**A Project On**

**EVENT MANAGEMENT SYSTEM**

**partial fulfillment of**

**Bachelor of Business Administration**

**(Computer Application)**

**Submitted by**

**UMAR KHAN**

**Under the Guidance**

**OF**

**In the**

**Academic Year**

**2022-2023**

**INTRODUCTION**

Event management is a process of organizing a professional and focused event, for a particular target audience. It involves visualizing concepts, planning, budgeting, organizing and executing events such as wedding, musical concerts, corporate seminars, exhibitions, birthday celebrations, theme parties, etc. Event Management is a multi-million dollar industry, growing rapidly, with events hosted regularly. Surprisingly, there is no formalized research conducted to access the growth of this industry. The industry includes fields such as the MICE (Meetings, Incentives and Events), exhibitions, conferences and seminars as well as live music and sporting events. On the profession side, event management is a glamorous and exciting profession that demands a lot of hard work and dynamism. The logistics side of the industry is paid less than the sales/sponsorship side, though some may say that these are two different industries.

Event management is the application of [project management](https://en.wikipedia.org/wiki/Project_management) to the creation and development of large scale events. The process of planning and coordinating the event is usually referred to as event planning and which can include budgeting, scheduling, site selection, acquiring necessary [permits](https://en.wikipedia.org/wiki/License), coordinating transportation and parking, arranging for speakers or entertainers, arranging decor, event security, [catering](https://en.wikipedia.org/wiki/Catering), coordinating with third party vendors, and emergency plans. The events industry now includes events of all sizes from the [Olympics](https://en.wikipedia.org/wiki/Olympics) down to business breakfast meetings. Many industries, [charitable organizations](https://en.wikipedia.org/wiki/Charitable_organization), and interest groups hold events in order to market themselves, build business relationships, raise money, or celebrate achievement. An event refers to a social gathering or activity, such as a festival,( for example a musical festival), a ceremony( for example a marriage ) and a party(for example a birthday party).There are mainly 3 types of event management:

* Corporate Event Management
* Product Launch Event Management
* Special Event Management
  1. **EVENT MANAGER**

The Event Manager is the person who plans and executes the event. Event managers and their teams are often behind-the-scenes running the event. Event managers may also be involved in more than just the planning and execution of the event, but also brand building, marketing and communication strategy. The event manager is an expert at the creative, technical and logistical elements that help an event succeed. This includes event design, audiovisual production, scriptwriting, logistics, budgeting, negotiation and, of course, client service. It is a multidimensional profession.

* 1. **EVENT MANAGEMENT PROCESS**

There are 2 stages of event management process namely, Event planning and Event control.

* Event Planning: To plan an event we must consider the following areas of an event, viz, feasibility, promotion, site choice/design, staging, shutdown, site map, event proposal.
* Event Control: To control an event we must look on the following areas logistics, negotiations, costing & cash flow, event manual, I.T, decision making and change, risk management.
  1. **SCOPE OF THE PROJECT**

The objective of this application is to develop a system that effectively manages all the data related to the various events that take place in an organization. The purpose is to maintain a centralized database of all event related information. The goal is to support various functions and processes necessary to manage the data efficiently.

* 1. **EXISTING SYSTEM**

This existing system is not providing secure registration and profile management of all the users properly. This system is not providing on-line help. This system doesn’t provide tracking of users activities and their progress. This manual system gives us very less security for saving data and some data may be lost due to mismanagement. This system is not providing event management through internet. This system is not providing proper events information. The system is giving manual information through the event management executer.

* 1. **FEASIBILITY STUDY**

A feasibility study is a low-level capsule version of the entire System analysis and Design Process. The study begins by classifying the problem definition. Feasibility is to determine if it’s worth doing. Once an acceptance problem definition has been generated, the analyst develops a logical model of the system. A search for alternatives is analyzed carefully. There are 3 parts in feasibility study.

* + 1. **Operational Feasibility**

Operational feasibility is the measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.The operational feasibility assessment focuses on the degree to which the proposed development projects fits in with the existing business environment and objectives with regard to development schedule, delivery date, [corporate culture](https://en.wikipedia.org/wiki/Corporate_culture) and existing business processes.To ensure success, desired operational outcomes must be imparted during design and development. These include such design-dependent parameters as reliability, maintainability, supportability, usability, producibility, disposability, sustainability, affordability and others. These parameters are required to be considered at the early stages of design if desired operational behaviours are to be realised. A system design and development requires appropriate and timely application of engineering and management efforts to meet the previously mentioned parameters. A system may serve its intended purpose most effectively when its technical and operating characteristics are engineered into the design. Therefore, operational feasibility is a critical aspect of systems engineering that needs to be an integral part of the early design phases.

* + 1. **Technical Feasibility**

This involves questions such as whether the technology needed for the system exists, how difficult it will be to build, and whether the firm has enough experience using that technology. The assessment is based on outline design of system requirements in terms of input, processes, output, fields, programs and procedures. This can be qualified in terms of volume of data, trends, frequency of updating inorder to give an introduction to the technical system. The application is the fact that it has been developed on windows XP platform and a low configuration of 1GB RAM on Intel Pentium Dual core processor. This is technically feasible .The technical feasibility assessment is focused on gaining an understanding of the present technical resources of the organization and their applicability to the expected needs of the proposed system. It is an evaluation of the hardware and software and how it meets the need of the proposed system.

* + 1. **Economical Feasibility**

Establishing the cost-effectiveness of the proposed system i.e. if the benefits do not outweigh the costs then it is not worth going ahead. In the fast paced world today there is a great need of online social networking facilities. Thus the benefits of this project in the current scenario make it economically feasible. The purpose of the economic feasibility assessment is to determine the positive economic benefits to the organization that the proposed system will provide. It includes quantification and identification of all the benefits expected. This assessment typically involves a cost/benefits analysis.

**REQUIREMENT ANALYSIS**

**2.1 HARDWARE REQUIREMENTS**

|  |  |
| --- | --- |
| * Hardware | **:** Processor Intel dual core and above |
| * Clock speed | **:** 3.0 GHz |
| * RAM size | **:** 512 MB |
| * Hard Disk capacity | **:** 400 GB |
| * Monitor type | **:** 15 inch color monitor |

**2.2 SOFTWARE REQUIREMENTS**

|  |  |
| --- | --- |
| * Operating System | **:**  Windows 7, Windows 8,Windows 10 |
| * Internet connection | **:** Existing telephone lines, Data card. |
| * Browser | **:** Google chrome latest version |
| * Database | **:** MySQL. |
| * Performance | **:** The turn-around time of the project will be medium. |
| * Documentation | **:** MS-Office |

**PROPOSED SYSTEM**

Event Management System is an Online event management software project that serves the functionality of an event manager. The system allows only registered users to login and new users are allowed to register on the application. This is a web application but desktop application of the same application is also available. The project provides most of the basic functionality required for an event. It allows the user to select from a list of event types. Once the user enters an event type eg(Marriage, Stage Show etc), the system then allows the user to select the date and time of event, place and the event equipment’s. All this data is logged in the database and the user is setting up his username and password while registering . The data is then sent to the administrator (website owner) and they may interact with the client as per his requirements and his contact data stored in the database.

**3.1 ADVANTAGES**

* The system is useful as it calculates an exact cost for all the resources required during the event.
* The user gets all the resources at a single place instead of wandering around for these.
* This system is effective and saves time and cost of the users.

**DESIGN AND ARCHITECTURES**

**4.1 DESIGN**

Design is the first step in development phase for any techniques and principles for the purpose of defining a device , a process or system in sufficient detail to permit its physical realization. Once the software requirement have been analyzed and specified the software design involves three technical activities-Design, Coding, Implementation, Testing that are required to build and verify the software.The design activities are of main importance in this phase, because in this activities decisions ultimately affecting the success of the software implementation and its ease of maintenance are made. These decision has the final bearing upon reliability and maintainability of the a system. Design is only way to accurately transfer the customers requirements into finished software or system .Design is the place where quality is fostered in development. Software design is the process through which requirements are translated into a representation of software. Software requirement is conducted in two steps. Preliminary design is concerned with the transformation of requirements into data.

**4.2 MODULE DESCRIPTION**

The system after careful analysis has been identified to be presented with the following modules.

**4.2.1 User Module**

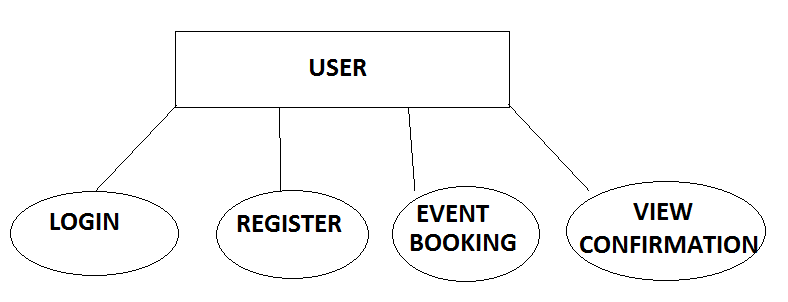
****

Figure 4.1 User Module

**4.2.2 Administrator Module**

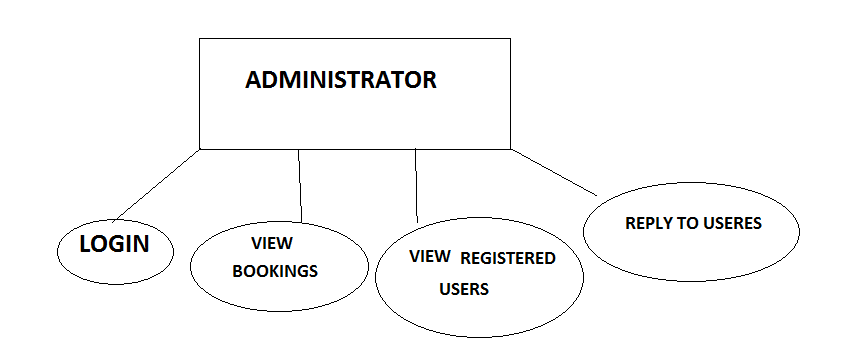


Figure 4.2Adminiistrator Module

**4.3ARCHITECTURE**

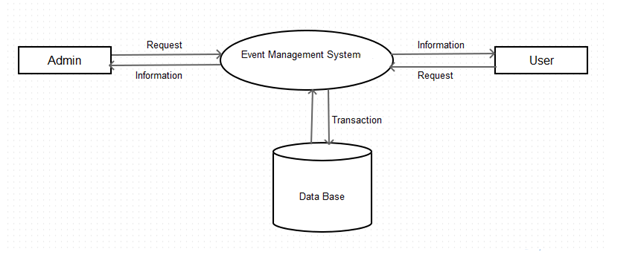
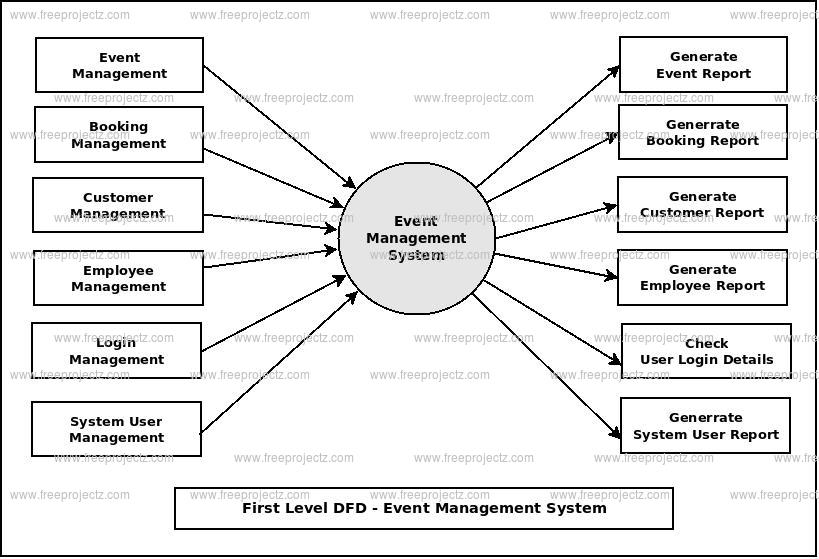
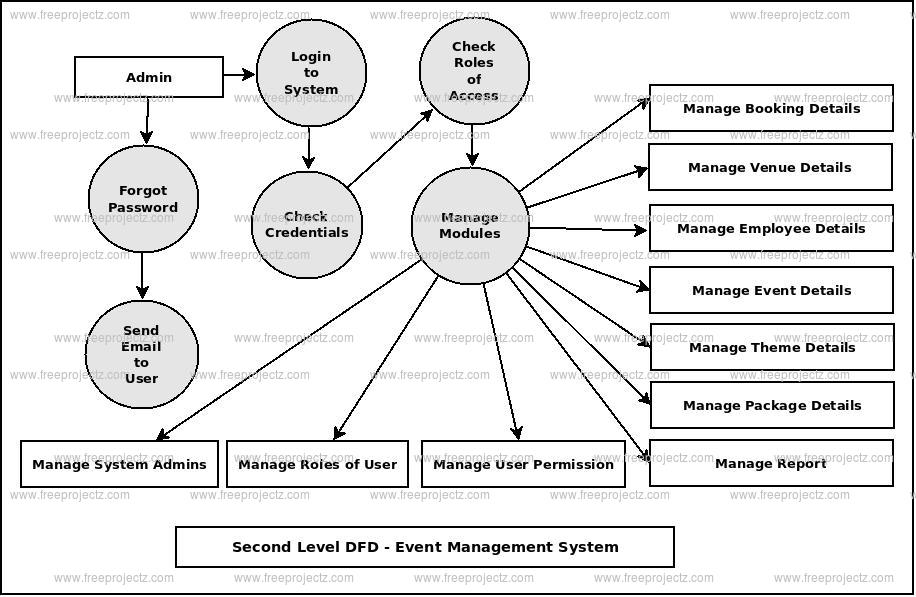
****

Figure 4.3 Architecture of Event Management System

**First Level DFD**



**SECOUND LEVEL DFD**



**DEVELOPMENT AND CODING**

**5.1 TECHNOLOGY DESCRIPTION**

**5.1.1.Php**

PHP is a [server-side scripting](https://en.wikipedia.org/wiki/Server-side_scripting) language designed primarily for [web development](https://en.wikipedia.org/wiki/Web_development) but also used as a [general-purpose programming language](https://en.wikipedia.org/wiki/General-purpose_programming_language). Originally created by [Rasmus Lerdorf](https://en.wikipedia.org/wiki/Rasmus_Lerdorf)  in 1994,the PHP [reference implementation](https://en.wikipedia.org/wiki/Reference_implementation) is now produced by The PHP Development Team. PHP originally stood for personal home page but it now stands for the [recursive acronym](https://en.wikipedia.org/wiki/Recursive_acronym) PHP: Hypertext Preprocessor.

PHP code may be embedded into [HTML](https://en.wikipedia.org/wiki/HTML) or HTML5 code, or it can be used in combination with various [web template systems](https://en.wikipedia.org/wiki/Web_template_system), [web content management systems](https://en.wikipedia.org/wiki/Web_content_management_system) and [web frameworks](https://en.wikipedia.org/wiki/Web_framework). PHP code is usually processed by a PHP [interpreter](https://en.wikipedia.org/wiki/Interpreter_(computing)) implemented as a [module](https://en.wikipedia.org/wiki/Plugin_(computing)) in the web server or as a [Common Gateway Interface](https://en.wikipedia.org/wiki/Common_Gateway_Interface) (CGI) executable. The web server combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a [command-line interface](https://en.wikipedia.org/wiki/Command-line_interface) (CLI) and can be used to implement [standalone](https://en.wikipedia.org/wiki/Computer_software) [graphical applications](https://en.wikipedia.org/wiki/Graphical_user_interface).

* **Php 3 and 4**

[Zeev Suraski](https://en.wikipedia.org/wiki/Zeev_Suraski) and [Andi Gutmans](https://en.wikipedia.org/wiki/Andi_Gutmans) rewrote the [parser](https://en.wikipedia.org/wiki/Parser) in 1997 and formed the base of PHP 3, changing the language's name to the  [recursive acronym](https://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](https://en.wikipedia.org/wiki/Rewrite_(programming)) of PHP's core, producing the [Zend Engine](https://en.wikipedia.org/wiki/Zend_Engine) in 1999. They also founded [Zend Technologies](https://en.wikipedia.org/wiki/Zend_Technologies) in [Ramat Gan](https://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.

* **Php 5**

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](https://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](https://en.wikipedia.org/wiki/Late_static_binding) had been missing from PHP and was added in version 5.3.

Many low-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](https://en.wikipedia.org/wiki/32-bit) and [64-bit](https://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](https://en.wikipedia.org/wiki/Microsoft_Windows) binary distributions were 32-bit [x86](https://en.wikipedia.org/wiki/X86) builds, requiring Windows 32-bit compatibility mode while using [Internet Information Services](https://en.wikipedia.org/wiki/Internet_Information_Services) (IIS) on a 64-bit Windows platform. PHP version 5.5 made the 64-bit [x86-64](https://en.wikipedia.org/wiki/X86-64) builds available for Microsoft Windows.

* **Php 6 and Unicode**

PHP has received criticism due to lacking native [Unicode](https://en.wikipedia.org/wiki/Unicode) support at the core language level, instead only supporting byte strings. In 2005, a project headed by Andrei Zmievski was initiated to bring native Unicode support throughout PHP, by embedding the [International Components for Unicode](https://en.wikipedia.org/wiki/International_Components_for_Unicode) (ICU) library, and representing text strings as [UTF-16](https://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 have been adopted.

* **Php 7**

During 2014 and 2015, a new major PHP version was developed, which was numbered PHP 7. The numbering of this version involved some debate. While the PHP 6 Unicode experiment had never been released, several articles and book titles referenced the PHP 6 name, which might have caused confusion if a new release were to reuse the name. After a vote, the name PHP 7 was chosen.The foundation of PHP 7 is a PHP [branch](https://en.wikipedia.org/wiki/Branching_(revision_control)) that was originally dubbed PHP next generation (phpng). It was authored by Dmitry Stogov, Xinchen Hui and Nikita Popov, and aimed to optimize PHP performance by refactoring the Zend Engine to use more compact [data structures](https://en.wikipedia.org/wiki/Data_structures) with improved [cache locality](https://en.wikipedia.org/wiki/Locality_of_reference) while retaining near-complete language compatibility. As of 14 July 2014, [WordPress](https://en.wikipedia.org/wiki/WordPress)-based benchmarks, which served as the main benchmark suite for the phpng project, showed 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](https://en.wikipedia.org/wiki/Just-in-time_compilation) (JIT) compiler. Because of the significant changes, the reworked Zend Engine is called Zend Engine 3, succeeding Zend Engine 2 used in PHP 5. Because of major internal changes in phpng, it must receive a new [major version](https://en.wikipedia.org/wiki/Software_versioning) number of PHP, rather than a minor PHP 5 release, according to PHP's release process. Major versions of PHP are allowed to break backward-compatibility of code and therefore PHP 7 presented an opportunity for other improvements beyond phpng that require backward-compatibility breaks, including wider use of [exceptions](https://en.wikipedia.org/wiki/Exception_(computer_science)), reworking variable syntax to be more consistent and complete, and the deprecation or removal of various legacy features. PHP 7 also introduced new language features, including return type declarations for functions, which complement the existing parameter type declarations, and support for the scalar types (integer, float, string, and boolean) in parameter and return type declarations.

* **Data Types**

PHP stores integers in a platform-dependent range, either a 64-bit or 32-bit [signed](https://en.wikipedia.org/wiki/Signed_number_representations) [integer](https://en.wikipedia.org/wiki/Integer_(computer_science)) equivalent to the [C-language long type](https://en.wikipedia.org/wiki/C_variable_types_and_declarations). Unsigned integers are converted to signed values in certain situations; this behavior is different from that of other programming languages. Integer variables can be assigned using decimal (positive and negative), [octal](https://en.wikipedia.org/wiki/Octal), [hexadecimal](https://en.wikipedia.org/wiki/Hexadecimal), and [binary](https://en.wikipedia.org/wiki/Binary_code) notations.[Floating point](https://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](https://en.wikipedia.org/wiki/Scientific_notation). PHP has a native [Boolean](https://en.wikipedia.org/wiki/Boolean_datatype) type that is similar to the native Boolean types in [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and [C++](https://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](https://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 other arrays. Order is preserved in lists of values and in [hashes](https://en.wikipedia.org/wiki/Hash_table) with both keys and values, and the two can be intermingled. PHP also supports [strings](https://en.wikipedia.org/wiki/String_(computing)), which can be used with single quotes, double quotes, nowdoc or [heredoc](https://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 defines a large array of functions in the core language and many are also available in 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](https://en.wikipedia.org/wiki/PHP#ORGANIC) above.

In lieu of [function pointers](https://en.wikipedia.org/wiki/Function_pointer), functions in PHP can be referenced by a string containing their name. In this manner, normal PHP functions can be used, for example, as [callbacks](https://en.wikipedia.org/wiki/Callback_function) or within [function tables](https://en.wikipedia.org/wiki/Function_table). User-defined functions may be created at any time without being [prototyped](https://en.wikipedia.org/wiki/Function_prototype). Functions may be defined inside code blocks, permitting a [run-time decision](https://en.wikipedia.org/wiki/Dynamic_dispatch) as to whether or not a function should be defined. There is a function\_exists function that determines whether a function with a given name has already been defined. Function calls must use parentheses, with the exception of zero-argument class [constructor](https://en.wikipedia.org/wiki/Constructor_(computer_science)) functions called with the PHP operator new, in which case parentheses are optional.

Until PHP 5.3, support for [anonymous functions](https://en.wikipedia.org/wiki/Anonymous_functions) and [closures](https://en.wikipedia.org/wiki/Closure_(computer_science)) 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. PHP 5.3 added syntax to define an anonymous function or "closure"which can capture variables from the surrounding scope. 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.

Unusually for a dynamically typed language, PHP supports type declarations on function parameters, which are enforced at runtime. This has been supported for classes and interfaces since PHP 5.0, for arrays since PHP 5.1, for "callables" since PHP 5.4, and scalar (integer, float, string and boolean) types since PHP 7.0. PHP 7.0 also has type declarations for function return types, expressed by placing the type name after the list of parameters, preceded by a colon. For example, the getAdder function from the earlier example could be annotated with types like so in PHP 7.

**5.1.2 Html**

Hypertext Markup Language (HTML) is the standard [markup language](https://en.wikipedia.org/wiki/Markup_language) for creating [web pages](https://en.wikipedia.org/wiki/Web_page) and [web applications](https://en.wikipedia.org/wiki/Web_application). With [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets) (CSS) and [JavaScript](https://en.wikipedia.org/wiki/JavaScript) it forms a triad of cornerstone technologies for the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web). [Web browsers](https://en.wikipedia.org/wiki/Web_browser) receive HTML documents from a [web server](https://en.wikipedia.org/wiki/Webserver) or from local storage and render them into multimedia web pages. HTML describes the structure of a web page [semantically](https://en.wikipedia.org/wiki/Semantic) and originally included cues for the appearance of the document.[HTML elements](https://en.wikipedia.org/wiki/HTML_element) are the building blocks of HTML pages. With HTML constructs, [images](https://en.wikipedia.org/wiki/Img_(HTML_element)) and other objects, such as [interactive forms,](https://en.wikipedia.org/wiki/Fieldset) may be embedded into the rendered page. It provides a means to create [structured documents](https://en.wikipedia.org/wiki/Structured_document) by denoting structural [semantics](https://en.wikipedia.org/wiki/Semantics) for text such as headings, paragraphs, lists, [links](https://en.wikipedia.org/wiki/Hyperlink), quotes and other items. HTML elements are delineated by tags, written using [angle brackets](https://en.wikipedia.org/wiki/Bracket#Angle_brackets). Tags such as <img /> and <input /> introduce content into the page directly. Others such as <p>...</p> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.HTML can embed programs written in a [scripting language](https://en.wikipedia.org/wiki/Scripting_language) such as [JavaScript](https://en.wikipedia.org/wiki/JavaScript) which affect the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The [World Wide Web Consortium](https://en.wikipedia.org/wiki/World_Wide_Web_Consortium) (W3C), maintainer of both the HTML and the CSS standards, has encouraged the use of CSS over explicit presentational HTML.

**5.1.3 MySQL**

MySQL  is an [open-source](https://en.wikipedia.org/wiki/Open-source) [relational database management system](https://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS). Its name is a combination of "My", the name of co-founder [Michael Widenius](https://en.wikipedia.org/wiki/Michael_Widenius)' daughter, and "[SQL](https://en.wikipedia.org/wiki/SQL)", the abbreviation for [Structured Query Language](https://en.wikipedia.org/wiki/Structured_Query_Language). The MySQL development project has made its [source code](https://en.wikipedia.org/wiki/Source_code) available under the terms of the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), as well as under a variety of [proprietary](https://en.wikipedia.org/wiki/Proprietary_software) agreements. MySQL was owned and sponsored by a single [for-profit](https://en.wikipedia.org/wiki/Business) firm, the [Swedish](https://en.wikipedia.org/wiki/Sweden) company [MySQL AB](https://en.wikipedia.org/wiki/MySQL_AB), now owned by [Oracle Corporation](https://en.wikipedia.org/wiki/Oracle_Corporation).

Figure 5.1.3 MySqigh

For proprietary use, several paid editions are available, and offer additional functionality.MySQL is a central component of the [LAMP](https://en.wikipedia.org/wiki/LAMP_(software_bundle)) open-source web application software stack (and other "[AMP](https://en.wikipedia.org/wiki/List_of_AMP_packages)" stacks). LAMP is an acronym for "[Linux](https://en.wikipedia.org/wiki/Linux), [Apache](https://en.wikipedia.org/wiki/Apache_HTTP_Server), MySQL, [Perl](https://en.wikipedia.org/wiki/Perl)/[PHP](https://en.wikipedia.org/wiki/PHP)/[Python](https://en.wikipedia.org/wiki/Python_(programming_language))". Applications that use the MySQL database include: [TYPO3](https://en.wikipedia.org/wiki/TYPO3), [MODx](https://en.wikipedia.org/wiki/MODx), [Joomla](https://en.wikipedia.org/wiki/Joomla), [WordPress](https://en.wikipedia.org/wiki/WordPress), [phpBB](https://en.wikipedia.org/wiki/PhpBB), [MyBB](https://en.wikipedia.org/wiki/MyBB), and [Drupal](https://en.wikipedia.org/wiki/Drupal). MySQL is also used in many low-profile, large-scale [websites](https://en.wikipedia.org/wiki/Website), including [Google](https://en.wikipedia.org/wiki/Google) (though not for searches), [Facebook](https://en.wikipedia.org/wiki/Facebook), [Twitter](https://en.wikipedia.org/wiki/Twitter), [Flickr](https://en.wikipedia.org/wiki/Flickr), and [YouTube](https://en.wikipedia.org/wiki/YouTube).

**TESTING AND IMPLEMENTATION**

**6.1 THE TESTING SPECTRUM**

The term implementation has different meanings ranging from the conversation of a basic application to a complete replacement of a computer system. The procedures however, are virtually the same. Implementation includes all those activities that take place to convert from old system to new. The new system may be totally new replacing an existing manual or automated system or it may be major modification to an existing system. The method of implementation and time scale to be adopted is found out initially. Proper implementation is essential to provide a reliable system to meet organization requirement.

**6.1.1 Unit Testing**

In [computer programming](https://en.wikipedia.org/wiki/Computer_programming), unit testing is a [software testing](https://en.wikipedia.org/wiki/Software_testing) method by which individual units of [source code](https://en.wikipedia.org/wiki/Source_code), sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application. In [procedural programming](https://en.wikipedia.org/wiki/Procedural_programming), a unit could be an entire module, but it is more commonly an individual function or procedure. In [object-oriented programming](https://en.wikipedia.org/wiki/Object-oriented_programming), a unit is often an entire interface, such as a class, but could be an individual method. Unit tests are short code fragmentscreated by programmers or occasionally by [white box testers](https://en.wikipedia.org/wiki/White-box_testing) during the development process. It forms the basis for component testing. Ideally, each [test case](https://en.wikipedia.org/wiki/Test_case) is independent from the others. Substitutes such as [method stubs](https://en.wikipedia.org/wiki/Method_stub), [mock objects](https://en.wikipedia.org/wiki/Mock_object), [fakes](https://en.wikipedia.org/wiki/Mock_object#Mocks.2C_fakes.2C_and_stubs), and [test harnesses](https://en.wikipedia.org/wiki/Test_harness) can be used to assist testing a module in isolation. Unit tests are typically written and run by [software developers](https://en.wikipedia.org/wiki/Software_developer) to ensure that code meets its design and behaves as intended.

**6.1.1.1 Benefits**

The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written [contract](https://en.wikipedia.org/wiki/Design_by_Contract) that the piece of code must satisfy. As a result, it affords several benefits.

* **Find problems early**

Unit testing finds problems early in the [development cycle](https://en.wikipedia.org/wiki/Development_cycle). In [test-driven development](https://en.wikipedia.org/wiki/Test-driven_development) (TDD), which is frequently used in both [extreme programming](https://en.wikipedia.org/wiki/Extreme_programming) and [scrum](https://en.wikipedia.org/wiki/Scrum_(software_development)), unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is considered to be a bug either in the changed code or the tests themselves. The unit tests then allow the location of the fault or failure to be easily traced. Since the unit tests alert the development team of the problem before handing the code off to testers or clients, it is still early in the development process.

* **Facilitates Change**

Unit testing allows the programmer to [refactor](https://en.wikipedia.org/wiki/Refactoring) code or upgrade system libraries at a later date, and make sure the module still works correctly (e.g., in [regression testing](https://en.wikipedia.org/wiki/Regression_testing)). The procedure is to write test cases for all [functions](https://en.wikipedia.org/wiki/Subroutine) and [methods](https://en.wikipedia.org/wiki/Method_(computer_science)) so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes which may break a [design contract](https://en.wikipedia.org/wiki/Design_by_contract).

* **Simplifies Integration**

Unit testing may reduce uncertainty in the units themselves and can be used in a [bottom-up](https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design) testing style approach. By testing the parts of a program first and then testing the sum of its parts, [integration testing](https://en.wikipedia.org/wiki/Integration_testing) becomes much easier.

* **Documentation**

Unit testing provides a sort of living documentation of the system. Developers looking to learn what functionality is provided by a unit, and how to use it, can look at the unit tests to gain a basic understanding of the unit's interface ([API](https://en.wikipedia.org/wiki/Application_programming_interface)).Unit [test cases](https://en.wikipedia.org/wiki/Test_case) embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviors that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

**6.1.2 Integration Testing**

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in [software testing](https://en.wikipedia.org/wiki/Software_testing) in which individual software modules are combined and tested as a group. It occurs after [unit testing](https://en.wikipedia.org/wiki/Unit_testing) and before [validation testing](https://en.wikipedia.org/wiki/Verification_and_validation_(software)). Integration testing takes as its input [modules](https://en.wikipedia.org/wiki/Module_(programming)) that have been unit tested, groups them in larger aggregates, applies tests defined in an integration [test plan](https://en.wikipedia.org/wiki/Test_plan) to those aggregates, and delivers as its output the integrated system ready for [system testing](https://en.wikipedia.org/wiki/System_testing).

* **Purpose**

The purpose of integration testing is to verify functional, performance, and reliability [requirements](https://en.wikipedia.org/wiki/Requirement) placed on major design items. These "design items", i.e., assemblages (or groups of units), are exercised through their interfaces using [black-box testing](https://en.wikipedia.org/wiki/Black-box_testing), success and error cases being simulated via appropriate parameter and data inputs. Simulated usage of shared data areas and [inter-process communication](https://en.wikipedia.org/wiki/Inter-process_communication) is tested and individual [subsystems](https://en.wikipedia.org/wiki/Subsystem) are exercised through their input interface. [Test cases](https://en.wikipedia.org/wiki/Test_case) are constructed to test whether all the components within assemblages interact correctly, for example across procedure calls or process activations, and this is done after testing individual modules, i.e., unit testing. The overall idea is a "building block" approach, in which verified assemblages are added to a verified base which is then used to support the integration testing of further assemblages.Software integration testing is performed according to the software development life cycle (SDLC) after module and functional tests. The cross-dependencies for software integration testing are: schedule for integration testing, strategy and selection of the tools used for integration, define the cyclomatical complexity of the software and software architecture, reusability of modules and life-cycle and versioning management.Some different types of integration testing are big-bang, [top-down, and bottom-up](https://en.wikipedia.org/wiki/Top-down_and_bottom-up_design), mixed (sandwich) and risky-hardest. Other Integration Patterns[[2]](https://en.wikipedia.org/wiki/Integration_testing#cite_note-2) are: collaboration integration, backbone integration, layer integration, client-server integration, distributed services integration and low-frequency integration.

.

* **Top-down And Bottom-up**

Bottom-up testing is an approach to integrated testing where the lowest level components are tested first, then used to facilitate the testing of lower level components. The process is repeated until the component at the top of the hierarchy is tested.All the bottom or low-level modules, procedures or functions are integrated and then tested. After the integration testing of lower level integrated modules, the next level of modules will be formed and can be used for integration testing. This approach is helpful only when all or most of the modules of the same development level are ready. This method also helps to determine the levels of software developed and makes it easier to report testing progress in the form of a percentage.Top-down testing is an approach to integrated testing where the top integrated modules are tested and the branch of the module is tested step by step until the end of the related module.Sandwich testing is an approach to combine top down testing with bottom up testing.

**6.1.3 Software Verification And Validation**

In [software project management](https://en.wikipedia.org/wiki/Software_project_management), [software testing](https://en.wikipedia.org/wiki/Software_testing), and [software engineering](https://en.wikipedia.org/wiki/Software_engineering), verification and validation (V&V) is the process of checking that a software system meets specifications and that it fulfills its intended purpose. It may also be referred to as [software quality control](https://en.wikipedia.org/wiki/Software_quality_control). It is normally the responsibility of [software testers](https://en.wikipedia.org/wiki/Software_testing) as part of the [software development lifecycle](https://en.wikipedia.org/wiki/Software_development_process). Validation checks that the product design satisfies or fits the intended use (low-level checking), i.e., the software meets the user requirements.This is done through [dynamic testing](https://en.wikipedia.org/wiki/Dynamic_testing) and other forms of review.Verification and validation are not the same thing, although they are often confused. [Boehm](https://en.wikipedia.org/wiki/Barry_Boehm) succinctly expressed the difference between

* Validation: Are we building the right product?
* Verification: Are we building the product right?

According to the [Capability Maturity Model](https://en.wikipedia.org/wiki/Capability_Maturity_Model) (CMMI-SW v1.1)

* Software Verification: The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.
* Software Validation: The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.

In other words, software verification is ensuring that the product has been built according to the requirements and design specifications, while software validation ensures that the product meets the user's needs, and that the specifications were correct in the first place. Software verification ensures that "you built it right". Software validation ensures that "you built the right thing". Software validation confirms that the product, as provided, will fulfill its intended use.

From testing perspective:

* Fault – wrong or missing function in the code.
* Failure – the manifestation of a fault during execution.
* Malfunction – according to its specification the system does not meet its specified functionality.

Both verification and validation are related to the concepts of [quality](https://en.wikipedia.org/wiki/Quality_(business)) and of [software quality assurance](https://en.wikipedia.org/wiki/Software_quality_assurance). By themselves, verification and validation do not guarantee software quality; planning, [traceability](https://en.wikipedia.org/wiki/Traceability), configuration management and other aspects of software engineering are required.Within the [modeling and simulation](https://en.wikipedia.org/wiki/Modeling_and_simulation) (M&S) community, the definitions of verification, validation and accreditation are similar:

* M&S Verification is the process of determining that a [computer model](https://en.wikipedia.org/wiki/Computer_model), simulation, or federation of models and simulations implementations and their associated data accurately represent the developer's conceptual description and specifications.
* M&S Validation is the process of determining the degree to which a model, simulation, or federation of models and simulations, and their associated data are accurate representations of the real world from the perspective of the intended use(s).
* [Accreditation](https://en.wikipedia.org/wiki/Accreditation) is the formal certification that a model or simulation is acceptable to be used for a specific purpose.

The definition of M&S validation focuses on the accuracy with which the M&S represents the real-world intended use(s). Determining the degree of M&S accuracy is required because all M&S are approximations of reality, and it is usually critical to determine if the degree of approximation is acceptable for the intended use(s). This stands in contrast to software validation.

* **Classification of Methods**

In [mission-critical](https://en.wikipedia.org/wiki/Mission-critical) software systems, where flawless performance is absolutely necessary, [formal methods](https://en.wikipedia.org/wiki/Formal_methods) may be used to ensure the correct operation of a system. However, often for non-mission-critical software systems, formal methods prove to be very costly and an alternative method of software V&V must be sought out. In such cases, [syntactic methods](https://en.wikipedia.org/wiki/Syntactic_methods) are often used.

* **Test Cases**

A test case is a tool used in the process. Test cases may be prepared for software verification and software validation to determine if the product was built according to the requirements of the user. Other methods, such as reviews, may be used early in the life cycle to provide for software validation.

**6.1.4 Black-Box Testing**

Black-box testing is a method of [software testing](https://en.wikipedia.org/wiki/Software_testing) that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: [unit](https://en.wikipedia.org/wiki/Unit_test), [integration](https://en.wikipedia.org/wiki/Integration_testing), [system](https://en.wikipedia.org/wiki/System_testing) and [acceptance](https://en.wikipedia.org/wiki/Acceptance_test). It typically comprises most if not all lower level testing, but can also dominate unit testing as well.

* **Test Procedures**

Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of what the software is supposed to do but is not aware of how it does it. For instance, the tester is aware that a particular input returns a certain, invariable output but is not aware of how the software produces the output in the first place.

* **Test Cases**

Test cases are built around specifications and requirements, i.e., what the application is supposed to do. Test cases are generally derived from external descriptions of the software, including specifications, requirements and design parameters. Although the tests used are primarily functional in nature, non-functional tests may also be used. The test designer selects both valid and invalid inputs and determines the correct output, often with the help of an [oracle](https://en.wikipedia.org/wiki/Oracle_(software_testing)) or a previous result that is known to be good, without any knowledge of the test object's internal structure.

* **Test Design Techniques**

Typical black-box test design techniques include:

* [Decision table](https://en.wikipedia.org/wiki/Decision_table) testing
* [All-pairs testing](https://en.wikipedia.org/wiki/All-pairs_testing)
* [Equivalence partitioning](https://en.wikipedia.org/wiki/Equivalence_partitioning)
* [Boundary value analysis](https://en.wikipedia.org/wiki/Boundary_value_analysis)
* [Cause–effect graph](https://en.wikipedia.org/wiki/Cause%E2%80%93effect_graph)
* [State transition](https://en.wikipedia.org/wiki/State_transition) testing
* [Use case](https://en.wikipedia.org/wiki/Use_case) testing
* [Domain analysis](https://en.wikipedia.org/wiki/Domain_analysis)
* Combining technique

**6.1.5 White-Box Testing**

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing [software](https://en.wikipedia.org/wiki/Software) that tests internal structures or workings of an application, as opposed to its functionality (i.e. [black-box testing](https://en.wikipedia.org/wiki/Black-box_testing)). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. [in-circuit testing](https://en.wikipedia.org/wiki/In-circuit_test) (ICT). White-box testing can be applied at the [unit](https://en.wikipedia.org/wiki/Unit_testing), [integration](https://en.wikipedia.org/wiki/Integration_testing) and [system](https://en.wikipedia.org/wiki/System_testing) levels of the [software testing](https://en.wikipedia.org/wiki/Software_testing) process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

White-box test design techniques include the following [code coverage](https://en.wikipedia.org/wiki/Code_coverage) criteria:

* [Control flow](https://en.wikipedia.org/wiki/Control_flow) testing
* Data flow testing
* Branch testing
* Statement coverage
* Decision coverage
* [Modified condition/decision coverage](https://en.wikipedia.org/wiki/Modified_condition/decision_coverage)
* Prime path testing
* Path testing

White-box testing is a method of testing the application at the level of the source code. These test cases are derived through the use of the design techniques mentioned above: [control flow](https://en.wikipedia.org/wiki/Control_flow) testing, data flow testing, branch testing, path testing, statement coverage and decision coverage as well as modified condition/decision coverage. White-box testing is the use of these techniques as guidelines to create an error free environment by examining any fragile code. These White-box testing techniques are the building blocks of white-box testing, whose essence is the careful testing of the application at the source code level to prevent any hidden errors later on.[[1]](https://en.wikipedia.org/wiki/White-box_testing#cite_note-level-1) These different techniques exercise every visible path of the source code to minimize errors and create an error-free environment. The whole point of white-box testing is the ability to know which line of the code is being executed and being able to identify what the correct output should be.

**6.1.5.1 Levels**

1. [Unit testing](https://en.wikipedia.org/wiki/Unit_testing). White-box testing is done during unit testing to ensure that the code is working as intended, before any integration happens with previously tested code. White-box testing during unit testing catches any defects early on and aids in any defects that happen later on after the code is integrated with the rest of the application and therefore prevents any type of errors later on.
2. [Integration testing](https://en.wikipedia.org/wiki/Integration_testing). White-box testing at this level are written to test the interactions of each interface with each other. The Unit level testing made sure that each code was tested and working accordingly in an isolated environment and integration examines the correctness of the behaviour in an open environment through the use of white-box testing for any interactions of interfaces that are known to the programmer.
3. [Regression testing](https://en.wikipedia.org/wiki/Regression_testing). White-box testing during regression testing is the use of recycled white-box test cases at the unit and integration testing levels.

**6.1.5.2 Basic Procedures**

White-box testing's basic procedures involves the tester having a deep level of understanding of the source code being tested. The programmer must have a deep understanding of the application to know what kinds of test cases to create so that every visible path is exercised for testing. Once the source code is understood then the source code can be analyzed for test cases to be created. These are the three basic steps that white-box testing takes in order to create test cases:

1. Input involves different types of requirements, functional specifications, detailed designing of documents, proper source code, security specifications. This is the preparation stage of white-box testing to layout all of the basic information.
2. Processing involves performing risk analysis to guide whole testing process, proper test plan, execute test cases and communicate results. This is the phase of building test cases to make sure they thoroughly test the application the given results are recorded accordingly.
3. Output involves preparing final report that encompasses all of the above preparations and results.

**6.1.5.3 Advantages**

White-box testing is one of the two biggest testing methodologies used today. It has several major advantages:

1. Side effects of having the knowledge of the source code is beneficial to thorough testing.
2. Optimization of code by revealing hidden errors and being able to remove these possible defects.
3. Gives the programmer introspection because developers carefully describe any new implementation.
4. Provides traceability of tests from the source, allowing future changes to the software to be easily captured in changes to the tests.
5. White box tests are easy to automate.
6. White box testing give clear, engineering-based, rules for when to stop testing.

**6.1.5.4 Disadvantages**

Although white-box testing has great advantages, it is not perfect and contains some disadvantages:

1. White-box testing brings complexity to testing because the tester must have knowledge of the program, including being a programmer. White-box testing requires a programmer with a low level of knowledge due to the complexity of the level of testing that needs to be done.
2. On some occasions, it is not realistic to be able to test every single existing condition of the application and some conditions will be untested.
3. The tests focus on the software as it exists, and missing functionality may not be discovered.

**6.1.6 System Testing**

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified [requirements](https://en.wikipedia.org/wiki/Requirements). System testing falls within the scope of [black-box testing](https://en.wikipedia.org/wiki/Black-box_testing), and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have passed [integration testing](https://en.wikipedia.org/wiki/Integration_testing) and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

* **Testing The Whole System**

System testing is performed on the entire system in the context of a [Functional Requirement](https://en.wikipedia.org/wiki/Functional_requirements) Specification(s) (FRS) and/or a [System Requirement](https://en.wikipedia.org/wiki/Requirements_analysis) Specification (SRS). System testing tests not only the design, but also the behavior and even the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).