Contat

Contents

[Front-End: 2](#_Toc164012177)

[Back-End: 2](#_Toc164012178)

[Phase 1: System Study 3](#_Toc164012179)

[Preliminary Investigation 4](#_Toc164012180)

[System Development Life Cycle 5](#_Toc164012181)

[Phase 2: System Analysis 9](#_Toc164012182)

[Importance of Computerized 10](#_Toc164012183)

[System Design 14](#_Toc164012184)

[Data Modeling 18](#_Toc164012185)

[HARDWARE 18](#_Toc164012186)

[SOFTWARE 19](#_Toc164012187)

[Phase: System Planning 19](#_Toc164012188)

[System Planning 20](#_Toc164012189)

[Estimates of Effort 20](#_Toc164012190)

[Code Efficiency 21](#_Toc164012191)

[Phase: System Testing 22](#_Toc164012192)

[Testing Phase 23](#_Toc164012193)

[Levels of testing 24](#_Toc164012194)

[Test Cases 29](#_Toc164012195)

[Verification and Validation (V&V) 30](#_Toc164012196)

[Phase: System Implementation 31](#_Toc164012197)

[System implementation maintenance and review 32](#_Toc164012198)

[Hardware Evaluation Factors: 33](#_Toc164012199)

[Software Evaluation Factors 34](#_Toc164012200)

[Gantt Chart 38](#_Toc164012201)

[Scope of future application 40](#_Toc164012202)

[Bibliography 42](#_Toc164012203)

[Glossary 43](#_Toc164012204)

# Front-End:

Python programming tools are complete programming environments. It allows programmers to build a GUI program using the various on-screen controls such as buttons, text, menus, boxes etc. These controls are placed on a form and then the processing details related with each control are filled in.

In the business world, competitive strategies have become the order of the day to improve quality, cut costs and provide a high response customer service base. Most organizations today need to be market driven and do a lot of value addition to their products and services. This naturally calls for rational decision making, which requires information. Information Technology or IT provides that effective channel to support and implement this strategy. Client/Server is the technology that empowers the desktop, thus setting a trend for the way successful organizations will use technology in the next decade.

# Back-End:

My SQL is an application used to create computer databases for the Microsoft Windows family of server operating systems. It provides an environment used to generate databases that can be accessed from workstations, the web, or other media such as a personal digital assistant (PDA). MY SQL is probably the most accessible and the most documented enterprise database environment right now. This also means that you can learn it a little quicker than most other database environments on the market

To start, you must have a computer that runs an appropriate operating system like Microsoft Windows >= 2007Home Edition: In this case, you must install MY SQL.

# Phase 1: System Study

## Preliminary Investigation

System development, a process consisting of two major steps of system analysis and design, start when management or sometimes system development personnel feel that a new system or an improvement in the existing system is required. The system development life cycle is classically thought of as the set of activities that analysts, designers and users carry out to develop and implement an information system. The system development life cycle consists of the following activities:

* Preliminary investigation
* Determination of system requirements
* Design of system
* Development of software
* System testing
* Implementation, evaluation, and maintenance

A request to take assistance from information system can be made for many reasons, but in each case someone in the organization initiates the request is made, the first system activity the preliminary investigation begins. This activity has three parts:

1. Request clarification
2. Feasibility study
3. Request approval

Request clarification: Many requests from employees and users in the organizations are not clearly defined, therefore it becomes necessary that project request must be examined and clarified properly before considering systems investigation.

## System Development Life Cycle

Systems are created to solve problems. One can think of the systems approach as an organized way of dealing with a problem. In this dynamic world, the subject System Analysis and Design (SAD), mainly deals with the software development activities.

DEFINING A SYSTEM

A collection of components that work together to realize some objective forms a system. Basically there are three major components in every system, namely input, processing and output.

In a system the different components are connected with each other and they are interdependent. For example, human body represents a complete natural system. We are also bound by many national systems such as political system, economic system, educational system and so forth. The objective of the system demands that some output is produced as a result of processing the suitable inputs.

SYSTEM LIFE CYCLE

System life cycle is an organizational process of developing and maintaining systems. It helps in establishing a system project plan, because it gives overall list of processes and subprocesses required for developing a system.

System development life cycle means combination of various activities. In other words we can say that various activities put together are referred as system development life cycle. In the System Analysis and Design terminology, the system development life cycle means software development life cycle.

Following are the different phases of software development cycle:

* System study
* Feasibility study
* System analysis
* System design
* Coding
* Testing
* Implementation
* Maintenance

The Different Phases Of Software Development Life Cycle Are Shown Below.

PRELIMINARY

INVESTIGATION

DETERMINATION OF

REQUIREMENTS

REVIEW RUNNING

SYSTEM AND SYSTEM

MAINTENANCE

SYSTEM

IMPLEMENTATION

SYSTEM TESTING

DEVELOPMENT OF

SOFTWARE AND CODING

DESIGN OF SYSTEM

DEVELOPMENT OF

PROTOTYPE SYSTEM

SYSTEM

ANALYSIS &

DESIGN 50%

DESIGN

OF

SYSTEM

30

%

CODING

20%

FIG: SHOWING GENERAL LIFE CYCLE PROCESS AND PERCENTAGE OF TIME

DEVOTED

A system analysis is a separation of a substance into parts for study and their implementation and detailed examination.

Before designing any system it is important that the nature of the business and the way it currently operates are clearly understood. The detailed examination provides the specific data required during designing in order to ensure that all the client's requirements are fulfilled. The investigation or the study conducted during the analysis phase is largely based on the feasibility study. Rather it would not be wrong to say that the analysis and feasibility phases overlap. High-level analysis begins during the feasibility study. Though analysis is represented as one phase of the system development life cycle (SDLC), this is not true. Analysis begins with system initialization and continues until its maintenance. Even after successful implementation of the system, analysis may play its role for periodic maintenance and up gradation of the system.

One of the main causes of project failures is inadequate understanding, and one of the main causes of inadequate understanding of the requirements is the poor planning of system analysis.

Analysis requires us to recall the objectives of the project and consider following three questions:

* What type of information is required?
* What are the constraints on the investigation?
* What are the potential problems that may make the task more difficult?

# Phase 2: System Analysis

## Importance of Computerized

There are several attributes in which the computer based information works. Broadly the working of computer system is divided into two main groups:

* Transaction System
* Decision Support System

Transaction System:

A transaction is a record of some well-defined single and usually small occurrence in a system. Transactions are input into the computer to update the database files. It checks the entering data for its accuracy. This means that numeric data appears in numeric field and character data in character field. Once all the checks are made, transaction is used to update the database. Transaction can be inputted in on-line mode or batch mode. In on-line mode, transactions are entered and updated into the database almost instantaneously. In batch mode, transactions are collected into batches, which may be held for a while and inputted later.

Decision Support System:

It assists the user to make analytical decision. It shows the various data in organized way called analysis. This analysis can be made to syrdy preferences and help in making decisions.

Computer system works out best with record maintenance. It will tell you which customer would get how much pending/reports statements. It will also help to search the information about a particular person by simply entering his telephone number. User can store information as per requirement, which can be used for comparison with other reports.

Principles of System Analysis

Principles:

1. Understand the problem before you begin to create the analysis model.
2. Develop prototypes that enable a user to understand how human machine interaction will occur.
3. Record the origin of and the reason for every requirement.
4. Use multiple views of requirements like building data, function and behavioral models.
5. Work to eliminate ambiguity.

A Complete Structure:

The limited time and resources have restricted us to incorporate, in this project, only the main activities that are performed in news sites, but utmost care has been taken to make the system efficient and user friendly.

For the optimum use of practical time it is necessary that every session is planned. Planning of this project will include the following things:

* Topic Understanding.
* Modular Break – Up of the System.
* Processor Logic for Each Module.
* Database Requirements.

Topic Understanding:

It is vital that the field of application as introduced in the project may be totally a new field. So as soon as the project was allocated to me, I carefully went through the project to identify the requirements of the project.

Modular Break –Up of the System:

* Identify The Various Modules In The System.
* List Them In The Right Hierarchy.  Identify Their Priority Of Development  Description Of The Modules:

## System Design

The design document that we will develop during this phase is the blueprint of the software. It describes how the solution to the customer problem is to be built. Since solution to complex problems isn’t usually found in the first try, iterations are most likely required. This is true for software design as well. For this reason, any design strategy, design method, or design language must be flexible and must easily accommodate changes due to iterations in the design. Any technique or design needs to support and guide the partitioning process in such a way that the resulting sub-problems are as independent as possible from each other and can be combined easily for the solution to the overall problem. Sub-problem independence and easy combination of their solutions reduces the complexity of the problem. This is the objective of the partitioning process. Partitioning or decomposition during design involves three types of decisions: -

Define the boundaries along which to break;

Determine into how money pieces to break; and

Identify the proper level of detail when design should stop and implementation should start.

Basic design principles that enable the software engineer to navigate the design process suggest a set of principles for software design, which have been adapted and extended in the following list:

Free from the suffer from "tunnel vision." A good designer should consider alternative approaches, judging each based on the requirements of the problem, the resources available to do the job.

The design should be traceable to the analysis model. Because a single element of the design model often traces to multiple requirements, it is necessary to have a means for tracking how requirements have been satisfied by the design model.

The design should not repeat the same thing. Systems are constructed using a set of design patterns, many of which have likely been encountered before. These patterns should always be chosen as an alternative to reinvention. Time is short and resources are limited! Design time should be invested in representing truly new ideas and integrating those patterns that already exist.

The design should "minimize the intellectual distance" between the software and the problem as it exists in the real world. That is, the structure of the software design should (whenever possible) mimic the structure of the problem domain.

The design should exhibit uniformity and integration. A design is uniform if it appears that one person developed the entire thing. Rules of style and format should be defined for a design team before design work begins. A design is integrated if care is taken in defining interfaces between design components.

The design activity begins when the requirements document for the software to be developed is available. This may be the SRS for the complete system, as is the case if the waterfall model is being followed or the requirements for the next "iteration" if the iterative enhancement is being followed or the requirements for the prototype if the prototyping is being followed. While the requirements specification activity is entirely in the problem domain, design is the first step in moving from the problem domain toward the solution domain. Design is essentially the bridge between requirements specification and the final solution for satisfying the requirements.

The design of a system is essentially a blueprint or a plan for a solution for the system. We consider a system to be a set of components with clearly defined behavior that interacts with each other in a fixed defined manner to produce some behavior or services for its environment. A component of a system can be considered a system, with its own components. In a software system, a component is a software module.

The design process for software systems, often, has two levels. At the first level, the focus is on deciding which modules are needed for the system, the specifications of these modules, and how the modules should be interconnected. This is what is called the system design or top-level design. In the second level, the internal design of the modules, or how the specifications of the module can be satisfied, is decided. This design level is often called detailed design or logic design. Detailed design essentially expands the system design to contain a more detailed description of the processing logic and data structures so that the design is sufficiently complete for coding.

Because the detailed design is an extension of system design, the system design controls the major structural characteristics of the system. The system design has a major impact on the testability and modifiability of a system, and it impacts its efficiency. Much of the design effort for designing software is spent creating the system design.

The input to the design phase is the specifications for the system to be designed. Hence, reasonable entry criteria can be that the specifications are stable and have been approved, hoping that the approval mechanism will ensure that the specifications are complete, consistent, unambiguous, etc. The output of the top-level design phase is the architectural design or the system design for the software system to be built. This can be produced with or without using a design methodology. Reasonable exit criteria for the phase could be that the design has been verified against the input specifications and has been evaluated and approved for quality.

A design can be object-oriented or function-oriented. In function-oriented design, the design consists of module definitions, with each module supporting a functional abstraction. In objectoriented design, the modules in the design represent data abstraction (these abstractions are discussed in more detail later). In the function-oriented methods for design and describe one particular methodology the structured design methodology in some detail. In a function- oriented design approach, a system is viewed as a transformation function, transforming the inputs to the desired outputs. The purpose of the design phase is to specify the components for this transformation function, so that each component is also a transformation function. Hence, the basic output of the system design phase, when a function oriented design approach is being followed, is the definition of all the major data structures in the system, all the major modules of the system, and how the modules interact with each other.Once the designer is satisfied with the design he has produced, the design is to be precisely specified in the form of a document. To specify the design, specification languages are used. Producing the design specification is the ultimate objective of the design phase. The purpose of this design document is quite different from that of the design notation. Whereas a design represented using the design notation is largely to be used by the designer, a design specification has to be so precise and complete that it can be used as a basis of further development by other programmers. Generally, design specification uses textual structures, with design notation helping in understanding

# 

# Data Modeling

Introduction to data dictionary:

Data dictionaries are an integral component of structured analysis, since data flow diagrams by themselves do not fully describe the subject of the investigation . The data flow diagrams provide the additional details about the project/system.

Data Dictionary (Definition):

A data dictionary is a catalog- a repository- of the elements in a system. These elements center on the data and the way they are structured to meet user requirements and organization needs. A data dictionary consists of a list of all the elements composing the data flowing through a system. The major elements are data flows , data stores , and processes. The data dictionary stores details and descriptions of these elements.

ER Diagram What is ER Diagram

DFD What is DFD

Class Diagram What is Class Diagram

Modules

Tools / Platform, Hardware and Software Requirement Specification

## HARDWARE

|  |  |
| --- | --- |
| Processor | : CORE i3 and above |
| Memory | : 2GB or more |
| Cache Memory | : - |
| Hard Disk | : 520 GB or more |
| Pen Drive | : Optional |
| Printer | : Not required as of now |

## SOFTWARE

|  |  |  |
| --- | --- | --- |
| Operating System |  | : Windows 10 |
| Font-End Tool |  | : Tkinter |
| Back-End |  | : MySQL |
| Ide |  | : Python Ide, PyCharm, vscode |
|  |  |  |

# Phase: System Planning

## System Planning

Scheduling of a software project does not differ greatly from scheduling of any multi- task engineering effort. Therefore, generalized project scheduling tools and techniques can be applied with little modification to software projects.

Program evaluation and review technique (PERT) and critical path method (CPM) are two project scheduling methods that can be applied to software development. Both techniques are driven by information already developed in earlier project planning activities.

# Estimates of Effort

* A decomposition of the product function.
* The selection of the appropriate process model and task set.
* Decomposition of tasks.

Interdependencies among tasks may be defined using a task network. Tasks, sometimes called the project Work Breakdown Structure (WBS) are defined for the product as a whole or for individual functions.

Both PERT and CPM provide quantitative tools that allow the software planner to (1) determine the critical path-the chain of tasks that determines the duration of the project; (2) establish "most likely" time estimates for individual tasks by applying statistical models; and (3) calculate "boundary times" that define a time window" for a particular task.

Boundary time calculations can be very useful in software project scheduling. Slippage in the design of one function, for example, can retard further development of other functions. It describes important boundary times that may be discerned from a PERT or CPM network: (I) the earliest time that a task can begin when preceding tasks are completed in the shortest possible time, (2) the latest time for task initiation before the minimum project completion time is delayed, (3) the earliest finish-the sum of the earliest start and the task duration, (4) the latest finish- the latest start time added to task duration, and (5) the total float-the amount of surplus time or leeway allowed in scheduling tasks so that the network critical path maintained on schedule. Boundary time calculations lead to a determination of critical path and provide the manager with a quantitative method for evaluating progress as tasks are completed.

Both PERT and CPM have been implemented in a wide variety of automated tools that are available for the personal computer. Such tools are easy to use and take the scheduling methods described previously available to every software project manager.

Screen Designs of the project output

Source Code

## Code Efficiency

Reviewing of Code efficiency for a module is carried out after the module is successfully compiled and all the syntax errors eliminated. Code efficiency review is extremely cost-effective strategies for reduction in coding errors in order to produce high quality code. Normally, two types of efficiency are carried out on the code of a module -code optimization and code inspection. The procedure and final objective of these two efficiency techniques are very different as discussed below.

Optimization of Code:

Code optimization is an informal code analysis technique. In this technique, after a module has been coded, it is successfully compiled and all syntax errors are eliminated. Some members of the development team are given the code a few days before the optimization meeting to read and understand the code. Each member selects some test cases and simulates execution of the code by hand (i.e. trace execution through each statement and function execution). The main objectives of the optimization are to discover the algorithmic and logical errors in the code. The members note down their findings to discuss these in a optimization meeting where the coder of the module is also present.

Even though a code optimization is an informal analysis technique, several guidelines have evolved over the years for making this naïve technique more effective and useful. Of course, these guidelines are based on personal experience, common sense, and several subjective factors. Therefore are based on personal experience, common sense, and several subjective factors. Therefore, guidelines should be considered as examples rather than as rules to be applied dogmatically. Some of these guidelines are the following:

The team performing the code optimization should not be either too big or too small. Ideally, it should consist of three to seven members.

# Phase: System Testing

UNIT TESTING

VALIDATION

INTEGRATION

SYSTEM

## Testing Phase

One of the purposes of the testing is to validate and verify the system. Verification means checking the system to ensure that it is doing what the function is supposed to do and Validation means checking to ensure that system is doing what the user wants it to do.

No program or system design is perfect; communication between the user and the designer is not always complete or clear, and time is usually short. The result is errors and more errors. Theoretically, a newly designed system should have all the pieces in working order, but in reality, each piece works independently. Now is the time to put all the pieces into one system and test it to determine whether it meets the user's requirements. This is the best chance to detect and correct errors before the system is implemented. The purpose of system testing is to consider all the likely variations to which it will be subjected and then push the system to its limits. If we implement the system without proper testing then it might cause the problems.

1. Communication between the user and the designer.
2. The programmer's ability to generate a code that reflects exactly the system specification.
3. The time frame for the design.

Theoretically, a new designed system should have all the pieces in working order, but in reality, each piece works independently. Now is the time to put all the pieces into one system and test it to determine whether it meets the requirements of the user.The process of system testing and the steps taken to validate and prepare a system for final implementation are:

## Levels of testing

The different types of testing are as follows:

UNIT TESTING

VALIDATION

INTEGRATION

SYSTEM

1. UNIT TESTING:

This is the smallest testable unit of a computer system and is normally tested using the white box testing. The author of the programs usually carries out unit tests.

1. INTEGRATION TESTING:

In integration testing, the different units of the system are integrated together to form the complete system and this type of testing checks the system as whole to ensure that it is doing what is supposed to do. The testing of an integrated system can be carried out topdown, bottom-up, or big-bang. In this type of testing, some parts will be tested with white box testing and some with black box testing techniques. This type of testing plays very important role in increasing the systems productivity. We have checked our system by using the integration testing techniques.

1. SYSTEM TESTING:

A part from testing the system to validate the functionality of software against the requirements, it is also necessary to test the non-functional aspect of the system. Some examples of non-functional tools include tests to check performance, data security, usability/user friendliness, volume, load/stress that we have used in our project to test the various modules.

System testing consists of the following steps:

* + 1. Program(s) testing.
    2. String testing.
    3. System testing.
    4. System documentation.
    5. User acceptance testing.

1. FIELD TESTING:

This is a special type of testing that may be very important in some projects. Here the system is tested in the actual operational surroundings. The interfaces with other systems and the real world are checked. This type of testing is very rarely used. So far our project is concerned; we haven't tested our project using the field testing.

1. ACCEPTANCE TESTING:

After the developer has completed all rounds of testing and he is satisfied with the system, then the user takes over and re-tests the system from his point of view to judge whether it is acceptable according to some previously identified criteria. This is almost always a tricky situation in the project because of the inherent conflict between the developer and the user. In this project, it is the job of the bookstores to check the system that whether the made system fulfills the goals or not.

Why System Testing?

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. Inadequate testing results in two types of problems:

* 1. The time lag between the cause and the appearance of the problem.
  2. The effect of system errors on the files and records within the system.

Activity Network for System Testing

The test plan entails the following activities:

* 1. Prepare test plan.
  2. Specify conditions for user acceptance testing.
  3. Prepare test data for program testing.
  4. Prepare test data for transaction path testing.
  5. Plan user training.
  6. Compile/assemble programs.
  7. Prepare job performance aids.
  8. Prepare operational documents.

PREPARE TEST: A workable test plan must be prepared in accordance with established design specifications. It includes the following items:

* + - Outputs expected from the system.
    - Criteria for evaluating outputs.
    - A volume of test data.
    - Procedure for using test data.
    - Personnel and training requirements.

SPECIFY CONDITIONS FOR USER ACCEPTANCE TESTING

Planning for user acceptance testing calls for the analyst and the user to agree on conditions for the test.

PREPARE TEST DATA FOR PROGRAM TESTING

As each program is coded, test data are prepared and documented to ensure that all aspects of the program are properly tested.

PREPARE TEST DATA FOR TRANSACTION PATH TESTING

This activity develops the data required for testing every condition and transactions to be introduced into the system. The path of each transaction from origin to destination is carefully tested reliable results.

PLAN USER TRAINING

User training is designed to prepare the user for testing and converting the system. User involvement and training take place parallel with programming for three reasons:

* The system group has time available to spend on training while the programs are being written.
* Initiating a user-training program gives the systems group a clearer image of the user's interest in the new system.
* A trained user participates more effectively in system testing.

The training plan is followed by preparation of the user training manual and other text materials.

COMPILE / ASSEMBLE PROGRAMS

All programs have to be compiled / assembled for testing.

PREPARE JOB PERFORMANCE AIDS

In this activity the materials to be used by personnel to run the system are specified and scheduled. This includes a display of materials.

PREPARE OPERATIONAL DOCUMENTS

During the test plan stage, all operational documents are finalized including copies of the operational formats required by the candidate system.

SYSTEMS TESTING

The computer department to ensure that the system functions as specified does this testing. This testing is important to ensure that a working system is handed over to the user for acceptance testing.

ACCEPTANCE TESTING

The user to ensure that the system functions, as the user actually wanted performs this testing. With prototyping techniques, this stage becomes very much a formality to check the accuracy and completeness of processing. The screen layouts and output should already have been tested during the prototyping phase.

An error in the program code can remain undetected indefinitely. To prevent this from happening the code was tested at various levels. To successfully test a system, each condition, and combinations of conditions had to be tested. Each program was tested and linked to other programs. This unit of program is tested and linked to other units and so on until the complete system has been tested.

The purpose of testing is to ensure that each program is fully tested. To do so a test plan had to be created. The test plan consists of a number of test runs such as the valid paths through the code, and the exception and error handling paths. For each test run there is a list of conditions tested, the test data used and the result expected. The test plan was then reviewed to check that each path through the code is tested correctly. It is the responsibility of the programmer to collect the data that will produce the required test condition.

## Test Cases

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | TEST DATA  Specifications for Customer and Inventory Management System | | | | | |
|  | Test Date | 08 Mar 21 | Programmer name: | Pradeep Singh | | |
|  | Tested By: | Pradeep Singh | Project Name | CUSTOMER AND INVENTORY MANAGEMENT SYSTEM | | |
|  |  |  |  |  | | |
|  | Test Cases for Login Page | | | | |  |
|  | User ID the field is required to match set value | | | | | |
|  | Password the field is required to match set value | | | | | |
|  | |  | | --- | | LOGIN |     Will direct to home page after validation | | | | | |
|  | Positive Test cases for Login Page | | | | | |
| T.C ID | PRE-CONDITION | T.C  DESCRIPTION | T.C DATA | EXPECTED | ACTUAL | RESULT |
| 1 | User should be on Login page | Check the  functionality of  User ID | 0000 | Will accept only  Value which is set in System | Ok | Pass |
| 2 | User should be on Login page | Check the  functionality of  Password | 1111 | Will accept only  Value which is set in System | Ok | Pass |
|  | Negative Test cases for User Details form | | | | | |
| T.C ID | PRE-CONDITION | T.C  DESCRIPTION | T.C DATA | EXPECTED | ACTUAL | RESULT |

## Verification and Validation (V&V)

The objectives of verification, validity activities are to assess and improve the quality of the work products generated during development and modification of the software. Quality depends upon the various attributes like correctness, completeness, consistency, reliability, usefulness, usability, efficiency and conformance to standards.

The terms verification and validation are used synonymously. These are defined as under: -

Verification: “Are we building the product right?”

Validation: “Are we building the right product?”

Verification activities include proving, testing, and reviews. Validation is the process of evaluating software at the end of the software development to ensure compliance with the software requirements. Testing is a common method of validation. Clearly, for high reliability we need to perform both activities. Together, they are often called V&V activities.

The major V&V activities for software development are inspection, reviews, and testing (both static and dynamic). The V&V plan identifies the different V&V tasks for the different phases and specifies how these tasks contribute to the project V&V goals. The methods to be used for performing these V&V activities, the responsibilities and milestones for each of these activities, inputs and outputs for each V&V task, and criteria for evaluating the outputs are also specified.

The two major V&V approaches are testing and inspections. Testing is an activity that can be generally performed only on code. It is an important activity and is discussed in detail in a later chapter. Inspection is a more general activity that can be applied to any work product, including code. Many of the V&V tasks are such that for them, an inspection type of activity is the only possible way to perform the tasks (e.g. trace ability and document evaluation). Due to this, inspections play a significant role in verification.

# Phase: System Implementation

## System implementation maintenance and review

As we know, creating software is one thing and the implementation of the created software is another. The process of implementing software is much difficult as compared to the task of creating the project. First we have to implement the software on a small scale for removing the bugs and other errors in the project and after removing them we can implement the software on a large scale. Before we think in terms of implementing the Software on a large basis, we must consider the Hardware requirements.

Whenever we develop software or project a certain hardware and software is being used by the programmer for developing the project. The hardware and software to be used by the programmer for developing the project should be such that it would result in the development of a project, which would satisfy all the basic needs for which the project has been created by the programmer. The Hardware should be such that cost constraints of the Client should also be taken into account without affecting the performance.

## Hardware Evaluation Factors:

When we evaluate computer hardware, we should first investigate specific physical and performance characteristics for each hardware component to be acquired. These specific questions must be answered concerning many important factors. These hardware evaluation factors questions are summarized in the below figure.

Notice that there is much more to evaluating hardware than determining the fastest and cheapest computing device. For e.g. the question of possible obsolescence must be addressed by making a technology evaluation. The factor of ergonomics is also very important. Ergonomics is the science and technology that tries to ensure that computers and other technologies are "user-friendly", that is safe, comfortable and easy to use. Connectivity is another important evaluation factor, since so many computer systems are now interconnected within wide area or local area telecommunications networks.

1. Performance
2. Cost
3. Reliability
4. Availability
5. Compatibility
6. Modularity
7. Technology
8. Ergonomics
9. Connectivity
10. Environmental requirements
11. Software
12. Support

## Software Evaluation Factors

Software can be evaluated according to many factors similar to the hardware evaluation. Thus the factors of performance, cost, reliability, compatibility, modularity, technology, ergonomics, and support should be used to evaluate proposed software acquisitions. In addition, however, the software evaluation factors are summarized in below figure. For e.g. some software packages require too much memory capacity and are notoriously slow, hard to use, or poorly documented. They are not a good selection for most end users, even if offered at attractive prices.

SOFTWARE EVALUATION FACTORS:

1. EFFICIENCY: Efficiency is the software a well-written system of computer instructions that does not use much memory capacity or CPU time?
2. FLEXIBILITY: can it handle its processing assignments easily without major modifications?
3. SECURITY: Security does it provide control procedures for errors, malfunctions and improper use?
4. LANGUAGE: Language do our computer programmers and users write it in a programming language that is used?
5. DOCUMENTATION: Documentation is the s/w well documented? Does it include helpful user instructions?
6. HARDWARE: Hardware does existing hardware have the features required to best use this software?
7. Other characteristics of hardware such as its performance, what about the cost, how much is reliable and etc.

Conversion and Training:

An important aspect of is to make sure that the new design is implemented to establish standards. The term implementation has different meanings, ranging form the conversion of a basic application to a complete replacement of a computer system. Implementation is used here to mean the process of converting a new or revise system into an operational one. Conversion is one aspect of implementation. Conversion means changing form one system to another. The objective is to put the tested system into operation while holding costs, risks, and personnel irritation to a minimum. It involves creating computer-compatible files, training the operation staff, and installing terminal and hardware. A critical aspect of conversion is not disrupting the functioning of the organization.

When a new system is used over and old, existing and running one, there are always compatibility errors. These errors are caused because of the lack of equipment or personnel to work the new system. Running any specified system at an organization does require some or other hardware or, in this case, software requirement as well.

Conversion is one aspect of implementation review & software maintenance.

There are three types of implementation:

1. Implementation of a computer system to replace a manual system. The problems encountered are converting files, training users, creating accurate files and verifying printouts for integrity.
2. Implementation of a new computer system to replace an existing one. This is usually a difficult conversion. If not properly planned there can be many problems. Some large computer systems have taken as long as year to convert.
3. Implementation of a modified application to replace an existing one, using the same computer. This type of conversion is relatively easy to handle, provided there are no major changes in the files.

Training Needs:

Training needs refer to the gaining of knowledge required for running the system.

First of all the system is a computer based system therefore the person should have good knowledge about computer and its working.

He should know how to use software's on the computer.

For a better usage and working of the software the organization should appoint a person who has good knowledge of all the required software. The organization gets a person trained through different institutes present in the market. The training should be as per the above requirements.

Cost Estimation of the Project:

Cost in a project is due to the requirements for software, hardware, and human resources. Hardware resources are computer time, terminal time and memory required for the project. Software resources include the tools and compilers needed during development. The bulk of cost of software development is due to human resources needed. Cost estimates are determined in terms of person-months (PM).

Total No. Of Persons Involved In This Project:

1. Project Manager
2. Senior Programmer

Since this Project will complete in 6 weeks

Wage/hour=100

Working hours a day= 8

Total number of days=16

Total Wage = 100 x 8 x 16 x 2 = Rs. 25,600/- Installation Charges= Rs. 1000 Cost Estimate:

= Salary of Person Involved + Deployment Fees + Rs. 30,000(Desktop) = 25,600+1000+30,000\*

 Rs. 25,600/- to 56,600/-

## Gantt Chart

Gantt charts mainly used to allocate resources to activities. The resources allocated to activities include staff, hardware, and software. Gantt charts (named after its developer Henry Gantt) are useful for resource planning. A Gantt chart is special type of bar chart where each bar represents an activity. The bars are drawn along a timeline. The length of each bar is proportional to the duration of the time planned for the corresponding activity.

Gantt chart is a project scheduling technique. Progress can be represented easily in a Gantt chart, by coloring each milestone when completed. The project started in the month of December 2023 and end in the February 2024.

Security and Validation Checks:

In this project we have used following validation checks.

* While entering the data into the form it will check for the name of the client is properly filled & it should not be null.

* Whenever we enter the data for the new customer, company, or user will automatically check the details from the database tables and also generate the connection number automatically.

* Similarly in the complaint table complaint number will generate automatically.

* Entered text / number should not exceed the limit (width).

Almost for all fields we have used the validation for example if name of the fields requires the text type of data then it will check for the string and if the data is numeric then it will check if the number entered is proper numeric or not.

# Scope of future application

This Software will be developed for the Online interested in information about customer and inventory. Utmost care and back-up procedures must be established to ensure 100% successful implementation of the Customer and inventory Management System. In case of system failure, the organization should be in a position to process the transaction with another organization or if the worst comes to the worst, it should be in a position to complete it manually.

The decision to automate generally depends on the needs to have accurate, consistent and timely data in a variety of reporting formats. But the most important factor that should be considered is –

* Be sure to take future growth into consideration and evaluate whether the software package could be modified if the organization expands in the future or you need to revise the system.

* The next generation of Windows-based computerized systems would be changing the way owners and management think about running their businesses. It would a powerful tool that gives them more ways to get their financial Information so that they can better manage and grow their business.

* The next generation of systems would take advantage of the latest technologies, including Microsoft Windows>=2007 and Orcale11i to offer access and integrate with all aspects of a business. Keeping this in view we could develop systems that would work efficiently and integrate seamlessly in virtually any industry setting, even up to mid-sized corporations and divisions of larger organizations.

* Consider another setting, where the users are not comfortable on computers, for such users we can develop systems where the users can have their own personalized menu setup in the accounting software so that they no longer have to search around to find what they need.

* Now consider the Internet. This wide open information exchange network comes with many benefits for businesses, including breaking down physical barriers as to where he/she can consult to the companies.

* Adding other capabilities can be added time to time. These are  Application may be installed client server.
  + Application can upgraded according to User Requirements with little changes made.
  + Application may be transferred to latest RDMBS like Oracle with little changes in current code. Use for Security purpose

NOTE: In a nutshell we have a lot of scope to further enhance the futures and functionalities of the proposed Solution

# Bibliography

* Foundation, P. S. (n.d.). The Python Tutorial. Retrieved from Python Documentation: https://docs.python.org/3/tutorial/

* Foundation., P. S. (n.d.). tkinter-Python interface to Tk. Retrieved from Python Documentation:

* https://docs.python.org/3/library/tkinter.html

* freepick.com. (n.d.). Retrieved from freepik.com

* Jain, S. Core Python.

* Python Tutorial. (n.d.). Retrieved from TutorialPoint: https://www.tutorialspoint.com/python/index.htm

* Rao, N. Core Python.

# Glossary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.No. | Module | Methods | | | Other |
| 1 | Tkinter ttk  messagebox | 1. Tk() 2. Geometry() 3. Resizable() 4. Mainloop() 5. .get() 6. .set() 7. .config() 8. .askyesno() 9. .bind() | | | Widgets:   1. Button 2. Label 3. Entry 4. Radiobutton 4. Canvas:    * create\_image()    * create\_window()    * create\_text()    * canvas.delete() 5. Treeview    * delete()    * get\_children()    * insert()    * heading()    * column()    * .indentify\_row(event.y)  .item(.focus())     Container:   1. frame 2. labelframe 3. canvas Layouts: 4. place() 5. pack() 6. grid() |
| 2 | mysql.connector | | 1. | mysql.connector.connect() | |
|  |  | | 2. | con.cursor() | |
|  |  | | 3. | con.commit() | |
|  |  | | 4. | con.rollback() | |
|  |  | | 5. | con.is\_connected() | |
|  |  | | 6. | cur.execute() | |
|  |  | | 7. | cur.fetchone() | |
|  |  | | 8. | cur.fetchall() | |
| 3 | PIL(Image,  ImageTk) | | 1. 2. | image.open()  resize() | |
|  |  | | 3. | image.antialias | |
|  |  | | 4. | imagetk.photoimage() | |
| 4 | Os | | 1. | remove() | |
|  |  | | 2. | rename() | |
| 5 | File Handling | | 1. | f=open(“file.txt”) | |
|  |  | | 2. | f.readline() | |
|  |  | | 3. | f.write() | |
|  |  | | 4. | f. close() | |
| 6 | String | |  | str.split(‘delimiter’) | |
| 7 | Datetime(date) | |  | .today() | |