://www.tizag.com/phpT/>
* <http://www.phpbuddy.com/>
* <http://www.php.net/>

Code

<?php

include\_once("config/config.php");

include('Adminheader.php');

include\_once('phpmailer/class.smtp.php');

include\_once('phpmailer/class.pop3.php');

include\_once('email.class.inc.php');

error\_reporting(0);

$configVars = array();

$configVars['my\_email'] = 'packtemp1@gmail.com';

$configVars['user\_name'] = 'packtemp1@gmail.com';

$configVars['password'] = 'owjmhdvikpxlunts';

function sendMail($param){

$message = '

<html>

<body bgcolor="#DCEEFC">

<center>

'.$param['message'].'

Thanks

</center>

</body>

</html>';

sendOrderMail($message);

}

mysql\_connect("localhost","root","") or die(mysql\_error());

mysql\_select\_db("placement") or die(mysql\_error());

if(isset($\_POST['register']))

{

if ($\_POST['p11']=="")

{

$query = "select \* from student where tenth\_CGPA>=".$\_POST['p8']." and twelth\_CGPA>=".$\_POST['p9']." and UG\_CGPA>=".$\_POST['p10']." and Specialization1 ='".$\_POST['p12']."' and Year\_of\_passing3='".$\_POST['p13']."'";

}

else

{

$query = "select \* from student where tenth\_CGPA>=".$\_POST['p8']." and twelth\_CGPA>=".$\_POST['p9']." and UG\_CGPA>=".$\_POST['p10']." and PG\_CGPA>=".$\_POST['p11']." and Specialization2='".$\_POST['p12']."' and Year\_of\_passing='".$\_POST['p13']."'";

}

//echo $query;exit;

$result = mysql\_query($query) or die(mysql\_error());

$result = mysql\_query($query) or die(mysql\_error());

}

if(isset($\_POST['send']))

{

if ($\_POST['p11']=="")

{

$query = "select \* from student where tenth\_CGPA>=".$\_POST['p8']." and twelth\_CGPA>=".$\_POST['p9']." and UG\_CGPA>=".$\_POST['p10']." and Specialization1 ='".$\_POST['p12']."' and Year\_of\_passing3='".$\_POST['p13']."'";

}

else

{

$query = "select \* from student where tenth\_CGPA>=".$\_POST['p8']." and twelth\_CGPA>=".$\_POST['p9']." and UG\_CGPA>=".$\_POST['p10']." and PG\_CGPA>=".$\_POST['p11']." and Specialization2='".$\_POST['p12']."' and Year\_of\_passing='".$\_POST['p13']."'";

}

echo $query;

//exit;

$result = mysql\_query($query) or die(mysql\_error());

$result = mysql\_query($query) or die(mysql\_error());

while($row=mysql\_fetch\_array($result))

{

$html="";

// $html .= 'Name:'.$row['first\_name'].'<br />';

// $html .= 'Phone NO:'.$row['phone'].'<br />';

// $html .= 'Address:'.$row['address'].'<br />';

//$html .= 'Country:'.$row['country'].'<br />';

$html .= 'Congratulations '.$row['First\_Name'].' Your Interview Details<br />';

$html .= 'Company Name : '.$\_POST['p2'].'<br />';

$html .= 'Address : '.$\_POST['p3'].'<br />';

$html .= 'Website : '.$\_POST['p4'].'<br />';

$html .= 'Contact Number : '.$\_POST['p5'].'<br />';

$html .= 'Interview Date : '.$\_POST['p6'].'<br />';

$email = new Email();

$email->set\_from($configVars['my\_email']);

$email->set\_from\_name('Acknowledgement Mail');

$email->set\_subject('Interview Notification');

$email->set\_message(html\_entity\_decode($html.'<br /><br />'));

$email->add\_to($row['Email\_ID']);

$sent\_flag = $email->send();

}

}

?>

<div align="center">

<script type="text/javascript">

$(document).ready(function(){

$("#register\_form").validate();

});

</script>

<form action="adminquery.php" name="register\_form" id="register\_form" method="post">

<table border="0" cellspacing="4" cellspadding="4" bgcolor="White" class="displaycontent" width="500">

<caption></caption>

<tr>

<th colspan="2"> Short List Details</th>

<br>

</tr>

<tr>

<td class="col">Shortlist Id:</td>

<td><input type="text" name="p1" value="" /></td>

</tr>

<tr>

<td class="col">Company Name:</td>

<td><input type="text" name="p2" value="<?php echo $\_POST['p2']; ?>" class="required" /></td>

</tr>

<tr>

<td class="col">Address:</td>

<td><input type="text" name="p3" value="<?php echo $\_POST['p3']; ?>" class="required" /></td>

</tr>

<tr>

<td class="col">Website:</td>

<td><input type="text" name="p4" value="<?php echo $\_POST['p4']; ?>" class="required" /></td>

</tr>

<tr>

<td class="col">Contact Number:</td>

<td><input type="text" name="p5" value="<?php echo $\_POST['p5']; ?>" class="required" /></td>

</tr>

<tr>

<td class="col">Interview Date:</td>

<td><input type="text" name="p6" value="<?php echo $\_POST['p6']; ?>" class="required" /></td>

</tr>

<tr>

<td class="col" >Venue:</td>

<td><input type="text" class="required" name="p7" value="<?php echo $\_POST['p7']; ?>" /></td>

</tr>

<tr>

<td class="col">10th %:</td>

<td><input type="text" name="p8" value="<?php echo $\_POST['p8']; ?>" class="required" /></td>

</tr>

<tr>

<td class="col">12th %:</td>

<td><input type="text" name="p9" value="<?php echo $\_POST['p9']; ?>" class="required" /></td>

</tr>

<tr>

<td class="col" >UG %:</td>

<td><input type="text" class="required" name="p10" value="<?php echo $\_POST['p10']; ?>" /></td>

</tr>

<tr>

<td class="col" >PG %:</td>

<td><input type="text" class="required" name="p11" value="<?php echo $\_POST['p11']; ?>" /></td>

</tr>

<tr>

<td class="col" >Specialization:</td>

<td><input type="text" class="required" name="p12" value="<?php echo $\_POST['p12']; ?>" /></td>

</tr>

<tr>

<td class="col" >Year Of Passing:</td>

<td><input type="text" class="required" name="p13" value="<?php echo $\_POST['p13']; ?>" /></td>

</tr>

<tr>

<td>&nbsp;</td>

<td >

<input type="submit" name="register" value="Apply Filter" />

<input type="submit" name="send" value="Submit" />

</td>

</tr>

</table>

<table border="2" cellspacing="6" class="displaycontent" width="800" style="border:10px solid Darkblue;" align="center">

<tr>

<table border="2" cellspacing="6" class="displaycontent" width="800" style="border:10px solid Darkblue;" align="center">

<tr>

<th bgcolor=Orange><font color=white size=2>Roll No</font></th>

<th bgcolor=Orange><font color=white size=2>First Name </font></th>

<th bgcolor=Orange><font color=white size=2>Last Name </font></th>

<th bgcolor=Orange><font color=white size=2>Mobile Number</font></th>

<th bgcolor=Orange><font color=white size=2>Email ID </font></td>

<th bgcolor=Orange><font color=white size=2>Date of Birth </font></th>

</tr>

<?php

while($row=mysql\_fetch\_array($result))

{

?>

<tr>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Roll\_no']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['First\_Name']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Last\_Name']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Mobile\_Number']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Email\_ID']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Date\_of\_birth']; ?></font></td>

</tr>

<?php

}

?>

</table>

<div align="center">

<center >

<?php include\_once("footer.php")?>

</div> <!-- End container -->

</div> <!-- End row -->

</div>

</div>

<?php

include\_once("config/config.php");

include\_once("Userheader.php");

error\_reporting(0);

?>

<form action="companyrview.php" name="ViewInterview" class="row" method="post">

<div align="center">

<center >

<h2> Company Details </h2>

<table border="2" cellspacing="6" class="displaycontent" width="800" style="border:10px solid Darkblue;" align="center">

<tr>

<th bgcolor=Orange><font color=white size=2>Company Id</font></th>

<th bgcolor=Orange><font color=white size=2>Company Name</font></th>

<th bgcolor=Orange><font color=white size=2>Contact Person</font></th>

<th bgcolor=Orange><font color=white size=2>Contact Number </font></th>

<th bgcolor=Orange><font color=white size=2>Website</font></th>

<th bgcolor=Orange><font color=white size=2>Number of Employees </font></th>

<th bgcolor=Orange><font color=white size=2>Address </font></th>

</tr>

<?php

$query = "select \* from companydetails";

$result = mysql\_query($query) or die(mysql\_error());

while($row = mysql\_fetch\_assoc($result))

{

?>

<tr>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['CompanyId']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['CompanyName']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['ContactPerson']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['ContactNumber']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Website']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Numberofemp']; ?></font></td>

<td bgcolor=white><font color=#000000 size=2><?php echo $row['Address']; ?></font></td>

</tr>

<?php } ?>

</table>

</div>

</form>

<?php

include\_once("footer.php")

?>

</div> <!-- End row -->

</div>

</div>