**2. Software Requirement Analysis and Specifications**

**2.1 Product perspective**

**User Details**

In this project the system can store and maintain the record in database of the users’ information related to the project, which will specify the details of who are registered and authorized to access the data.

**Data Provider Details**

This record will maintain the details of all the data providers who are able to right to upload the data to the server. This data will provide the details of what the data provider can upload into the server.

**2.2 User Characteristics**

Every user should be comfortable of working with computer and net browsing. She/he must have basic knowledge of English too. The administrators of various phases must be aware of basic networking security and must handle the passwords carefully.

**2.3 Functional Requirements**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs in general are:

* External Outputs, whose destination is outside the organization,.
* Internal Outputs whose destination is within organization and they are the User’s main interface with the computer.
* Operational outputs whose use is purely within the computer department.
* Interface outputs, which involve the user in communicating directly.
* Understanding user’s preferences, expertise level and his business requirements through a friendly questionnaire.
* Input data can be in four different forms - Relational DB, text files, .xls and xml files. For testing and demo you can choose data from any domain. User-B can provide business data as input.

**2.4 Non-Functional Requirements**

Non-functional requirements describe user-visible aspects of the system that are not directly related to functionality of the system.

**User Interface**

A menu interface has been provided to the user to be user friendly.

**Performance Constraints**

* Requests should be processed within no time.
* Users should be authenticated for accessing the requested data

**Error Handling and Extreme Conditions**

* In case of User Error, the System should display a meaningful error message to the user, such that the user can correct his Error.
* The high level components in proposed system should handle exceptions that occur while connecting to various database servers, IO Exceptions etc.

**Quality Issues**

* Quality issues mainly refers to reliable, available and robust system developing the proposed system the developer must be able to guarantee the reliability transactions so that they will be processed completely and accurately.

**2.5 Software Requirements:**

* Operating System : Windows 10.
* Programming Language : ASP.Net, C#.
* Front-end : Visual Studio 2015 Professional.
* Database : SQL Server 2014.
* Application server : IIS 7.0

**2.6 Hardware Requirements:**

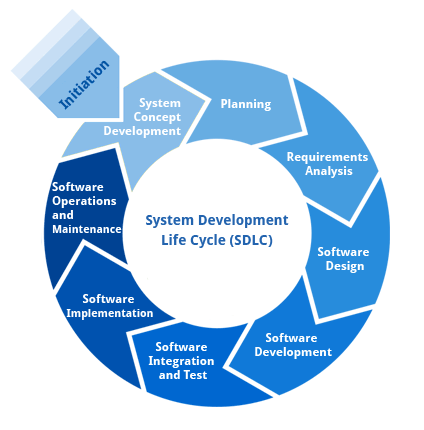
* Hard Disk : 1 TB.
* RAM : 8 GB.
* Processor : Intel Core i3.

**2.7 SDLC Methodologies**

This document play a vital role in the development of life cycle (SDLC) as it describes the complete requirement of the system. It means for use by developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

**2.7.1Software Development Life Cycle**

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies use to develop these systems.



**Figure2.7.1: System Development Life Cycle**

**Requirement Analysis and Design**

Analysis gathers the requirements for the system. This stage includes a detailed study of the business needs of the organization. Options for changing the business process may be considered. The design focuses on high level design like, what programs are needed and how are they going to interact, low-level design (how the individual programs are going to work), interface design (what are the interfaces going to look like) and data design (what data will be required). During these phases, the software's overall structure is defined. Analysis and Design are very crucial in the whole development cycle.

**Implementation**

In this phase the designs are translated into code. Computer programs are written using a conventional programming language or an application generator. Programming tools like Compilers, Interpreters and Debuggers are used to generate the code. Different high-level programming languages like C, C++, Pascal, Java, .Net are used for coding. With respect to the type of application, the right programming language is chosen.

**Testing**

In this phase the system is tested. Normally programs are written as a series of individual modules, these subjects to separate and detailed test. The system is then tested as a whole. The separate modules are brought together and tested as a complete system.

**Maintenance**

Inevitably the system will need maintenance. The software will definitely undergo change once it is delivered to the customer. There are many reasons for the change. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations.

**2.7.2 Spiral Model**

It was defined by Barry Boehm in his 1988 article, A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models. Each phase starts with a design goal and ends with a client reviewing the progress thus far. Analysis and engineering efforts are applied at each phase of the project, with an eye toward the end goal of the project.

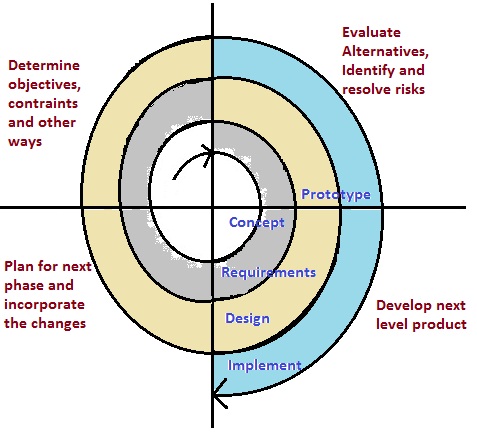
The steps for Spiral Model can be generalized as follows:

* The new system requirements are defined in as much details as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
* A preliminary design is created for the new system.
* A first prototype of the new system is constructed from the preliminary design.

**Advantages**

* Estimation becomes more realistic as work progresses, because important issues discovered earlier.
* It is more able to cope with the changes that are software development generally entails.
* Software engineers can get their hands on and start working on the core of a project earlier.

The following diagram shows how a spiral model acts like:



**Figure 2.7.2: Spiral Model.**

**2.8 Feasibility Study**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out.

Four key considerations involved in the feasibility analysis are:

* Economic Feasibility.
* Technical Feasibility.
* Social Feasibility.
* Operational Feasibility.

**2.8.1 Economic Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available.

**2.8.2 Technical Feasibility**

This study is carried out to check the technical feasibility that is the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client.

**2.8.3 Social Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it.

**2.8.4 Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information systems, which will meet the organization’s operating requirements. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

**2.9 Introduction to .Net Framework**

**Features of .NET**

Microsoft .NET is a set of Microsoft software technologies for rapidly building and integrating XML Web services, Microsoft Windows-based applications, and Web solutions. The .NET Framework is a language-neutral platform for writing programs that can easily and securely interoperate. There’s no language barrier with .NET: there are numerous languages available to the developer including managed C++, C#, Visual Basic and Java Script. The .NET framework provides the foundation for components to interact seamlessly, whether locally or remotely on different platforms. The following features of the .NET framework are also worth description:

**Common Type System**

The CLR uses something called the Common Type System (CTS) to strictly enforce type-safety. This ensures that all classes are compatible with each other, by describing types in a common way. CTS define how types work within the runtime, which enables types in one language to interoperate with types in another language, including cross-language exception handling.

**Common Language Specification**

The CLR provides built-in support for language interoperability. To ensure that you can develop managed code that can be fully used by developers using any programming language, a set of language features and rules for using them called the Common Language Specification (CLS) has been defined. Components that follow these rules and expose only CLS features are considered CLS-compliant.

**Languages Supported By .NET**

The multi-language capability of the .NET Framework and Visual Studio .NET enables developers to use their existing programming skills to build all types of applications and XML Web services. The .NET framework supports new versions of Microsoft’s old favorites Visual Basic and C++ (as VB.NET and Managed C++), but there are also a number of new additions to the family. Visual Basic .NET has been updated to include many new and improved language features that make it a powerful object-oriented programming language. These features include inheritance, interfaces, and overloading, among others.

* COBOL
* Eiffel

|  |  |
| --- | --- |
| ASP.NET  XML WEB SERVICES | Windows Forms |
| Base Class Libraries | |
| Common Language Runtime | |
| Operating System | |

**Figure 2.9,1: .NET Framework**

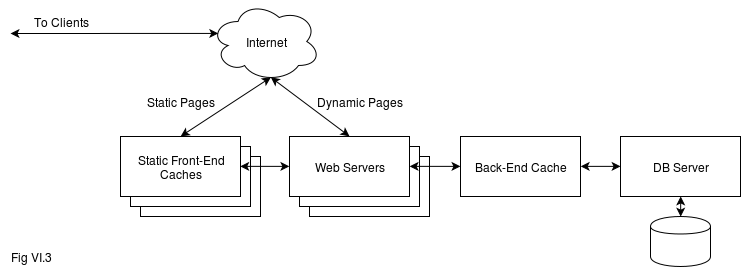
**Constructors and Destructors**

Constructors are used to initialize objects, whereas destructors are used to destroy them. In other words, destructors are used to release the resources allocated to the object. In C#.NET the sub finalize procedure is available. The sub finalize procedure is used to complete the tasks that must be performed when an object is destroyed. .

**Memory management**

The .NET Framework CLR frees the developer from the burden of managing memory (allocating and freeing up when done); instead it does the memory management itself. To this end, the memory allocated to instantiations of .NET types (objects) is done contiguously from the managed heap, a pool of memory managed by the CLR.

**Server-side managed code**

**Figure 2.9.1:** Server side manage code diagram