**A**

**PROJECT REPORT**

**SUBMITTED**

**IN PARTIAL FULLFILEMENT**

**FOR THE AWARD OF THE DEGREE**

**BACHELOR OF TECHNOLOGY**

**IN THE DEPARTMENT OF COMPUTER SCIENCE**



**Name:-Lucky Borana**

**Roll no. :- CS022**

**Name:- Mukesh Jangir**

**Roll no. :- CS027**

**MAHARAJA COLLEGE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE & IT ENGINEERING**

**RAJAASTHAN TECHNICAL UNIVERSITY**

**CONTENT**

1. ACKNOWLEDMENT
2. APPRAISAL SHEET
3. OBJECTIVE
4. HARDWARE AND SOFTWARE REQUIREMENT
5. TOOLS AND PLATFORM
6. PROJECT CATERGRY
7. PROJECT DESIGN
8. SYSTEM ANALYSIS AND DESIGN
9. TECHNOLOGY USED
10. FUTURE ASPECT
11. UTILITY OF PROJECT CLINIC MANAGEMENT
12. FLOW CHART
13. DATA TABLE
14. SOURCE CODE
15. BIBLIOGRAPHY

**ACKNOWLEDGEMENT**

**ACKNOWLEDGEMENT**

I am extremely grateful and remain indebted to my guide Mr. Bhavesh Gandhi for being a source of inspiration for his constant support in Design, implementation and Evolution of this project. I am thankful to him for his constant constructive criticism and invaluable suggestion, Which benefited me a lot while developing this project on “Clinic Management” he has been a constant source of inspiration and motivating to hard work. Through this column, it would be utmost pleasure to express my warm thanks to him.

**Lucky Borana & Mukesh Jangir**

**APPRASIAL SHEET**

**APPRAISAL SHEET**

This is to certified that the project Titled ”Clinic Management ” is submitted by Lucky Borana & Mukesh Jangir as a student of B.Tech from Maharaja college of engineering, Udaipur towards partial fulfillment of their Bachelor Degree.

The Project demonstrate a good hold of the student over the field of interest. This work reveals a high technical acumen as well as aptitude. This work is the product of their labor and hard work.

It is approved for the award of bachelor of computer science degree.

External Examiner Internal Examiner

**OBJECTIVES**

**Objectives**

**Clinic Management Systems** are computer software products that coordinate and integrate all the inherent activities involved in the management and running of a [healthcare](http://en.wikipedia.org/wiki/Healthcare) facility. They must meet specified security, technology and functionality standards for Patient Information, Medicine details, Bed details, Staff details Doctor Details coupled with the [Electronic Medical Records](http://en.wikipedia.org/wiki/Electronic_Medical_Records) technology & functional standards.

We with this project are trying to maintain are possible with ADO.NET connected Sql Database. Database of Clinic till yet patient , Medicine, Bed and Doctor details are maintaining in registry which are materialistic thing and its hard to say that they are proper r not. You can choose SQL Server or MS Access as the database for this project.

Let us design a table corresponding to each entity so that we can save those entities. When a new patient arrives in a hospital and register, we will add a record to this table. A patient will have only one record in this table. This is a master table, where we maintain the list of all available doctors in the hospital. This is a master table where maintain the list of all beds (rooms) in the hospital. Before we start assigning a bed to a patient, we must have created all available beds in the hospital.

**HARDWARE**

**&**

**SOFTWARE**

**REQUIRMENT**

**Configuration**

**HARDWARE USED:-**

PROCESSOR : Pentium and Higher

RAM : 256MB and Higher

HARDDISK : 280MB and Higher

**OUTPUT DEVICES:-**

MONITOR : MONOCROMO/COLOUR

MONITER

PRINTER : DMP/DESJET/LASER

MINIMUM RESLUTION : 1024×768 PIXELS/32BITS

**SOFTWARE USED:-**

OPEARATING SYSTEM : WINDOWS XP/7

PROGRAMMING PACKAGE : VISUAL BASIC STUDIO 2008

DATA BASE : MICROSOFT SQL SERVER 2005

**TOOLS & PLATFORMS**

**TOOLS AND PLATFORM**

Stay with us on the payroll presentation, a “Clinic Management”. This project is develop on visual basic 2008 using the database technology at its core. As the visual images and graphics more appealing affect to the eyes on the user then they other conventional format, so as to make the project more effective and to make the project to able to store the information about various entities involved. The project incorporates the use of tools like:-

* MS VISUAL BASIC - FRONT END
* SQL DATABASE SERVER - BACK END

The front end is software, which manages the logical representation of data while back end manages manipulate & store data.

**PROJECT CATEGORY**

**INTRODUCTION OF COMPUTING**

**BASIC ORGANIZATION OF A COMPUTER SYSTEM:-**

A Computer is an electronic device, which accepts data from input devices and process, data according to the instructive stored in it and then, communicates the result output devices.

**COMPONANTS OF A COMPUTER SYSTEM:-**

1. **INPUT UNITS:**

Devices which accepts data from the user and communication them to central processing unit.

1. **CENTRAL PROCESSING UNIT (CPU):-**

It performs all the arithmetic calculation and makes logical decision and control the activities of all other units i.e.

1. **Arithmetical Logic unit**
2. **Control unit**
3. **Memory**

**Storage devices: –** There are two types of storage devices.

* **PRIMARY STORAGE DEVICES** :-

The term of storage devices refer to the main memory of computer . the primary storage devices is volatile means that the data and instruction are lst as the computer is switched off.

Example of primary storage device is ROM.

* **SECOUNDARY STORAGE DEVICES:-**

These are called auxiliary storage devices, secondary storage is non-volatile, meaning that the data and instruction remain intact when the computer is turned off.

Example of secondary storage devices floppy disk, hard disk.

1. **OUTPUT DEVICES:-**

These obtain the machine coded output result from the CPU and convert into a from as desired by the user.

**INTODUCTION OF MS VISUAL BASIC 2008**

Visual Basic .NET (VB.NET) is an [object-oriented](http://en.wikipedia.org/wiki/Object-oriented_programming) [computer programming language](http://en.wikipedia.org/wiki/Programming_language) that can be viewed as an evolution of the classic [Visual Basic](http://en.wikipedia.org/wiki/Visual_Basic) (VB) which is implemented on the [.NET Framework](http://en.wikipedia.org/wiki/.NET_Framework). Microsoft currently supplies two major implementations of Visual Basic: [Microsoft Visual Studio](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio), which is [commercial software](http://en.wikipedia.org/wiki/Commercial_software) and [Microsoft Visual Studio Express](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio_Express).

## Versions :-

There are four versions and five releases of Visual Basic .NET implemented by the Visual Basic Team.

### Visual Basic .NET 2003 (VB 7.1):-

Visual Basic .NET 2003 was released with *version 1.1* of the .NET Framework. New features included support for the [.NET Compact Framework](http://en.wikipedia.org/wiki/.NET_Compact_Framework) and a better VB upgrade [wizard](http://en.wikipedia.org/wiki/Wizard_%28software%29). Improvements were also made to the performance and reliability of the .NET IDE (particularly the [background compiler](http://msdn.microsoft.com/msdnmag/issues/05/06/AdvancedBasics/default.aspx)) and runtime. In addition, Visual Basic .NET 2003 was available in the *Visual Studio.NET Academic Edition* (VS03AE). VS03AE is distributed to a certain number of scholars from each country without cost.

### Visual Basic 2005 (VB 8.0):-

Visual Basic 2005 is the name used to refer to the Visual Basic .NET, Microsoft having decided to drop the .NET portion of the title.

### Visual Basic 2008 (VB 9.0):-

Visual Basic 9.0 was released together with the [Microsoft .NET Framework 3.5](http://en.wikipedia.org/wiki/.NET_Framework#.NET_Framework_3.5) on 19 November 2007.

For this release, Microsoft added many features, including:

* A true [conditional operator](http://en.wikipedia.org/wiki/Conditional_operator), "I If (condition as Boolean, true part, false part)", to replace the "I If" function.
* [Anonymous types](http://en.wikipedia.org/wiki/Anonymous_type)
* Support for [LINQ](http://en.wikipedia.org/wiki/LINQ)
* [Lambda expressions](http://en.wikipedia.org/wiki/Lambda_expression)
* [XML Literals](http://en.wikipedia.org/wiki/XML_Literals)
* [Type Inference](http://en.wikipedia.org/wiki/Type_inference)
* [Extension methods](http://en.wikipedia.org/wiki/Extension_method)

### Visual Basic 2010 (VB 10.0):-

### In April 2010, Microsoft released Visual Basic 2010. Microsoft had planned to use the [Dynamic Language Runtime (DLR)](http://en.wikipedia.org/wiki/Dynamic_Language_Runtime) for that release but shifted to a co-evolution strategy between Visual Basic and sister language C# to bring both languages into closer parity with one another. Visual Basic's innate ability to interact dynamically with CLR and COM objects has been enhanced to work with dynamic languages built on the DLR such as [Iron Python](http://en.wikipedia.org/wiki/IronPython) and [Iron Ruby](http://en.wikipedia.org/wiki/IronRuby). The Visual Basic compiler was improved to infer line continuation in a set of common contexts, in many cases removing the need for the "\_" line continuation character. Also, existing support of inline Functions was complemented with support for inline Subs as well as multi-line versions of both Sub and Function lambdas.

### Visual Studio Express:-

[Visual Studio Express Editions](http://en.wikipedia.org/wiki/Microsoft_Visual_Studio_Express) are a set of free lightweight individual IDEs which are provided as stripped-down versions of the Visual Studio IDE on a per-platform basis or per-language basis, i.e., it installs the development tools for the supported platforms (web, Windows, phone) or supported development languages (VB, C#) onto individual Visual Studio Shell Aphids. It includes only a small set of tools as compared to the other systems. It does not include support for plug-ins. [x64](http://en.wikipedia.org/wiki/X86-64) compilers are not included in the Visual Studio Express edition IDEs, but are available as part of a Windows Software Development Kit that can be installed separately. Microsoft targets the Express IDEs at students and hobbyists. Express editions do not use the full [MSDN Library](http://en.wikipedia.org/wiki/MSDN_Library) but use the MSDN Essentials Library. The languages available as part of the Express IDEs is:

* **Visual Basic Express**
* **Visual C++ Express**
* **Visual C# Express**
* **Visual Web Developer Express**
* **Express for Windows Phone**

### Visual Studio Light Switch:-

Microsoft Visual Studio Light Switch is an IDE specifically tailored for creating line-of-business applications built on existing .NET technologies and Microsoft platforms. The applications produced are architecturally [3-tier](http://en.wikipedia.org/wiki/Multitier_architecture): the user interface runs on [Microsoft Silver light](http://en.wikipedia.org/wiki/Silverlight); the logic and data-access tier is built on WCF RIA Services and Entity Framework, hosted in ASP.NET; and the primary data storage supports [Microsoft SQL Server Express](http://en.wikipedia.org/wiki/SQL_Server_Express), [Microsoft SQL Server](http://en.wikipedia.org/wiki/Microsoft_SQL_Server) and [Microsoft SQL Azure](http://en.wikipedia.org/wiki/SQL_Azure). Light Switch also supports other data sources including [Microsoft SharePoint](http://en.wikipedia.org/wiki/SharePoint). Light Switch includes graphical designers for designing entities and entity relationships, entity queries, and UI screens. Business logic may be written in either [Visual Basic](http://en.wikipedia.org/wiki/Visual_Basic_.NET) or [Visual C#](http://en.wikipedia.org/wiki/Visual_C_Sharp). The tool can be installed as a stand-alone SKU or as an integrated add-in to Visual Studio 2010 Professional and higher.

### Visual Studio Professional:-

Visual Studio Professional Edition provides an IDE for all supported development languages. As of Visual Studio 2010, the Standard edition was renamed to the Professional edition. MSDN support is available as MSDN Essentials or the full MSDN library depending on licensing. It supports [XML](http://en.wikipedia.org/wiki/XML) and [XSLT](http://en.wikipedia.org/wiki/XSLT) editing, and can create deployment packages that only use [Click Once](http://en.wikipedia.org/wiki/ClickOnce) and [MSI](http://en.wikipedia.org/wiki/Windows_Installer). It includes tools like Server Explorer and integration with [Microsoft SQL Server](http://en.wikipedia.org/wiki/Microsoft_SQL_Server) also. Windows Mobile development support was included in Visual Studio 2005 Standard; however, with Visual Studio 2008, it is only available in Professional and higher editions. Windows Phone 7 development support was added to all editions in Visual Studio 2010. Development for all versions of Windows Mobile prior to Windows Mobile 7 is no longer supported in Visual Studio 2010.

### Visual Studio Premium:-

Visual Studio Premium Edition includes all of the tools in Visual Studio Professional and adds additional functionality such as code metrics, profiling, static code analysis, and database unit testing.

### Visual Studio Tools for Office:-

[Visual Studio Tools for Office](http://en.wikipedia.org/wiki/Visual_Studio_Tools_for_Office) is an SDK and an add-in for Visual Studio that includes tools for developing for the [Microsoft Office](http://en.wikipedia.org/wiki/Microsoft_Office) suite. Previously (for Visual Studio .NET 2003 and Visual Studio 2005) it was a separate SKU that supported only [Visual C#](http://en.wikipedia.org/wiki/Visual_C_Sharp) and [Visual Basic](http://en.wikipedia.org/wiki/Visual_Basic_.NET) languages or was included in the Team Suite. With Visual Studio 2008, it is no longer a separate SKU but is included with Professional and higher editions. A separate runtime is required when deploying VSTO solutions.

### Visual Studio Ultimate:-

Visual Studio Ultimate provides set of software and database development, collaboration, metrics, architecture, testing and reporting tools in addition to the features provided by Visual Studio Premium. As of Visual Studio 2010, the Team Suite edition was renamed to the Ultimate edition. Visual Studio Ultimate offers a superset of toolsets based on the [Application Lifecycle Management](http://en.wikipedia.org/wiki/Application_Lifecycle_Management) (ALM) role it is being used for.

#### Visual Studio Team System:-

Prior to Visual Studio 2010, [Visual Studio Team System](http://en.wikipedia.org/wiki/Visual_Studio_Team_System) provided four role-specific editions are:

* **Team Explorer** (basic TFS client)
* **Architecture Edition**
* **Database Edition**
* **Development Edition**
* **Test Edition**

The combined functionality of the four Team System Editions is provided in a **Team Suite** Edition. The Database Edition, codenamed "Data Dude", was initially released as a separate edition after Visual Studio 2005's initial release. It is included with Visual Studio 2008 as a separate edition, but Microsoft did roll its functionality into the Premium Edition with Visual Studio 2010.

### Test Professional:-

Visual Studio Test Professional is an edition which was introduced with Visual Studio 2010. Its focus is aimed at the dedicated tester role and includes support for the management of test environments, the ability to start and report on tests and to connect to Team Foundation Server. It does not include support for development or authoring of tests.

**Using ActiveX Control in .Net:-**

ActiveX control is a special type of COM component that supports a User Interface. Using ActiveX Control in your .Net Project is even easier than using COM component. They are bundled usually in .ocx files. Again a proxy assembly is made by .Net utility AxImp.exe (which we will see shortly) which your application (or client) uses as if it is a .Net control or assembly.

**ADVANTAGES OF USING VB.NET: -**

1. First of all, VB.NET provides managed code execution that runs under the Common Language Runtime (CLR), resulting in robust, stable and secure applications. All features of the .NET framework are readily available in VB.NET.
2. VB.NET is totally object oriented. This is a major addition that VB6 and other earlier releases didn't have.
3. The .NET framework comes with ADO.NET, which follows the disconnected paradigm, i.e. once the required records are fetched the connection no longer exists. It also retrieves the records that are expected to be accessed in the immediate future. This enhances Scalability of the application to a great extent.
4. VB.NET uses XML to transfer data between the various layers in the DNA Architecture i.e. data are passed as simple text strings.
5. Error handling has changed in VB.NET. A new Try-Catch-Finally block has been introduced to handle errors and exceptions as a unit, allowing appropriate action to be taken at the place the error occurred thus discouraging the use of ON ERROR GOTO statement. This again credits to the maintainability of the code.
6. Another great feature added to VB.NET is free threading against the VB single-threaded apartment feature. In many situations developers need spawning of a new thread to run as a background process and increase the usability of the application. VB.NET allows developers to spawn threads wherever they feel like, hence giving freedom and better control on the application.
7. Security has become more robust in VB.NET. In addition to the role-based security in VB6, VB.NET comes with a new security model, Code Access security. This security controls on what the code can access. For example you can set the security to a component such that the component cannot access the database. This type of security is important because it allows building components that can be trusted to various degrees.
8. The CLR takes care of garbage collection i.e. the CLR releases resources as soon as an object is no more in use. This relieves the developer from thinking of ways to manage memory. CLR does this for them.

**WHY I USED “VISUAL BASIC”?**

In addition to addition to advantages of the visual basic mentioned above, there are certain regions to use the visual basic in development of this application:-

* The visual basic in technology committed to improved itself continually and grows along with the changes and the improvements in the operating systems used.
* Visual basic enables rapid prototyping as is built tools, making programming with visual basic easy.
* The developers can create interactive application and web based application with ease.

**INTRODUCTION OF MICROSOFT SQL SERVER 2005**

After a long gap in the release of SQL Server databases, Microsoft recently released SQL Server 2005 (formerly code-named Yukon). In this substantial upgrade, they've packed the new database engine full of features. Probably the most significant one that will catch your attention is the price tag – it’s up to 25% higher than SQL Server 2000. A single processor license for SQL Server 2005 Enterprise Edition will set you back approximately $25,000. That’s not cheap, but Microsoft has made some great advances in functionality that make up the difference.   
  
In this first part of our series on this new product, let’s take a look at the four different editions of SQL Server 2005 that Microsoft plans to release:

* **SQL Server 2005 Express** replaces the Microsoft Data Engine (MSDE) as the free version of SQL Server for application development and lightweight use. It remains free and retains the limitations of MSDE with respect to client connections and performance. It’s a great tool for developing and testing applications and extremely small implementations, but that’s about as far as you can run with it.
* **SQL Server 2005 Workgroup** is the new entrant in the product line. It’s billed as a “small business SQL Server” and it offers an impressive array of functionality for a tag per processor. Workgroup edition maxes out at 2 CPUs with 3GB of RAM and allows for most of the functionality you’d expect from a server-based relational database. It offers limited replication capabilities as well.
* The workhorse **SQL Server 2005 Standard Edition** remains the staple of the product line for serious database applications. It can handle up to 4 CPUs with an unlimited amount of RAM. Standard Edition 2005 introduces database mirroring and integration services. The big kid on the block is **SQL Server 2005 Enterprise Edition**. With the release of 2005, Enterprise Edition allows unlimited scalability and partitioning. It’s truly an enterprise-class database and it’s hefty price tag reflects its value.

Those are the basics of SQL Server 2005! In future weeks, we’ll explore some of the new functionality offered by this powerful database.

[**Hardware and Software Requirements (32-Bit and 64-Bit)**](javascript:void(0))**:-**

### Monitor

SQL Server graphical tools require VGA or higher resolution: at least 1,024x768 pixel resolution.

### Pointing Device

A Microsoft mouse or compatible pointing device is required.

### CD or DVD Drive

A CD or DVD drive, as appropriate, is required for installation from CD or DVD media.

### Cluster Hardware Requirements

On 32-bit and 64-bit platforms, eight-node cluster installations (that is, the maximum number of nodes supported by Microsoft Windows Server 2003) are supported. For more information on high availability solutions, see [Configuring High Availability](http://msdn.microsoft.com/en-us/library/ms190202%28v=sql.90%29.aspx) and [Before Installing Failover Clustering](http://msdn.microsoft.com/en-us/library/ms189910%28v=sql.90%29.aspx).

### Network Software Requirements

Network software requirements for the 64-bit versions of SQL Server 2005 are the same as the requirements for the 32-bit versions. Windows Server 2003, Windows XP, and Windows 2000 have built-in network software.

Stand-alone named and default instances support the following network protocols:

* Shared memory
* Named pipes
* TCP/IP
* VIA

### Software Requirements

SQL Server Setup requires Microsoft Windows Installer 3.1 or later and Microsoft Data Access Components (MDAC) 2.8 SP1 or later. You can download MDAC 2.8 SP1 from this [Microsoft Web site](http://go.microsoft.com/fwlink/?LinkId=50233).

SQL Server Setup installs the following software components required by the product:

* Microsoft .NET Framework 2.0
* Microsoft SQL Server Native Client
* Microsoft SQL Server Setup support files

### Cross-Language Support

**For upgrades:**

* English-language versions of SQL Server can be upgraded to any localized version of SQL Server 2005.
* Localized versions of SQL Server can be upgraded to localized versions of SQL Server 2005 of the same language.
* Localized version of SQL Server cannot be upgraded to the English-language version of SQL Server 2005.
* Localized versions of SQL Server cannot be upgraded to localized SQL Server 2005 versions of a different localized language.

All localized upgrades must follow supported upgrade paths. For specific information on supported upgrade paths, see [Version and Edition Upgrades](http://msdn.microsoft.com/en-us/library/ms143393%28v=sql.90%29.aspx).

**Additional cross-language support:**

* The English-language version of SQL Server 2005 is supported on all localized versions of supported operating systems.
* Localized versions of SQL Server 2005 are supported on localized operating systems that are the same language as the localized SQL Server version.
* Localized versions of SQL Server 2005 are also supported on English-language versions of supported operating systems through the use of Windows Multilingual User Interface Pack (MUI) settings. However, you must verify certain operating system settings before installing a localized version of SQL Server 2005 on a server that is running an English-language operating system with a non-English MUI setting. You must verify that the following operating system settings match the language of the localized SQL Server to be installed:
  + The operating system user interface setting
  + The operating system user locale setting
  + The system locale setting

If these operating system settings do not match the language of the localized SQL Server, then you must correctly set these operating system settings as described in [How to: Change Operating System Settings to Support Localized Versions](http://msdn.microsoft.com/en-us/library/ms144258%28v=sql.90%29.aspx).

### Virtual Operating System Support

You can install SQL Server 2005 on Microsoft Virtual Server and on Microsoft Virtual PC on supported operating systems and hardware. For information about Virtual Server, see this [Microsoft Web site](http://go.microsoft.com/fwlink/?LinkId=50379). For information about Virtual PC, see this [Microsoft Web site](http://go.microsoft.com/fwlink/?LinkId=50378).

**component of ms sql server:-**

Microsoft SQL Server 2005 can be installed through use of the Installation Wizard, or from the command prompt. The Installation Wizard provides a graphical user interface that guides you through each install-time decision, and is the preferred method for most users. The Installation Wizard provides guidance for initial setup of SQL Server 2005, including feature selection, instance naming rules, service account configuration, strong password guidelines, and scenarios for setting collations.

### SQL Server Integration Services

This is a revamped version of the SQL Server 2000 based Data Transformation Services. The improvements have been made in the performance, usability and manageability aspects of the tool. It contains a Business Intelligence Workbench and SQL server Workbench which enables extracting data for the data warehouse simpler. Analysis Services has been enhanced to give better performance.

**Relational and XML data support**

Both relational and XML structured data are supported. The XML data becomes available through the addition of the XML data type and allows storing of XML fragments and documents in SQL Server databases. It has new data types such as VARCHAR(MAX), NVARCHAR(MAX) and VARBINARY(MAX) which has an ability to store up to 2 GB of data supplementing TEXT, NTEXT and IMAGE data types.

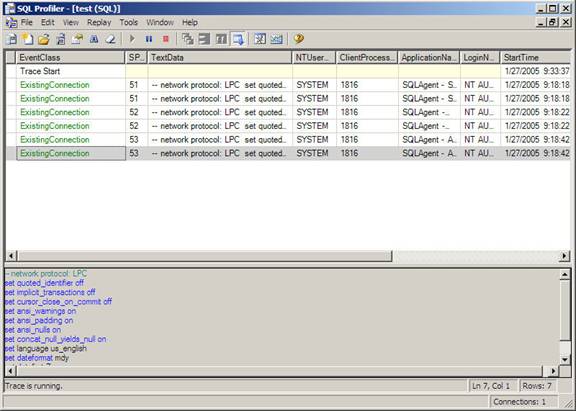
**SQL Computer Manager**

This utility is accessible from the Programs menu and is implemented as a Microsoft Management console snap in. It has a functionality that is based on Windows Management Instrumentation. It combines the features of SQL Server Manager, Server Network Configuration and Client Network configuration. It is backward compatible

### SQL Profiler

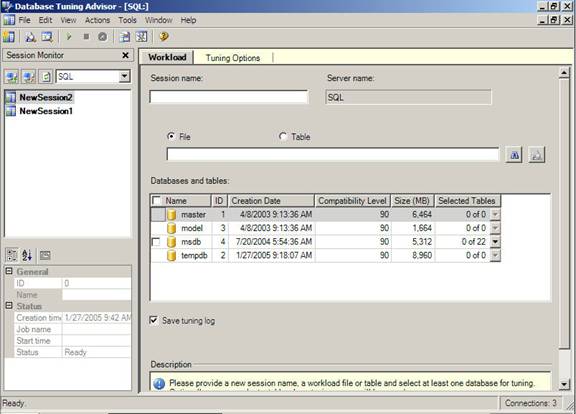
The SQL Profiler has been enhanced with a few new features. The ability to analyze MDX Analysis Services statements and monitoring the operation of the Data Transformation services are significant changes to the tool. Access to the utility is no longer limited to the SysAdmin fixed server role. It can be granted to designated logins in the Analysis server role.

Aggregated views allow grouping of recorded events on the basis of arbitrarily selected criteria. Special file types can be used to extract and store events for ease of troubleshooting. The Create Trace Wizard enables the creation of traces. Deadlocks can be displayed in a graphical view and integration with the Windows System monitor helps troubleshoot problems of performance.

****

### Database Tuning Advisor

This is an enhanced version of the Index tuning Wizard. In SQL Server 2005 it integrates with the SQL Profiler and reduces the time spent on tuning. It also helps determine whether aligned or non-aligned layout should be used to benefit performance in dealing with partitioning related tables. It analyzes workload and physical implementation of one or more databases. It uses trace files, trace tables or Transact-SQL scripts as workload input when tuning databases. It can recommend the optimum physical design structure for the database and statistics that should be collected to backup physical design structures.



**Command Line Utilities**

The SQLWB.EXE launches the SQL Server Management Studio from the command prompt. The server type can be specified by using the various switches of this command. Databases can also be specified or queries, projects and solutions can be specified. The default values of this command is stored in Tools>options menu. Profiler90.Exe is used as a command line utility that launches the SQL Profiler and provides an ability to connect to a server or Analysis Services or database, table, template or file etc.

DTA.EXE is a command line version of the Database Tuning Advisor. It analyzes performance and provides recommendations regarding improvements. It can run on the same server or on a different server. Connection settings will have to be specified using the switches. Other functions such as table tuning, workload in the form of SQL Profiler Trace, SQL file or SQL Server Trace file etc can also be performed using this utility along with the switches.

**SQLCMD.EXE**

This is a new and improved version of the OSQL and ISQL programs of the earlier versions of this server. It helps connect to any version of SQL Server via OLE DB. It runs batches of T-SQL statements. It can also connect to a server via the Dedicated Administrative connection when invoked with an –A switch from the Command prompt.

A number of functional enhancements have also been made to SQLCMD.EXE. Creation of elaborate scripts, provision for setting the SQLCMDPASSWORD environment variable, assignment of custom and predefined scripting variable for use within scripts and execution of SQLCMD scripts from within the query editor are some of the new features added in SQL Server 2005.

The above details were only intended to give the reader a flavor of the new SQL Server 2005. We will study the features in greater detail later in this tutorial and also learn how to use them to an advantage.

**PROJECT DESIGN**

**PROJECT DESIGN**

**Module Mycs:-**

This module is provide network string for dataset. In this module we define data sure connection**.**

**FRM CLINIC:-**

This form is working as main form whole project. It provides the following option (command button)-

* Patient details
* Doctor details
* Bed details
* Medicine details
* Staff details
* Exit

To on desire d form you have select and click on he option.

**Patient details:-**

This form is used to insert new record of the patient.

**Medicine details:-**

This form is used to insert new record medicine.

**Bed details:-**

This form is used to insert bed details.

**Staff details:-**

This form is used to insert employee details.

**Doctor details:-**

This form is used to insert doctor information.

**Exit :-**

This menu is used to close.

**SYSTEM ANALYSIS**

**&**

**DESIGN**

**Introduction to System Analysis and Design**

**INTRODUCTION**

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, mainly deals with the software development activities.

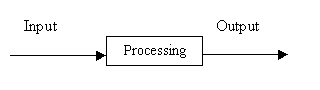
**OBJECTIVES**

After going through this lesson, you should be able to:

* understand a system
* understand the different phases of system developments life cycle
* know the components of system analysis
* know the components of system designing

**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 demand that some output is produced as a result of processing the suitable inputs.

**SYSTEM LIFE CYCLE**

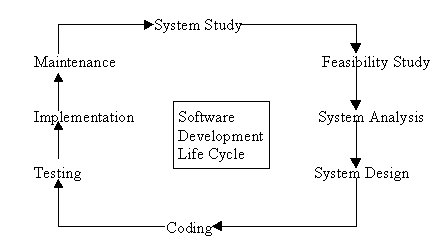
System life cycle is an organisational process of developing and maintaining systems. It helps in establishing a system project plan, because it gives overall list of processes and sub-processes required developing a system.

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 is shown in Fig.29.1

****

Different phases of Software development Life Cycle

**PHASES OF SYSTEM DEVELOPMENT LIFE CYCLE**

Let us now describe the different phases and the related activities of system development life cycle in detail.

**(a) System Study**

System study is the first stage of system development life cycle. This gives a clear picture of what actually the physical system is? In practice, the system study is done in two phases. In the first phase, the preliminary survey of the system is done which helps in identifying the scope of the system. The second phase of the system study is more detailed and in-depth study in which the identification of user’s requirement and the limitations and problems of the present system are studied. After completing the system study, a system proposal is prepared by the System Analyst (who studies the system) and placed before the user.

**(b) Feasibility Study**

On the basis of result of the initial study, feasibility study takes place. The feasibility study is basically the test of the proposed system in the light of its workability, meeting user’s requirements, effective use of resources and .of course, the cost effectiveness. The main goal of feasibility study is not to solve the problem but to achieve the scope. In the process of feasibility study, the cost and benefits are estimated with greater accuracy.

**(c) System Analysis**

Assuming that a new system is to be developed, the next phase is system **analysis**. Analysis involved a detailed study of the current system, leading to specifications of a new system. Analysis is a detailed study of various operations performed by a system and their relationships within and outside the system. During analysis, data are collected on the available files, decision points and transactions handled by the present system. Interviews, on-site observation and questionnaire are the tools used for system analysis. Using the following steps it becomes easy to draw the exact boundary of the new system under consideration:

* Keeping in view the problems and new requirements
* Workout the pros and cons including new areas of the system

All procedures, requirements must be analysed and documented in the form of detailed data flow diagrams (DFDs), data dictionary, logical data structures and miniature specifications. System Analysis also includes sub-dividing of complex process involving the entire system, identification of data store and manual processes.

The main points to be discussed in system analysis are:

* Specification of what the new system is to accomplish based on the user requirements.
* Functional hierarchy showing the functions to be performed by the new system and their relationship with each other.
* Function network which are similar to function hierarchy but they highlight the those functions which are common to more than one procedure.
* List of attributes of the entities - these are the data items which need to be held about each entity (record)

**(d) System Design**

Based on the user requirements and the detailed analysis of a new system, the new system must be designed. This is the phase of **system designing**. It is a most crucial phase in the development of a system. Normally, the design proceeds in two stages :

* preliminary or general design
* Structure or detailed design

Preliminary or general design: In the preliminary or general design, the features of the new system are specified. The costs of implementing these features and the benefits to be derived are estimated. If the project is still considered to be feasible, we move to the detailed design stage.

Structure or Detailed design: In the detailed design stage, computer oriented work begins in earnest. At this stage, the design of the system becomes more structured. Structure design is a blue print of a computer system solution to a given problem having the same components and inter-relationship among the same components as the original problem. Input, output and processing specifications are drawn up in detail. In the design stage, the programming language and the platform in which the new system will run are also decided.

There are several tools and techniques used for designing. These tools and techniques are:

* Flowchart
* Data flow diagram (DFDs)
* Data dictionary
* Structured English
* Decision table
* Decision tree

Each of the above tools for designing will be discussed in detailed in the next lesson.

**(e) Coding**

After designing the new system, the whole system is required to be converted into computer understanding language. **Coding** the new system into computer programming language does this. It is an important stage where the defined procedure are transformed into control specifications by the help of a computer language. This is also called the programming phase in which the programmer converts the program specifications into computer instructions, which we refer as **programs**. The programs coordinate the data movements and control the entire process in a system.

It is generally felt that the programs must be modular in nature. This helps in fast development, maintenance and future change, if required.

**(f) Testing**

Before actually implementing the new system into operations, a test run of the system is done removing all the bugs, if any. It is an important phase of a successful system. After codifying the whole programs of the system, a test plan should be developed and run on a given set of test data. The output of the test run should match the expected results.

Using the test data following test run are carried out:

* Unit test
* System test

Unit test: When the programs have been coded and compiled and brought to working conditions, they must be individually tested with the prepared test data. Any undesirable happening must be noted and debugged (error corrections).

System Test: After carrying out the unit test for each of the programs of the system and when errors are removed, then system test is done. At this stage the test is done on actual data. The complete system is executed on the actual data. At each stage of the execution, the results or output of the system is analysed. During the result analysis, it may be found that the outputs are not matching the expected out of the system. In such case, the errors in the particular programs are identified and are fixed and further tested for the expected output.

When it is ensured that the system is running error-free, the users are called with their own actual data so that the system could be shown running as per their requirements.

**(g) Implementation**

After having the user acceptance of the new system developed, the implementation phase begins. Implementation is the stage of a project during which theory is turned into practice. During this phase, all the programs of the system are loaded onto the user's computer. After loading the system, training of the users starts. Main topics of such type of training are:

* How to execute the package
* How to enter the data
* How to process the data (processing details)
* How to take out the reports

After the users are trained about the computerised system, manual working has to shift from manual to computerised working. The following two strategies are followed for running the system:

1. **Parallel run:** In such run for a certain defined period, both the systems i.e. computerized and manual are executed in parallel. This strategy is helpful because of the following:
   * Manual results can be compared with the results of the computerized system.
   * Failure of the computerized system at the early stage, does not affect the working of the organization, because the manual system continues to work, as it used to do.
2. **Pilot run:** In this type of run, the new system is installed in parts. Some part of the new system is installed first and executed successfully for considerable time period. When the results are found satisfactory then only other parts are implemented. This strategy builds the confidence and the errors are traced easily.

**(h) Maintenance**

Maintenance is necessary to eliminate errors in the system during its working life and to tune the system to any variations in its working environment. It has been seen that there are always some errors found in the system that must be noted and corrected. It also means the review of the system from time to time. The review of the system is done for:

* knowing the full capabilities of the system
* knowing the required changes or the additional requirements
* studying the performance

If a major change to a system is needed, a new project may have to be set up to carry out the change. The new project will then proceed through all the above life cycle phases.

**DESIGN**

Software design is a process of problem solving and planning for a [software](http://en.wikipedia.org/wiki/Software) solution. After the purpose and specifications of software are determined, [software developers](http://en.wikipedia.org/wiki/Software_developer) will [design](http://en.wikipedia.org/wiki/Design) or employ [designers](http://en.wikipedia.org/wiki/Designer) to develop a plan for a solution. It includes low-level component and [algorithm](http://en.wikipedia.org/wiki/Algorithm) implementation issues as well as the [architectural](http://en.wikipedia.org/wiki/Software_architecture) view.

A software design may be [platform-independent](http://en.wikipedia.org/wiki/Platform-independent_model) or [platform-specific](http://en.wikipedia.org/wiki/Platform-specific_model), depending on the availability of the technology called for by the design.

Software design can be considered as putting solution to the problem(s) in hand using the available capabilities. Hence the main difference between Software analysis and design is that the output of the analysis of a software problem will be smaller problems to solve and it should not deviate so much even if it is conducted by different team members or even by entirely different groups. But since design depends on the capabilities, we can have different designs for the same problem depending on the capabilities of the environment that will host the solution (whether it is some OS, web , mobile or even the new cloud computing paradigm). The solution will depend also on the used development environment (Whether you build a solution from scratch or using reliable [frameworks](http://en.wikipedia.org/wiki/Frameworks) or at least implement some suitable [design patterns](http://en.wikipedia.org/wiki/Design_patterns)).

## Software Design Topics:-

### Design concepts

The design concepts provide the software designer with a foundation from which more sophisticated methods can be applied. A set of fundamental design concepts has evolved. They are:

1. [**Abstraction**](http://en.wikipedia.org/wiki/Abstraction) - Abstraction is the process or result of generalization by reducing the information content of a concept or an observable phenomenon, typically in order to retain only information which is relevant for a particular purpose.
2. [**Refinement**](http://en.wikipedia.org/wiki/Program_refinement) - It is the process of elaboration. A hierarchy is developed by decomposing a macroscopic statement of function in a stepwise fashion until programming language statements are reached. In each step, one or several instructions of a given program are decomposed into more detailed instructions. Abstraction and Refinement are complementary concepts.
3. [**Modularity**](http://en.wikipedia.org/wiki/Modularity) - Software architecture is divided into components called modules.
4. [**Software Architecture**](http://en.wikipedia.org/wiki/Software_Architecture)- It refers to the overall structure of the software and the ways in which that structure provides conceptual integrity for a system. A good software architecture will yield a good return on investment with respect to the desired outcome of the project, e.g. in terms of performance, quality, schedule and cost.
5. [**Control Hierarchy**](http://en.wikipedia.org/w/index.php?title=Control_Hierarchy&action=edit&redlink=1)- A program structure that represent the organization of a program components and implies a hierarchy of control.
6. [**Structural Partitioning**](http://en.wikipedia.org/w/index.php?title=Structural_Partitioning&action=edit&redlink=1) - The program structure can be divided both horizontally and vertically. Horizontal partitions define separate branches of modular hierarchy for each major program function. Vertical partitioning suggests that control and work should be distributed top down in the program structure.
7. [**Data Structure**](http://en.wikipedia.org/wiki/Data_Structure) - It is a representation of the logical relationship among individual elements of data.
8. [**Software Procedure**](http://en.wikipedia.org/w/index.php?title=Software_Procedure&action=edit&redlink=1) - It focuses on the processing of each modules individually
9. [**Information Hiding**](http://en.wikipedia.org/wiki/Information_Hiding) - Modules should be specified and designed so that information contained within a module is inaccessible to other modules that have no need for such information.

### Design considerations:-

There are many aspects to consider in the design of a piece of software. The importance of each should reflect the goals the software is trying to achieve. Some of these aspects are:

* **Compatibility** - The software is able to operate with other products that are designed for interoperability with another product. For example, a piece of software may be backward-compatible with an older version of itself.
* [**Extensibility**](http://en.wikipedia.org/wiki/Extensibility) - New capabilities can be added to the software without major changes to the underlying architecture.
* [**Fault-tolerance**](http://en.wikipedia.org/wiki/Fault-tolerance) - The software is resistant to and able to recover from component failure.
* [**Maintainability**](http://en.wikipedia.org/wiki/Maintainability) - The software can be restored to a specified condition within a specified period of time. For example, antivirus software may include the ability to periodically receive virus definition updates in order to maintain the software's effectiveness.
* [**Modularity**](http://en.wikipedia.org/wiki/Modularity) - the resulting software comprises well defined, independent components. That leads to better maintainability. The components could be then implemented and tested in isolation before being integrated to form a desired software system. This allows division of work in a software development project.
* [**Packaging**](http://en.wikipedia.org/wiki/Packaging) - Printed material such as the box and manuals should match the style designated for the target market and should enhance usability. All compatibility information should be visible on the outside of the package. All components required for use should be included in the package or specified as a requirement on the outside of the package.
* **Reliability** - The software is able to perform a required function under stated conditions for a specified period of time.
* [**Reusability**](http://en.wikipedia.org/wiki/Reusability) - the software is able to add further features and modification with slight or no modification.
* [**Robustness**](http://en.wikipedia.org/wiki/Fault-tolerant_system) - The software is able to operate under stress or tolerate unpredictable or invalid input. For example, it can be designed with a resilience to low memory conditions.
* [**Security**](http://en.wikipedia.org/wiki/Computer_security) - The software is able to withstand hostile acts and influences.
* [**Usability**](http://en.wikipedia.org/wiki/Usability) - The software [user interface](http://en.wikipedia.org/wiki/User_interface) must be usable for its target user/audience. Default values for the parameters must be chosen so that they are a good choice for the majority of the users.

### Modeling language

A [modeling language](http://en.wikipedia.org/wiki/Modeling_language) is any artificial language that can be used to express information or knowledge or systems in a structure that is defined by a consistent set of rules. The rules are used for interpretation of the meaning of components in the structure. A modeling language can be graphical or textual. Examples of graphical modeling languages for software design are:

* [Business Process Modeling Notation](http://en.wikipedia.org/wiki/Business_Process_Modeling_Notation) (BPMN) is an example of a [Process Modeling](http://en.wikipedia.org/wiki/Process_Modeling) language.
* [EXPRESS](http://en.wikipedia.org/wiki/EXPRESS_%28data_modeling_language%29) and EXPRESS-G (ISO 10303-11) is an international standard general-purpose [data modeling](http://en.wikipedia.org/wiki/Data_modeling) language.
* [Extended Enterprise Modeling Language](http://en.wikipedia.org/wiki/Extended_Enterprise_Modeling_Language) (EEML) is commonly used for business process modeling across a number of layers.
* [Flowchart](http://en.wikipedia.org/wiki/Flowchart) is a schematic representation of an algorithm or a stepwise process,
* [Fundamental Modeling Concepts](http://en.wikipedia.org/wiki/Fundamental_Modeling_Concepts) (FMC) modeling language for software-intensive systems.
* [IDEF](http://en.wikipedia.org/wiki/IDEF) is a family of modeling languages, the most notable of which include [IDEF0](http://en.wikipedia.org/wiki/IDEF0) for functional modeling, [IDEF1X](http://en.wikipedia.org/wiki/IDEF1X) for information modeling, and [IDEF5](http://en.wikipedia.org/wiki/IDEF5) for modeling ontologism.
* [Jackson Structured Programming](http://en.wikipedia.org/wiki/Jackson_Structured_Programming) (JSP) is a method for structured programming based on correspondences between data stream structure and program structure
* [LePUS3](http://en.wikipedia.org/wiki/Lepus3) is an [object-oriented](http://en.wikipedia.org/wiki/Object-oriented) visual Design Description Language and a [formal specification](http://en.wikipedia.org/wiki/Formal_specification) language that is suitable primarily for modelling large object-oriented ([Java](http://en.wikipedia.org/wiki/Java_%28programming_language%29), [C++](http://en.wikipedia.org/wiki/C%2B%2B), [C#](http://en.wikipedia.org/wiki/C_Sharp_%28programming_language%29)) programs and [design patterns](http://en.wikipedia.org/wiki/Design_patterns).
* [Unified Modeling Language](http://en.wikipedia.org/wiki/Unified_Modeling_Language) (UML) is a general modeling language to describe software both structurally and behaviorally. It has a graphical notation and allows for extension with a [Profile (UML)](http://en.wikipedia.org/wiki/Profile_%28UML%29).
* [Alloy (specification language)](http://en.wikipedia.org/wiki/Alloy_%28specification_language%29) is a general purpose specification language for expressing complex structural constraints and behavior in a software system. It provides a concise language based on first-order relational logic.
* [Systems Modeling Language](http://en.wikipedia.org/wiki/Systems_Modeling_Language) (SysML) is a new general-purpose modeling language for systems engineering.

### Design patterns

A software designer or architect may identify a design problem which has been solved by others before. A template or pattern describing a solution to a common problem is known as a [design pattern](http://en.wikipedia.org/wiki/Design_pattern_%28computer_science%29). The reuse of such patterns can speed up the software development process, having been tested and proved in the past.

TECHNOLOGY USED

**Technologies used**

ADO.NET:-

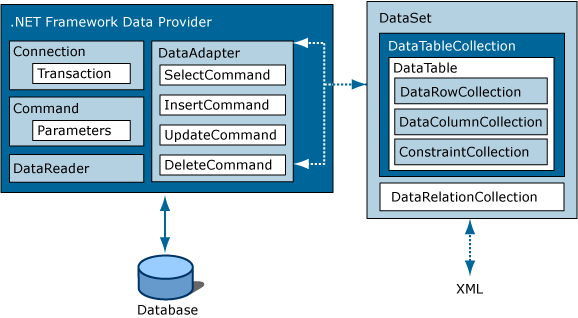
ADO.NET provides consistent access to data sources such as SQL Server and XML and to data sources exposed through OLE DB and ODBC. Data-sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, handle, and update the data that they contain.

ADO.NET separates data access from data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET Framework data providers for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, placed in an ADO.NET [Dataset](http://msdn.microsoft.com/en-us/library/system.data.dataset%28v=VS.90%29.aspx) object in order to be exposed to the user in an ad hoc manner, combined with data from multiple sources, or passed between tiers. The DataSet object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML.

The ADO.NET classes are found in System.Data.dll, and are integrated with the XML classes found in System.Xml.dll. For sample code that connects to a database, retrieves data from it, and then displays that data in a console window, see [ADO.NET Code Examples](http://msdn.microsoft.com/en-us/library/dw70f090%28v=VS.90%29.aspx).

ADO.NET provides functionality to developers who write managed code similar to the functionality provided to native component object model (COM) developers by ActiveX Data Objects (ADO). We recommend that you use ADO.NET, not ADO, for accessing data in your .NET applications.

**ADO.NET architecture:-**



### Choosing a Data Reader or a Dataset:-

When you decide whether your application should use a Data Reader (see [Retrieving Data Using a Data Reader (ADO.NET)](http://msdn.microsoft.com/en-us/library/haa3afyz%28v=VS.90%29.aspx)) or a Dataset (see [Datasets, Data Tables, and Data Views (ADO.NET)](http://msdn.microsoft.com/en-us/library/ss7fbaez%28v=VS.90%29.aspx)), consider the type of functionality that your application requires. Use a Dataset to do the following:

* Cache data locally in your application so that you can manipulate it. If you only need to read the results of a query, the Data Reader is the better choice.
* Remote data between tiers or from an XML Web service.
* Interact with data dynamically such as binding to a Windows Forms control or combining and relating data from multiple sources.
* Perform extensive processing on data without requiring an open connection to the data source, which frees the connection to be used by other clients.

If you do not require the functionality provided by the Dataset, you can improve the performance of your application by using the Data Reader to return your data in a forward-only, read-only manner. Although the Data Adapter uses the Data Reader to fill the contents of a Dataset (see [Populating a Dataset from a Data Adapter (ADO.NET)](http://msdn.microsoft.com/en-us/library/bh8kx08z%28v=VS.90%29.aspx)), by using the Data Reader, you can boost performance because you will save memory that would be consumed by the Dataset, and avoid the processing that is required to create and fill the contents of the Dataset.

[**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))LINQ to Dataset**](javascript:void(0))**:-**

LINQ to Dataset provides query capabilities and compile-time type checking over data cached in a Dataset object. It allows you to write queries in one of the .NET Framework development language, such as C# or Visual Basic. For more information, see [LINQ to Dataset](http://msdn.microsoft.com/en-us/library/bb386977%28v=VS.90%29.aspx).

[**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))LINQ to SQL**](javascript:void(0))

LINQ to SQL supports queries against an object model that is mapped to the data structures of a relational database without using an intermediate conceptual model. Each table is represented by a separate class, tightly coupling the object model to the relational database schema. LINQ to SQL translates language-integrated queries in the object model into Transact-SQL and sends them to the database for execution. When the database returns the results, LINQ to SQL translates the results back into objects. For more information, see [LINQ to SQL](http://msdn.microsoft.com/en-us/library/bb386976%28v=VS.90%29.aspx).

[**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))ADO.NET Entity Framework**](javascript:void(0))

The ADO.NET Entity Framework is designed to enable developers to create data access applications by programming against a conceptual application model instead of programming directly against a relational storage schema. The goal is to decrease the amount of code and maintenance required for data-oriented applications. For more information, see [ADO.NET Entity Framework](http://msdn.microsoft.com/en-us/library/bb399572%28v=VS.90%29.aspx).

[**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))ADO.NET Data Services**](javascript:void(0))

The ADO.NET Data Services framework is used to deploy data services on the Web or an intranet. The data is structured as entities and relationships according to the specifications of the Entity Data Model. Data deployed on this model is addressable by standard HTTP protocol. For more information, see [ADO.NET Data Services Framework](http://msdn.microsoft.com/en-us/library/cc668792%28v=VS.90%29.aspx).

[**[http://i.msdn.microsoft.com/Hash/030c41d9079671d09a62d8e2c1db6973.gif](javascript:void(0))XML and ADO.NET**](javascript:void(0))

ADO.NET leverages the power of XML to provide disconnected access to data. ADO.NET was designed hand-in-hand with the XML classes in the .NET Framework; both are components of a single architecture.

ADO.NET and the XML classes in the .NET Framework converge in the DataSet object. The Dataset can be populated with data from an XML source, whether it is a file or an XML stream. The Dataset can be written as World-Wide Web Consortium (W3C) compliant XML that includes its schema as XML schema definition language (XSD) schema, regardless of the source of the data in the DataSet. Because of the native serialization format of the Dataset is XML, it is an excellent medium for moving data between tiers, making the Dataset an optimal choice for remoting data and schema context to and from an XML Web service. For more information, see [XML Documents and Data](http://msdn.microsoft.com/en-us/library/2bcctyt8%28v=VS.90%29.aspx).

**TIMER CONTROL:-**

A timer control can execute code at regular interval by causing a timer event to occur.

The timer control, invisible to the user is useful for background processing.

You can’t set enabled property of a timer for a multiple section of control other than timer control.

**TOOLBAR CONTROL:-**

A toolbar control contains a collection of button object used to create a toolbar that is associated with application.

Typically, toolbar contain buttons that correspond to item in an application menu, providing a graphic interface for the user to access an application’s most frequently used function an command.

**IMAGE LIST CONTROL:-**

An image list control contains a collection of list image objects, each of which can be referred to by its index or key. The image list control is not meant to be used alone , but has a central responsibility to conveniently supply other control with image

**FRAME CONTROL:-**

Frame control identifiable grouping for control you can also use a frame to subdivide a form functionality – for example to separate group of button controls.

**TEXT BOX CONTROL:-**

A text box control some time called an edit field or edit control, display information enter in design time, entered by the user r assign to the controls in code at run time.

**LABEL CONTROL;-**

Label control is a graphical control: you can used to display text that a user can’t change directly.

**COMMAND BUTTON CONTROL:-**

Use a command button control to begin interrupt or end a process when choose a command button appears pushed in and so is sometimes called a push button.

**COMBO BOX CONTROL:-**

A combo box combine the feature of a text box control & a list box control user can enter in the text box portion or select an item from the list box pattern of control.

**CHECK BOX CONTROL:-**

A check box control display an X when selected: the X disappear when check box is cleared use this control to give the user a true/false or yes/no option. You can use check box control in group to display multiple choice from which the user can select one or more you can also set the value of check box programmatically with the value properly.

**FLOW CHART**

**Patient details**

Patient

Edit

Information inserted

INSERT

OUTPUT

**DATA FLOW DIAGRAM & ER. DIAGRAM**

Conform massage

Patient details form

Confirm details

Medicine

Verify

Ability

inserted

Patient name

Accepted id

**Patient details**

**FLOW CHART TO INSERT PATIENT DEAILS**

Click patient detail

Enter the data

Record does not exist

Save the record

No

yes

Enter new record

No

yes

**Data table**

**DATA TABLE FOT PATIENT DETAILS**

|  |  |  |
| --- | --- | --- |
| **S.no.** | **Field** | **type** |
| 1. | Id | int |
| 2. | Name | nvarchar(MAX) |
| 3. | Address | nvarchar(MAX) |
| 4. | Date of birth | Date time |
| 5. | Phone No. | Int |
| 6. | Emergency contact no. | int |
| 7. | Date of registration | Date time |
| 8. | gender | nvarchar(MAX) |

**DATA TABLE FOT DOCTOR DETAILS**

|  |  |  |
| --- | --- | --- |
| **S.no.** | **Field** | **type** |
| 1. | Id | Int |
| 2. | Name | Varchar(max) |
| 3. | Address | Varchar(max) |
| 4. | Phone no. | int |

**DATA TABLE FOT BED DETAILS**

|  |  |  |
| --- | --- | --- |
| **S.no.** | **Field** | **type** |
| 1. | Id | int |
| 2. | Bed patient name | Varchar(max) |
| 3. | Bed rate | Money |
| 4. | Bed type | Varchar (max) |

**DATA TABLE FOT MEDICINE DETAILS**

|  |  |  |
| --- | --- | --- |
| **S.no.** | **Field** | **Type** |
| 1. | Medicine name | Vharchar(max) |
| 2. | Medicine cost | Money |
| 3. | Date of dispensed | Datetime |
| 4. | Expiry date | Datetime |
| 5. | Method | Vharchar(max) |
| 6. | Special precaution | Vharchar(max) |

**DATA TABLE FOR STAFF DETAILS**

|  |  |  |
| --- | --- | --- |
| S.no. | Field | type |
| 1. | Name | Vharchar(max) |
| 2. | Address | Vharchar(max) |
| 3. | Phone no, | int |
| 4. | Post | Vharchar(max) |
| 5. | Salary | Money |
| 6. | Work time | vharchar |
| 7. | Date of joining | Date time |

SOURCE CODE

&

VIEW

**SOURCE CODE FOR FORM CLINIC DETAILS**

Public Class clinic

Private Sub PatientDetailsToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles PatientDetailsToolStripMenuItem.Click

For Each f As Form In Me.MdiChildren

f.Close()

Next

Dim x As New frmpatient

x.MdiParent = Me

x.Show()

x.Location = New Point(0, 0)

x.Size = New Size(1024, 720)

End Sub

Private Sub DoctorDetailsToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

For Each f As Form In Me.MdiChildren

f.Close()

Next

Dim x As New doctor

x.MdiParent = Me

x.Show()

x.Location = New Point(0, 0)

x.Size = New Size(1024, 720)

End Sub

Private Sub BedDetailsToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

For Each f As Form In Me.MdiChildren

f.Close()

Next

Dim x As New Bed

x.MdiParent = Me

x.Show()

x.Location = New Point(0, 0)

x.Size = New Size(1024, 720)

End Sub

Private Sub ExitToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

For Each f As Form In Me.MdiChildren

f.Close()

Next

Dim x As New frmmedicine

x.MdiParent = Me

x.Show()

x.Location = New Point(0, 0)

x.Size = New Size(1024, 720)

End Sub

Private Sub StaffDetailsToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

For Each f As Form In Me.MdiChildren

f.Close()

Next

Dim x As New staffdetails

x.MdiParent = Me

x.Show()

x.Location = New Point(0, 0)

x.Size = New Size(1024, 720)

End Sub

Private Sub ExitToolStripMenuItem1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)

Me.Close()

End Sub

Private Sub cliniclogin\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load

End Sub

End Class

**SOURCE CODE FOR FORM PATIENT DETAILS**

Imports System.Windows.Forms

Imports System

Imports Microsoft.VisualStudio.TestTools.UnitTesting

Imports clinic

'''<summary>

'''This is a test class for frmpatientTest and is intended

'''to contain all frmpatientTest Unit Tests

'''</summary>

<TestClass()> \_

Public Class frmpatientTest

Private testContextInstance As TestContext

'''<summary>

'''Gets or sets the test context which provides

'''information about and functionality for the current test run.

'''</summary>

Public Property TestContext() As TestContext

Get

Return testContextInstance

End Get

Set(ByVal value As TestContext)

testContextInstance = Value

End Set

End Property

#Region "Additional test attributes"

'

'You can use the following additional attributes as you write your tests:

'

'Use ClassInitialize to run code before running the first test in the class

'<ClassInitialize()> \_

'Public Shared Sub MyClassInitialize(ByVal testContext As TestContext)

'End Sub

'

'Use ClassCleanup to run code after all tests in a class have run

'<ClassCleanup()> \_

'Public Shared Sub MyClassCleanup()

'End Sub

'

'Use TestInitialize to run code before running each test

'<TestInitialize()> \_

'Public Sub MyTestInitialize()

'End Sub

'

'Use TestCleanup to run code after each test has run

'<TestCleanup()> \_

'Public Sub MyTestCleanup()

'End Sub

'

#End Region

'''<summary>

'''A test for txtphone

'''</summary>

<TestMethod()> \_

Public Sub txtphoneTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As TextBox = Nothing ' TODO: Initialize to an appropriate value

Dim actual As TextBox

target.txtphone = expected

actual = target.txtphone

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for txtname

'''</summary>

<TestMethod()> \_

Public Sub txtnameTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As TextBox = Nothing ' TODO: Initialize to an appropriate value

Dim actual As TextBox

target.txtname = expected

actual = target.txtname

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for txtid

'''</summary>

<TestMethod()> \_

Public Sub txtidTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As TextBox = Nothing ' TODO: Initialize to an appropriate value

Dim actual As TextBox

target.txtid = expected

actual = target.txtid

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for txtcontactno

'''</summary>

<TestMethod()> \_

Public Sub txtcontactnoTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As TextBox = Nothing ' TODO: Initialize to an appropriate value

Dim actual As TextBox

target.txtcontactno = expected

actual = target.txtcontactno

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for txtaddress

'''</summary>

<TestMethod()> \_

Public Sub txtaddressTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.txtaddress = expected

actual = target.txtaddress

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for txtadd

'''</summary>

<TestMethod()> \_

Public Sub txtaddTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As TextBox = Nothing ' TODO: Initialize to an appropriate value

Dim actual As TextBox

target.txtadd = expected

actual = target.txtadd

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for rdbmale

'''</summary>

<TestMethod()> \_

Public Sub rdbmaleTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As RadioButton = Nothing ' TODO: Initialize to an appropriate value

Dim actual As RadioButton

target.rdbmale = expected

actual = target.rdbmale

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for rdbfemale

'''</summary>

<TestMethod()> \_

Public Sub rdbfemaleTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As RadioButton = Nothing ' TODO: Initialize to an appropriate value

Dim actual As RadioButton

target.rdbfemale = expected

actual = target.rdbfemale

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for lblphone

'''</summary>

<TestMethod()> \_

Public Sub lblphoneTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.lblphone = expected

actual = target.lblphone

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for lblname

'''</summary>

<TestMethod()> \_

Public Sub lblnameTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.lblname = expected

actual = target.lblname

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for lblid

'''</summary>

<TestMethod()> \_

Public Sub lblidTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.lblid = expected

actual = target.lblid

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for lbldob

'''</summary>

<TestMethod()> \_

Public Sub lbldobTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.lbldob = expected

actual = target.lbldob

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for Label8

'''</summary>

<TestMethod()> \_

Public Sub Label8Test()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.Label8 = expected

actual = target.Label8

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for Label7

'''</summary>

<TestMethod()> \_

Public Sub Label7Test()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.Label7 = expected

actual = target.Label7

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for Label2

'''</summary>

<TestMethod()> \_

Public Sub Label2Test()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.Label2 = expected

actual = target.Label2

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for Label1

'''</summary>

<TestMethod()> \_

Public Sub Label1Test()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Label = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Label

target.Label1 = expected

actual = target.Label1

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for dtpdor

'''</summary>

<TestMethod()> \_

Public Sub dtpdorTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As DateTimePicker = Nothing ' TODO: Initialize to an appropriate value

Dim actual As DateTimePicker

target.dtpdor = expected

actual = target.dtpdor

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for dtpdob

'''</summary>

<TestMethod()> \_

Public Sub dtpdobTest()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As DateTimePicker = Nothing ' TODO: Initialize to an appropriate value

Dim actual As DateTimePicker

target.dtpdob = expected

actual = target.dtpdob

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for Button2

'''</summary>

<TestMethod()> \_

Public Sub Button2Test()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Button = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Button

target.Button2 = expected

actual = target.Button2

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for Button1

'''</summary>

<TestMethod()> \_

Public Sub Button1Test()

Dim target As frmpatient = New frmpatient ' TODO: Initialize to an appropriate value

Dim expected As Button = Nothing ' TODO: Initialize to an appropriate value

Dim actual As Button

target.Button1 = expected

actual = target.Button1

Assert.AreEqual(expected, actual)

Assert.Inconclusive("Verify the correctness of this test method.")

End Sub

'''<summary>

'''A test for InitializeComponent

'''</summary>

<TestMethod(), \_

DeploymentItem("clinic.exe")> \_

Public Sub InitializeComponentTest()

Dim target As frmpatient\_Accessor = New frmpatient\_Accessor ' TODO: Initialize to an appropriate value

target.InitializeComponent()

Assert.Inconclusive("A method that does not return a value cannot be verified.")

End Sub

'''<summary>

'''A test for frmpatient\_Load

'''</summary>

<TestMethod(), \_

DeploymentItem("clinic.exe")> \_

Public Sub frmpatient\_LoadTest()

Dim target As frmpatient\_Accessor = New frmpatient\_Accessor ' TODO: Initialize to an appropriate value

Dim sender As Object = Nothing ' TODO: Initialize to an appropriate value

Dim e As EventArgs = Nothing ' TODO: Initialize to an appropriate value

target.frmpatient\_Load(sender, e)

Assert.Inconclusive("A method that does not return a value cannot be verified.")

End Sub

'''<summary>

'''A test for Dispose

'''</summary>

<TestMethod(), \_

DeploymentItem("clinic.exe")> \_

Public Sub DisposeTest()

Dim target As frmpatient\_Accessor = New frmpatient\_Accessor ' TODO: Initialize to an appropriate value

Dim disposing As Boolean = False ' TODO: Initialize to an appropriate value

target.Dispose(disposing)

Assert.Inconclusive("A method that does not return a value cannot be verified.")

End Sub

'''<summary>

'''A test for Button2\_Click

'''</summary>

<TestMethod(), \_

DeploymentItem("clinic.exe")> \_

Public Sub Button2\_ClickTest()

Dim target As frmpatient\_Accessor = New frmpatient\_Accessor ' TODO: Initialize to an appropriate value

Dim sender As Object = Nothing ' TODO: Initialize to an appropriate value

Dim e As EventArgs = Nothing ' TODO: Initialize to an appropriate value

target.Button2\_Click(sender, e)

Assert.Inconclusive("A method that does not return a value cannot be verified.")

End Sub

'''<summary>

'''A test for Button1\_Click

'''</summary>

<TestMethod(), \_

DeploymentItem("clinic.exe")> \_

Public Sub Button1\_ClickTest()

Dim target As frmpatient\_Accessor = New frmpatient\_Accessor ' TODO: Initialize to an appropriate value

Dim sender As Object = Nothing ' TODO: Initialize to an appropriate value

Dim e As EventArgs = Nothing ' TODO: Initialize to an appropriate value

target.Button1\_Click(sender, e)

Assert.Inconclusive("A method that does not return a value cannot be verified.")

End Sub

'''<summary>

'''A test for frmpatient Constructor

'''</summary>

<TestMethod()> \_

Public Sub frmpatientConstructorTest()

Dim target As frmpatient = New frmpatient

Assert.Inconclusive("TODO: Implement code to verify target")

End Sub

End Class

**SOURCE CODE FOR FORM DOCTOR DETAILS**

Imports System.Data

Imports System.Data.SqlClient

Public Class doctor

Dim con As New SqlConnection(mycs.getconnection())

Private Sub btnenter\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnenter.Click

Try

If (txtid.TextLength <= 0) Then

MsgBox("id can not be blank")

Return

ElseIf (txtname.TextLength <= 0) Then

MsgBox("name can not be blank")

Return

ElseIf (txtaddress.TextLength <= 0) Then

MsgBox("Address can not be blank")

Return

ElseIf (txtphone.TextLength <= 0) Then

MsgBox(" Phone no. can not be blank")

Return

ElseIf Not IsNumeric(txtphone.Text) Then

MsgBox("Mobile No Should be numeric no")

Return

ElseIf (txtphone.TextLength <> 10) Then

MsgBox("Mobile Should be 10 digit Numeric no.")

Return

End If

Dim cmd As New SqlCommand

cmd.Connection = con

cmd.CommandType = CommandType.Text

cmd.CommandText = "Insert into tbldoctor values(@a,@b,@c,@d)"

cmd.Parameters.Add("@a", SqlDbType.Int).Value = txtid.Text

cmd.Parameters.Add("@b", SqlDbType.VarChar).Value = txtname.Text

cmd.Parameters.Add("@c", SqlDbType.VarChar).Value = txtaddress.Text

cmd.Parameters.Add("@d", SqlDbType.VarChar).Value = txtphone.Text

con.Open()

Dim i As Integer = cmd.ExecuteNonQuery

MsgBox("Record Entered.")

txtid.Clear()

txtname.Clear()

txtaddress.Clear()

txtphone.Clear()

Catch ex As SqlException

MsgBox(ex.Number & " " & ex.Message)

Finally

con.Close()

End Try

End Sub

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

Me.Close()

End Sub

**SOURCE CODE FOR FORM MEDICINE DETAILS**

Imports System.Data

Imports System.Data.SqlClient

Public Class frmmedicine

Dim con As New SqlConnection(mycs.getconnection())

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Try

If (txtmname.TextLength <= 0) Then

MsgBox("Madicine name can not be blank")

Return

ElseIf (txtmcost.TextLength <= 0) Then

MsgBox("Cost can not be entered.")

ElseIf (txtdosage.TextLength <= 10) Then

MsgBox("Dosage and methods can not be blank.")

Return

End If

Dim cmd As New SqlCommand

cmd.Connection = con

cmd.CommandType = CommandType.Text

cmd.CommandText = "Insert into medical values(@a,@b,@c,@d,@e,@f)"

cmd.Parameters.Add("@a", SqlDbType.VarChar).Value = txtmname.Text

cmd.Parameters.Add("@b", SqlDbType.Money).Value = txtmcost.Text

cmd.Parameters.Add("@c", SqlDbType.DateTime).Value = dtpdespensed.Text

cmd.Parameters.Add("@d", SqlDbType.DateTime).Value = dtpexpirey.Value

cmd.Parameters.Add("@e", SqlDbType.VarChar).Value = txtdosage.Text

cmd.Parameters.Add("@f", SqlDbType.VarChar).Value = txtspecial.Text

con.Open()

Dim i As Integer = cmd.ExecuteNonQuery

MsgBox("Record Entered.")

txtmname.Clear()

txtmcost.Clear()

txtdosage.Clear()

txtspecial.Clear()

Catch ex As SqlException

MsgBox(ex.Number & " " & ex.Message)

Finally

con.Close()

End Try

End Sub

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

Me.Close()

End Sub

End Class

**SOURCE CODE FOR FORM STAFF DETAILS**

Imports System.Data

Imports System.Data.SqlClient

Public Class staffdetails

Dim con As New SqlConnection(mycs.getconnection())

Private Sub btnenter\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnenter.Click

Try

If (txtename.TextLength <= 0) Then

MsgBox("Name can not be blank")

Return

ElseIf Not IsNumeric(txtphone.Text) Then

MsgBox("Mobile No Should be numeric no")

Return

ElseIf (txtphone.TextLength <> 10) Then

MsgBox("Mobile Should be 10 digit Numeric no.")

Return

End If

Dim cmd As New SqlCommand

cmd.Connection = con

cmd.CommandType = CommandType.Text

cmd.CommandText = "Insert into staff values(@a,@b,@c,@d,@e,@f,@g)"

cmd.Parameters.Add("@a", SqlDbType.VarChar).Value = txtename.Text

cmd.Parameters.Add("@b", SqlDbType.VarChar).Value = txtaddress.Text

cmd.Parameters.Add("@c", SqlDbType.VarChar).Value = txtpost.Text

cmd.Parameters.Add("@d", SqlDbType.Money).Value = txtsalary.Text

cmd.Parameters.Add("@e", SqlDbType.VarChar).Value = txtphone.Text

cmd.Parameters.Add("@f", SqlDbType.VarChar).Value = cmbtime.Text

cmd.Parameters.Add("@g", SqlDbType.Date).Value = dtpjoin.Value

con.Open()

Dim i As Integer = cmd.ExecuteNonQuery

MsgBox("Record Inserted.")

txtename.Clear()

txtaddress.Clear()

txtpost.Clear()

txtsalary.Clear()

txtphone.Clear()

Catch ex As SqlException

MsgBox(ex.Number & " " & ex.Message)

Finally

con.Close()

End Try

End Sub

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

Me.Close()

End Sub

End Class

**SOURCE CODE FOR FORM BED DETAILS**

Imports System.Data

Imports System.Data.SqlClient

Public Class Bed

Dim con As New SqlConnection(mycs.getconnection())

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Try

If (txtid.TextLength <= 0) Then

MsgBox("id can not be blank")

Return

ElseIf (txtname.TextLength <= 0) Then

MsgBox("name can not be blank")

Return

ElseIf (txtrate.TextLength <= 0) Then

MsgBox("Bed price can not be blank")

Return

ElseIf (txttype.TextLength <= 0) Then

MsgBox(" Phone no. can not be blank")

Return

End If

Dim cmd As New SqlCommand

cmd.Connection = con

cmd.CommandType = CommandType.Text

cmd.CommandText = "Insert into tblbed values(@a,@b,@c,@d)"

cmd.Parameters.Add("@a", SqlDbType.Int).Value = txtid.Text

cmd.Parameters.Add("@b", SqlDbType.VarChar).Value = txtname.Text

cmd.Parameters.Add("@c", SqlDbType.Money).Value = txtrate.Text

cmd.Parameters.Add("@d", SqlDbType.VarChar).Value = txttype.Text

con.Open()

Dim i As Integer = cmd.ExecuteNonQuery

MsgBox("Record Entered.")

txtid.Clear()

txtname.Clear()

txtrate.Clear()

txttype.Clear()

Catch ex As SqlException

MsgBox(ex.Number & " " & ex.Message)

Finally

con.Close()

End Try

End Sub

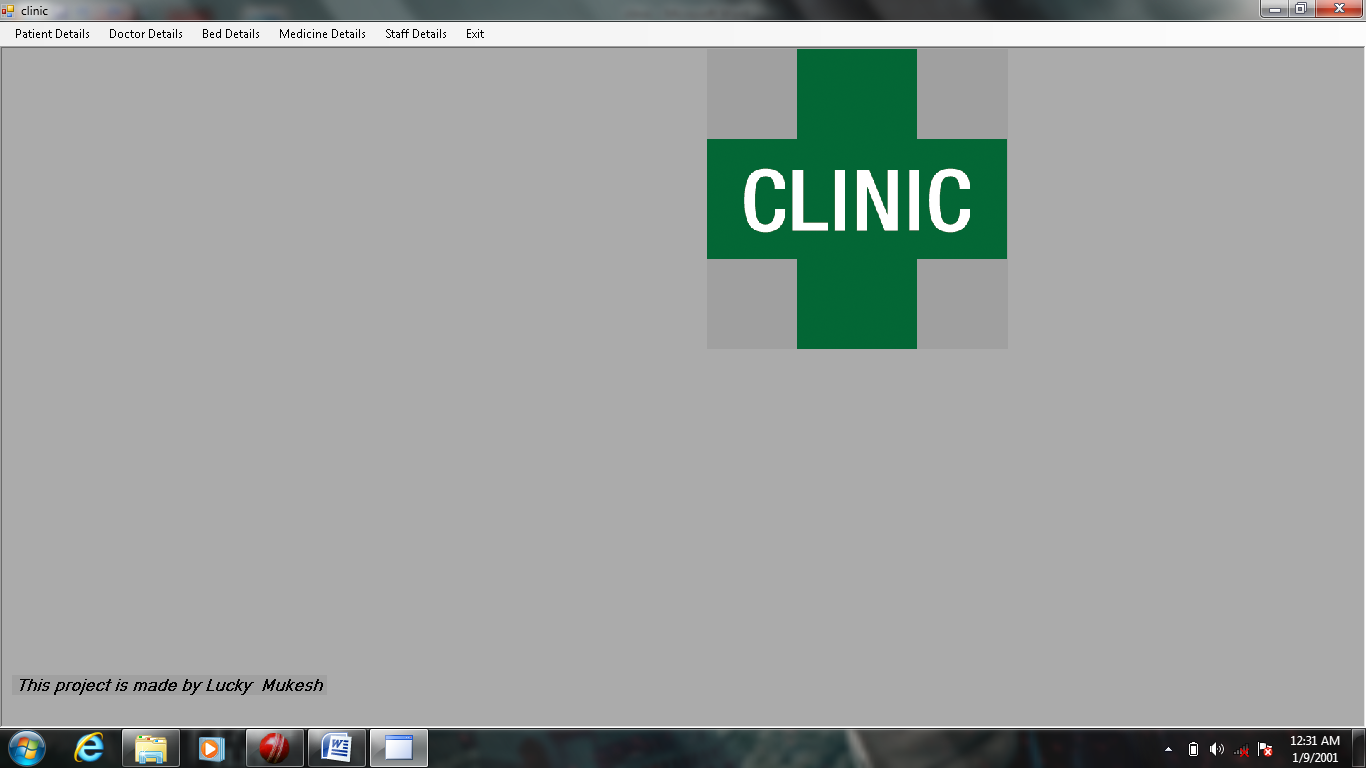
Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

Me.Close()

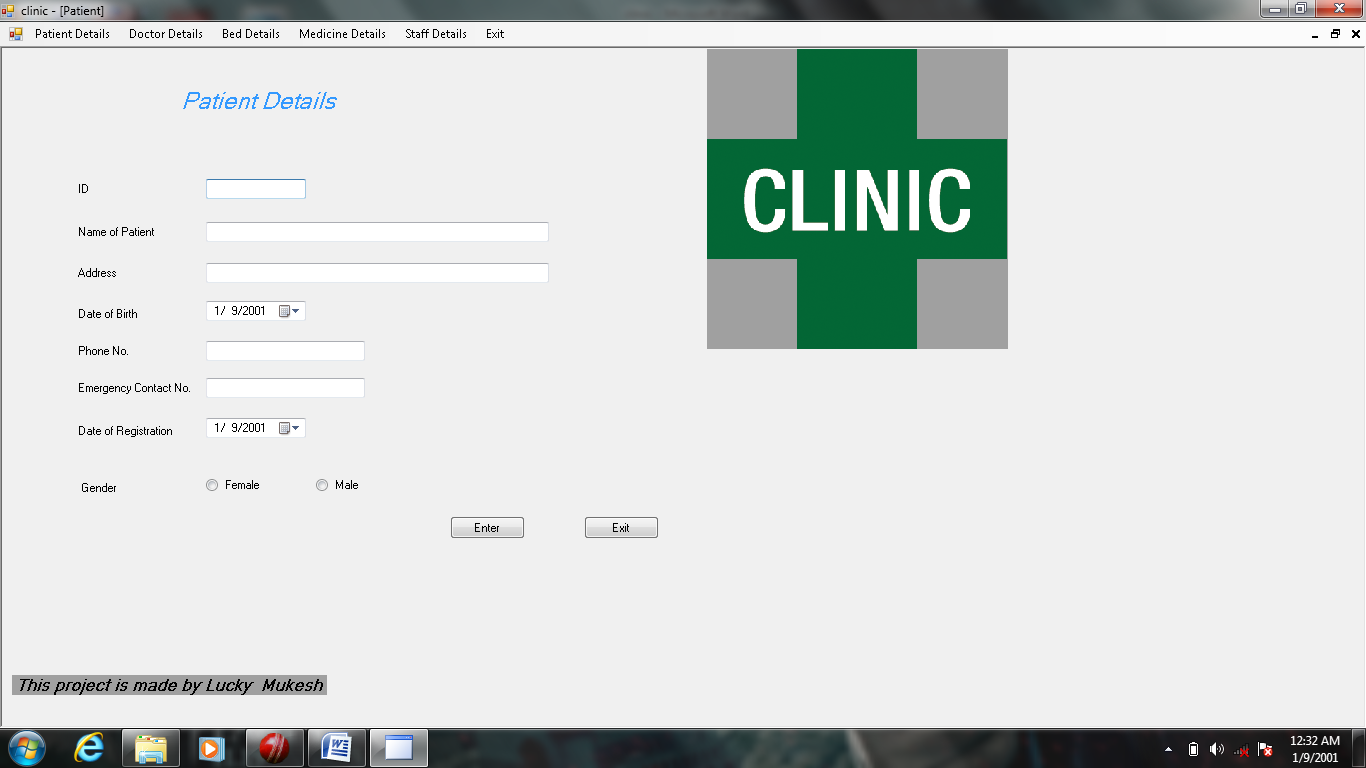
End Sub

**VIEW**

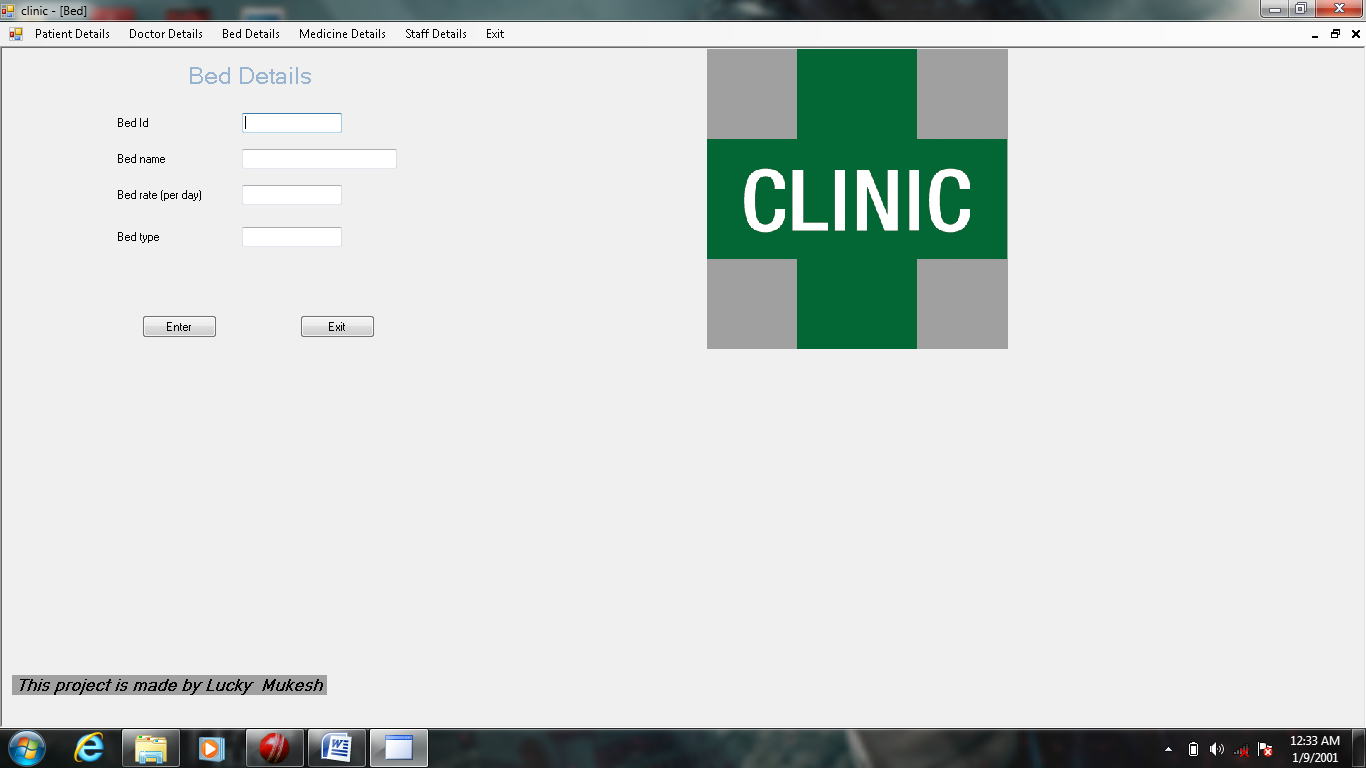
**Form clinic view:-**

****

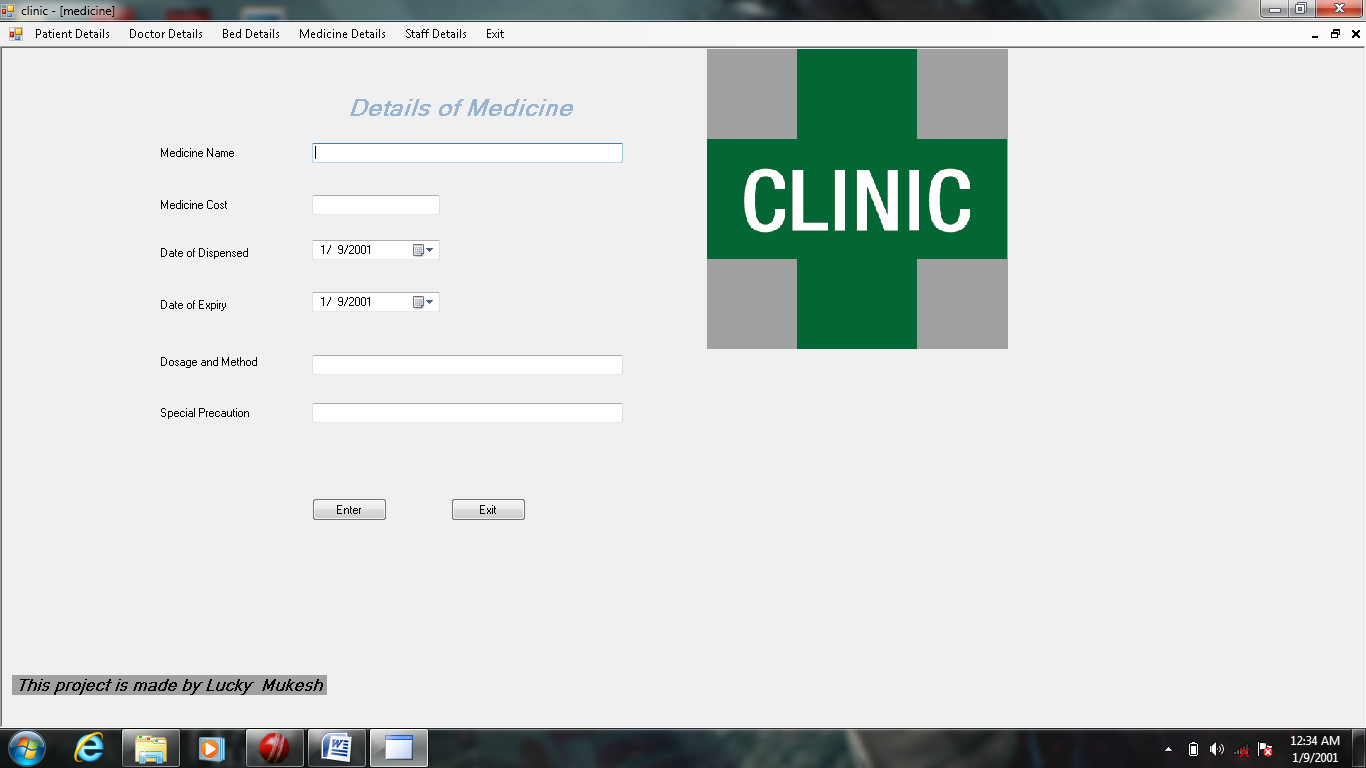
**Form patient view:-**

****

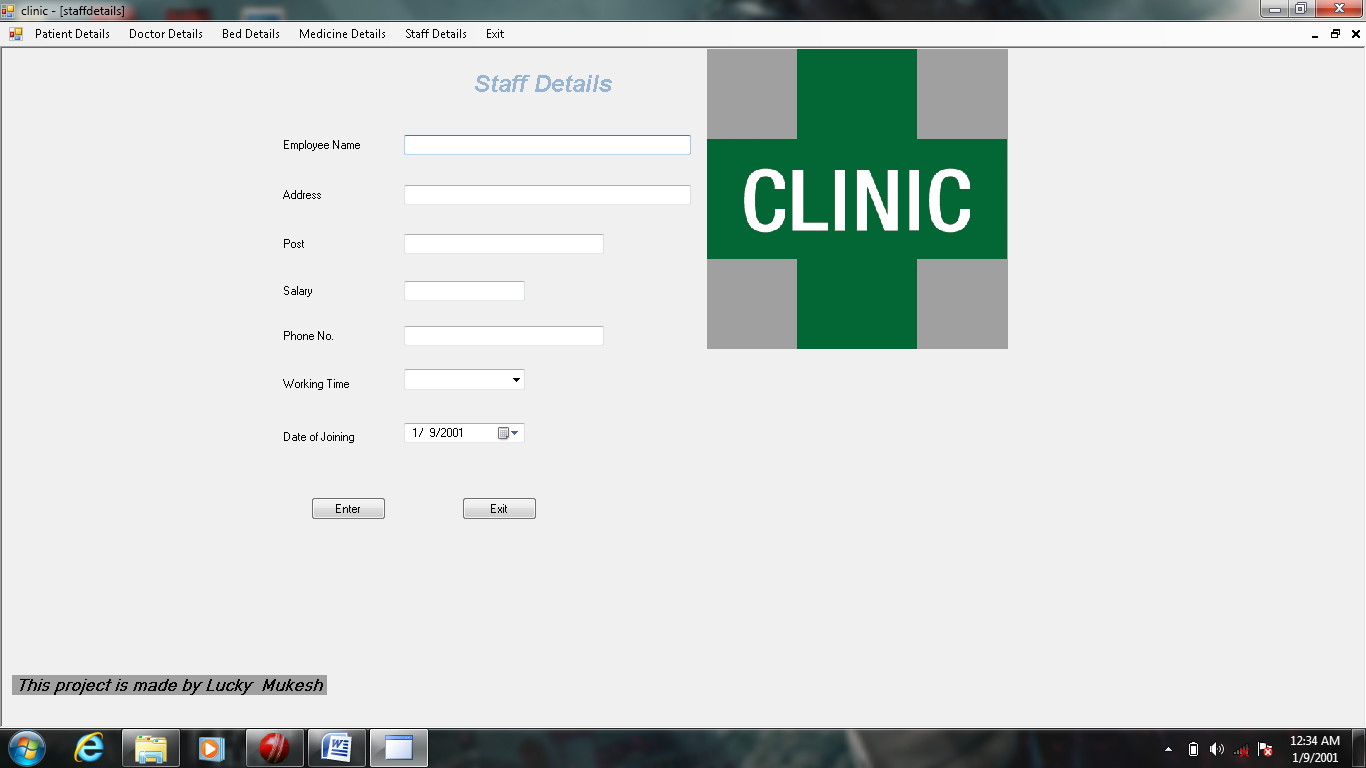
**Form Bed details view:-**

****

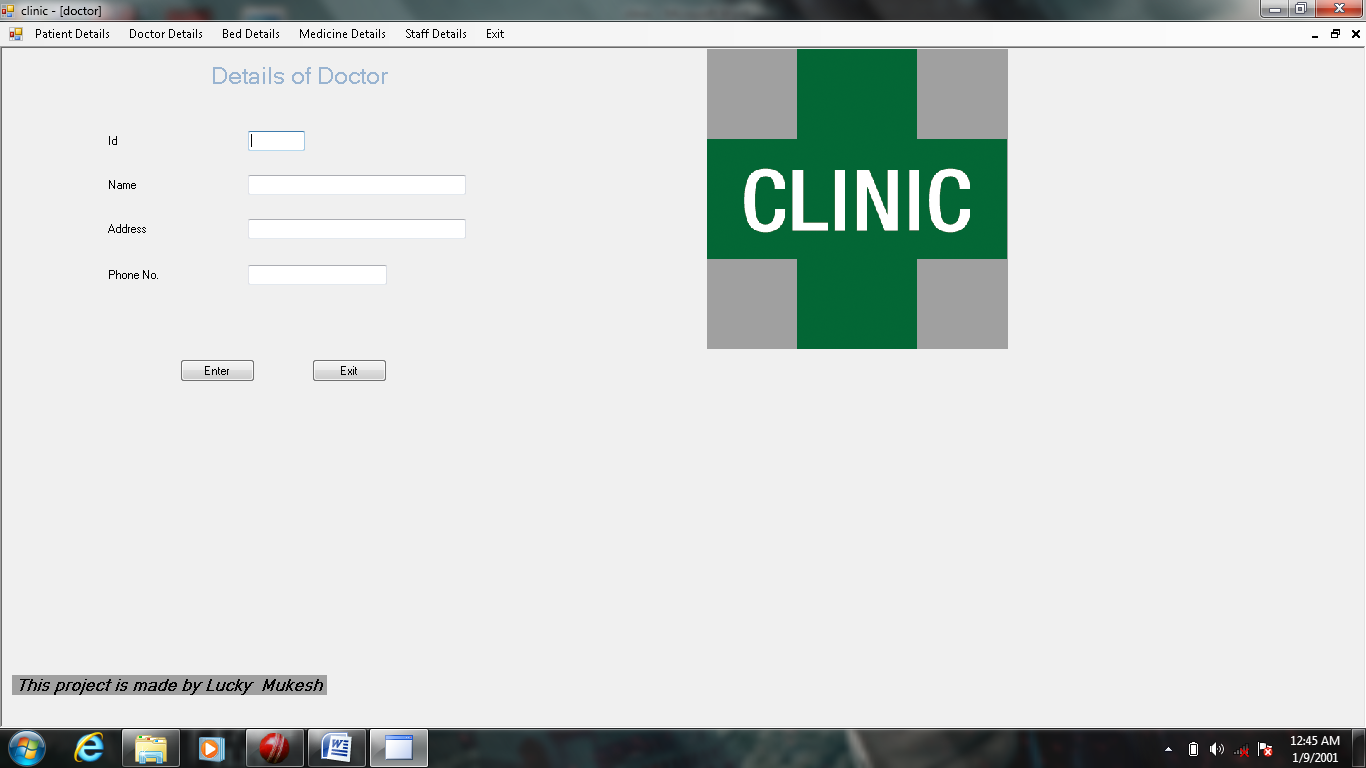
**Form Medicine details view:-**

****

**Form staff detail view:-**

****

**Form Doctor form view:-**

****

**BIBLIGRAPHY**

* A to Z visual basic
* Mastering visual basic 2008
* Computer science visual basic
* Programming with visual basic 2008
* Introduction to programming using visual basic