CONTENTS

1:Introduction

1.1 Summary

Project Title: Medical Store Management

Project Concept: Industrial project(Sales management, generation of reports and bills)

Type of Project: Desktop/Windows Application

My Medical Store Management System t is very helpful to manage sales information of Medical Store.It can easily keep the record of Hospitals who done regular business deals.This software is comprehensiveness and the simple by which it handles complex tasks easily.

1.2 purpose

The main purpose of the system is to make the sales information manage simply and effectively.customers and companies' management and transactions are entered on computers and saved on computers and can be accessed as in the form of the reports and can be updated very easily on computers.

It is quite difficult to maintain the stoke of the medicines in the store and reflect it with the database. So, the owner can place order for new Medicines And replace expired medicines.

This software can do that very easily it also provides many facilities.

The attention to the objectives:

1. To automate the manual system
2. To make the system User-friendly a simple ,A just English known person can easily handle the system
3. Proper Validation technique to be used
4. Reports should be maintained by user choice
5. Report& Bill Printing Facilities

1.3 scope

This software can be used by the small medical stores to maintain daily stoke of Medicine for the small unit.

1.4 Technologies and Literature Review

Front End – C#.Net 2008

C#, the new language introduced in the .NET Framework. However, C# is a modern, objected-oriented (from the ground up) type-safe language.

Language features

The following sections take a quick look at some of the features of the C# language.

Classes

All code and data in C# must be enclosed in a class. You can't define a variable outside of a class, and you can't write any code that's not in a class. Classes can have constructors, which execute when an object of the class is created, and a destructor, which executes when an object of the class is destroyed. Classes support single inheritance, and all classes ultimately derive from a base class called object. C# supports versioning techniques to help your classes evolve over time while maintaining compatibility with code that uses earlier versions of your classes.

Data types

C# lets you work with two types of data: value types and reference types. Value types hold actual values. Reference types hold references to values stored elsewhere in memory.
Primitive types such as char, int and float, as well as enumerated values and structures, are value types. Reference types hold variables that deal with objects and arrays. C# comes with predefined reference types (object and string), as well as predefined value types (sbyte, short, int, long, byte, ushort, uint, ulong, float, double, bool, char, and decimal). You can also define your own value and reference types in your code. All value and reference types ultimately derive from a base type called object. C# allows you to convert a value of one type into a value of another type. You can work with both implicit conversions and explicit conversions. Implicit conversions always succeed and don't lose any information. Explicit conversions may cause you to lose data. You must write a cast operator into your code to make an explicit conversion happen.

Variables

Variables can be defined as constants. Constants have values that cannot change during the execution of your code. C# provides a built-in mechanism for defining and handling events. If you write a class that performs a lengthy operation, you may want to invoke an event when the operation is completed. Clients can subscribe to that event and catch the event in their code, which enables them to be notified when you have completed your lengthy operation. The event handling mechanism in C# uses delegates, which are variables that reference a function.
Note An event handler is a procedure in your code that determines the actions to be
performed when an event occurs, such as the user clicking a button. If your class holds a set of values, clients may want to access the values as if your class were an array. You can write a piece of code called an indexer to enable your class to be accessed
as if it were an array.

Interfaces

C# supports interfaces, which are groups of properties, methods, and events that specify a set of functionality. C# classes can implement interfaces, which tell users that the class supports the set of functionality documented by the interface. You can develop implementations of interfaces without interfering with any existing code, which minimizes compatibility problems. Once an interface has been published, it cannot be changed, but it can evolve through inheritance. C# classes can implement many interfaces, although the classes can only inherit from a single base class.

Attributes

Attributes declare additional information about your class to the CLR. Attributes can also be used to bind runtime information to a class, defining how it should act when used. The possibilities are endless, which is why Microsoft includes many predefined attributes within the .NET Framework.

Back End :-SQL Server

DATABASE

A database management, or DBMS, gives the user access to their data and helps them transform the data into information. Such database management systems include dBase, paradox, IMS, SQL Server and SQL Server. These systems allow users to create, update and extract information from their database.

A database is a structured collection of data. Data refers to the characteristics of people, things and events. SQL Server stores each data item in its own fields. In SQL Server, the fields relating to a particular person, thing or event are bundled together to form a single complete unit of data, called a record (it can also be referred to as raw or an occurrence). Each record is made up of a number of fields. No two fields in a record can have the same field name.

During an SQL Server Database design project, the analysis of your business needs identifies all the fields or attributes of interest. If your business needs change over time, you define any additional fields or change the definition of existing fields.

SQL Server Tables

SQL Server stores records relating to each other in a table. Different tables are created for the various groups of information. Related tables are grouped together to form a database.

Primary Key

Every table in SQL Server has a field or a combination of fields that uniquely identifies each record in the table. The Unique identifier is called the Primary Key, or simply the Key. The primary key provides the means to distinguish one record from all other in a table. It allows the user and the database system to identify, locate and refer to one particular record in the database.

Relational Database

Sometimes all the information of interest to a business operation can be stored in one table. SQL Server makes it very easy to link the data in multiple tables. Matching an employee to the department in which they work is one example. This is what makes SQL Server a relational database management system, or RDBMS. It stores data in two or more tables and enables you to define relationships between the table and enables you to define relationships between the tables.

Foreign Key

When a field is one table matches the primary key of another field is referred to as a foreign key. A foreign key is a field or a group of fields in one table whose values match those of the primary key of another table.

A major purpose of a database system is to provide users with an abstract view of the data. This system hides certain details of how the data is stored and maintained. Data abstraction is divided into three levels.

Physical level: This is the lowest level of abstraction at which one describes how the data are actually stored.
Conceptual Level: At this level of database abstraction all the attributed and what data are actually stored is described and entries and relationship among them.
View level: This is the highest level of abstraction at which one describes only part of the database.

Advantages of RDBMS

- Redundancy can be avoided
- Inconsistency can be eliminated
- Data can be Shared
- Standards can be enforced
- Security restrictions can be applied
- Integrity can be maintained
- Conflicting requirements can be balanced
- Data independence can be achieved.

Disadvantages of DBMS

A significant disadvantage of the DBMS system is cost. In addition to the cost of purchasing of developing the software, the hardware has to be upgraded to allow for the extensive programs and the workspace required for their execution and storage. While centralization reduces duplication, the lack of duplication requires that the database be adequately backed up so that in case of failure the data can be recovered.

FEATURES OF SQL SERVER (RDBMS)

SQL SERVER is one of the leading database management systems (DBMS) because it is the only Database that meets the uncompromising requirements of today's most demanding information systems. From complex decision support systems (DSS) to the most rigorous online transaction processing (OLTP) application, even application that require simultaneous DSS and OLTP access to the same critical data, SQL Server leads the industry in both performance and capabilitySQL SERVER is a truly portable, distributed, and open DBMS that delivers unmatched performance, continuous operation and support for every database.

The row level lock manager

The unrivaled portability and connectivity of the SQL SERVER DBMS enables all the systems in the organization to be linked into a singular, integrated computing resource.
Portability

SQL SERVER is fully portable to more than 80 distinct hardware and operating systems platforms, including UNIX, MSDOS, OS/2, Macintosh and dozens of proprietary platforms. This portability gives complete freedom to choose the database server platform that meets the system requirements.

Distributed Data Sharing

SQL Server's networking and distributed database capabilities to access data stored on remote server with the same ease as if the information was stored on a single local computer. A single SQL statement can access data at multiple sites. You can store data where system requirements such as performance, security or availability dictate.
Unmatched PerformanceThe most advanced architecture in the industry allows the SQL SERVER DBMS to deliver unmatched performance.

No I/O Bottlenecks

SQL Server's fast commit groups commit and deferred write technologies dramatically reduce disk I/O bottlenecks. While some database write whole data block to disk at commit time, SQL Server commits transactions with at most sequential log file on disk at commit time, On high throughput systems, one sequential writes typically group commit multiple transactions. Data read by the transaction remains as shared memory so that other transactions may access that data without reading it again from disk. Since fast commits write all data necessary to the recovery to the log file, modified blocks are written back to the database independently of the transaction commit, when written from memory to disk.

2. projectmanagement

2.1 Project Planning and scheduling

2.1.1 Project Development Approach

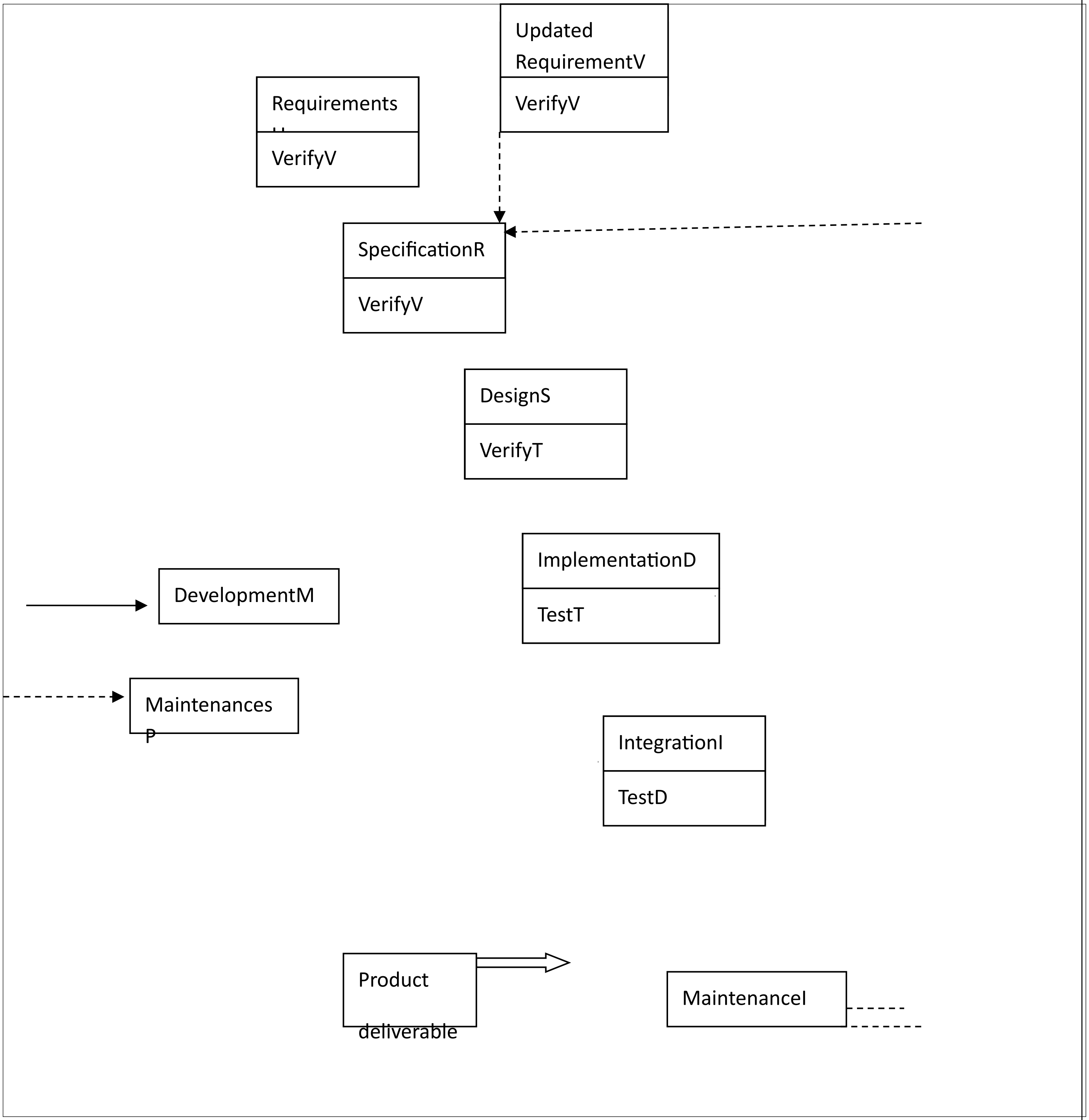
To solve actual problems in an industry setting, a software development strategy must be incorporated that encompasses the process, methods and tools for software engineering. This strategy is often referred to as **software process model** and **software engineering paradigm.** Asoftware process model for software engineering is chosen based on the nature of project and application, the methods and tool to be used and the controls and deliverables that are required.

For the development and implementation of windows based module several distinct approaches are in practice occurs. Among them, a very popular one is the classical **system development life cycle model (SDLC)** orthe **waterfall model. The waterfall model** has following phase of its development:

1) System/Information Engineering and modeling
2) Software Requirement Analysis
3) System Analysis and Design
4) Code Generation
5) Testing and Maintenance

- **System Information Engineering and Modeling:**

As software is always of a large system (or business), work begins by establishing requirements for all system elements and then allocating some subset of these requirements to software. This system view is essential when software must interface with other elements such as adware, people and other resources. System is the basic and very critical requirement for the existence of software in any entity. So if the system is not in place, the system should be engineering and put in place. In some cases to extract the maximum output, system should be re-engineered and spices up.



Interactive Water fall Model

- **Software Requirement Analysis**

This is also known as feasibility study. In this phase, the development team visits the need for possible software automation in the given system. By the end of the feasibility study, the team furnishes a document that holds the different specific recommendations for the candidate system.

- **System Analysis and Design**

In this phase, the software's overall structure and its nuances are defined. In terms of the end user/server technology, the number of tiers needed for the package architecture, the database design, the data structure design etc are all defined in this phase. Any glitch in the design phase could be very expensive to solve in the later stage of the software development. The logical system of the product is developed in this phase.

- **Code Generation**

The design must be translated into a machine-readable form. The code generation step performs this task. If design is performed in detailed manner, code generation can be accomplished without much complication. Programming tools like Compilers, Interpreters, and Debuggers are used to generate the code. Different high level programming languages like C, C++, Pascal, Java, ASP.NET and VB.NET are used for coding. Here I have used c# .NET for the implementation.

- **Testing and Maintenance**

Once the code is generated, the program testing begins. Different testing methodologies are available to unravel the bugs that were committed during the previous phases. Different testing tools and methodologies are already available.

Software will definitely undergo change once it is delivered to the end user. 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. The software should be developed to accommodate changes that could happen during the implementation period.

2.1.2 Project Plan

A plan is drawn up at the start of the project, should be used as the driver of the project. The project planning consists of:

- Selection of suitable software development process model which I have selected Interactive Water Fall Model.
- Risk Management Plan, which involves the risk identification and risk assessments.
- Project Scheduling, which involves the tasks and duration required for performing tasks. This is described by task representation and the Timeline chart representation.
- Cost and Effort estimation, which involves estimation of cost as well as effort applied by the developers.

- **Milestones and Deliverables**

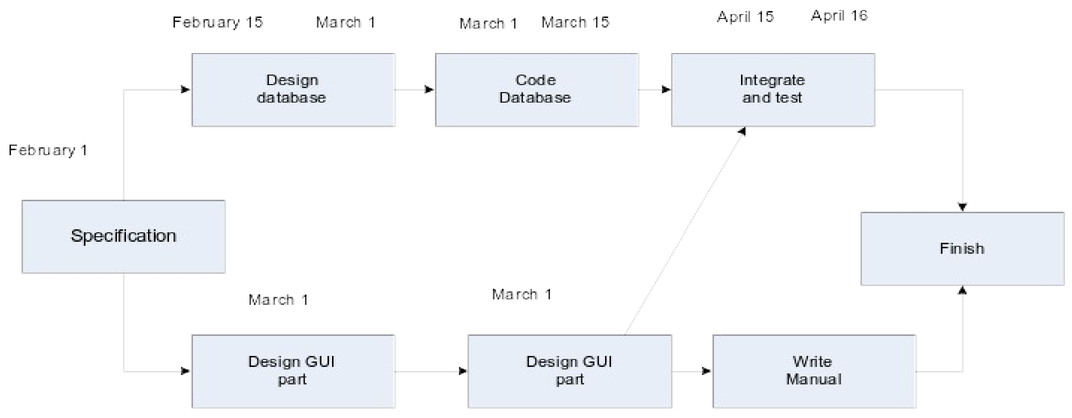
Management needs information. As software is intangible, this information can only be provided as a document that describes the state of the software being developed. Without this information, it is impossible to judge progress and cost estimates and schedules cannot be updated.

When planning a project series of milestones are established.

- **Milestones:**
    - Milestone is an end-point of the software process activity.
    - At each milestone there should be formal output, such as report, that can be represented to the management.
    - Milestone report need not be large document; they are the short report of achievements in software project activity.
    - Milestone represents the end of the distinct, logical stage in the project.

- **Deliverables:**
- Deliverable is a project report that is delivered to customer.
    - Deliverables are delivered to the customer at the end of the same major project phase such as specification, design, etc.
    - Deliverables are usually milestones.
    - Milestones may be internal project results that are used by the project manager to check progress but which are not delivered to the customer.

Pert Chart Representation:



2.2risk management

2.2.1 Risk Identification

Project Risks

Project Risks threaten the project plan. That is, if project risks become real, it is likely that project schedule will slip and that costs will increase.

Project risks identify potential budgetary, schedule, personnel (staffing and organization), resource, stockholder and requirements problems and their impact on a software project. Project complexity, size and the degree of structural uncertainty were also defined as project risk factors.

Technical Risks

Technical risks threaten the quality and timeliness of the software to produce. If a technical risk becomes a reality implementation may become difficult or impossible. Technical risks identify potential design, implementation, interface, verification, and maintenance problems.

Business Risks

Business risks threaten the visibility of the software to be built. Business risks often jeopardize the project or the product. Candidates for **top fivebusiness risks are,**

- Building an excellent product or system that no one really wants.
- Building a product that no longer fits into the overall business strategy for the company.
- Building a project that the sales force doesn't understand how to sell.
- Losing the support of senior management due to change in focus or a change in people.
- Losing budgetary or personnel commitment.

2.2.2 Risk Analysis

Our project is threat to following known and predictable risks:

Effectiveness

This is one of the major risks because it is not worthwhile if the project developed does not serve for what it is developed. So Effectiveness (Usability Risk) is one of major risk involved.

Efficiency

Efficiency is also major risk because the project developed should be efficient to the functionality it provides. So we have to consider this threat also.

Confidentiality

Because author of system should gets access to system according to his/her authorization. So confidentiality Risk is also considerable threat.

Integrity

The application should also threat by Integrity Risk .The data related to the project should be preserve qualities like consistency.

Compliance

Project is threat by this risk because the project should follow specific standards.

Reliability

The application to be developed is also threat by Reliability Risk because the processing done and information should be reliable.

2.2.3 Risk Planning

To assist the project team in developing a strategy for dealing with risk. An effective strategy must consider three issues:

- ➢ Risk avoidance
- ➢ Risk Monitoring
- ➢ Risk Management

Risk Mitigating

- Meet with current staff to determine causes for turnover.
- Mitigate those causes that are under our control before the project starts.
- Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave.
- Organize project teams so that information about each development activity is widely dispersed.
- Define documentation standards and establish mechanisms to ensure that documents are developed in a timely manner.
- Assign a backup staff member for every critical technologist.

Risk Monitoring

- General attitude of team members based on project pressures.
- The degree to which the team has jelled.
- Interpersonal relationships among team members.
- Potential problems with compensation and benefits.
- The availability of jobs within the company and outside it.

RMMM-Risk Management Plan

A risk management strategy can be included in the software project plan or the risk management steps can be organized into a separate Risk Mitigation, Monitoring and Management Plan. The RMMM plan documents all performed as part of risk analysis and are used by project manager as part of the overall project plan.

2.3Estimation

2.3.1 EFFORT ESTIMATON

Analysis

We complete this analysis after gathering all requirements about this topic and preparing diagrams like Entity Relationship, Use case, Context, Data Flow within one month

Database design and coding

We had worked on creating designing first by on paper drawing and then make it on system within few days and then I had done coding for it.

Form Designing

First we have designed splash form and other admin related forms then I have designed reports that are related to all information of companies, replacement orders, and validation of sold part in the system.

Testing

From last 10 to 15 days we had completed testing.

2.3.2cost analysis

There are mainly two types of costs.

1. Direct cost
2. Indirect cost

1. Direct cost:

In direct cost, cost of the software's and toll are included. In our project we use Visual Studio 2005 and SQL server 2005. The prices of all of this are shown below.

Microsoft Visual Studio 2008 – 1000$

SQL server 2005 – 400$

2. Indirect cost

In indirect cost, cost of man power is included for requirement analysis, project development and training of the project given to the user.

For requirement analysis we spend 20 days of time in starting of project.

For development we spend 5 hours per day for 1 months.

And for training of the project we will provide 1 week of one trainer to the user.

So the cost of all this are also included in the project.

3. SYSTEM REQUIREMENT STUDY

3.1 USER CHARACTERISTICS

The user characteristics of the user show which kind of user are dealing wit the system.

Administrator is the person who is taking care of whole the organization system. The admin is having rights to decide whether to allow the insertion, deletion, modification etc performed on details of customers. The admin can also apply changes in rent as well as area setting as per situations.

This software provides many facilities to admin like he/she can easily get information of any customer or any staff details just put their and select IDs in the combo box. By just one clicking the admin can get whole information about the pending money list and model no wise, name wise, date wise report generation is also main task of the system.

Without Login into the system no one can be change or modify the database that is set by the owner of the project.

Thus all the activities which are necessary for the proper functioning of system are performed by this user called **OWNER** who had made this project.

3.2 HARDWARE AND SOFTWARE CONFIGURATION

Software required for development:

✓ Operating System	:	Windows 7
✓ Front End	:	C#.NET
✓ Back End	:	Microsoft SQL Server 2005
✓ Technology	:	Microsoft .net 2005

Hardware requirement for development:

- ✓ Intel i3 (Processor).
- ✓ Memory Ram :4 Ram
- ✓ 10 MB Cache Memory
- ✓ Hard disk : 500GB
- ✓ Microsoft Compatible 101 or more Key Board

3.3 CONSTRAINTS

1. Interfaces to other Applications

Payment management and contact management modules in Property Listing India are very important modules. So it has interface with other modules of Property Listing India. In this system all modules are interrelated with each other .

2. Parallel Operations

In this system updating database for proper management of any event is needed to be carried out parallel. Whenever any user deletes his/her account, all data related to that user will be deleted simultaneously.

3. Higher Order Language Requirements

There is no need of any higher order languagerequirements in this application.

4. Reliabilities Requirement

In this project the reliability requirements. There should be reliability of the database and the data that are entered into the database. Even there is small error in the management of events done can be wrong.

5. Performance Requirement

Here all the events being registered must be managed properly. It must be taken care that no two events should be overlapped means multiple entries about the part or model should not happen.

4. SYSTEM ANALYSIS

The initial analysis is made by knowing the user requirements. In analysis phase, we have analyzed the user's requirement such as:-

Deletion of the medicinewhen stoke is empty. It also keeps the data about medicine which is sold , addition of the record of new medicine which are imported, printing and calculation of the bill, generate reports, search by user choice etc. In this project we have also analyzed that the product or software should not be very costly but its quality and interface must be attractive. If any wrong operation is being performed then the software must invoke the operator accordingly.

The most important phase of developing any system is system analysis. Because of the analysis phase decides that what type of requirements, medicinnes are required.

Analysis of System is the process of gathering facts, solution of problems and to decide over all constitution of the desired system. In System analysis, we have to analyze all the processes, related features, required functions, available sources and the time which should be specified for the analysis stage.

All these things are depend upon our system that what type of outputs of our system or functions, we desired from the system is also responsible for defining the above factors. So the overall structure of system that we want to implement will be decided in analysis part of a system development by analyst.

4.1 STUDY OF CURRENT SYSTEM

In current system we can generate the report & bill. This project is very useful to small medical firms like medical store. It manages all sales &stoke of medicines& deal with management of database.

Where search option is very useful feature to find particular medicines details quickly.

4.2 PROBLEM AND WEEKNESSES OF CURRENT SYSTEM

It is not operable by multiuser. It is access by unique operator who organized by company.

4.3 REQUIREMENT OF NEW SYSTEM

In past work system was fully manual. All type of transactions, accounts details were done by man.

For saving of time and manpower there is a need for the system that performs the entire task that done by a man who is working in industry. The new system that is more secure than the manually working system.In this system there are many customers in the number of hundreds. So, it is quite difficult to manage their requirement in account book.

So there is need of good computer and configuration which fulfill the requirement of this windows application.

4.4FEASIBILITY STUDY

An important outcome of the preliminary investigation is the determination that the system requested is feasible or not. There are three study aspects in the feasibility study portion of the preliminary investigation.

Technical Feasibility:

Technical Feasibility deals with the availability of the required technology for implementing the system means it examines whether the current technical resources or technology is available in the organization or in the current market which is capable of handling the user's requirement. It includes these:

- The system is opened by nature and can be easily expanded in near feature.
- The proposed system has capacity to hold data required.
- The use of reliable Microsoft Access with Server Design and Coding Standards followed guarantees accuracy, reliability, ease of data access and data security.

Operational Feasibility:

Proposed project is beneficial only if it can be turned into information systems that will meet the organization's operating requirements. Operational Feasibility examines whether the proposed system can fit in with existing operations and whether the right information at the right time is provided to the users. It includes these:

- The system is well supported by the service developers with technical guide and tasking special interest in the development process.
- The proposed system makes best efforts to satisfy the requirement of the user keeping in mind certain constraint. Since the most trivial issued assume a major problematic state later in the development cycle. Every possible aspect of Operational Feasibility

is checked.

- The clear advantage of being core gives better advantage.
- The services developers are always ready to test the functionality provided system.

Behavioral Feasibility:

Which have special efforts to educate, sell train on new we have to consider the cost of staffways.

Economical Feasibility:

It examines whether economically the system's cost is effective. That is, whether finance is available for implementing the proposed solution and whether the returns are proportionate with the cost of the project. It includes these:

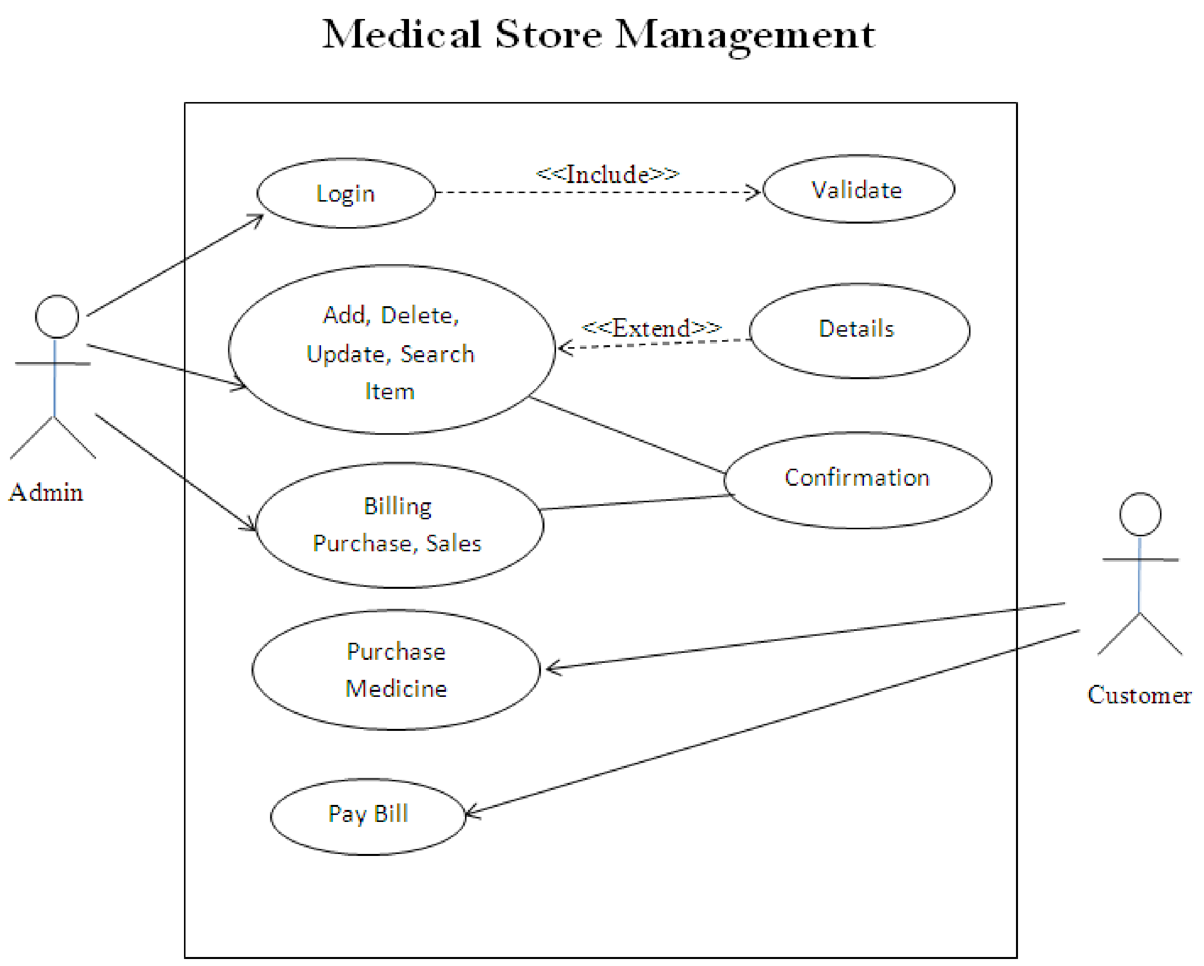
- The cost of Hardware/Software for the application already exists.
- The use of existing software developing new application.
- Using it only increase the gain.
- The cost if nothing were to change from the present system would not necessarily increase but the net gain from the system being comparatively better.

4.5 REQUIREMENT VALIDATION

- Username and Password validation that is compulsory to enter within system (Not for visitor).
- Editing into combo box not allowed.
- For every new entry of enter all data manually, made entry by adjustment form. So it reduces mismatch in data.
- The field denoted by (*) are compulsory.
- Most of the data are enter from the master table so it also reduces the chances of mismatch data.

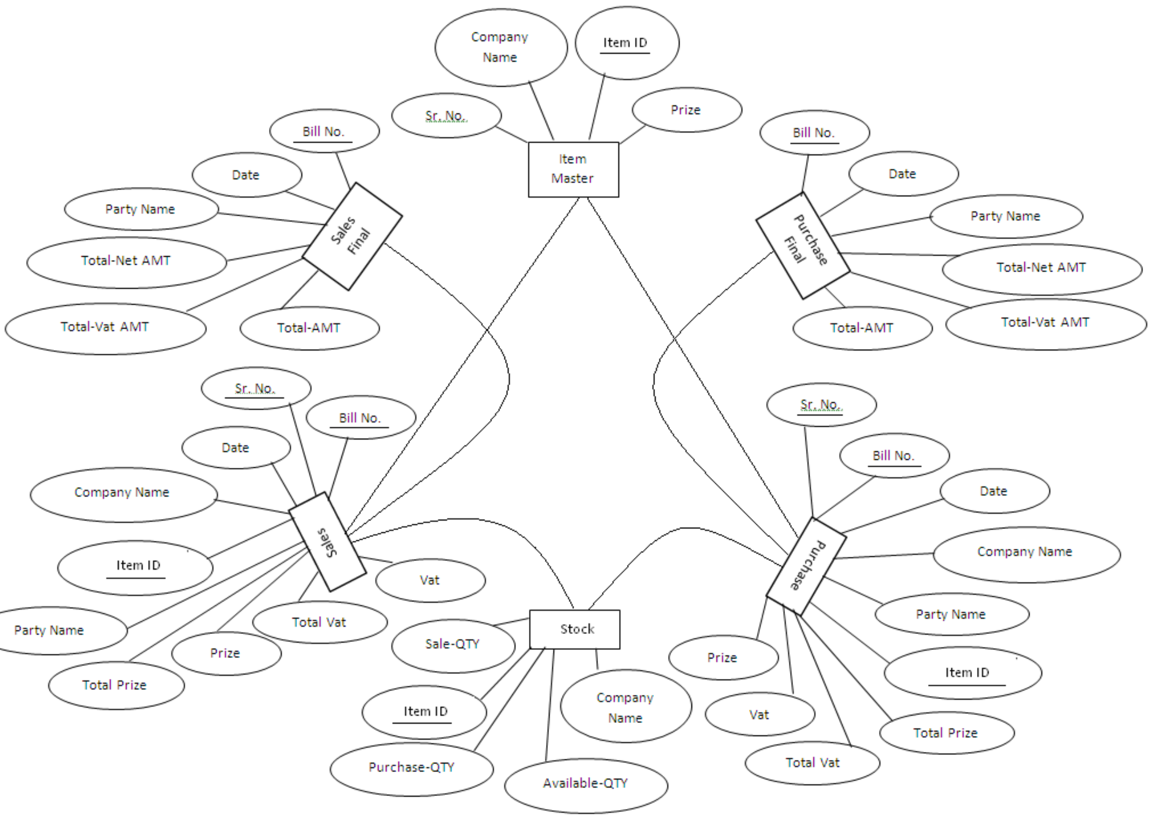
4.6 Functions Of System

4.6.1Use Cases Diagram:

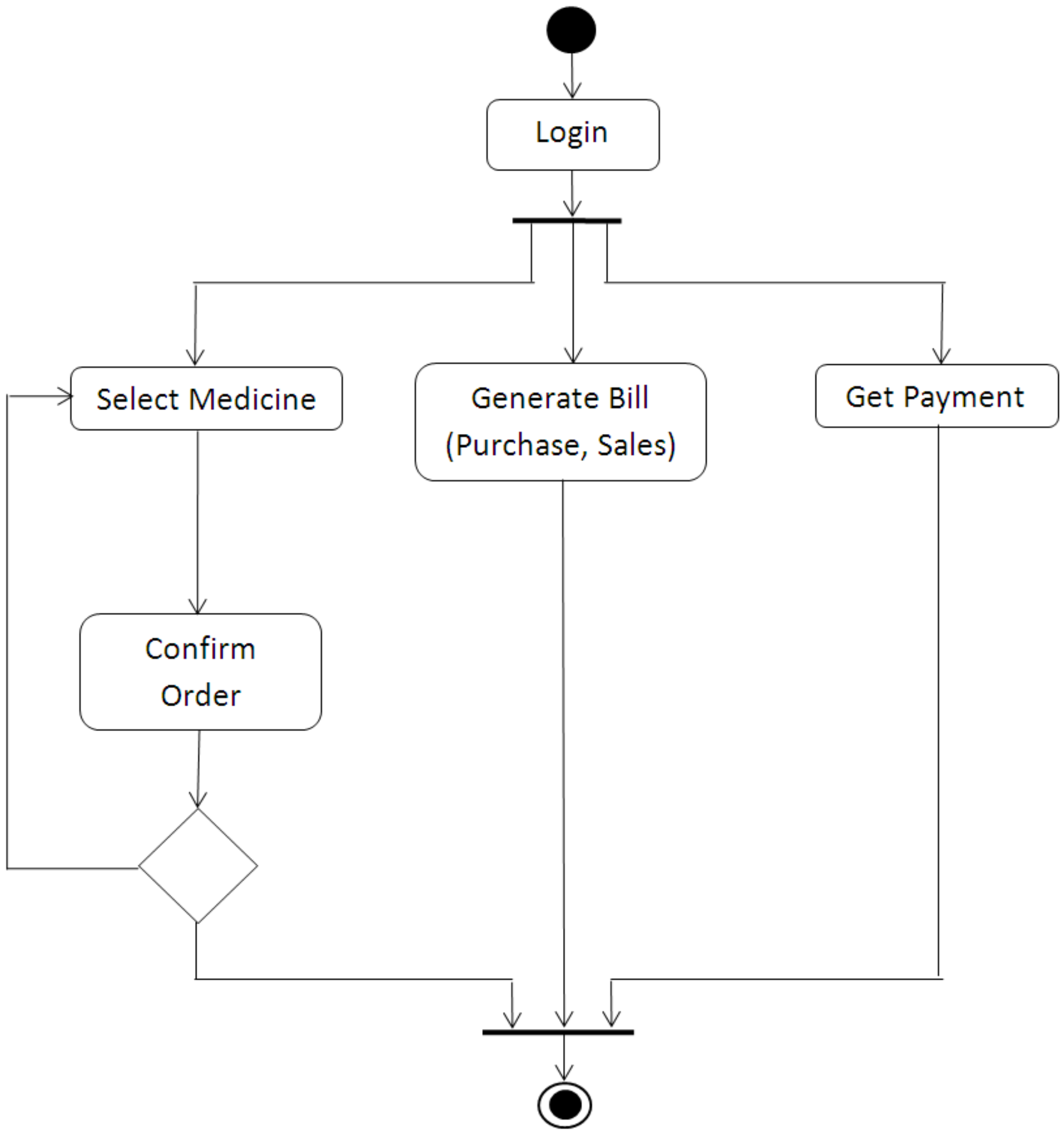


4.7 Data Modeling:

4.7.1E-R diagrams:



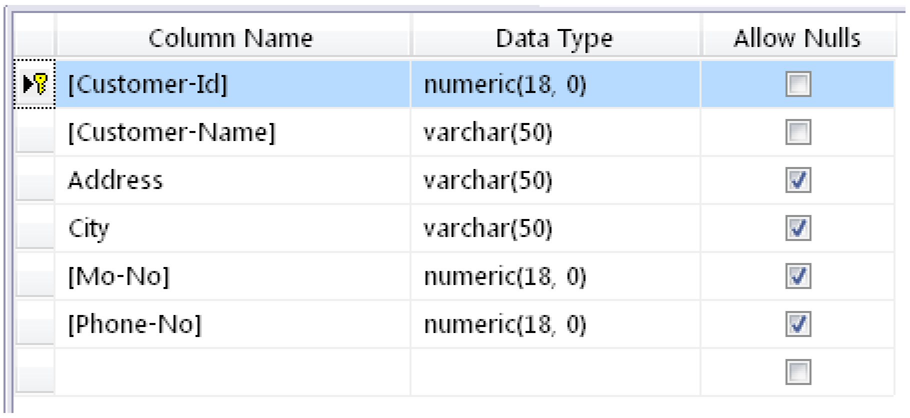
4.7.2 System Activity or Object interaction Diagram:



4.7.3 DATA DICTIONARY

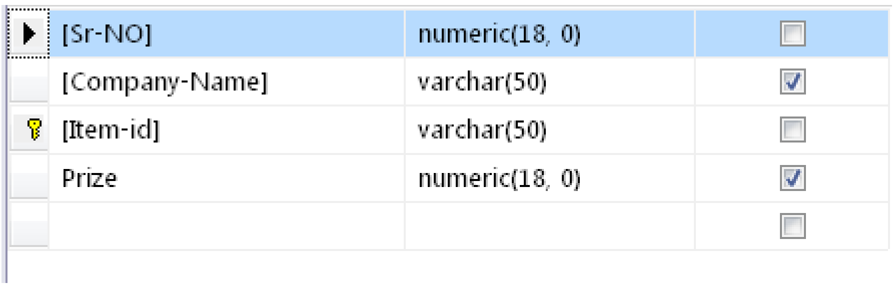
In order to find the data elements required in the various documents, we work backwards. In other words, we ask what data elements are required to meet the information needs and find out the data elements required in the input documents and the required to be maintained at the receiving office. The data elements in the output are also determined by the specifications of information requirements.

Customer Master:



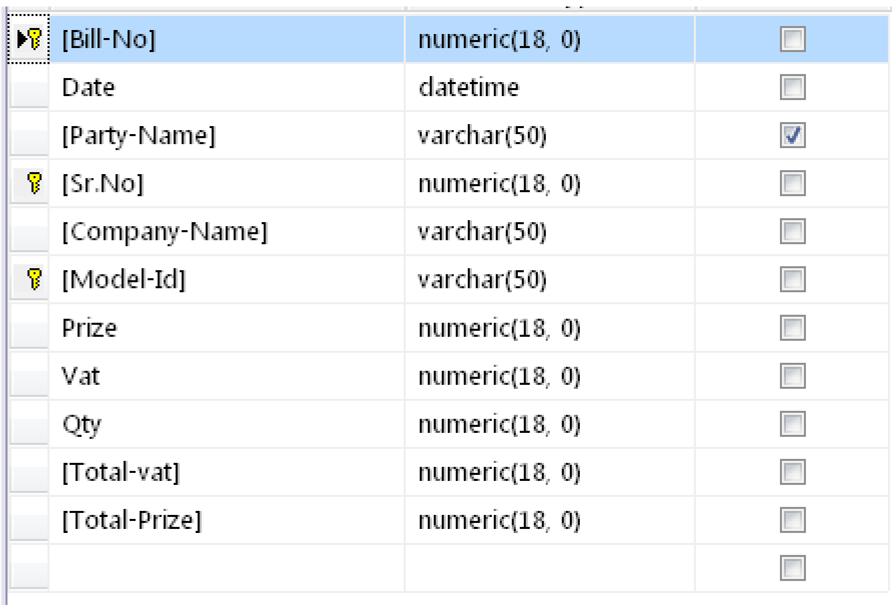
Column Name	Data Type	Allow Nulls
[Customer-Id]	numeric(18, 0)	☐
[Customer-Name]	varchar(50)	☐
Address	varchar(50)	✓
City	varchar(50)	✓
[Mo-No]	numeric(18, 0)	✓
[Phone-No]	numeric(18, 0)	✓
		☐

Item Master:



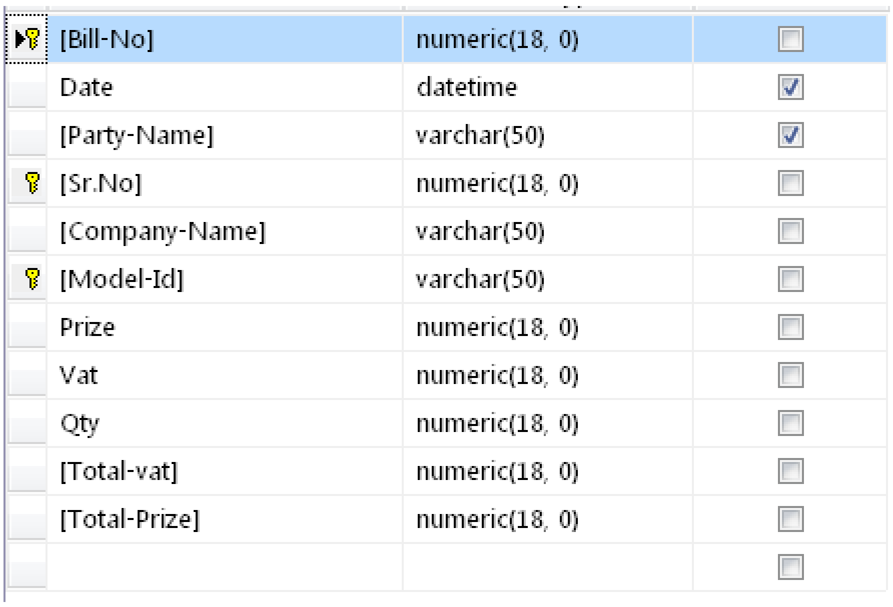
[Sr-NO]	numeric(18, 0)	☐
[Company-Name]	varchar(50)	✓
[Item-id]	varchar(50)	☐
Prize	numeric(18, 0)	✓
		☐

Purchase:



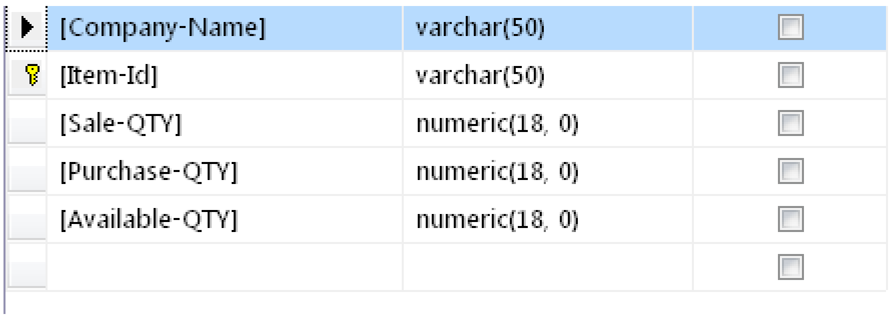
▶🔑	[Bill-No]	numeric(18, 0)	☐
	Date	datetime	☐
	[Party-Name]	varchar(50)	✓
🔑	[Sr.No]	numeric(18, 0)	☐
	[Company-Name]	varchar(50)	☐
🔑	[Model-Id]	varchar(50)	☐
	Prize	numeric(18, 0)	☐
	Vat	numeric(18, 0)	☐
	Qty	numeric(18, 0)	☐
	[Total-vat]	numeric(18, 0)	☐
	[Total-Prize]	numeric(18, 0)	☐
			☐

Sales:



▶🔑	[Bill-No]	numeric(18, 0)	☐
	Date	datetime	✓
	[Party-Name]	varchar(50)	✓
🔑	[Sr.No]	numeric(18, 0)	☐
	[Company-Name]	varchar(50)	☐
🔑	[Model-Id]	varchar(50)	☐
	Prize	numeric(18, 0)	☐
	Vat	numeric(18, 0)	☐
	Qty	numeric(18, 0)	☐
	[Total-vat]	numeric(18, 0)	☐
	[Total-Prize]	numeric(18, 0)	☐
			☐

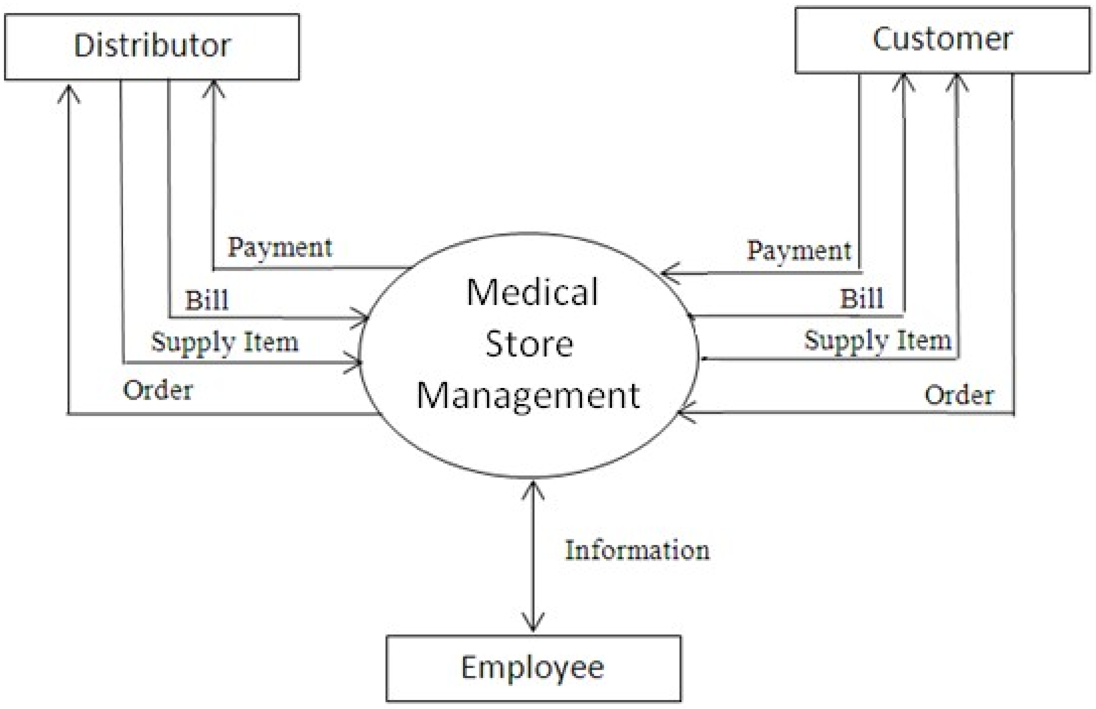
Stock:



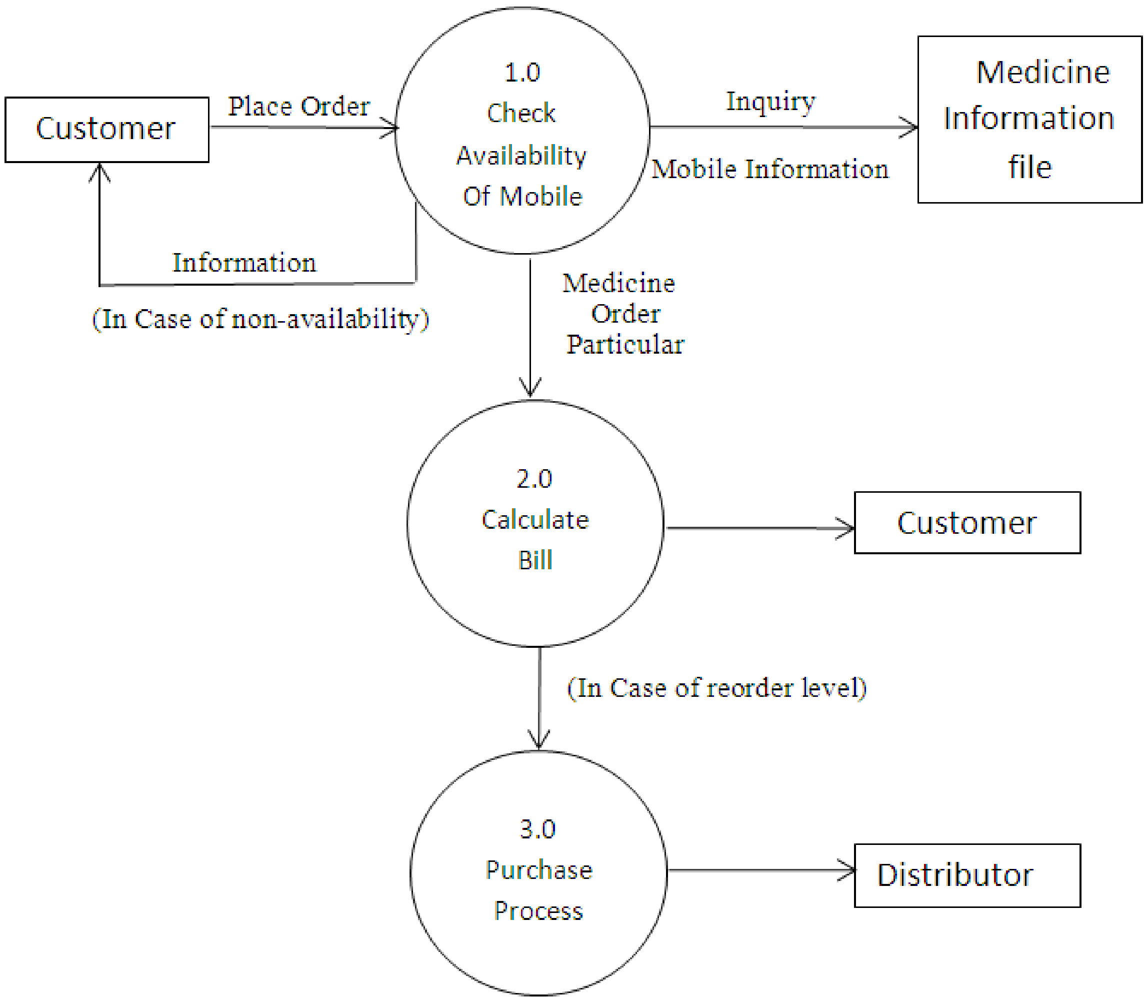
▶	[Company-Name]	varchar(50)	☐
🔑	[Item-Id]	varchar(50)	☐
	[Sale-QTY]	numeric(18, 0)	☐
	[Purchase-QTY]	numeric(18, 0)	☐
	[Available-QTY]	numeric(18, 0)	☐
			☐

4.8 Functional and Behavioral Modeling:

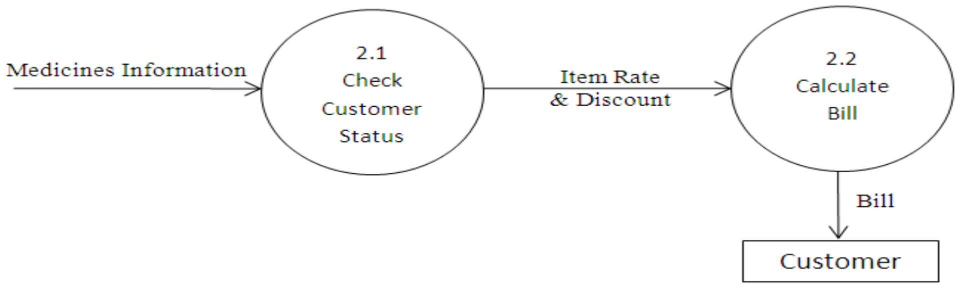
4.8.1 Context Diagram:



4.8.2 Data Flow Diagram (level 0):



Data Flow Diagram level 1:



5. IMPLEMENTATION PLANNING AND DETAILS

5.1 IMPLEMENTATION ENVIRONMENT

Single User vs. Multi User:

In the single user environment, the whole system is used by only a single person of the organization. The person who is using that system will have to look for all the procedures which are held within the organization. And managing all the activities by a single hand is a very-difficult task

While if Multi User environment is there then the user of the system will be more than one. Different people can handle different task to perform .This kind of system will reduce work load and can give better results.

In this project I have provided single user environment. Only one person can have access to the system up to some extent which is decided by the owner.

Thus, in my project there is only one user that is administrators that can handle whole system due to this the administrator have to do work more because of single user system.

GUI vs. Non GUI

The word stands for Graphical User Interface COBOL, C/C++, etc are the example of non GUI language, means they do not provide graphical interface.

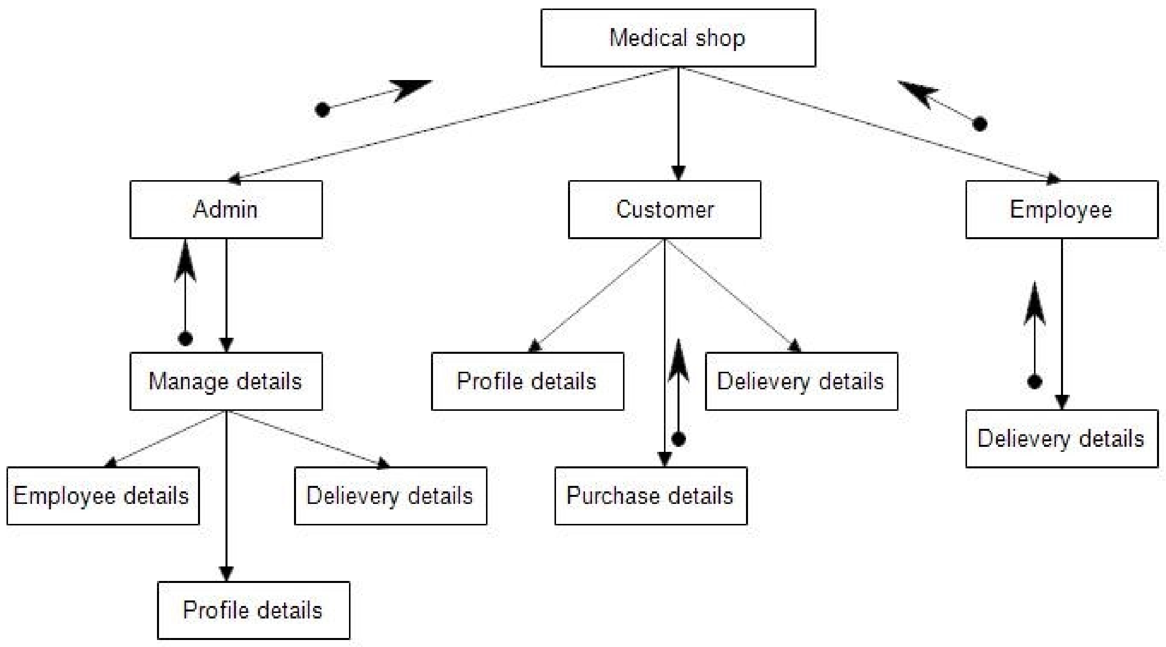
Graphical interface provides ease to the user of the system. User can have better understanding with GUI working system.GUI working system also looks attractive than Non GUI working system. PHP, VB, .NET, etc are the examples of the GUI languages.

COMS developed under C#.net that provides Graphical Interface to the user. Thus it is better to have GUI system rather than Non GUI system.

To understand any system and if there is Graphical user interface is used then the understanding is very easy compared to Non graphical user interface system.

5.2 PROGRAM/ MODULES SPECIFICATION

Flow Chart:-



5.3SECURITY FEATURES

GENERAL CONSIDERATION

- The application is well build with all constraints and validations like no any character can type to mobile no field and also same as no any digit can be typed into name and text fields.
- Also each focus lost event need to check whether the fields is empty or not
- So ne worry about to having null data in the database.

5.4 Sample Coding

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Medical_store
{
publicpartialclassitemStock : Form
    {
public itemStock(Medical_store.Mdi_parent1 parent)
        {
            InitializeComponent();
this.MdiParent = parent;
        }
privatevoid itemStock_Load(object sender, EventArgs e)
        {
this.stockTableAdapter.Fill(this.medicalDataSet1.Stock);

        }
privatevoid button1_Click(object sender, EventArgs e)
        {
this.Close();
        }
privatevoid dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
        {

        }
    }
}
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;



namespace MEDICAL

{

    public partial class ADDItem : Form

    {
        DS.DS_ITEM.StockInMst_SelectDataTable IDT = new
MEDICAL.DS.DS_ITEM.StockInMst_SelectDataTable();

        DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter IAdapter = new
MEDICAL.DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter();


        DS.DS_COMPANY.CompanyMst_SelectDataTable CDT = new
MEDICAL.DS.DS_COMPANY.CompanyMst_SelectDataTable();

        DS.DS_COMPANYTableAdapters.CompanyMst_SelectTableAdapter CAdapter = new
MEDICAL.DS.DS_COMPANYTableAdapters.CompanyMst_SelectTableAdapter();


        DS.DS_STOCK.StockMst_SelectDataTable SDT = new
MEDICAL.DS.DS_STOCK.StockMst_SelectDataTable();

        DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter SAdapter = new
MEDICAL.DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter();


        DS.DS_USER.UserMst_SelectDataTable UDT = new
DS.DS_USER.UserMst_SelectDataTable();

        DS.DS_USERTableAdapters.UserMst_SelectTableAdapter UAdapter = new
MEDICAL.DS.DS_USERTableAdapters.UserMst_SelectTableAdapter();

        public string username, strmenu;

        public ADDItem(string uname,string strmnu)

        {

            username = uname;

            strmenu = strmnu;

            InitializeComponent();

        }


        private void ADDItem_Load(object sender, EventArgs e)

        {

            if (strmenu == "Add")

            {


                tabControl1.SelectedIndex = 0;

            }
            else if (strmenu == "Update")
            {
                tabControl1.SelectedIndex = 1;
            }
            else if (strmenu == "Delete")
            {
                tabControl1.SelectedIndex = 2;
            }
            else if (strmenu == "View")
            {
                tabControl1.SelectedIndex = 3;
            }


            CDT = CAdapter.SelectComapny();
            cmbdompany.DataSource = CDT;
            cmbdompany.DisplayMember = "Cname";
            cmbdompany.ValueMember = "Cid";
            cmbdompany.Text = "SELECT";
            IDT = IAdapter.SelectItem();
            comboBox1.DataSource = IDT;
            comboBox1.DisplayMember = "I_name";
            comboBox1.ValueMember = "I_ID";
            comboBox1.Text = "SELECT";
        }

        private void button1_Click(object sender, EventArgs e)
        {
            int inst = IAdapter.Insert(txtiname.Text, txtidetail.Text,
Convert.ToInt32(txtiqnt.Text), Convert.ToDouble(txtiprice.Text), cmbdompany.Text,
Convert.ToDateTime(dateexpire.Text), txtlocation.Text);
```

```
            double price=Convert.ToDouble(txtiqnt.Text) *
Convert.ToDouble(txtiprice.Text);

            int addstock = SAdapter.Insert(txtiname.Text,
Convert.ToDouble(txtiprice.Text), Convert.ToInt32(txtiqnt.Text), 0,
Convert.ToInt32(txtiqnt.Text), price, 0, price, System.DateTime.Now.Date);


            MessageBox.Show("Item Added Successfully !!", "MEdical System");

            txtiname.Text = "";

            txtidetail.Text = "";

            txtiprice.Text = "";

            txtiqnt.Text = "";

            txtlocation.Text = "";

            cmbdompany.Text = "SELECT";

        }


        private void tabControl1_SelectedIndexChanged(object sender, EventArgs e)

        {

            if (tabControl1.SelectedIndex == 0)

            {

                CDT = CAdapter.SelectComapny();

                cmbdompany.DataSource = CDT;

                cmbdompany.DisplayMember = "Cname";

                cmbdompany.ValueMember = "Cid";

                cmbdompany.Text = "SELECT";

            }

            else if (tabControl1.SelectedIndex == 1)

            {

               // IDT = IAdapter.SelectItem();

                SDT = SAdapter.SelectStock();

                comboBox1.DataSource = SDT;

                comboBox1.DisplayMember = "Iname";
                comboBox1.ValueMember = "s_ID";

                comboBox1.Text = "SELECT";

            }

            else if (tabControl1.SelectedIndex == 2)

            {

              //  IDT = IAdapter.SelectItem();

                SDT = SAdapter.SelectStock();

                comboBox2.DataSource = SDT;

                comboBox2.DisplayMember = "Iname";

                comboBox2.ValueMember = "s_ID";

                comboBox2.Text = "SELECT";


            }

            else if (tabControl1.SelectedIndex == 3)

            {


                IDT = IAdapter.SelectItem();

                dataGridView1.DataSource = IDT;

            }

        }


        private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)

        {

            //if (comboBox1.SelectedText != "")

            //{

            //IDT = IAdapter.SelectBYID(Convert.ToInt32(comboBox1.SelectedValue));

           // txtcurentqnt.Text = IDT.Rows[0]["I_Quantity"].ToString();

            //}

        }


        private void button2_Click(object sender, EventArgs e)
```

```
        {
            if (MessageBox.Show("Are you sure !! You want to Update Stock !!",
"Medical system", MessageBoxButtons.OKCancel) == DialogResult.OK)
            {

                IDT = IAdapter.SelectByINmae(comboBox1.Text);
               //
IAdapter.StockInMst_Updateitem(Convert.ToInt32(comboBox1.SelectedValue),
Convert.ToInt32(txtnewqnt.Text));
                int inst = IAdapter.Insert(comboBox1.Text, IDT.Rows[0]
["I_Descrip"].ToString(), Convert.ToInt32(txtnewqnt.Text),
Convert.ToDouble(IDT.Rows[0]["I_Price"].ToString()), IDT.Rows[0]["c_name"].ToString(),
Convert.ToDateTime(IDT.Rows[0]["I_Ex_date"].ToString()), IDT.Rows[0]
["I_location"].ToString());




                double tprice = Convert.ToDouble(txtnewqnt.Text) *
Convert.ToDouble(IDT.Rows[0]["I_Price"].ToString());
                SAdapter.StockMst_ADD_Update_Quantity(Convert.ToInt32(txtnewqnt.Text),
tprice, comboBox1.Text);

                MessageBox.Show("Quantity Updated Successfully !!", "Medical System");
                txtnewqnt.Text = "";
                txtcurentqnt.Text = "";

                comboBox1.Text = "SELECT";
            }
        }

        private void button3_Click(object sender, EventArgs e)
        {
            if (MessageBox.Show("Are you sure !! You want to Delete Stock !!",
"Medical system", MessageBoxButtons.OKCancel) == DialogResult.OK)
            {
                int del = SAdapter.Delete(Convert.ToInt32(comboBox2.SelectedValue));
                int dellitem = IAdapter.Delete(comboBox2.Text);
                MessageBox.Show("Item Deleted Successfully !!", "Medical System");
                SDT = SAdapter.SelectStock();
                comboBox2.DataSource = SDT;
                comboBox2.DisplayMember = "Iname";
                comboBox2.ValueMember = "s_ID";
                comboBox2.Text = "SELECT";
            }

        }

        private void button4_Click(object sender, EventArgs e)
        {
            SDT = SAdapter.SelectBY_INAME(comboBox1.Text);
            txtcurentqnt.Text = SDT.Rows[0]["totalQuantity"].ToString();
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace MEDICAL
```

```
{
    public partial class Client : Form
    {
        public string strmenu;
        DS.DS_CLIENT.ClientMst_SelectDataTable CDT = new
MEDICAL.DS.DS_CLIENT.ClientMst_SelectDataTable();
        DS.DS_CLIENTTableAdapters.ClientMst_SelectTableAdapter CAdapter = new
MEDICAL.DS.DS_CLIENTTableAdapters.ClientMst_SelectTableAdapter();
        public Client( string strmnu)
        {
            strmenu = strmnu;
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            int inst = CAdapter.Insert(txtame.Text, txtsurname.Text,
Convert.ToDouble(txtmobile.Text), txtadd.Text, txtcity.Text);
            MessageBox.Show("Client Added Susscessfully !!", "Medical System");
            txtame.Text = "";
            txtadd.Text = "";
            txtcity.Text = "";
            txtmobile.Text = "";
            txtsurname.Text = "";
            txtame.Focus();
        }

        private void tabControl1_SelectedIndexChanged(object sender, EventArgs e)
        {
            if (tabControl1.SelectedIndex == 0)
            {
            }
            else if (tabControl1.SelectedIndex == 1)
            {

                CDT = CAdapter.SelectClient();
                comboBox1.DataSource = CDT;
                comboBox1.DisplayMember = "cu_name";
                comboBox1.ValueMember = "cu_id"; comboBox1.Text = "SELECT";

            }
            else if (tabControl1.SelectedIndex == 2)
            {
                CDT = CAdapter.SelectClient();
                dataGridView1.DataSource = CDT;
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            int del = CAdapter.Delete(Convert.ToInt32(comboBox1.SelectedValue));
            MessageBox.Show("Client Deleted !", "Medical System");
            CDT = CAdapter.SelectClient();
            comboBox1.DataSource = CDT;
            comboBox1.DisplayMember = "cu_name";
            comboBox1.ValueMember = "cu_id";
            comboBox1.Text = "SELECT";

        }

        private void Client_Load(object sender, EventArgs e)
        {
            if (strmenu == "Add")
            {
                tabControl1.SelectedIndex = 0;
            }
            else if (strmenu == "Delete")
            {
                tabControl1.SelectedIndex = 1;
            }
            else if (strmenu == "View")
            {
                tabControl1.SelectedIndex = 2;
            }
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace MEDICAL
{
    public partial class Company : Form
    {
        DS.DS_COMPANY.CompanyMst_SelectDataTable CDT = new
MEDICAL.DS.DS_COMPANY.CompanyMst_SelectDataTable();
```

```
        DS.DS_COMPANYTableAdapters.CompanyMst_SelectTableAdapter CAdapter = new
MEDICAL.DS.DS_COMPANYTableAdapters.CompanyMst_SelectTableAdapter();

        public string strmenu;

        public Company(string strmnu)

        {

            strmenu = strmnu;

            InitializeComponent();

        }


        private void button1_Click(object sender, EventArgs e)

        {

            int ist = CAdapter.Insert(txtname.Text, txtpersn.Text, txtadd.Text,
txtmobile.Text);

            MessageBox.Show("Company Detail Addedd !!", "Medical System");


            txtname.Text = "";

            txtpersn.Text = "";

            txtadd.Text = "";

            txtmobile.Text = "";

            txtname.Focus();

        }


        private void tabControl1_SelectedIndexChanged(object sender, EventArgs e)

        {

            if (tabControl1.SelectedIndex == 0)

            {


            }

            else if (tabControl1.SelectedIndex == 1)

            {
                CDT = CAdapter.SelectComapny();

                comboBox1.DataSource = CDT;

                comboBox1.DisplayMember = "Cname";

                comboBox1.ValueMember = "CID";

                comboBox1.Text = "SELECT";

            }

            else if (tabControl1.SelectedIndex == 2)

            {

                CDT = CAdapter.SelectComapny();

                dataGridView1.DataSource = CDT;

            }

        }


        private void button2_Click(object sender, EventArgs e)

        {

            if (MessageBox.Show("Are you sure !! You want to Delete !!", "Medical
system", MessageBoxButtons.OKCancel) == DialogResult.OK)

            {

                int del = CAdapter.Delete(Convert.ToInt32(comboBox1.SelectedValue));

                MessageBox.Show("Delete Company !!", "Medical System");

                CDT = CAdapter.SelectComapny();

                comboBox1.DataSource = CDT;

                comboBox1.DisplayMember = "Cname";

                comboBox1.ValueMember = "CID";

                comboBox1.Text = "SELECT";


            }


        }


        private void Company_Load(object sender, EventArgs e)
        {
            if (strmenu == "Add")
            {
                tabControl1.SelectedIndex = 0;
            }
            else if (strmenu == "Delete")
            {
                tabControl1.SelectedIndex = 1;

            }
            else if (strmenu == "View")
            {
                tabControl1.SelectedIndex = 2;
            }

        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace MEDICAL
{
    public partial class Form1 : Form

    {

        DS.DS_USER.UserMst_SelectDataTable UDT = new
DS.DS_USER.UserMst_SelectDataTable();

        DS.DS_USERTableAdapters.UserMst_SelectTableAdapter UAdapter = new
MEDICAL.DS.DS_USERTableAdapters.UserMst_SelectTableAdapter();

        public string username;

        public Form1()

        {

            InitializeComponent();

        }


        private void Form1_Load(object sender, EventArgs e)

        {


            lbltime.Text = System.DateTime.Now.ToString();

            lblday.Text = System.DateTime.Now.DayOfWeek.ToString();




            foreach (Control ctl in this.Controls)

            {

                try

                {

                    System.Windows.Forms.Control Mdi = (MdiClient)ctl;


                    Mdi.BackColor = System.Drawing.Color.DarkSeaGreen;

                }

                catch (Exception a)

                {
                }
            }
        }

        private void btnlogin_Click(object sender, EventArgs e)
        {
            if (txtname.Text == "")
            {

                MessageBox.Show("Enter Login Name !", "Medical System");
            }
            else if (txtpass.Text == "")
            {
                MessageBox.Show("Enter Login Password !", "Medical System");
            }
            else
            {
                UDT = UAdapter.SelectForLOGIN(txtname.Text, txtpass.Text);
                if(UDT.Rows.Count>0)
                { lblname.Text = "welcome " + txtname.Text;
                    username = txtname.Text;
                    txtname.Text = "";
                    txtpass.Text = "";
                    gplogin.Visible = false;
                    menulogout.Visible = true;
                    mENUToolStripMenuItem.Enabled = true;
                    sELLToolStripMenuItem.Enabled = true;
                    cLIENTSToolStripMenuItem.Enabled = true;
                    rEPORTSToolStripMenuItem.Enabled = true;
                    mANAGEUSERToolStripMenuItem.Enabled = true;
                    cOMPANYToolStripMenuItem.Enabled = true;
```

```
                }
                else
                {
                    MessageBox.Show("Invalid LoginName OR Password !", "Medical
System");
                }
            }
        }

        private void menulogout_Click(object sender, EventArgs e)
        {
            closeExistingForm();
            gplogin.Visible = true;
            menulogout.Visible = false;
            mENUToolStripMenuItem.Enabled = false;
            sELLToolStripMenuItem.Enabled = false;
            cLIENTSToolStripMenuItem.Enabled = false;
            rEPORTSToolStripMenuItem.Enabled = false;
            mANAGEUSERToolStripMenuItem.Enabled = false;
            cOMPANYToolStripMenuItem.Enabled = false;

            lblname.Text = "";
        }

        private void nEWToolStripMenuItem_Click(object sender, EventArgs e)
        {
            closeExistingForm();
            Form additem = new ADDItem(username,"Add");
            additem.MdiParent = this;
            additem.Show();
        }
        private void closeExistingForm()
        {
            try
            {
                this.ActiveMdiChild.Close();
            }
            catch (Exception)
            {

            }

        }
        private void eXITToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void aDDNEWToolStripMenuItem_Click(object sender, EventArgs e)
        {
            closeExistingForm();
            Form additem = new UserMst(username,"Add");
            additem.MdiParent = this;
            additem.Show();
        }

        private void aDDNEWToolStripMenuItem1_Click(object sender, EventArgs e)
        {
            closeExistingForm();
            Form additem = new Company("Add");
            additem.MdiParent = this;
            additem.Show();


        }

        private void aDDNEWToolStripMenuItem2_Click(object sender, EventArgs e)
        {
            closeExistingForm();
            Form additem = new Client("Add");
            additem.MdiParent = this;
            additem.Show();


        }

        private void sELLToolStripMenuItem_Click(object sender, EventArgs e)
        {
            DS.DS_SALES.SALES_SELECTDataTable SDT = new
MEDICAL.DS.DS_SALES.SALES_SELECTDataTable();
            DS.DS_SALESTableAdapters.SALES_SELECTTableAdapter SAdapter = new
MEDICAL.DS.DS_SALESTableAdapters.SALES_SELECTTableAdapter();
            int del = SAdapter.Delete();
            closeExistingForm();
            Form additem = new SELL();
            additem.MdiParent = this;
            additem.Show();
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            lbltime.Text = System.DateTime.Now.ToString();


        }
        private void stockReportToolStripMenuItem_Click(object sender, EventArgs e)

        {

            closeExistingForm();

            Form srpt = new StockReport();

            srpt.MdiParent = this;

            srpt.Show();

        }


        private void sellReportToolStripMenuItem_Click(object sender, EventArgs e)

        {

            closeExistingForm();

            Form sellrpt = new SellReport();

            sellrpt.MdiParent = this;

            sellrpt.Show();

        }


        private void updateStockToolStripMenuItem_Click(object sender, EventArgs e)

        {

            closeExistingForm();

            Form additem = new ADDItem(username,"Update");

            additem.MdiParent = this;

            additem.Show();

        }


        private void totalStockReportToolStripMenuItem_Click(object sender, EventArgs
e)

        {

            closeExistingForm();

            Form tsrpt = new TotalStockReport();

            tsrpt.MdiParent = this;
```

```
    tsrpt.Show();

}

private void deleteStockToolStripMenuItem_Click(object sender, EventArgs e)
{
    closeExistingForm();
    Form additem = new ADDItem(username, "Delete");
    additem.MdiParent = this;
    additem.Show();
}

private void repotsToolStripMenuItem_Click(object sender, EventArgs e)
{
    closeExistingForm();
    Form additem = new ADDItem(username, "View");
    additem.MdiParent = this;
    additem.Show();
}

private void dELETEToolStripMenuItem2_Click(object sender, EventArgs e)
{
    closeExistingForm();
    Form additem = new Client("Delete");
    additem.MdiParent = this;
    additem.Show();
}

private void vIEWToolStripMenuItem_Click(object sender, EventArgs e)
{
    closeExistingForm();
```

```
            Form additem = new Client("View");

            additem.MdiParent = this;

            additem.Show();

        }


        private void dELETEToolStripMenuItem_Click(object sender, EventArgs e)

        {

            closeExistingForm();

            Form additem = new UserMst(username, "Delete");

            additem.MdiParent = this;

            additem.Show();

        }


        private void rEPORTSToolStripMenuItem1_Click(object sender, EventArgs e)

        {

            closeExistingForm();

            Form additem = new UserMst(username, "View");

            additem.MdiParent = this;

            additem.Show();

        }


        private void cHNAEGPASSWORDToolStripMenuItem_Click(object sender, EventArgs e)

        {

            closeExistingForm();

            Form additem = new UserMst(username, "Password");

            additem.MdiParent = this;

            additem.Show();

        }


        private void dELETEToolStripMenuItem1_Click(object sender, EventArgs e)

        {
            closeExistingForm();

            Form additem = new Company("Delete");

            additem.MdiParent = this;

            additem.Show();

        }


        private void rEPORTSToolStripMenuItem2_Click(object sender, EventArgs e)

        {

            closeExistingForm();

            Form additem = new Company("View");

            additem.MdiParent = this;

            additem.Show();

        }

    }

}


ausing System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;


namespace MEDICAL

{

    static class Program

    {

        /// <summary>

        /// The main entry point for the application.

        /// </summary>

        [STAThread]

        static void Main()

        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace MEDICAL
{
    public partial class SELL : Form
    {
        DS.DS_SALES.SALES_SELECT_SUM_TOTALDataTable SUMDT = new
MEDICAL.DS.DS_SALES.SALES_SELECT_SUM_TOTALDataTable();
        DS.DS_SALESTableAdapters.SALES_SELECT_SUM_TOTALTableAdapter SUMAdapter = new
MEDICAL.DS.DS_SALESTableAdapters.SALES_SELECT_SUM_TOTALTableAdapter();


        DS.DS_CLIENT.ClientMst_SelectDataTable CDT = new
MEDICAL.DS.DS_CLIENT.ClientMst_SelectDataTable();
        DS.DS_CLIENTTableAdapters.ClientMst_SelectTableAdapter CAdapter = new
MEDICAL.DS.DS_CLIENTTableAdapters.ClientMst_SelectTableAdapter();


        DS.DS_SALES.SALES_SELECTDataTable SDT = new
MEDICAL.DS.DS_SALES.SALES_SELECTDataTable();
```

```
        DS.DS_SALESTableAdapters.SALES_SELECTTableAdapter SAdapter = new
MEDICAL.DS.DS_SALESTableAdapters.SALES_SELECTTableAdapter();


        DS.DS_ITEM.StockInMst_SelectDataTable IDT = new
MEDICAL.DS.DS_ITEM.StockInMst_SelectDataTable();

        DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter IAdapter = new
MEDICAL.DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter();


        DS.DS_STOCK.StockMst_SelectDataTable StockDT = new
MEDICAL.DS.DS_STOCK.StockMst_SelectDataTable();

        DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter StockAdapter = new
MEDICAL.DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter();


        DS.DS_SOUT.StockOutMst_SelectDataTable SOotDT = new
MEDICAL.DS.DS_SOUT.StockOutMst_SelectDataTable();

        DS.DS_SOUTTableAdapters.StockOutMst_SelectTableAdapter SOutAdapter = new
MEDICAL.DS.DS_SOUTTableAdapters.StockOutMst_SelectTableAdapter();

        public SELL()

        {

            InitializeComponent();

        }


        private void groupBox1_Enter(object sender, EventArgs e)

        {


        }


        private void SELL_Load(object sender, EventArgs e)

        {

            int del = SAdapter.Delete();

            SqlConnection con = new SqlConnection("Data Source='.\\
SQLEXPRESS';Integrated Security='true';Initial Catalog='MEDICAL'");

            SqlCommand cmd = new SqlCommand("SELECT I_name FROM StockInMst", con);

            con.Open();
            SqlDataReader reader = cmd.ExecuteReader();

            AutoCompleteStringCollection SCollection = new
    AutoCompleteStringCollection();

            while (reader.Read())

            {

                SCollection.Add(reader.GetString(0));

            }

            txtiname.AutoCompleteCustomSource = SCollection;

            con.Close();



            CDT = CAdapter.SelectClient();

            comboBox1.DataSource = CDT;

            comboBox1.DisplayMember = "cu_name";

            comboBox1.ValueMember = "cu_id";

        }


        private void gvsales_CellContentClick(object sender, DataGridViewCellEventArgs
    e)

        {

            try

            {

                if (gvsales.Rows[e.RowIndex].Cells[4].Selected == true)

                {



                    string namee = gvsales.Rows[e.RowIndex].Cells[0].Value.ToString();

                    SAdapter.SALES_DELETE_by_name(namee);

                  //  STADapter.DeleteById(Convert.ToInt32(m));

                   // BindGrid();
                    SDT = SAdapter.SelectBy_PNAME(txtpname.Text);

                    gpdispatch.Visible = true;

                    gvsales.AutoGenerateColumns = false;

                    gvsales.DataSource = SDT;


                    SUMDT = SUMAdapter.SelectTotla();

                    lblqnt.Text = SUMDT.Rows[0]["qnt"].ToString();

                    lbltprice.Text = SUMDT.Rows[0]["Tprice"].ToString();

                }


            }

            catch (Exception )

            { }

        }


        private void txtiname_KeyDown(object sender, KeyEventArgs e)

        {


        }


        private void txtiname_TextChanged(object sender, EventArgs e)

        {



        }


        private void txtiname_KeyPress(object sender, KeyPressEventArgs e)

        {


        }
```

```
        private void txtiname_Leave(object sender, EventArgs e)
        {
            if (txtiname.Text != "")
            {
                label8.Text = txtiname.Text;
                StockDT = StockAdapter.SelectBY_INAME(txtiname.Text);
                IDT = IAdapter.SelectByINmae(txtiname.Text);
                if (StockDT.Rows.Count > 0)
                {
                    if (StockDT.Rows[0]["AvailableQuantity"].ToString() == "0")
                    {
                        MessageBox.Show("No Enought Quantity !!", "Medical System");
                    }
                    else
                    {

                        SDT = SAdapter.SelectBY_NAME(txtiname.Text);
                        if (SDT.Rows.Count > 0)
                        {

                            SDT = SAdapter.SelectBY_NAME(txtiname.Text);

                            int exiqnt =Convert.ToInt32( SDT.Rows[0]
["Quantity"].ToString());
                            if (Convert.ToInt32(StockDT.Rows[0]
["AvailableQuantity"].ToString()) > exiqnt)
                            {
                                int QNT = Convert.ToInt32(SDT.Rows[0]
["Quantity"].ToString()) + 1;
                                double TPRICE = Convert.ToInt32(SDT.Rows[0]
["Price"].ToString()) * QNT;
                                int existsalseupdate = SAdapter.Update(txtiname.Text,
QNT.ToString(), TPRICE);
```

```
                                }
                                else
                                {
                                    MessageBox.Show("No Enought Quantity", "Medical
system");
                                }

                            }
                            else
                            {
                                lblq.Text = StockDT.Rows[0]
["AvailableQuantity"].ToString();
                                lblp.Text = IDT.Rows[0]["I_Price"].ToString();
                                lbll.Text = IDT.Rows[0]["I_location"].ToString();
                                txtq.Text = "1";


                                double tprice = Convert.ToDouble(IDT.Rows[0]
["I_Price"].ToString());
                                int salinst = SAdapter.Insert(txtpname.Text,
txtiname.Text, tprice, 1, tprice);
                            }
                            SDT = SAdapter.SelectBy_PNAME(txtpname.Text);
                            gpdispatch.Visible = true;
                            gvsales.AutoGenerateColumns = false;
                            gvsales.DataSource = SDT;


                            SUMDT = SUMAdapter.SelectTotla();
                            lblqnt.Text = SUMDT.Rows[0]["qnt"].ToString();
                            lbltprice.Text = SUMDT.Rows[0]["Tprice"].ToString();


                        }
```

```
                gpdispatch.Visible = true;
            }
            else
            {
                MessageBox.Show("Item Not Available !!", "Medical System");
            }
        }
      //  txtiname.Text = "";
       // txtiname.Focus();
    }

    private void txtiname_DragEnter(object sender, DragEventArgs e)
    {



    }

    private void txtiname_Click(object sender, EventArgs e)
    {



    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {
        txtpname.Text = comboBox1.Text;
    }

    private void label1_Click(object sender, EventArgs e)
    {
    }


    private void txtq_TextChanged(object sender, EventArgs e)
    {



    }


    private void txtq_Leave(object sender, EventArgs e)
    {
        if (txtq.Text != "")
        {
            StockDT = StockAdapter.SelectBY_INAME(label8.Text);
            IDT = IAdapter.SelectByINmae(label8.Text);
            if (StockDT.Rows.Count > 0)
            {

                if (StockDT.Rows[0]["AvailableQuantity"].ToString() == "0")
                {
                    MessageBox.Show("No Enought Quantity !!", "Medical System");
                }
                else
                {


                    SDT = SAdapter.SelectBY_NAME(label8.Text);
                    if (SDT.Rows.Count > 0)
                    {
                        SDT = SAdapter.SelectBY_NAME(label8.Text);


                        int exiqnt = Convert.ToInt32(SDT.Rows[0]
["Quantity"].ToString());
```

```
                             int qntt = exiqnt + Convert.ToInt32(txtq.Text);

                             if (Convert.ToInt32(StockDT.Rows[0]
["AvailableQuantity"].ToString()) > qntt)

                             {

                                 // int QNT = Convert.ToInt32(SDT.Rows[0]
["Quantity"].ToString()) + 1;

                                 double TPRICE = Convert.ToInt32(SDT.Rows[0]
["Price"].ToString()) * qntt;

                                 int existsalseupdate = SAdapter.Update(label8.Text,
qntt.ToString(), TPRICE);

                                 SDT = SAdapter.SelectBy_PNAME(txtpname.Text);

                                 gpdispatch.Visible = true;

                                 gvsales.AutoGenerateColumns = false;

                                 gvsales.DataSource = SDT;


                                 SUMDT = SUMAdapter.SelectTotla();

                                 lblqnt.Text = SUMDT.Rows[0]["qnt"].ToString();

                                 lbltprice.Text = SUMDT.Rows[0]["Tprice"].ToString();

                             }

                             else

                             {

                                 MessageBox.Show("No Enought Quantity", "Medical
system");

                             }


                         }

                     }

                 }



             }
```

```
        }


        private void button1_Click(object sender, EventArgs e)
        {
            if (MessageBox.Show("Are you sure !! You want to Confirm this Order !!",
"Medical system", MessageBoxButtons.OKCancel) == DialogResult.OK)
            {
                if (gvsales.Rows.Count == 0)
                {
                    MessageBox.Show("Enter Some Item First !!");
                }
                else
                {
                    SDT = SAdapter.SelectBy_PNAME(txtpname.Text);



                        for (int i = 0; i < SDT.Rows.Count; i++)
                        {
                            int sout = SOutAdapter.Insert(txtpname.Text, SDT.Rows[i]
["IName"].ToString(), Convert.ToInt32(SDT.Rows[i]["Quantity"].ToString()),
Convert.ToDouble(SDT.Rows[i]["Price"].ToString()), Convert.ToDouble(SDT.Rows[i]
["TPrice"].ToString()), 0, System.DateTime.Now.Date);


StockAdapter.StockMst_SELL_Update_Quantity(Convert.ToInt32(SDT.Rows[i]
["Quantity"].ToString()), Convert.ToDouble(SDT.Rows[i]
["TPrice"].ToString()),SDT.Rows[i]["IName"].ToString());
                        }


                        int del = SAdapter.Delete();



                        MessageBox.Show("Your Order has been Submitted !!", "Medical
System");



                        gvsales.DataSource= null;
                }
                gpdispatch.Visible = false;
            }
        }

        private void button2_Click(object sender, EventArgs e)
        {
            if (MessageBox.Show("Are you sure !! You want to Cancel this Order !!",
"Medical system", MessageBoxButtons.OKCancel) == DialogResult.OK)
            {
                int del = SAdapter.Delete();
                gpdispatch.Visible = false;

            }
        }
    }
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace MEDICAL
{
    public partial class SellReport : Form
    {
        //DS.DS_STOCK.StockMst_SelectDataTable SDT = new
DS.DS_STOCK.StockMst_SelectDataTable();

        //DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter SAdapter = new
DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter();

        //DS.DS_ITEM.StockInMst_SelectDataTable IDT = new
MEDICAL.DS.DS_ITEM.StockInMst_SelectDataTable();

        //DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter IAdapter = new
MEDICAL.DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter();


        DS.DS_CLIENT.ClientMst_SelectDataTable CDT = new
DS.DS_CLIENT.ClientMst_SelectDataTable();

        DS.DS_CLIENTTableAdapters.ClientMst_SelectTableAdapter CAdapter = new
DS.DS_CLIENTTableAdapters.ClientMst_SelectTableAdapter();


        DS.DS_SOUT.StockOutMst_SelectDataTable SOotDT = new
MEDICAL.DS.DS_SOUT.StockOutMst_SelectDataTable();

        DS.DS_SOUTTableAdapters.StockOutMst_SelectTableAdapter SOutAdapter = new
MEDICAL.DS.DS_SOUTTableAdapters.StockOutMst_SelectTableAdapter();

        public SellReport()

        {

            InitializeComponent();

        }


        private void SellReport_Load(object sender, EventArgs e)

        {

            SqlConnection con = new SqlConnection("Data Source='.\\
SQLEXPRESS';Integrated Security='true';Initial Catalog='MEDICAL'");

            SqlCommand cmd = new SqlCommand("SELECT I_name FROM StockInMst", con);

            con.Open();

            SqlDataReader reader = cmd.ExecuteReader();

            AutoCompleteStringCollection SCollection = new
AutoCompleteStringCollection();

            while (reader.Read())

            {

                SCollection.Add(reader.GetString(0));
            }
            txtiname.AutoCompleteCustomSource = SCollection;
            con.Close();


            SqlCommand cmdd = new SqlCommand("SELECT cu_name FROM clientmst", con);
            con.Open();
            SqlDataReader readerr = cmdd.ExecuteReader();
            AutoCompleteStringCollection SCollectionn = new
AutoCompleteStringCollection();
            while (readerr.Read())
            {
                SCollectionn.Add(readerr.GetString(0));
            }
            txtcname.AutoCompleteCustomSource = SCollectionn;
            con.Close();
        }

        private void btncustsearch_Click(object sender, EventArgs e)
        {
            txtiname.Text = "";
            SOotDT = SOutAdapter.Select_by_Clintname(txtcname.Text);
            GVReports.DataSource = SOotDT;
        }

        private void btnitemsearch_Click(object sender, EventArgs e)
        {
            txtcname.Text = "";
            SOotDT = SOutAdapter.Select_by_Iname(txtiname.Text);
            GVReports.DataSource = SOotDT;
        }
    }
```

```
}

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;
namespace MEDICAL
{

    public partial class StockReport : Form
    {
        DS.DS_STOCK.StockMst_SelectDataTable SDT = new
DS.DS_STOCK.StockMst_SelectDataTable();
        DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter SAdapter = new
DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter();
        DS.DS_ITEM.StockInMst_SelectDataTable IDT = new
MEDICAL.DS.DS_ITEM.StockInMst_SelectDataTable();
        DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter IAdapter = new
MEDICAL.DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter();



        public StockReport()
        {
            InitializeComponent();
        }
        private void btncomsearch_Click(object sender, EventArgs e)

        {

            txtiname.Text = "";

            //SDT = SAdapter.SelectBY_INAME(txtiname.Text);

            IDT = IAdapter.Select_By_CNAME(txtcname.Text);

            GVReports.DataSource = IDT;

        }


        private void btnitemsearch_Click(object sender, EventArgs e)

        {

            txtcname.Text = "";

            //SDT = SAdapter.SelectBY_INAME(txtiname.Text);

            IDT = IAdapter.SelectByINmae(txtiname.Text);

            GVReports.DataSource = IDT;


        }


        private void StockReport_Load(object sender, EventArgs e)

        {

            SqlConnection con = new SqlConnection("Data Source='.\\
SQLEXPRESS';Integrated Security='true';Initial Catalog='MEDICAL'");

            SqlCommand cmd = new SqlCommand("SELECT I_name FROM StockInMst", con);

            con.Open();

            SqlDataReader reader = cmd.ExecuteReader();

            AutoCompleteStringCollection SCollection = new
AutoCompleteStringCollection();

            while (reader.Read())

            {

                SCollection.Add(reader.GetString(0));

            }

            txtiname.AutoCompleteCustomSource = SCollection;
            con.Close();


            SqlCommand cmdd = new SqlCommand("SELECT cname FROM companymst", con);

            con.Open();

            SqlDataReader readerr = cmdd.ExecuteReader();

            AutoCompleteStringCollection SCollectionn = new
AutoCompleteStringCollection();

            while (readerr.Read())

            {

                SCollectionn.Add(readerr.GetString(0));

            }

            txtcname.AutoCompleteCustomSource = SCollectionn;

            con.Close();

        }

    }

}


using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.SqlClient;

namespace MEDICAL

{

    public partial class TotalStockReport : Form

    {
        DS.DS_STOCK.StockMst_SelectDataTable SDT = new
DS.DS_STOCK.StockMst_SelectDataTable();

        DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter SAdapter = new
DS.DS_STOCKTableAdapters.StockMst_SelectTableAdapter();

        DS.DS_ITEM.StockInMst_SelectDataTable IDT = new
MEDICAL.DS.DS_ITEM.StockInMst_SelectDataTable();

        DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter IAdapter = new
MEDICAL.DS.DS_ITEMTableAdapters.StockInMst_SelectTableAdapter();


        public TotalStockReport()

        {

            InitializeComponent();

        }


        private void btnitemsearch_Click(object sender, EventArgs e)

        {




            //SDT = SAdapter.SelectBY_INAME(txtiname.Text);

            SDT = SAdapter.SelectBY_INAME(txtiname.Text);

            GVReports.DataSource = SDT;

        }


        private void btncompsearch_Click(object sender, EventArgs e)

        {


        }


        private void TotalStockReport_Load(object sender, EventArgs e)

        {

            SqlConnection con = new SqlConnection("Data Source='.\\
SQLEXPRESS';Integrated Security='true';Initial Catalog='MEDICAL'");

            SqlCommand cmd = new SqlCommand("SELECT I_name FROM StockInMst", con);
            con.Open();

            SqlDataReader reader = cmd.ExecuteReader();

            AutoCompleteStringCollection SCollection = new
AutoCompleteStringCollection();

            while (reader.Read())

            {

                SCollection.Add(reader.GetString(0));

            }

            txtiname.AutoCompleteCustomSource = SCollection;

            con.Close();




            SDT = SAdapter.SelectStock();

            GVReports.DataSource = SDT;


            //SqlCommand cmdd = new SqlCommand("SELECT cname FROM companymst", con);

            //con.Open();

            //SqlDataReader readerr = cmdd.ExecuteReader();

            //AutoCompleteStringCollection SCollectionn = new
AutoCompleteStringCollection();

            //while (readerr.Read())

            //{

            //    SCollectionn.Add(readerr.GetString(0));

            //}

            //txtcname.AutoCompleteCustomSource = SCollectionn;

            //con.Close();

        }

    }

}


using System;
using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;


namespace MEDICAL

{

    public partial class UserMst : Form

    {

        DS.DS_USER.UserMst_SelectDataTable UDT = new
DS.DS_USER.UserMst_SelectDataTable();

        DS.DS_USERTableAdapters.UserMst_SelectTableAdapter UAdapter = new
MEDICAL.DS.DS_USERTableAdapters.UserMst_SelectTableAdapter();

        public string username, strmenu;

        public UserMst(string uname, string strmnu)

        {

            username = uname;

            strmenu = strmnu;

            InitializeComponent();

        }


        private void tabPage2_Click(object sender, EventArgs e)

        {


        }


        private void label7_Click(object sender, EventArgs e)

        {
        }

        private void button1_Click(object sender, EventArgs e)
        {
            if (txtname.Text == "")
            {
                MessageBox.Show("Error", "Medical System");
            }
            else if (txtpass.Text == "")
            {
                MessageBox.Show("Error", "Medical System");

            }
            else if (txtpass.Text != txtcpass.Text)
            {
                MessageBox.Show("Error", "Medical System");
            }
            else
            {
                int isrt = UAdapter.Insert(txtname.Text, txtpass.Text,
System.DateTime.Now.Date);
                txtpass.Text = "";
                txtname.Text = "";
                txtcpass.Text = "";
                MessageBox.Show("User Added Sucssesfully !!", "Medical System");

            }



        }
```

```
private void tabControl1_SelectedIndexChanged(object sender, EventArgs e)
{
    if (tabControl1.SelectedIndex == 0)
    {

    }
    else if (tabControl1.SelectedIndex == 1)
    {
        UDT = UAdapter.SelectUser();
        comboBox1.DataSource = UDT;
        comboBox1.DisplayMember = "U_Name";
        comboBox1.ValueMember = "U_ID";
        comboBox1.Text = "SELECT";

    }
    else if (tabControl1.SelectedIndex == 2)
    {
        UDT = UAdapter.SelectUser();
        dataGridView1.DataSource = UDT;

    }
    else if (tabControl1.SelectedIndex == 3)
    {


    }
}

private void button2_Click(object sender, EventArgs e)
{
    if (comboBox1.Text == username.ToString())

    {

        MessageBox.Show("You can't delete your profile !!", "Medical System");

    }

    else

    {

        int del = UAdapter.Delete(Convert.ToInt32(comboBox1.SelectedValue));

        MessageBox.Show("User Deleted Successfuly !!", "Medical System");

        UDT = UAdapter.SelectUser();

        comboBox1.DataSource = UDT;

        comboBox1.DisplayMember = "U_Name";

        comboBox1.ValueMember = "U_ID";

        comboBox1.Text = "SELECT";

    }

}


private void button3_Click(object sender, EventArgs e)

{

    if (textBox1.Text == "")

    {

        MessageBox.Show("Error", "Medical System");


    }

    else if (textBox1.Text != textBox2.Text)

    {

        MessageBox.Show("Error", "Medical System");

    }

    else

    {

        UAdapter.UserMst_Update_password(username.ToString(), textBox1.Text);

        MessageBox.Show("Password has been changed !!", "Medical System");
```

```
                txtcpass.Text = "";

                txtpass.Text = "";

            }

        }


        private void UserMst_Load(object sender, EventArgs e)

        {

            if (strmenu == "Add")

            {


                tabControl1.SelectedIndex = 0;

            }

            else if (strmenu == "Delete")

            {


                tabControl1.SelectedIndex = 1;

            }

            else if (strmenu == "View")

            {


                tabControl1.SelectedIndex = 2;

            }

            else if (strmenu == "Password")

            {


                tabControl1.SelectedIndex = 3;

            }




        }

    }
```

```
}
```

6. TESTING

6.1 Testing Plan

Once code has been generated, program testing begins. The testing process focuses on the logical internals of the software, ensuring that all statements have been tested, and on the functional externals; that is, conducting tests to uncover errors and ensure that defined input will produce actual results that agree with required results.

Need of Testing

A Successful test is one that finds an undiscovered error. If the testing is conducted successfully, it will uncover errors in the software.

Testing demonstrate that software function appear to be working according to specification, that behavioral and performance requirements appear to have been met.

Testing is conducted provide a good indication of the software reliability and software quality. Testing cannot show the absent of errors and defects, it can show only that software errors and defects are present.

With the project nearing completion, we have taken up the activity of testing the individuals' forms and making sure that the interaction among the various forms is smooth and without any glitches. Before actually beginning to test a few things should be born in mind. Among others these include:

1. All tests should be traceable to customer requirements
2. Test should be planned before the testing begins
3. To be effective, testing should be conducted by an independent third party

6.2 TESTING STRATEGY

1) SYSTEM TEST

The System tests will focus on the behavior of the system. User scenarios will be executed against the system as well as screen mapping and error message testing. Overall, the system tests will test the integrated system and verify that it meets the requirements defined in the requirements document

2) PERFORMANCE TEST

Performance test will be conducted to ensure that the system's response times meet the user expectation and do not exceed the specified performance criteria. During these tests, response times will be measured under heavy stress and/or volume.

3) SECURITY TEST

Security tests will determine how secure the system is. The tests will verify that unauthorized user access to confidential data is prevented.

4) AUTOMATED TEST

A suite of automated tests will be developed to test the basic functionality of the system and perform regression testing on areas of the systems that previously had critical/major defects. The tool will also assist us by executing user scenarios thereby emulating several users.

5) RECOVERY TEST

Recovery tests will force the system to fail in a various ways and verify the recovery is properly performed. It is vitally important that all payroll data is recovered after a system failure & no corruption of the data occurred.

6) DOCUMENTATION TEST

Tests will be conducted to check the accuracy of the user documentation. These tests will ensure that no features are missing, and the contents can be easily understood.

7) USER ACCEPTANCE TEST

Once the hotel management system is ready for implementation, the Payroll department will perform User Acceptance Testing. The purpose of these tests is to confirm that the system is developed according to the specified user requirements and is ready for operational use.

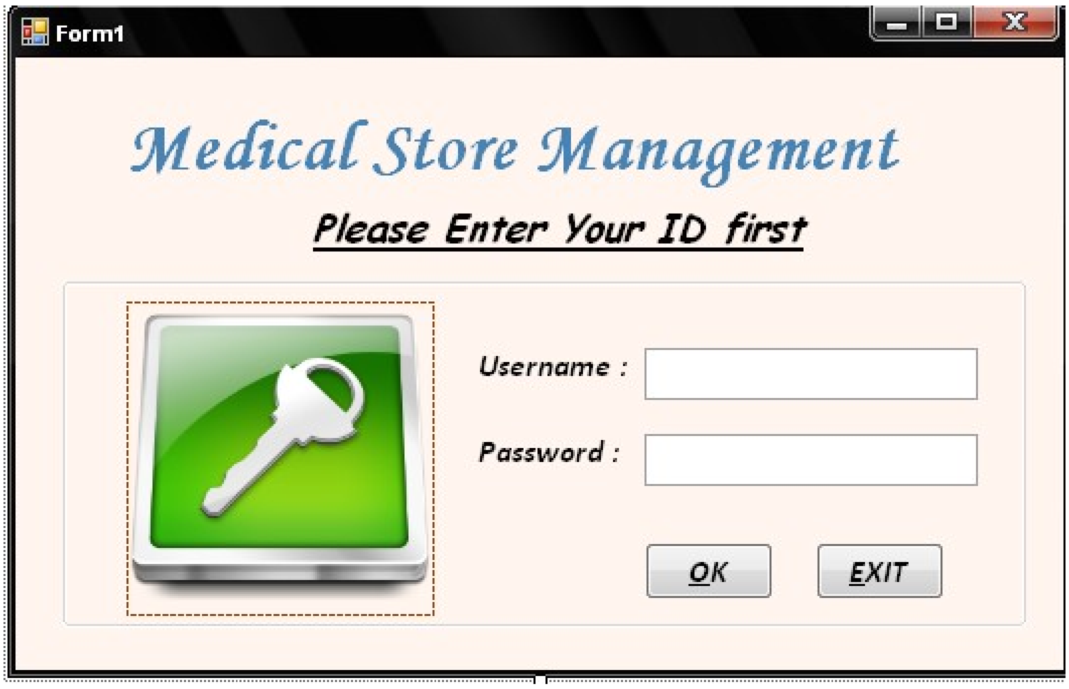
6.3 TESTING METHODS

- **BLACK-BOX TESTING**
  In using this strategy, the tester views the program as a black – box, tester doesn't see the code of the program: Equivalence partitioning, Boundary – value analysis, Error guessing.

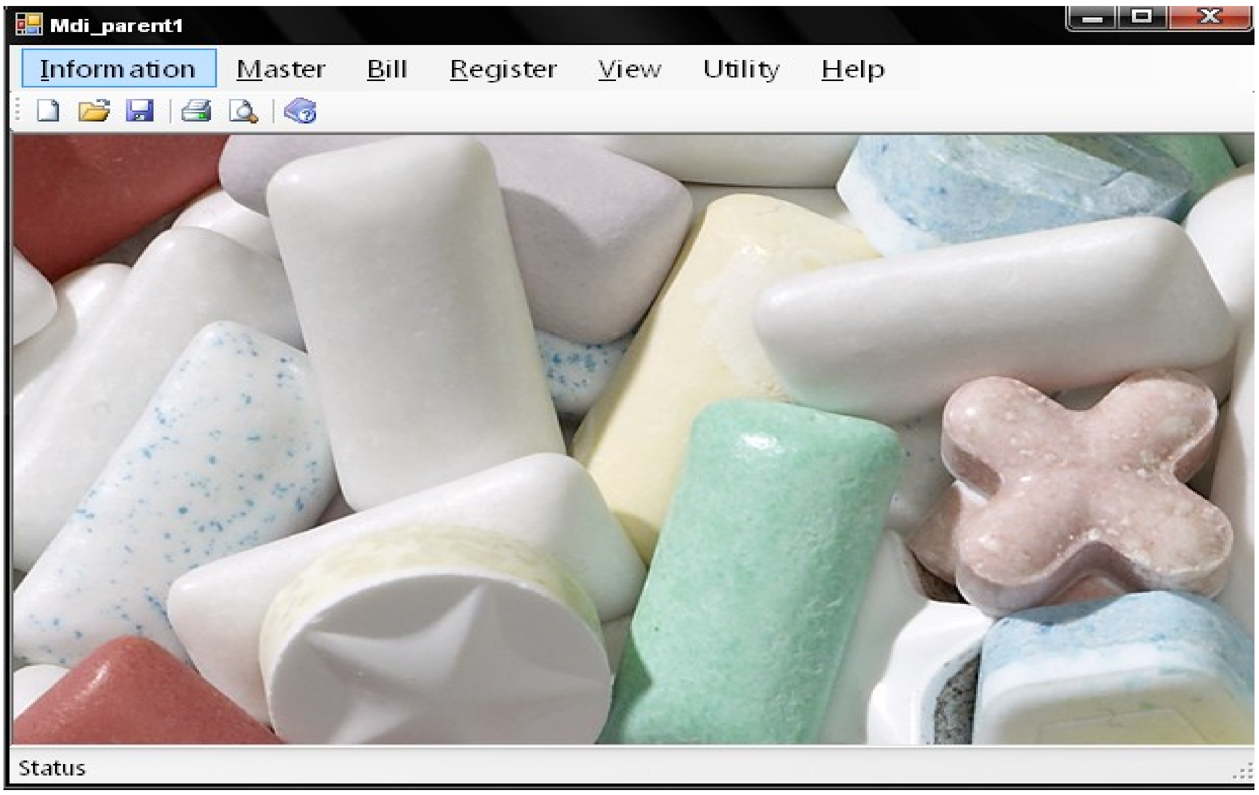
- **WHITE-BOX TESTING**
  In using this strategy, the tester examines the internal structure of the program: Statement coverage, Decision coverage, condition coverage, Decision/Conditional coverage, Multiple – condition coverage.
- **GRAY-BOX TESTING**
  In using this strategy Black box testing can be combine with knowledge of database validation, such as SQL for database query and adding/loading data sets to confirm functions, as well as query the database to confirm expected result.
- **TEST SCRIPT**
  It is type of test file. It is a set of instructions run automatically by a software or hardware test tool.

7. SCREEN SHOTS AND USER MANUAL

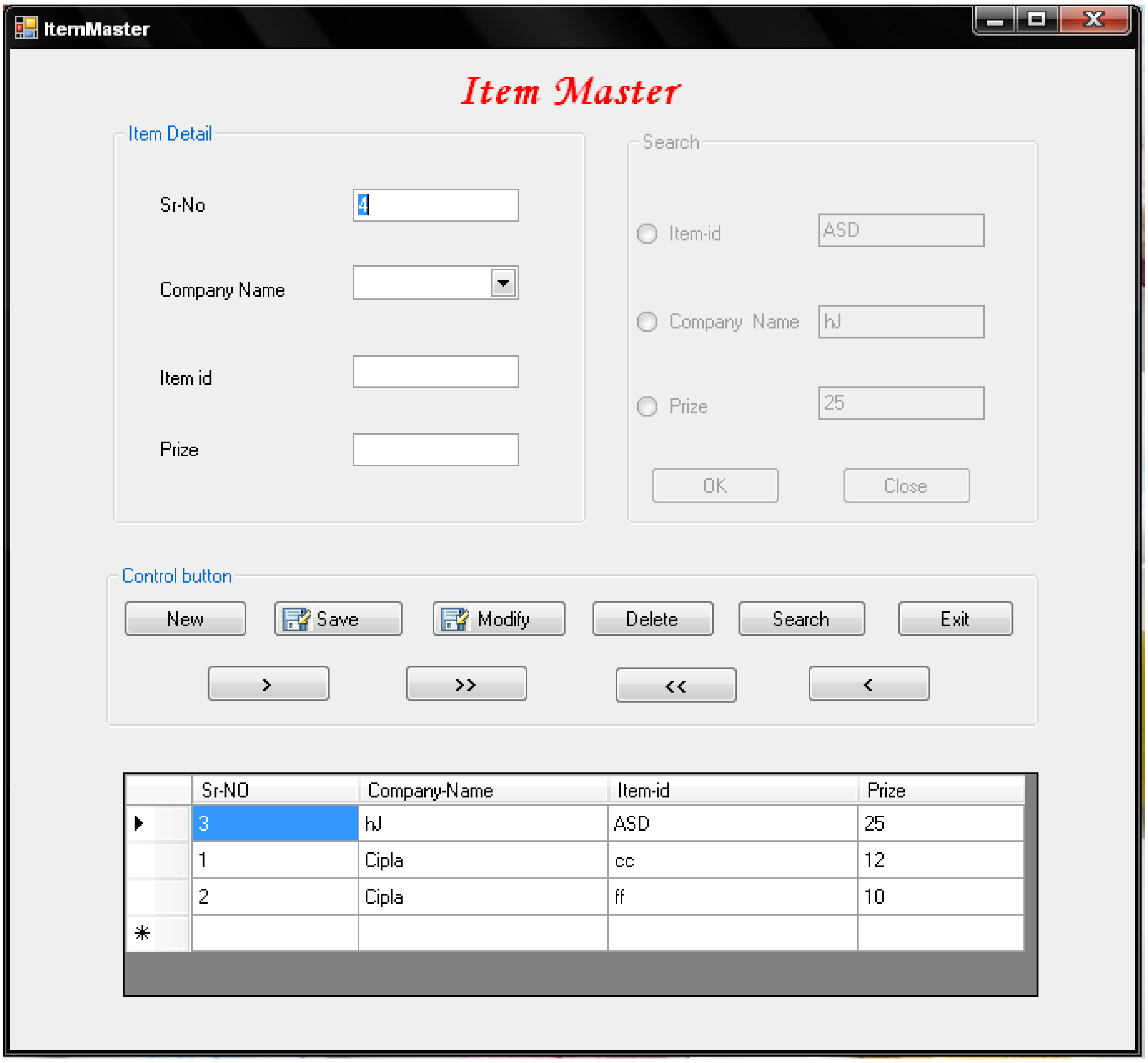
1. **Login Form:**



2. **Mdi Parent:**



3. Item Master:



Sr-No: 4

Company Name:

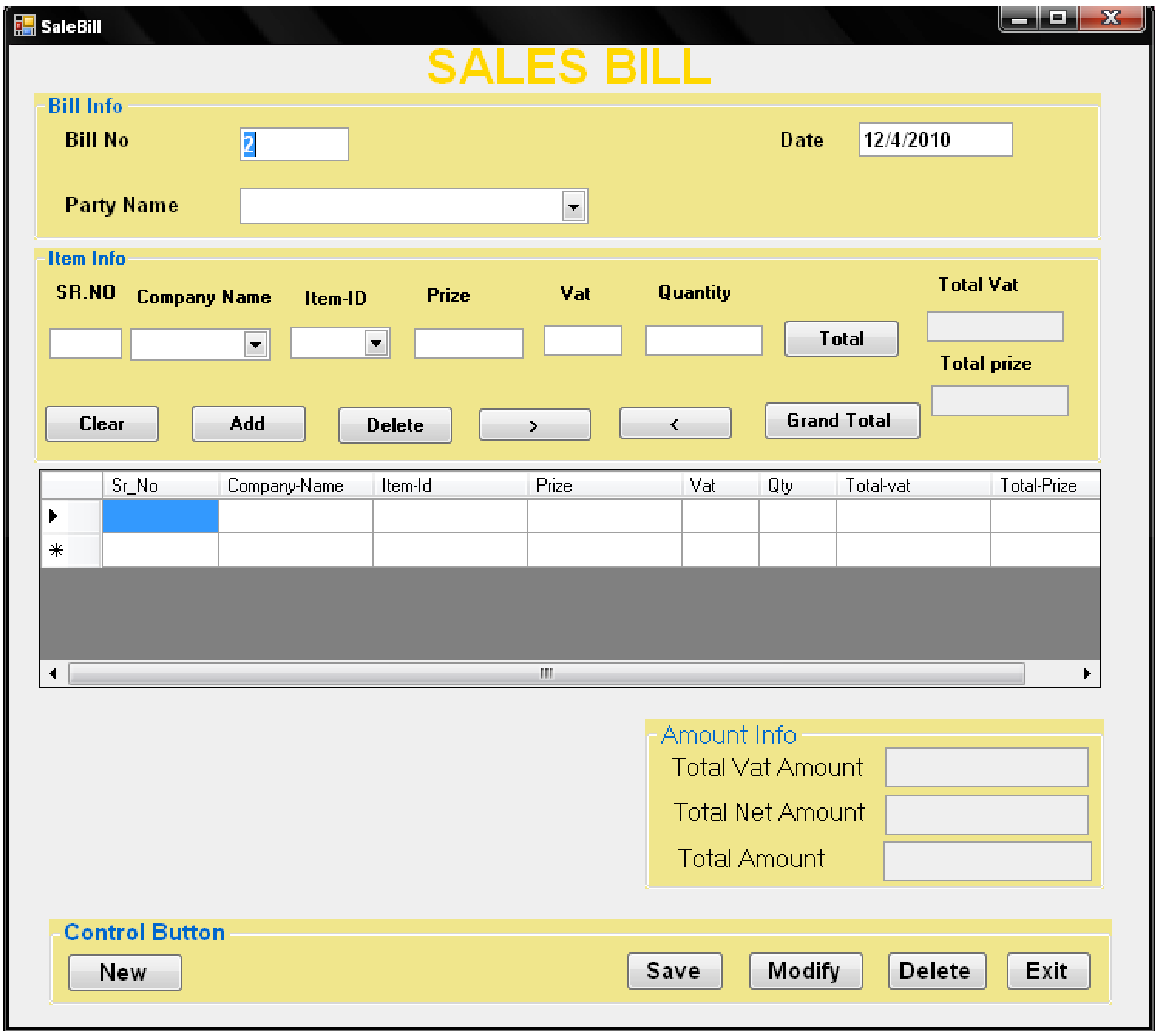
Item id:

Prize:

Search:
- ( ) Item-id: ASD
- ( ) Company Name: hJ
- ( ) Prize: 25

	Sr-NO	Company-Name	Item-id	Prize
▶	3	hJ	ASD	25
	1	Cipla	cc	12
	2	Cipla	ff	10
*				

4. Sales Bill:

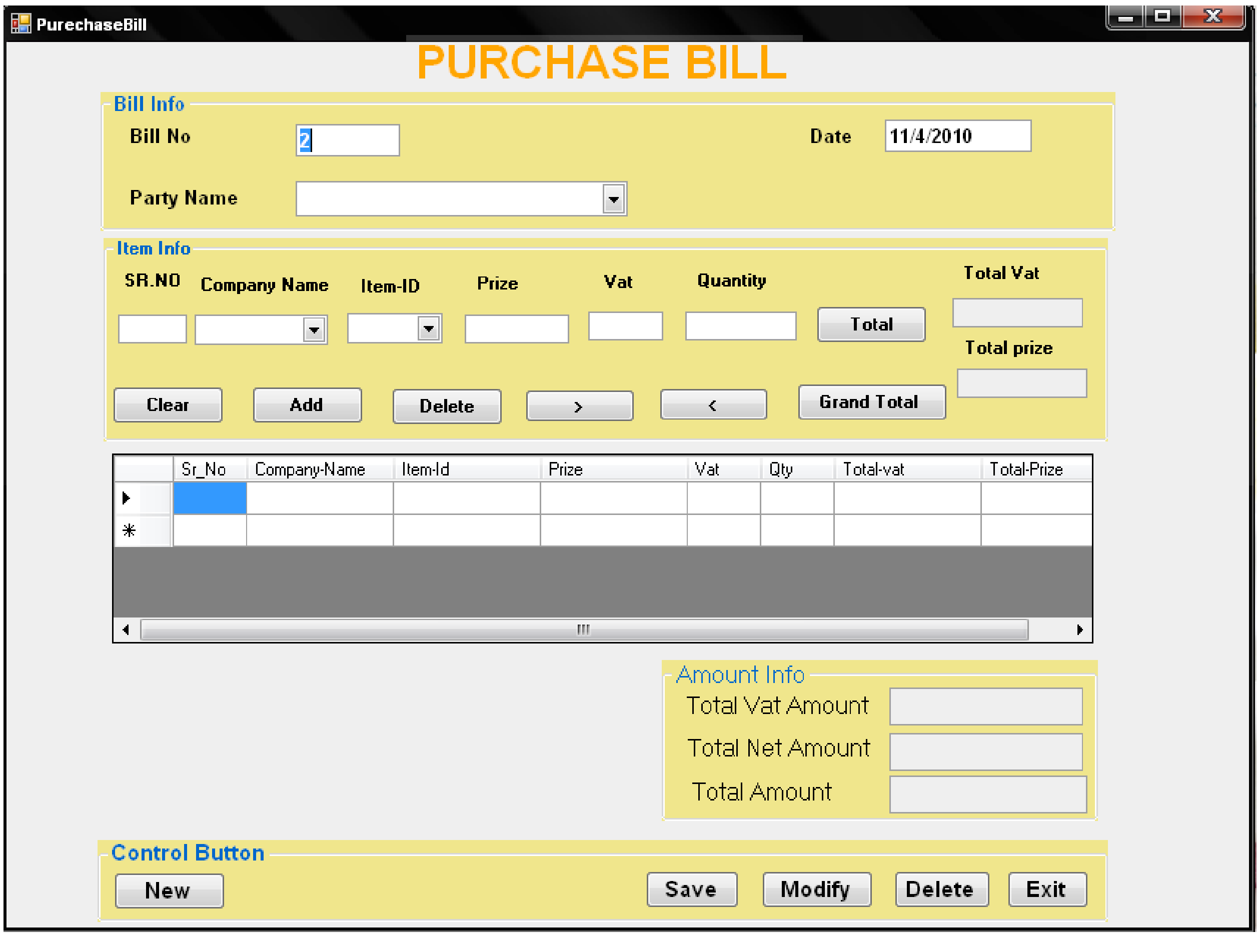


Bill No: 2

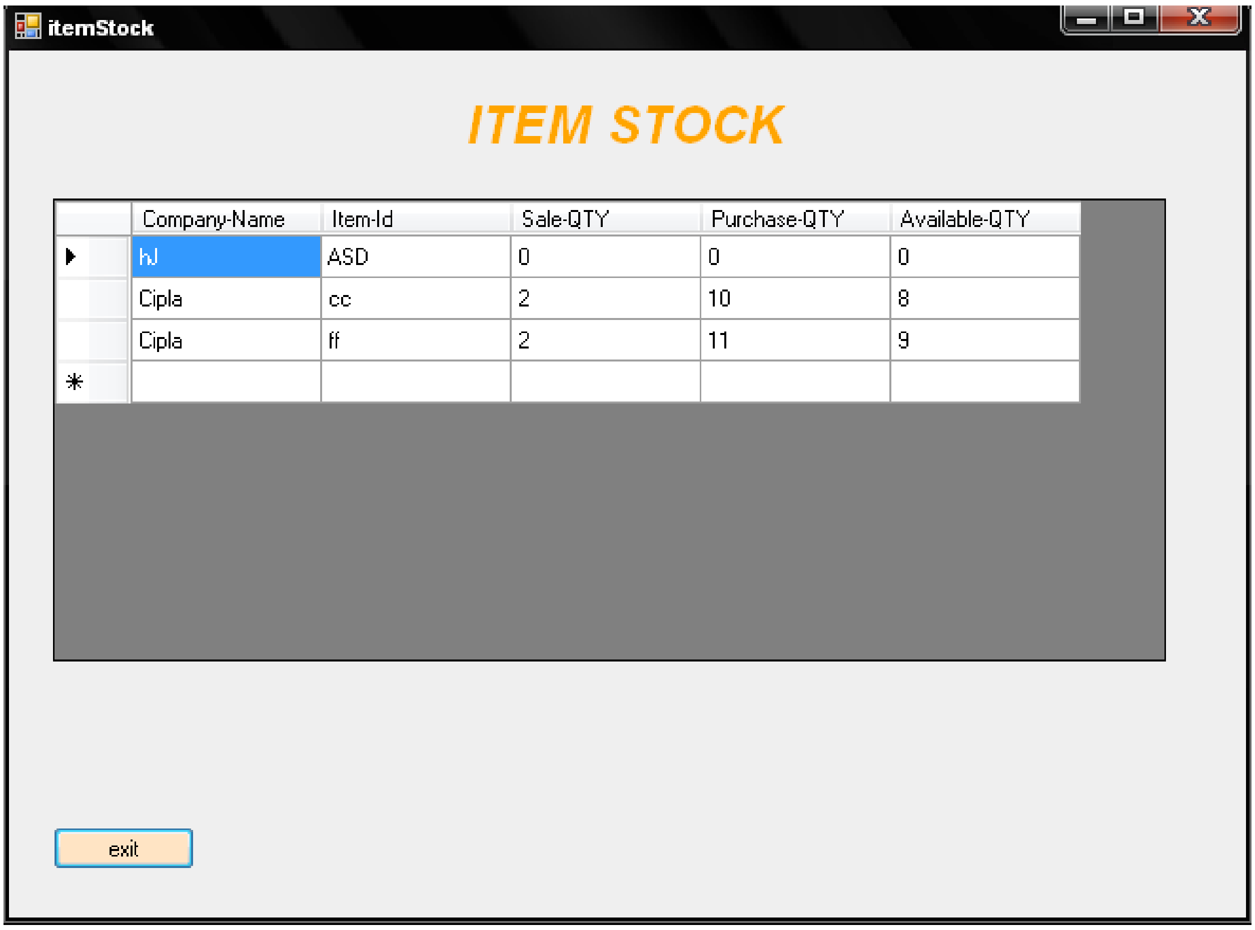
Date: 12/4/2010

Sr_No	Company-Name	Item-Id	Prize	Vat	Qty	Total-vat	Total-Prize

5. Purchase Bill:



6. Item Stoke:



ITEM STOCK

	Company-Name	Item-Id	Sale-QTY	Purchase-QTY	Available-QTY
▶	hJ	ASD	0	0	0
	Cipla	cc	2	10	8
	Cipla	ff	2	11	9
*					

8. Limitations and Future Enhancement

Limitation:

There are many kind of advantages for application but some amounts of limitation are available in this system.
These certain limitation is following here :

- SQL server is must be requiredin the software
- Itcan't be used big medical Firms, it only applicable for small Units.

No interactive designs are included in the software.

Future expansion:

- In next version master can change rate & type of Medicines company& rate.
- In next version master can change rate& type of the table.
- As well as they can also add new Medicines details.
- We will also include the new modules in next version.
- Solve all the limitations of the project.
- Create a data base in XML so system can be more flexible for various Operating Systems.

10. Conclusion:

After the process system that we have made we have to come to the following conclusion.

- This process system is more easy and reliable to use then manual system.
- There is a security system with password in the system that the manual system never had and there was theft of the data being stolen.
- There are difference forms for every record, there necessary to company to have maintained of data.
- There is also feature of showing the previous customer detail.
- Finally we have to come to the conclusion that whatever the medical store requirements is present in this software.

Reference:-

- www.projectguidance.com
- http://e-library.net/
- www.sourcecodeonline.com
- www.developers.net
- www.googel.com
- www.wikipedia.com
- www.medicalshop.com