NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

HOSPITAL AUTOMATION SYSTEM WITH V.B.NET

Graduation Project
COM400

Student: Kamil Selek(20011335)

Supervisor: Mr. Ümit İlhan

Nicosia-2006

ACKNOWLEDGMENTS

It is my pleasure to take this opportunity to express my greatest gratitude to man individuals who have given me a lot of supports during my four-year Undergraduation program in the **Near East University**. Without them, my Graduation Project would not have been successfully completed on time.

First of all, I would like to express my thanks to my supervisor **Mr. Ümit İlhan** for supervising my project. Under the guidance of him I successfully overcome many difficulties and I learned a lot about web designing. In each discussion, he used to explain the problems and answer my questions. He always helped me a lot and I felt remarkable progress during his supervisior. Also I thankes for giving his time during the my study and my advisering.

I also want to thank all my friends and specially **Adem Atçeken, Alper Karakuş, Yahya Göksay and Sinan Çıklaçevik** who supported and helped me all the time.

Finally, special thanks for my family, especially my parents for being patientfull during my undergraduate degree study. I could never have completed my study without their encouragement and endless support.

TABLE OF CONTENTS

ABSTRACT

Automation programs with the development of the technology became compulsory software to make easy the works of the human in large platforms. Because the computers take place in every part of our lives.

At the beginning Data holded on paper, it is moved to the computer with the aim of decrease the data loosing and after a time it spreat to data base usege as a result of being data security, accessibility, data management and ordering facilities.

The program which I prepared for dental department of medicine is a software that can record the patient personel informations safely, record the applied treatments easily with the help of visual interface and control the treatment cost and payments with the help of data base querying .

While preparing this project I used Visual Basic.Net as programing language and SQL Server for database.

INTRODUCTION

The technology is entered to every platform of our life, the usage of computer is spread day by day. Without software the machines are nothing therefore there are software need in many sectors human needed both software and hardware together. The main point is making the user's job easy.

My project is a practise of software which is prepared to facilitate the studies in dental department of medicine. software that can record the patient personel informations safely, record the applied treatments easily with the help of visual interface and control the treatment cost and payments with the help of data base querying .

Chapter 1 I explained the main structure, syntax, usage of Visual Basic.Net which I used in preparing my project.
Chapter2 I explained structure and usage of SQL Server which I used for database.
Cahpter3 I showed the codes which I wrote, methods that I applied and the function of project which I wrote.

1.VISUAL BASIC.NET

1.1. Why Should You Move toVisual Basic.NET?

One of the most common questions today is, "Why should I move to .NET?" .NET is new, and there are many questions about what it can do for you. From a Visual Basic standpoint, it's important to understand some of the dramatic benefits that can be achieved by moving to VB.NET.

1.2.The New Look of Visual Basic

In moving to VB.NET, Microsoft has ditched a number of older, arcane features like GoSub and default properties, and totally reworked features such as arrays and data types. Other native features like the MsgBox function and the Cxxx convert functions have been demoted. These demoted features are still in VB.NET but Microsoft is recommending that you move to using the .NET System classes instead. Of course, depending on your experience and base of existing legacy VB applications, some of the changes may cause considerable pain. More than likely, however, you will soon grow to appreciate the redesigned VB language.

What does the new Visual Basic.NET language mean to the average ASP developer who has written thousands of lines of VBScript code but who has had little exposure to VB proper? If you find yourself in this category of developer, you may experience a short period of bewilderment, as you get accustomed to the wealth of new features offered by VB.NET, features that VBScript never offered. But soon enough, you will start to forget the limited VBScript language and grow to appreciate and even love the much more nimble and full-featured VB.NET.

1.3.Getting Started with VB.NET

Compared to many programming languages, Visual Basic.NET is a fairly easy language to learn. Unlike the C family of languages, VB.NET prefers to use the English language rather than cryptic symbols like &&, ||, and %. Unlike prior versions of the VB language, however, VB.NET is a full-featured object-oriented language that can hold its own when compared to C++, C#, or Java. The remainder of this chapter consists of a walkthrough of the essential elements of the VB.NET language.

1.3.1.Statements and Lines

VB.NET statements can be placed on one or more lines. Unlike C++, C#, and Java, there is no statement terminator character in VB. When continuing a statement across more than one line, you must end continuation lines with a space followed by an underscore character (_).

For example, the following VB.NET statement spans two lines:

```
Function CreateFullName(LastName As String , _ FirstName As String)
```

1.3.2. Comments

You can add comments to your code using the apostrophe (') character. Everything to the right of an apostrophe is ignored by the VB.NET compiler:

```
x = y + 5 'Add 5 to the value of y
```

1.3.3.Operators

Like any programming language, VB.NET has its assortment of operators. The most common of these operators are summarized in Table 1.3.3.

Table 1.3.3

Continued Type	Operator	Purpose	Example
	^=	Exponentiates the value of a variable by an expression and assigns the result to the variable*	x ^= y
Comparison	=	Is equal to	If (x = y)
	<	Is less than	If (x < y)
	<=	Is less than or equal to	If (x <= y)
	>	Is greater than	If (x > y)
	>=	Is greater than or equal to	If (x >= y)
	<>	Is not equal to	If (x <> y)
	Like	Matches a pattern*	If (x Like "p??r")
	Is	Do object variables refer to same object	If (x Is y)
Logical	And	True if both expressions are true	If (x = 3 And y = 4)
	Or	True if one or both expressions are true	If (x = 3 Or y = 4)
	Not	True if the expression is False	If Not (x = 5)
	Xor	True if one expression is true, but not both*	If (x = 3 Xor y = 4)

* This operator was introduced in VB.NET.

You will find a number of examples that use the VB.NET operators scattered about the chapter.

1.3.4.Using Procedures

The basic unit of executable code in VB.NET, as in most programming languages, is the procedure. VB supports two basic types of procedures: the subroutine (or sub) and the function.

1.3.4.1.Subroutines

You declare a subroutine with the Sub statement. For example

Sub HelloWorld() Response.Write("Hello World") End Sub

You call a sub using either of the following statements:

HelloWorld() Call HelloWorld()

1.3.4.2.Functions

Functions in VB.NET are similar in functionality to subroutines with one difference: Functions can return a value to the calling program. You create a function with the Function statement. For example, the following function returns "Hello World" to the calling code:

Function SayHello() Return "Hello World" End Function

1.3.5.Using Variables and Parameters

You use the Dim, Private, Protected, Friend, or Public statements in VB.NET to declare a variable and its data type. Which statement you use depends on where you wish to declare the variable.

To declare a variable from within a subroutine or function, you use the Dim statement.

For example

Function DoSomething()

Dim Counter As Integer End Function

A variable declared using Dim is local to the procedure in which it is declared.

To declare a variable that's global to the entire page, you declare the variable outside of any subroutine or function using the Private statement. For backward compatibility,

Dim also works in this context, but it's best to use Private instead. New for VB.NET, you can both declare a variable and set its initial value in one statement.

For example

Dim Age As Integer = 23 Private Company As String = "Microsoft"

VB.NET supports the data types shown in Table 1.3.5.

Table 1.3.5

Visual Basic.NET Data Types Visual Basic Data Type	.NET Runtime Data Type	Storage Size	Range of Values
Boolean	System.Boolean	4 bytes	True or False
Byte	System.Byte	1 byte	0 to 255 (unsigned)
Char	System.Char	2 bytes	1 Unicode "" character
Date	System.DateTime	8 bytes	January 1, 0001 to December 31,9999 12:00:00 AM

1.3.6.Understanding Visual Basic.NET Syntax and Structure

You may have noticed that there is no entry for Variant in Table 1.3.5. That's because VB.NET no longer supports the Variant data type. However, you can use the generic Object type any place you would have used Variant in prior versions of VB. (In VB.NET, Variant is a synonym for Object.) Unlike prior versions of VB, if you use a declare statement as shown in the following example, all three variables will be declared as integers:

Dim x, y, z As Integer

In prior versions of VB, x and y would be declared as variant variables and only z would be declared as an Integer.

1.3.6.1. Constants

You can use the Const statement to declare a constant. Like a variable, a constant holds a value; however, a constant's value is set at design time and may not change. You can include the Private or Public keyword within the Const statement to alter the scooping of the constant declaration. Here are a few examples:

```
Const Pi As Double = 3.14159
 Private Const CmPerInch As Double = 2.54
Public Const BookTitle As String = "ASP for Developers"
```

In addition to user-defined constants, VB.NET and the .NET Framework define a number of intrinsic constants. For example, you can use the intrinsic constant CrLf anytime you wish to add a carriage return and line feed to a string:

```
MsgString = "An error has occurred in the program." & _ CrLf & "Click on OK to
continue or CANCEL to abort."
```

1.3.6.2.Implicit and Explicit Variable Declarations

VB has always supported implicit variable declarations, which means that you are not required to declare your variables or parameters before using them. However, most professional developers agree that you should not take advantage of this VB feature unless you like bugs in your code. The issue is best demonstrated with an example:

```
Function Multiply(number1, number2)
Return number1 * numbr2 End Function
```

The Multiply function will always return 0 because we misspelled one of the parameters. This happens because VB.NET implicitly declares numbr2 and initializes it to 0 because it is used in a numeric context. You can avoid this type of hard-to-find bug by using Option Explicit or Option Strict. In this example, if you had used either of these options, VB.NET would generate a compile-time error when the page was compiled.

1.3.6.3.Option Explicit Versus Option Strict

VB has always had the Option Explicit declaration, which forces you to declare all your variables, but VB.NET also introduces Option Strict, which goes one step further. In addition to forcing you to declare all your variables, Option Strict restricts the types of implicit conversions that the language allows. When you use Option Strict, VB won't allow conversions where data loss would occur. Option Strict also disallows implicit conversions between numeric and string data types.

To specify Option Explicit, you can use the following page directive at the top of the ASP page:

<%@ Page Explicit="True" %>

To specify Option Strict, you can use the following page directive at the top of the ASP page:

<%@ Page Strict="True" %>

1.3.6.4.Arrays

You create arrays in VB.NET using the Dim, Public, or Private statements. You use parentheses to specify that you wish to declare an array rather than a scalar variable. For example, the following statement creates an array of strings:

Dim Names() As String

Before using an array, you must specify the total number of elements in the array with the ReDim statement:

Dim Names() As String ReDim Names(2) Names(0) = "Mike" Names(1) = "Paul"

All arrays have a lower bound of zero. The number you place between the parentheses of the ReDim statement designates the total number of elements the array will hold. Thus, a value of 2 as shown in this example tells VB that the array will hold two string elements, numbered 0 and 1.

1.3.6.5Optional Parameters

VB.NET supports optional parameters. To create an optional parameter you insert the Optional keyword before the parameter name and you supply the parameter's default value after the data type, like this:

Optional parameter_name As data_type = default_value

The following function takes a string and makes it into an HTML heading of a level specified by the Level parameter. If Level is not specified, it is assumed to be 1:

The HelloWorld.aspx page calls CreateHead twice from the Page_Load subroutine:

<script language="VB" runat="server">

' ...

Sub Page_Load(Src as Object, E as EventArgs)

If Not Page.IsPostBack Then

DisplayMsg.Text = CreateHead("Hello World!", 3)

DisplayMsg.Text &= CreateHead("Hello Universe!")

End If

End Sub ' ... </script>

<asp:label id="DisplayMsg" runat="server" />

The first time the code calls CreateHead with the phrase "Hello World" and Level is equal to 3. The second time the code calls CreateHead with the phrase "Hello Universe" and Levelis not specified, which is interpreted to mean a heading level of 1. This produces a page like the one shown in Figure 1.3.6.5.

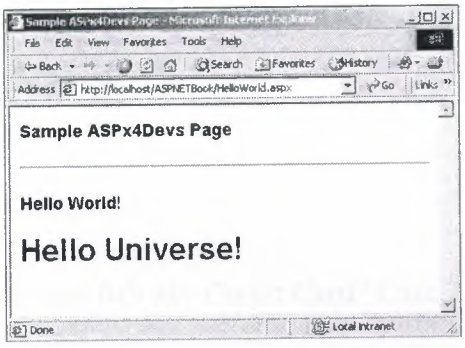

Figure 1.3.6.5

This sample page illustrates the use of optional parameters.

Every parameter to the right of an optional parameter must also be optional.

1.3.7.Using Branching and Looping Structures

More than likely, you'll want to be able to conditionally branch in your code based on the value of a variable or an expression. Or perhaps you'll want to repeatedly loop through a section of code. VB.NET supports several branching and looping structures.

1.3.7.1Branching in VB.NET

You can branch in your code using the If...Then...Else statement or the Select Case statement.

1.3.7.1.1The If...Then...Else Statement

You use the If...Then...Else statement (or simply the If statement) to conditionally execute a piece of code based on the value of some expression. The simplest form of the If statement contains an If clause without any Else clause. Such a statement was used in an earlier example (from HelloWorld.aspx):

If Not Page.IsPostBack Then

DisplayMsg.Text = CreateHead("Hello World!", 3)

DisplayMsg.Text &= CreateHead("Hello Universe!") End If

In this example, the two assignment statements are executed only when Page.IsPostBack is False. Otherwise, no statements are executed.

1.3.7.1.2.The Select...Case Statement

You can also use the Select...Case statement for branching in VB.NET. The Select...Case statement is useful when you wish to check the value of an expression against a list of possible values and execute a different set of code for each value.

This example checks the value of the integer variable PayMethod against a list of possible values. The Select...Case statement sets the value of two string variables to various values depending on the value of PayMethod.

Select Case PayMethod

Case 1

PayMethText = "Visa"

SubmitText = "Complete Order and Bill My Credit Card" Case 2

PayMethText = "Mastercard"

SubmitText = "Complete Order and Bill My Credit Card" Case 3

PayMethText = "American Express"

SubmitText = "Complete Order and Bill My Credit Card" Case 4

PayMethText = "Company PO"

SubmitText = "Complete Order" Case 5

PayMethText = "Check"

SubmitText = "Complete Order" Case Else

PayMethText = "Error"

SubmitText = "Illegal Payment Method: please correct." End Select

Notice the Case Else clause, which is executed if none of the other cases is true.

1.3.7.2Looping in VB.NET

You can loop using the Do...Loop statement, the While...End While statement, the For...Next statement, or the For...Each statement.

1.3.7.2.1The Do...Loop Statement

You can use the Do...Loop statement (or simply Do loop) to execute a set of statements repeatedly, either while some condition is true or until some condition becomes true.

For example, the following code from titles.aspx fills a dropdownlist control with records from the titles table of the pubs sample SQL Server database. We have used a Do loop to move through each of the records returned by the query and added them to the dropdownlist's ListItem collection. (See Chapter 10, "Designing Advanced User Interfaces with Web For List Controls and Custom Web Controls," for more on Web Form list controls and Chapter 15, "Accessing SQL Server Data with the SQL Managed Provider," for more on using ADO.NET with the SQL Managed Provider.)

Sub FillList() Dim ConnectString As String = _ "server=localhost;uid=sa;

pwd=;database=pubs"

Dim SQL As String

Dim PubsCnx As SQLConnection

Dim TitlesQry As SQLCommand

Dim TitlesRdr As SQLDataReader

Dim TitleItem As ListItem

PubsCnx = New SQLConnection(ConnectString)

PubsCnx.Open()

SQL = "SELECT title, title_id FROM titles ORDER BY title"

TitlesQry = New SQLCommand(SQL, PubsCnx)

TitlesQry.Execute(TitlesRdr)

Do While TitlesRdr.Read() TitleItem = New ListItem(TitlesRdr("title"), TitlesRdr("title_id"))

TitleList.Items.Add(TitleItem)

Loop End Sub

The Do...Loop statement from the FillList subroutine uses the SQLDataReader's Read method to retrieve the next record returned by the query. Read advances the current record pointer and returns True if it was able to successfully retrieve a record or False if there are no more records to retrieve.

1.3.7.2.2.The While...End While Statement

The While...End While statement (or simply the While loop) is very similar to the Do...Loop statement. You can use it to execute a set of statements repeatedly, while some condition is true. For example

While i<=Length If Mid(Phrase, i, 1) = "." Then

Exit While

End if

i += 1

End While

1.3.7.2.3.The For...Next Statement

You can use the For...Next statement (or simply the For loop) to repeatedly execute a block of statements a specified number of times. While similar in concept to the Do and While loops, the For loop differs in that it automatically increments a counter variable for you.

The For loop is especially useful for iterating through the items in an array. For example, the following code iterates through all of the elements in the Colors array and displays them on the page:

For i = 0 To UBound(Colors)

Response.Write("
" & i & "=" & Colors(i)) Next

1.3.7.2.4The For...Each Statement

The For...Each statement is a special kind of For...Next loop that is useful for iterating through members of a collection. A collection is an ordered set of items, usually objects, that you can refer to and manipulate as a unit. For example, when working with ADO.NET, you can work with the Errors collection of Error objects.

For example, the following function (from titles2.aspx) returns an HTML table containing a row for each record in the titles table of the SQL Server Pubs database. The DataSet's Table object contains a collection of rows and each row contains a collection of columns. The DisplayTitles function employs two nested For...Each loops to iterate through the TitlesSet dataset. (Datasets and ADO.NET are explained in more detail in Chapter 15 and Chapter 16, "Accessing Non-SQL Server Data with the OLE DB Managed Provider.")

1.3.8.Creating Objects

Prior versions of VB lacked many object-oriented programming (OOP) features that other languages such as C++, Java, and FoxPro have had for years. Fortunately, VB.NET includes strong support for OOP.

1.3.9.OOP Primer

Class, subclass, inheritance, constructor, polymorphism: Object-oriented programming uses lots of fancy new terms that undoubtedly confuse the non-OOP programmer. In this section, you'll find a 10-minute primer of OOP terminology.

1.3.9.1Objects and Classes

Objects are things that you want to represent in your code. Another way to think of an object is as a grouping of properties, methods, and events that are logically tied together. You work with an object by manipulating its properties and methods and reacting to its events.
A class is a template or schema for creating an object. At design time you create the class that serves as the template for creating objects at runtime. An object is thus an instance of a class. And that's one of the neat things about using classes: You can have as many instances of a class as you want, and VB automatically keeps each object's data independent of each other object's data. Another neat thing about classes is that they encapsulate the implementation of the object into a neat package. Encapsulation allows you to separate the implementation of the class (the code inside of the class that makes it work) from its interface (the public properties, methods, and events of the class).

1.3.9.2Inheritance and Polymorphism

One of the big additions to VB.NET is its support for inheritance. Inheritance allows you to create classes that are descendants of another class. When a class inherits from another class, the original class is termed the base class(also sometimes referred to as the superclass or parent class) and the class that inherits from the base class is called the derived class (also sometimes referred to as the subclass or child class).

VB.NET supports the overriding of a base class's methods with alternate implementations. Polymorphism is the ability of different classes to support the properties and methods with the

same name but with different implementations. VB.NET's support for overriding allows your classes to support polymorphism.

1.3.10.Windows Forms

Windows Forms is a set of classes that encapsulates the creation of the graphical user interface (GUI) portion of a typical desktop application. Previously, each programming language had its own way of creating windows, text boxes, buttons, etc. This functionality has all been moved into the .NET Framework class library—into the types located in the System.Windows.Forms namespace. Closely related is the System.Drawing namespace, which contains several types used in the creation of GUI
applications. The capabilities provided by the types in the System.Drawing namespace are commonly referred to as GDI+ (discussed more fully later in this chapter). In this chapter, we'll examine the form (or window) as the central component in a classic desktop application. We'll look at how forms are programmatically created and how they're hooked to events. We'll also examine how multiple forms in a single application relate to one another and how you handle forms in an application that has one or more child forms. Finally, we'll discuss two topics,printing and 2-D graphics, that are relevant to desktop application development.

1.3.10.1. Creating a Form

The easiest way to design a form is to use the Windows Forms Designer in Visual Studio .NET. The developer can use visual tools to lay out the form, with the designer translating the layout into Visual Basic .NET source code. If you don't have Visual Studio .NET, you can write the Visual Basic .NET code directly and not use the designer at all. This section will demonstrate both methods.

Programmatically, a form is defined by deriving a class from the Form class (defined in System.Windows.Forms). The Form class contains the know-how for displaying an empty form, including its title bar and other amenities that we expect from a Windows form. Adding members to the new class and overriding members inherited from the Form class add visual elements and behavior to the new form.

1.3.10.1.1 Creating a Form Using Visual Studio .NET

To create a GUI application in Visual Studio .NET:

1. Select File New Project. The New Project dialog box appears, as shown in Figure 1.3.10.1.1.

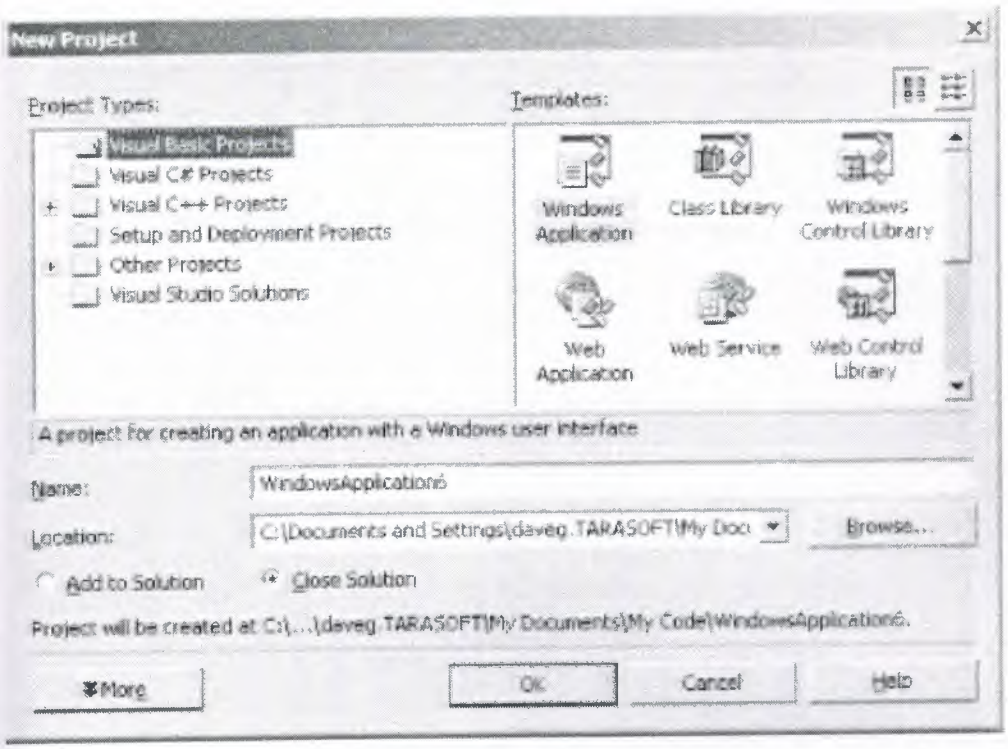

Figure 1.3.10.1.1

2. Select Visual Basic Projects in the Project Types pane on the left side of the dialog box.

3. Select Windows Application in the Templates pane on the right side of the dialog box.

4. Enter a name in the Name text box.

5. Click OK. Visual Studio .NET creates a project with a form in it and displays the form in a designer, as shown in Figure 1.3.10.1.2

.

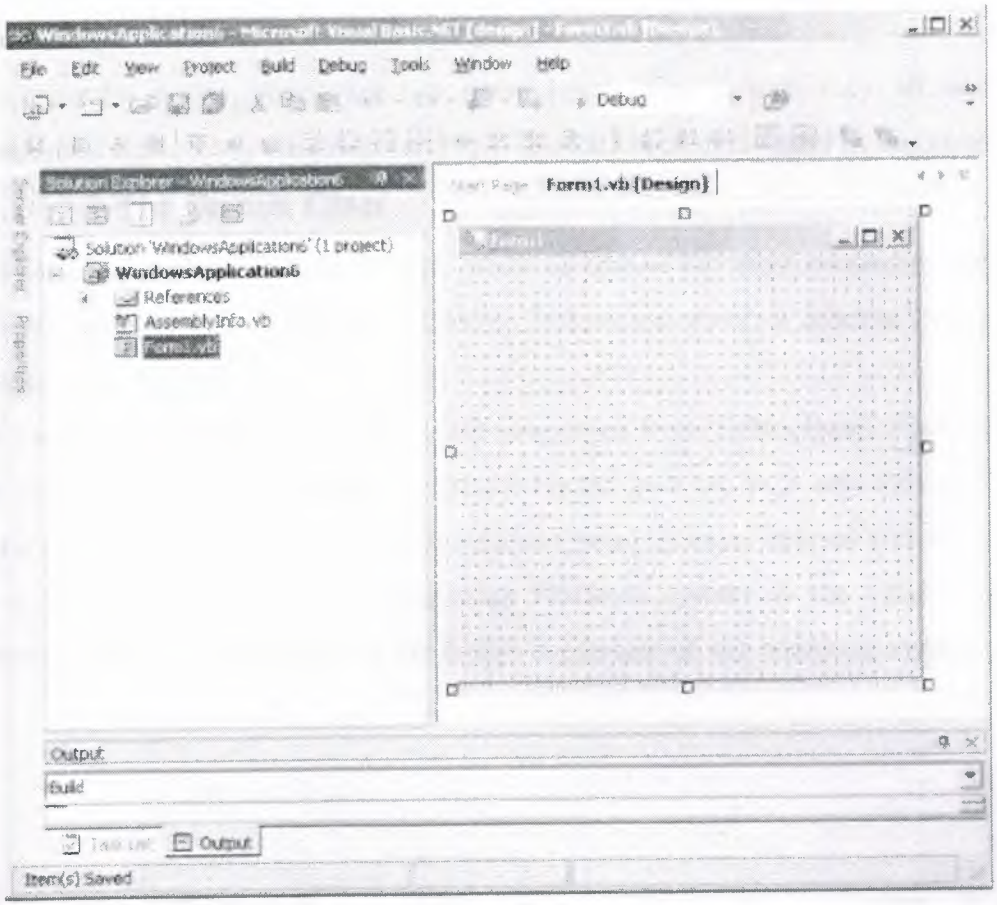

Figure 1.3.10.1.2

1.3.10.2.Controls, Common Dialog Boxes, and Menus

By themselves, one or more forms provide very little functionality to most desktop applications. For the most part, forms are valuable insofar as they serve as containers for controls. In this chapter, we'll complete our discussion of building desktop applications by focusing on the objects that forms contain—in particular, controls and components, common dialogs, and menus.

1.3.10.2.1 Common Controls and Components

This section contains a summary of the controls and components defined in the System.Windows.Forms namespace. *Components* are classes derived from the Component class(defined in the System.ComponentModel namespace). They may or may not provide a visual interface.

They are often used as elements of forms but don't have to be. *Controls* are classes derived from the Control class (defined in the System.Windows.Forms namespace). Controls

generally are used to build the visual appearance of a form. The Control class itself is derived from the Component class, so controls are also components.

The common dialog boxes are not listed here, even though they all derive from the Component class. They are given their own section, Section 5.4 later in this chapter.

1.3.10.2.2.The Button Class

This class represents a button control, which is one of the most commonly used controls in Windows applications. The Button class's Click event, which it inherits from Control, is its most commonly used event.

The Button class inherits two important properties from ButtonBase: FlatStyle and Image. The first determines the appearance of the button and can take any value of the FlatStyle enumeration: Flat, Popup, Standard (the default), and System. Buttons with these four settings are shown in Figure1.3.10.2.2. Assigning FlatStyle.System as the value of the FlatStyle property makes the appearance of the button dependent on the operating system

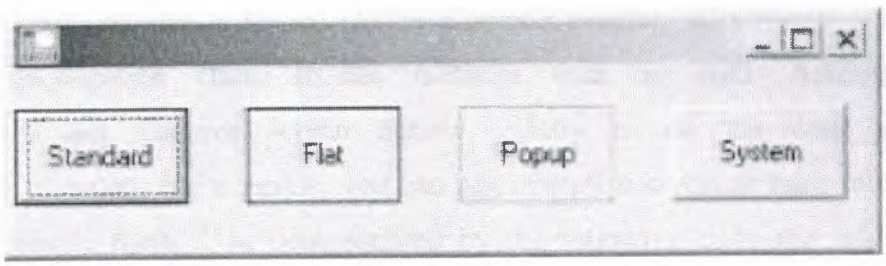

Figure1.3.10.2.2.

The Image property allows you to embed an image into a button. The following code shows how to programmatically set the Image property of Button:

Button1.Image = New System.Drawing.Bitmap(filepath)

1.3.10.2.3. The CheckBox Class

The CheckBox class represents a checkbox control. Its appearance is determined by its Appearance property, which can take either value of the Appearance enumeration: Button or Normal (the default). The Button value is rarely used because this setting makes the checkbox look like a Button to uncheck it.

1.3.10.2.4The ComboBox Class

Both the ComboBox and ListBox classes derive from the ListControl class; therefore, the ComboBox class is very similar to the ListBox class and has properties and methods similar to those of the ListBox class.

1.3.10.2.5.The DateTimePicker Class

The DateTimePicker class represents a control that allows users to select a date in the calendar, just like the MonthCalendar control. Unlike MonthCalendar, however, the DateTimePicker control only displays a box, which looks like a combo box, containing the selected date. When the user clicks the arrow, the control displays a drop-down calendar similar to the MonthCalendar control, from which the user can select a date. This drop-down portion closes as soon as the user selects a date. T he usercan also click on the day, date, month, or year portion of the control for editing. The DateTimePicker class has MinDate and MaxDate properties that are similar to the ones in the MonthCalendar class. To set the current date or to obtain the selected date, use the Value property of the DateTimePicker class. The selected date is readily available as a DateTime data type.

1.3.10.2.6.The GroupBox Class

As the name implies, a GroupBox control is used for grouping other controls, such as radio buttons or checkboxes; it corresponds to the Frame control in Visual Basic 6.0. A GroupBox grouping two radio buttons is shown in The Controls property of GroupBox represents a Control.ControlCollection class. It has methods such as Add, AddRange, Clear, GetEnumerator, and Remove, which behave exactly as do the same methods in Form.ControlCollection. For example, you can add several controls at once to a GroupBox using its AddRange method, as demonstrated by the following code that adds two radio buttons to a GroupBox named groupBox1:

groupBox1.Controls.AddRange(New Control( ) {radioButton1, radioButton2})

1.3.10.2.7.The ImageList Class

The ImageList class allows you to manage a collection of images. The most important property of this class is Images, which returns an ImageList.ImageCollection object. The ImageList.ImageCollection class has methods to add and remove images from the collection.

The Add method of the ImageList.ImageCollection class adds a bitmap image or an icon to the ImageList's image collection.

1.3.10.2.8.The Label Class

This class represents a Label control. Its appearance is determined by two properties: BorderStyle and FlatStyle. The BorderStyle property defines the appearance of the control's border and takes any of 190 the three members of the BorderStyle enumeration: None (the default), FixedSingle, and Fixed3D.

.

1.3.10.2.9.The LinkLabel Class

The LinkLabel class represents a label that can function as a hyperlink, which is a URL to a web site. Its two most important properties are Text and Links. The Text property is a String that defines the label of the LinkLabel object. You can specify that some or all of the Text property value is a hyperlink. For example, if the Text property has the value "Click here for more details", you can make the whole text a hyperlink, or you can make part of it (e.g., the word "here") a hyperlink. How to do this will become clear after the second property is explained. For a LinkLabel to be useful, it must contain at least one hyperlink. The Links property represents a LinkLabel.LinkCollection class of the LinkLabel object. You use the Add method of the LinkLabel.LinkCollection class to add a LinkLabel.Link object. is linked to the URL

The LinkLabel class has a number of properties that are related to the appearance of a LinkLabel.

1.3.10.2.10.The ListBox Class

The ListBox class represents a box that contains a list of items. The following are its more importantproperties:

MultiColumn

This is a Boolean that indicates whether the listbox has more than one column. Its default value is False.

ColumnWidth

In a multicolumn listbox, this property represents the width of each column in pixels. By default, the value of this property is zero, which makes each column have a default width.

Items

This is the most important property of the ListBox class. It returns the ListBox.ObjectCollection class, which is basically the Items collection in the ListBox. You can programmatically add an item using the Add method or add a range of items using the AddRange method of the ListBox.ObjectCollection class. For example, the following code adds the names of vegetables and fruits to a ListBox object named listBox1:

```
listBox1.Items.AddRange(New Object( ) _
{"apple", "avocado", "banana", "carrot", _
"mandarin", "orange"})
```

SelectionMode

This property determines whether multi-item selection is possible in a ListBox object. It can be assigned any member of the SelectionMode enumeration: None, One (the default value), MultiSimple, and MultiExtended. Both MultiSimple and MultiExtended allow the user to select more than one item. However, MultiExtended allows the use of the Shift, Ctrl, and arrow keys to make a selection.

SelectedIndex

This is the index of the selected item. The index is zero-based. If more than one item is selected, this property represents the lowest index. If no item is selected, the property returns -1.

SelectedIndices

This read-only property returns the indices to all items selected in a ListBox object in the form of a ListBox.SelectedIndexCollection object. The ListBox.SelectedIndexCollection class has a Count property that returns the number of selected indices and an Item property that returns the index number. For example, the following code returns the index number of all selected items in a ListBox control named listBox1:

```
Dim selectedIndices As ListBox.SelectedIndexCollection
' Obtain the selected indices.
selectedIndices = listBox1.SelectedIndices
' Get the number of indices.
Dim count As Integer = selectedIndices.Count
Dim i As Integer
For i = 0 To count - 1
Console.WriteLine(selectedIndices(i))
Next
```

SelectedItem

This read-only property returns the selected item as an object of type Object. You must cast the returned value to an appropriate type, which is normally String. If more than one item is selected, the property returns the item with the lowest index.

SelectedItems

This read-only property returns all items selected in a ListBox object in the form of a ListBox.SelectedObjectCollection object. The ListBox.SelectedObjectCollection class has a Count property that returns the number of items in the collection and an Item property that you can use to obtain the selected item. For example, the following code displays all the selected items of a ListBox control called listBox1:

```
Dim selectedItems As ListBox.SelectedObjectCollection
selectedItems = listBox1.SelectedItems
Dim count As Integer = selectedItems.Count
Dim i As Integer
For i = 0 To count - 1
Console.WriteLine(selectedItems(i))
Next
```

Sorted

A value of True means that the items are sorted. Otherwise, the items are not sorted. By default, the value of this property is False.

Text

This is the currently selected item's text.

TopIndex

This is the index of the first visible item in the ListBox. The value changes as the user scrolls through the items.

1.3.10.2.11 The ListBox.ObjectCollection Class

This class represents all the items in a ListBox object. It has a Count property that returns the number of items in the ListBox and an Item property that returns the item object in a certain index position. The following sample code reiterates all the items in a ListBox control named listBox1:

```
Dim items As ListBox.ObjectCollection
items = ListBox1.Items
Dim count As Integer = items.Count
Dim i As Integer
For i = 0 To count - 1
Console.WriteLine(items(i))
Next
```

In addition, the ListBox.ObjectCollection class has the following methods:

Add

Adds an item to the ListBox object. Its syntax is:

ListBox.ObjectCollection.Add(*item*)

where *item* is data of type Object that is to be added to the collection. The method returns the zero-based index of the new item in the collection.

AddRange

Adds one or more items to the ListBox object. Its most common syntax is:

ListBox.ObjectCollection.AddRange(*items( )*)

where *items* is an array of objects containing the data to be added to the ListBox.

Clear

Clears the ListBox, removing all the items. Its syntax is:

ListBox.ObjectCollection.Clear( )

Contains

Checks whether an item can be found in the list of items. Its syntax is:

ListBox.ObjectCollection.Contains(*value*)

where *value* is an Object containing the value to locate in the ListBox. The method returns True if *value* is found; otherwise, it returns False.

CopyTo

Copies all items to an object array. Its syntax is:

ListBox.ObjectCollection.CopyTo(*dest*( ), *arrayIndex*) where *dest* is the Object array to which the ListBox items are to be copied, and *arrayIndex* is the starting position within *dest* at which copying is to begin.

IndexOf

Returns the index of a particular item. Its syntax is: ListBox.ObjectCollection.IndexOf(*value*)

where *value* is an Object representing the item to locate in the collection. The method returns the item's index. If the item cannot be found, the method returns -1.

Insert

Inserts an item into the ListBox at the specified index position. Its syntax is:

ListBox.ObjectCollection.Insert(*index, item*)

where *index* is the zero-based ordinal position at which the item is to be inserted, and *item* is an Object containing the data to be inserted into the collection.

Remove

Removes the item that is passed as an argument to this method from the ListBox. Its syntax is:

ListBox.ObjectCollection.Remove(*value*)

where *value* is an Object representing the item to remove from the collection.

RemoveAt

Removes an item at the specified index position. Its syntax is:

ListBox.ObjectCollection.RemoveAt(*index*) where *index* is the zero-based ordinal position in the collection of the item to be removed.

1.3.10.2.12.The ListView Class

A ListView is a container control that can hold a collection of items. Each item in a ListView can have descriptive text and an image, and the items can be viewed in four modes. The righthand pane of Windows Explorer is a ListView control.

1.3.10.2.13.The MonthCalendar Class

The MonthCalendar class represents a control that displays days of a month. A MonthCalendar control is shown in Figure 5-5. By default, when first displayed, the control displays the current month on the user's computer system. Users can select a day by clicking on it or select a range of dates by holding the Shift key while clicking the date at the end of the desired range. Users can also scroll backward and forward to previous or upcoming months, or they can click on the month part and more quickly

select one of the 12 months. To change the year, users can click on the year part and click the scrollbar that appears.

1.3.10.2.14.The Panel Class

A panel is a container that can hold other controls. Panels are typically used to group related controls in a form. Like the PictureBox class, the Panel class has a BorderStyle property that defines the panel's border and can take as its value any member of the BorderStyle enumeration: None (the default value), FixedSingle, and Fixed3D.

You can add controls to a Panel object using the Add method or the AddRange method of the Control.ControlCollection class. The following code adds a button and a text box to a Panel control called panel1:

1.3.10.2.15.The PictureBox Class

The PictureBox class represents a control to display an image. Loading an image into this control is achieved by assigning a System.Drawing.Bitmap object to its Image property, as the following code does:

```
Dim pictureBox1 As PictureBox = New PictureBox( )
pictureBox1.Image = New System.Drawing.Bitmap("c:\tv.bmp")
pictureBox1.Location = New System.Drawing.Point(72, 64)
pictureBox1.Size = New System.Drawing.Size(144, 128)
```

Me.Controls.Add(pictureBox1)

In addition, the PictureBox class has the BorderStyle and SizeMode properties. The BorderStyle property determines the PictureBox object's border and can take as its value any member of the BorderStyle enumeration: None (the default value), FixedSingle, and Fixed3D. The SizeMode property determines how the image assigned to the Image property is displayed. The SizeMode property can take any of the members of the PictureBoxSizeMode enumeration: AutoSize, CenterImage, Normal (the default value), and StretchImage.

1.3.10.2.16.The RadioButton Class

The RadioButton class represents a radio button. When you add more than one radio button to a form, those radio buttons automatically become one group, and you can select only one button at a time. If you want to have multiple groups of radio buttons on a form, you need to use a GroupBox or Panel control to add radio buttons in the same group to a single GroupBox or Panel. The following code shows how you can add two radio buttons to a GroupBox and then add the GroupBox to a form. Notice that you don't need to add each individual radio button to a form:' Declare and instantiate a GroupBox and two radio buttons.

Dim groupBox1 As GroupBox = New GroupBox( )

Dim radioButton1 As RadioButton = new RadioButton( )

Dim radioButton2 As RadioButton = new RadioButton( )

1.4.Developing Database Applications

Many software applications benefit from storing their data in database management systems. A database management system is a software component that performs the task of storing and retrieving large amounts of data. Examples of database management systems are Microsoft SQL Server

All examples in this chapter assume that the following declaration appears in the same file as the code:

Imports System.Data

Examples that use SQL Server also assume this declaration:

Imports System.Data.SqlClient

and examples that use Access assume this declaration:

Imports System.Data.OleDb

1.4.1. A Brief History of Universal Data Access

Database management systems provide APIs that allow application programmers to create and access databases. The set of APIs that each manufacturer's system supplies is unique to that manufacturer. Microsoft has long recognized that it is inefficient and error prone for an applications programmer to attempt to master and use all the APIs for the various available database management systems. What's more, if a new database management system is released, an existing application can't make use of it without being rewritten to understand the new APIs. What is needed is a common database API.

Microsoft's previous steps in this direction included Open Database Connectivity (ODBC), OLE DB, and ADO (not to be confused with ADO.NET). Microsoft has made improvements with each new technology.

With .NET, Microsoft has released a new mechanism for accessing data: ADO.NET. The name is a carryover from Microsoft's ADO (ActiveX Data Objects) technology, but it no longer stands for ActiveX

Data Objects—it's just ADO.NET. To avoid confusion, I will refer to ADO.NET as ADO.NET and to ADO as *classic* ADO.

If you're familiar with classic ADO, be careful—ADO.NET is not a descendant, it's a new technology. In order to support the Internet evolution, ADO.NET is highly focused on disconnected data and on the ability for anything to be a source of data. While you will find many concepts in ADO.NET to be similar to concepts in classic ADO, it is not the same.

1.4.2. Managed Providers

When speaking of data access, it's useful to distinguish between providers of data and consumers of data. A *data provider* encapsulates data and provides access to it in a generic way. The data itself can be in any form or location. For example, the data may be in a typical database management system such as SQL Server, or it may be distributed around the world and accessed via web services. The data provider shields the data consumer from having to know how to reach the data. In ADO.NET, data providers are referred to as *managed providers*.

A *data consumer* is an application that uses the services of a data provider for the purposes of storing, retrieving, and manipulating data. A customer-service application that manipulates a customer database is a typical example of a data consumer. To consume data, the application must know how to access one or more data providers.

ADO.NET is comprised of many classes, but five take center stage:

Connection

Represents a connection to a data source.

Command

Represents a query or a command that is to be executed by a data source.

DataSet

Represents data. The DataSet can be filled either from a data source (using a DataAdapter object) or dynamically.

DataAdapter

Used for filling a DataSet from a data source.

DataReader

Used for fast, efficient, forward-only reading of a data source.

With the exception of DataSet, these five names are not the actual classes used for accessing data sources. Each managed provider exposes classes specific to that provider. For example, the SQL Server managed provider exposes the SqlConnection, SqlCommand, SqlDataAdapter, and SqlDataReader classes. The DataSet class is used with all managed providers.

Any data-source vendor can write a managed provider to make that data source available to ADO.NET data consumers. Microsoft has supplied two managed providers in the .NET Framework:SQL Server and OLE DB.

The examples in this chapter are coded against the SQL Server managed provider, for two reasons.

The first is that I believe that most programmers writing data access code in Visual Basic .NET will be doing so against a SQL Server database. Second, the information about the SQL Server managed provider is easily transferable to any other managed provider.

1.4.3 Connecting to a SQL Server Database

To read and write information to and from a SQL Server database, it is necessary first to establish a connection to the database. This is done with the SqlConnection object, found in the System.Data.SqlClient namespace. Here's an example:

```
' Open a database connection.
Dim strConnection As String = _
"Data Source=localhost;Initial Catalog=Northwind;" _
& "Integrated Security=True"
```

```
Dim cn As SqlConnection = New SqlConnection(strConnection)
cn.Open( )
```

This code fragment instantiates an object of type SqlConnection, passing its constructor a connection string. Calling the SqlConnection object's Open method opens the connection. A connection must be open for data to be read or written, or for commands to be executed. When you're finished accessing the database, use the Close method to close the connection: '

```
Close the database connection.
cn.Close( )
```

The connection string argument to the SqlConnection class's constructor provides information that allows the SqlConnection object to find the SQL Server database. The connection string shown in the earlier code fragment indicates that the database is located on the same machine that is running the code snippet (Data Source=localhost), that the database name is Northwind (Initial Catalog=Northwind), and that the user ID that should be used for logging in to SQL Server is the current Windows login account (Integrated Security=True)

1.4.5. Reading Data into a DataSet

The DataSet class is ADO.NET's highly flexible, general-purpose mechanism for reading and updating data. how to issue a SQL SELECT statement against the SQL Server Northwind sample database to retrieve and display the names of companies located in London. The resulting display is shown in Figure.

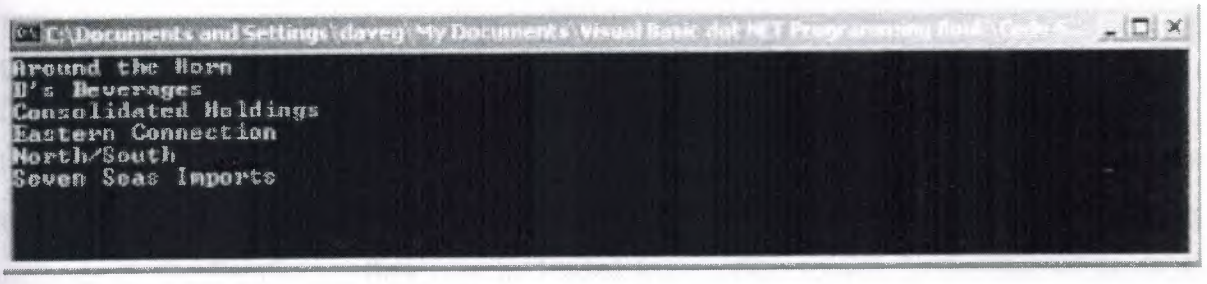

Figure 1.4.5.. The output generated by the code

Retrieving data from SQL Server using a SQL SELECT statement

```
' Open a connection to the database.
Dim strConnection As String = _
"Data Source=localhost; Initial Catalog=Northwind;" _
& "Integrated Security=True"
Dim cn As SqlConnection = New SqlConnection(strConnection)
cn.Open( )
```

```
' Set up a data set command object.
Dim strSelect As String = "SELECT * FROM Customers WHERE City = 'London'"
Dim dscmd As New SqlDataAdapter(strSelect, cn)
' Load a data set.
Dim ds As New DataSet( )
dscmd.Fill(ds, "LondonCustomers")
' Close the connection.
cn.Close( )
' Do something with the data set.
Dim dt As DataTable = ds.Tables.Item("LondonCustomers")
Dim rowCustomer As DataRow
For Each rowCustomer In dt.Rows
Console.WriteLine(rowCustomer.Item("CompanyName"))
Next
```

The code in Example 8-1 performs the following steps to obtain data from the database:

1. Opens a connection to the database using a SqlConnection object.

2. Instantiates an object of type SqlDataAdapter in preparation for filling a DataSet object. SQL SELECT command string and a Connection object are passed to the SqlDataAdapter object's constructor.

3. Instantiates an object of type DataSet and fills it by calling the SqlDataAdapter object's Fill method.

1.5. The DataSet Class

The DataSet class encapsulates a set of tables and the relations between those tables.

The DataSet is alwayscompletely disconnected from any data source. In fact, the DataSet has no knowledge of the source of its tables and relations. They may be dynamically created using methods on the DataSet, or they may be loaded from a data source. In the case of the SQL Server managed provider, a DataSet can be loaded from a SQL Server database using an SqlDataAdapter object.

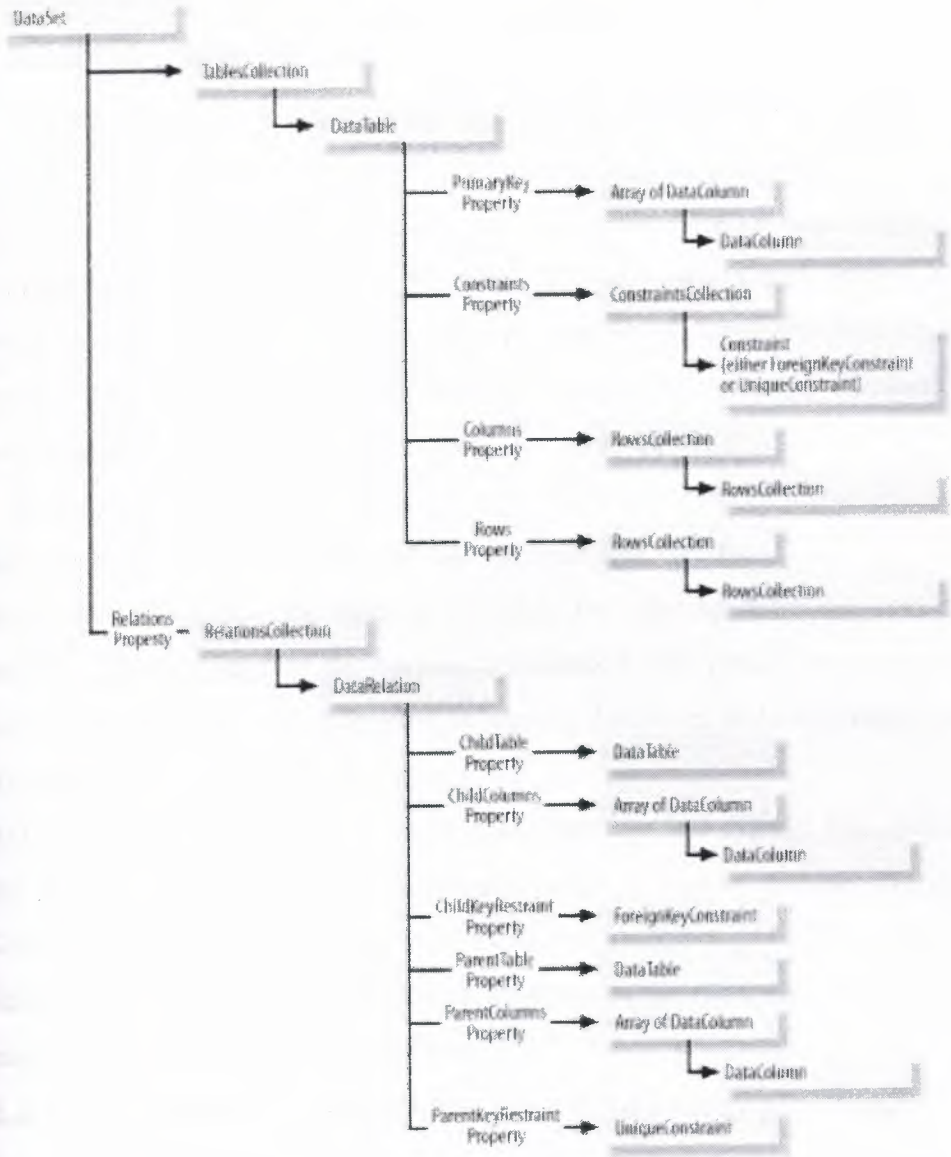
DataSet
TablesCollection
DataTable
PrimaryKey Property
Array of DataColumn
DataColumn
Constraints Property
ConstraintsCollection
Constraint (either ForeignKeyConstraint or UniqueConstraint)
Columns Property
RowsCollection
DataRow
Rows Property
RowsCollection
DataRow
Relations Property
RelationsCollection
DataRelation
ChildTable Property
DataTable
ChildColumns Property
Array of DataColumn
DataColumn
ChildKeyConstraint Property
ForeignKeyConstraint
ParentTable Property
DataTable
ParentColumns Property
Array of DataColumn
DataColumn
ParentKeyConstraint Property
UniqueConstraint

Figure 1.5

2.SQL Server 2000

2.1.Introduction

Every organization has data close to its heart. Storing and caring for data are the roles of SQL Server. SQL Server's robust RDBMS architecture supports hosting multiple, distinct databases and, since version 7.0, provides native file system support, simplifying the management of database files. Planning for the location and growth of these files is an important process in designing and creating your database. This chapter discusses the architecture of SQL Server, various storage systems for SQL database files, and how to design an efficient database file structure for your application.

Creating databases in SQL Server can be as simple as responding to dialog box prompts using the Create Database Wizard or as configurable as the T-SQL CREATE DATABASE statement. We review the options available for creating databases and configuring options such as database support for various collations,a new option in SQL Server 2000. Several features are available for tasks such as moving databases and supporting database growth with file autogrow and multiple data files.

Before you can begin creating the physical database and storage files, you need to understand the database structure and its requirements. This chapter reviews the architecture of SQL Server as well as the modeling techniques and tools available to design and implement your database solution. Having a clear understanding of the database model is essential to planning and modifying your physical database.

2.2.How Will SQL Server 2000 Benefit My Organization?

SQL Server 2000 includes many new and enhanced features that have proven beneficial to all types of organizations and applications, including e-commerce, business intelligence, and line-of-business applications. Integrated technologies such as XML support, OLAP, and data-mining engines offer an unprecedented list of features, allowing SQL Server to play an integral role in every aspect of your organization—from business-to-business integration and electronic commerce to back-office data analysis and decision support. The importance of technologies such as XML continues to increase as organizations work toward greater integration with business partners, providing higher levels of efficiency and access to new customers. OLAP and data-mining capabilities result in more successful business decisions based on the discovery of new information among your piles of data. Whether your organization is a small business or a multinational corporation, SQL Server 2000 offers advantages such as improved self-tuning, automatic file growth, and configuration wizards

through four-node fail-over clustering, federated servers, and support for up to 32 processors and 64GB of memory. SQL Server 2000 offers compatible platform support ranging from Windows CE to Windows 2000 Datacenter Server, allowing organizations to leverage existing SQL programming skills to deliver applications on every Windows platform.

2.3What language SQL Server uses to implement and maintain the relational model

Transact-SQL is a subset standard of SQL that SQL Server uses to implement, maintain, and access databases.

2.4.What software is used to access SQL Server

SQL Server comes with several utilities that allow you to access its services. You can use these utilities locally or remotely to manage a SQL Server system.

2.5.SQL Server 2000 Architecture

SQL Server 2000 consists of numerous components that interact to provide complete database application capabilities, including relational database management, OLAP, data mining, full-text indexing, data import and export, and replication, as well as client access, as depicted in Figure 2.5. In the later chapters of this book, we review each of these components in detail and assist you in configuring and using them in your applications. This chapter begins by exploring the base components of SQL Server and its databases. SQL Server 2000's relational database architecture is highly scalable and reliable and continues to meet the growing demands of thousands of customers with databases into the terabytes and users in the thousands. The foundation of every

SQL Server solution begins with the same component: the database. Each SQL Server instance can support up to 32,767 databases, each with a maximum database size of 1,048,516 terabytes (TB). SQL Server 2000 has also increased the maximum size of its log files from 4TB in version 7.0 to 32TB in SQL Server 2000, offering greater transactional capacity. Table 2.5 provides the maximum database properties of SQL Server 2000.

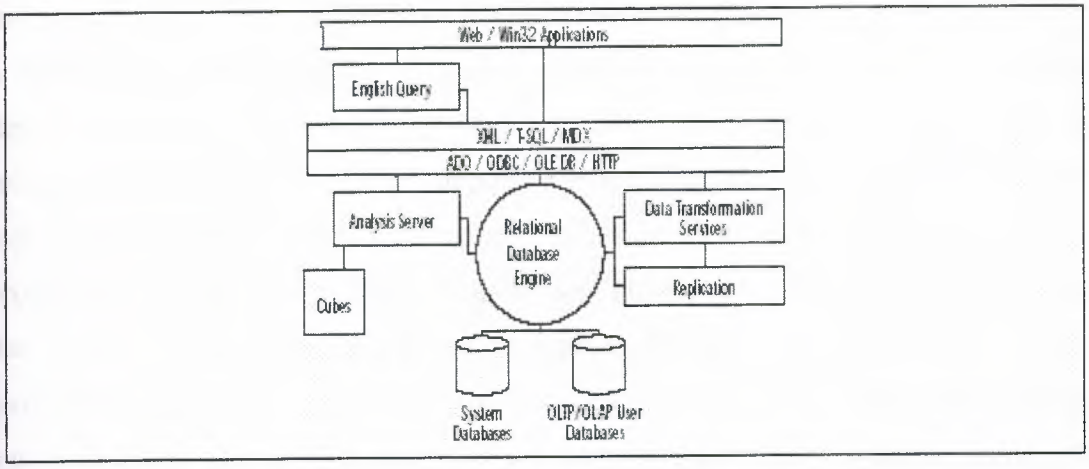

Figure 2.5 SQL Server architecture overview.

Table 2.5 SQL Server Maximum Capacities

Object/Property	SQL Server 2000
Instances per server	16
Databases per instance	32,767
Filegroups per database	256
Files per database	32,767
Database size	1,048,516TB
Data file size	32TB
Log file size	32TB
Total database objects	2,147,483,647
Columns per table	1,024

2.5.1.Relational Databases

Although there are numerous methods of storing information, the relational database model has grown to be recognized as the most efficient data storage model. Relational databases are based on the need to efficiently organize and store data. Data in a relational database are organized as entities and stored in tables. Each table consists of attributes, which result in the columns that make up a table. The process of identifying the primary data items or entities and efficiently laying out these tables and their attributes is accomplished through a process called *normalization*. The process of normalization works to reduce data redundancy and derivation, producing efficiency in both storage and data management. By eliminating redundant and derived data (data that are the result ofother attributes), the physical storage requirements are reduced. Additionally, the task of managing multiple copies of identical data as well as computed or derived data is eliminated, resulting in a more accurate and

manageable database. Complex data items can be broken down into multiple tables to produce a normalized database model, as shown in the example in Figure2.5.1 This simple model depicts a portion of an order-entry database. This collection of "related" tables makes up a relational database. SQL Server supports the requirements of a RDBMS by providing the logical and physical storage architecture that are needed. At the core of the logical storage architecture of SQL Server 2000 are databases, tables, and columns. The database represents the overall data grouping and is the foundation of SQL Server applications. Tables store individual data entities and consist of columns (attributes) that store the actual data item values. Beyond the base components are several higher-level database items, commonly referred to as database *objects*, such as views, triggers, indexes, keys, defaults, constraints, stored procedures, user-defined data types, and user-defined functions. Each of these objects plays a particular role in controlling the integrity of the data or effectively delivering it for application use. The following list outlines each of the components of a relational database solution in SQL Server 2000:

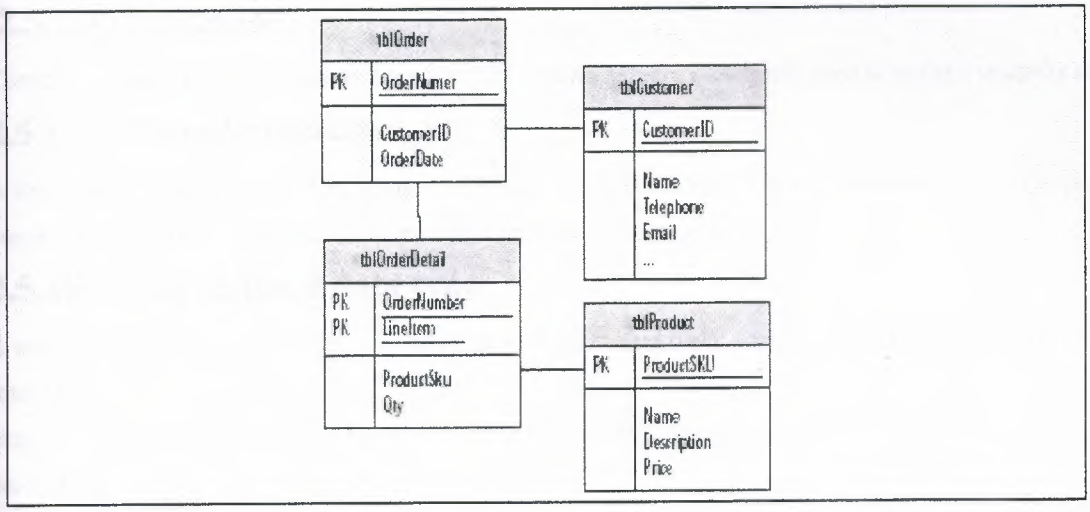

Figure 2.5.1 A normalized database example

2.5.1.1.Database

The database is the primary object and contains all the remaining database objects, such as tables, views, and the like.

2.5.1.2Table

Each table represents a data item or entity. Tables are typically logical data containers, such as a customer table or an order table in an order-entry database solution.

2.5.1.3Column

A column is a property of a table and represents an attribute or data item about the table.

2.5.1.4.View

A view is often called a *virtual table* and is typically used to combine several tables in to a meaningful representation of the data. Views do not physically store the data; rather, they provide a combined presentation of the underlying tables.

2.5.1.5.Trigger

Triggers are routines or stored procedures that execute automatically when an INSERT, UPDATE, or DELETE action occurs against a table. Triggers are often used to enforce business rules in a database application. For example, a trigger could send an e-mail message when the Orders table receives new data.

2.5.1.6. Index

An index relates to a table and speeds the retrieval of data from the table by providing a representation of the data that is more efficient for locating the requested information.

2.5.1.7.Key

A key is a property of a table and is one or more columns that uniquely identify each record or row in the table.

2.5.1..8.Default

A default specifies the value of a column during an insert if an explicit value is not supplied.

2.5.1.9.Constraint

Constraints specify the valid values for a specific column and are commonly used to enforce integrity rules.

2.5.1.10. Stored procedure

A stored procedure is a predefined group of Transact- SQL statements that is commonly used as a routine to manipulate or perform complex filter operations in order to retrieve specific data.

2.5.1.11 User-defined data type

A user-defined data type, or UDT, is used to enforce exact data types across multiple tables and is based on a standard
data type—for example, a Social Security number (SSN) data type that represents an 11-character (nnn-nn-nnnn) data type.

2.5.1.12. User-defined function

A user-defined function is a reusable logic set that can be called to perform set actions and return a given result. User-defined functions are similar to stored procedures except that they can return data type results, allowing them to be used in line with TSQL statements.

3. DATABASE DESIGN OF THE PROGRAM & INTERFACE

3.1Database Design of The Program

My project database consists of five tables those are id(personel informations),treatment (treatment informations treatment name price), illtr (appliedtreatment), payment, userk(login) tables.

ID table contains sixteen fields :

- illname
- illsurname
- illid
- hphone
- wphone
- mphone
- Address
- Sex
- Blood
- Birth place
- Mail
- Age
- Regdate
- Anamnes
- Notes
- Pic

ID table contains four fields :

- Trid
- Trtype
- Trname
- Trprice

ILLTRtable contains six fields :

- illname
- illsurname

- illid
- trdate
- tr
- cost
- tno

PAYMENT table contains six fields :

- payid
- illno
- payment
- paydate

USERK table contains three fields :

- userid
- username
- userpassword

The relationships between tables will as follows:

In Id Table illid field is a primary key.

In Treatment Table trid is a primary key.

In Illtr Table illtrn is a primary key.

In Payment Table Payid is a primary key.

In Userk Table userid is a primary key.

3.2.Interface

When you execute the program Login Form opens, then it will ask you username and password. If you do not know username and password you can not login this program You can see it in Fig 3.1.

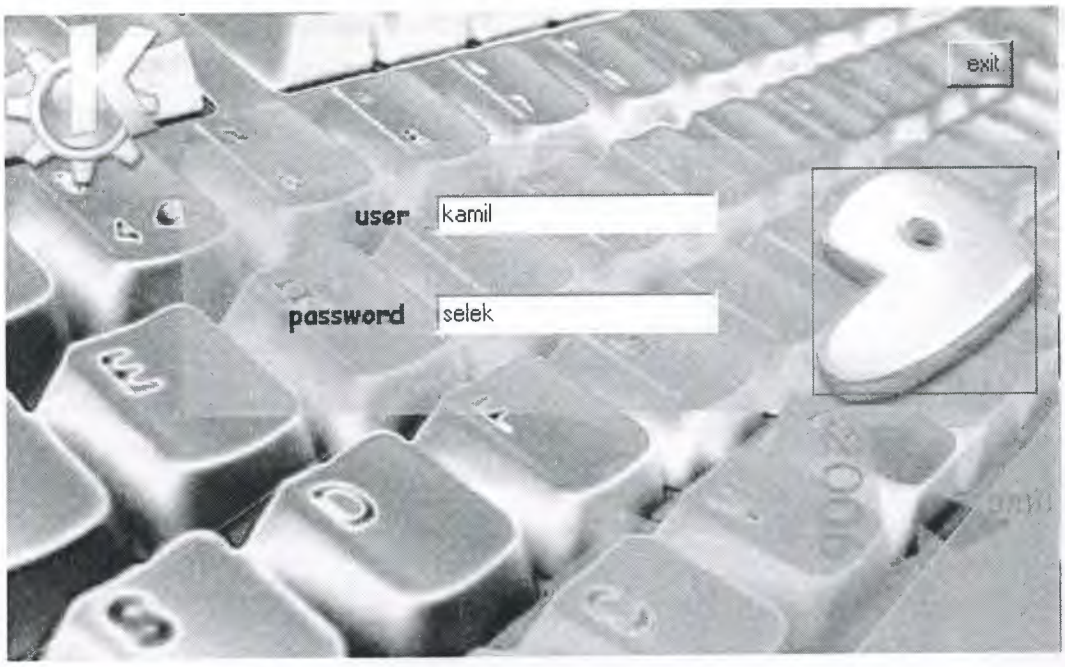

Fig 3.1.

If the entered user name and password is true than the program main pages aneble to use After authorization new record page comes on screen.you can see it in figure3.2

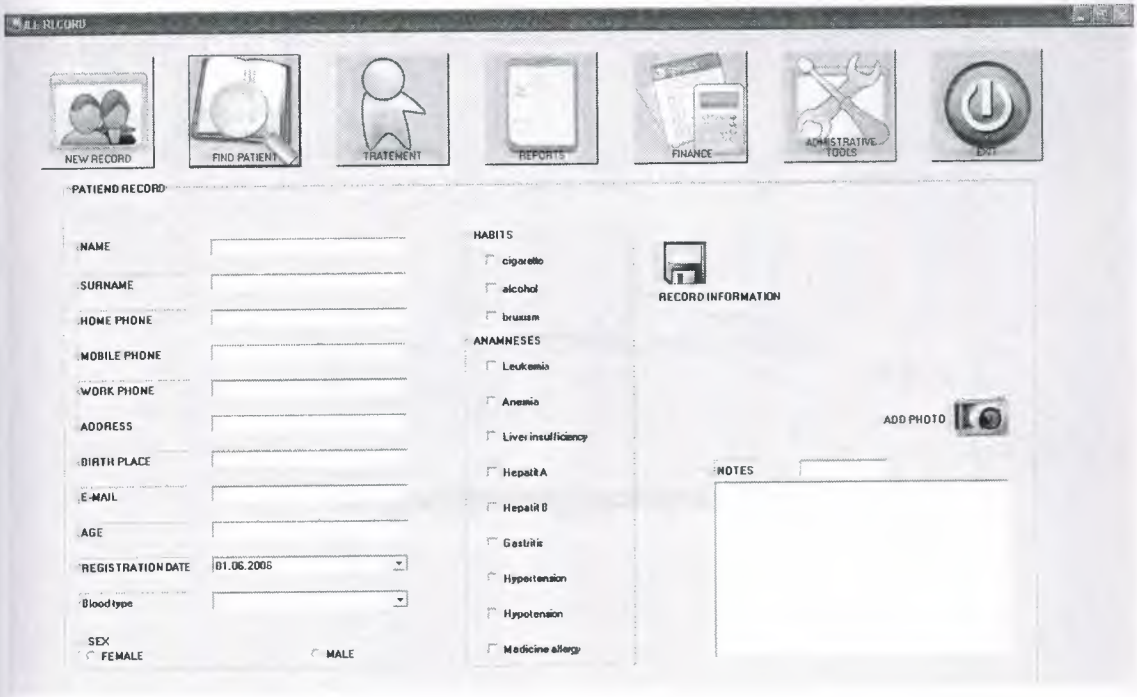

Fig 3.2.

Top of the page there is buttons for accessing other applications.This form prepared for recording new patiend personal informations

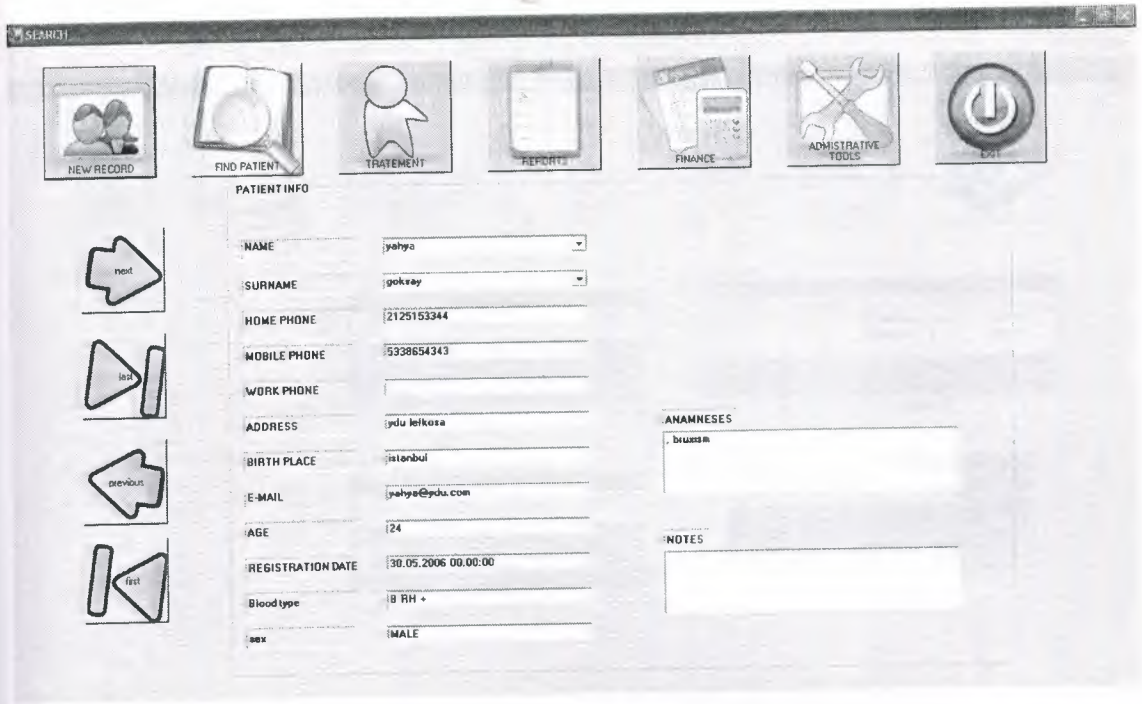

Fig 3.3.

This form for finding recorded patients informations

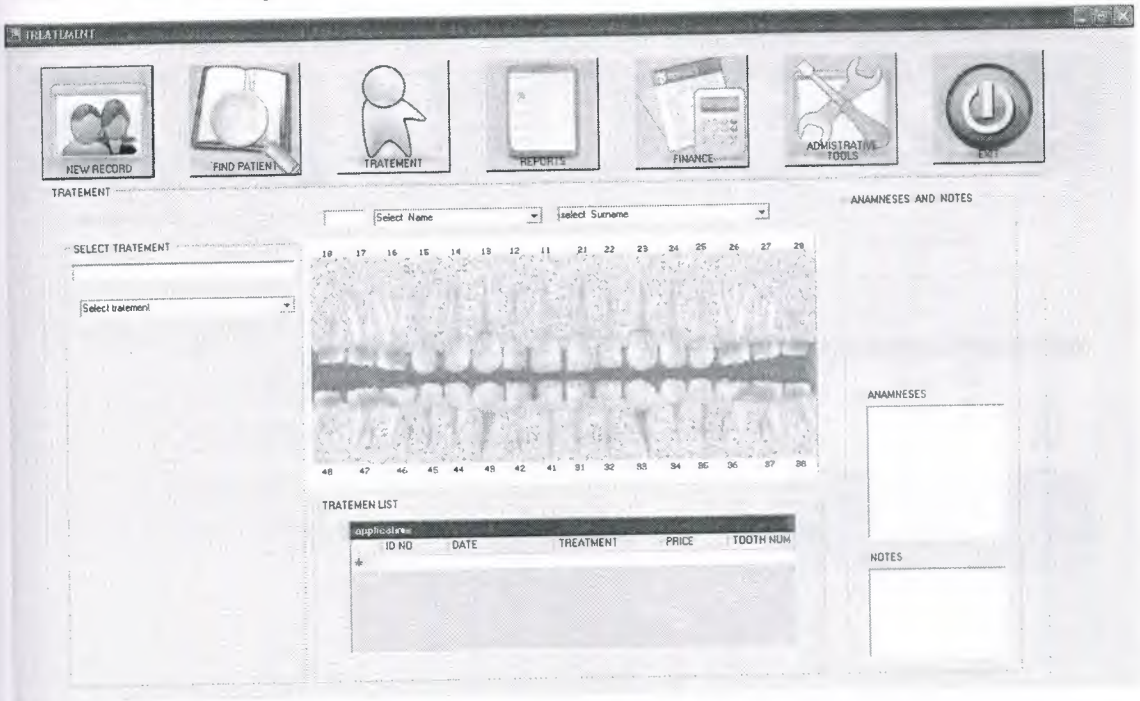

Fig 3.4.

This is main application form of my project you can select patient and treatment than click a tooth which treatment is applied and record it database, it can also show applied treatment and cost information on datagrid ,anamneses and notes

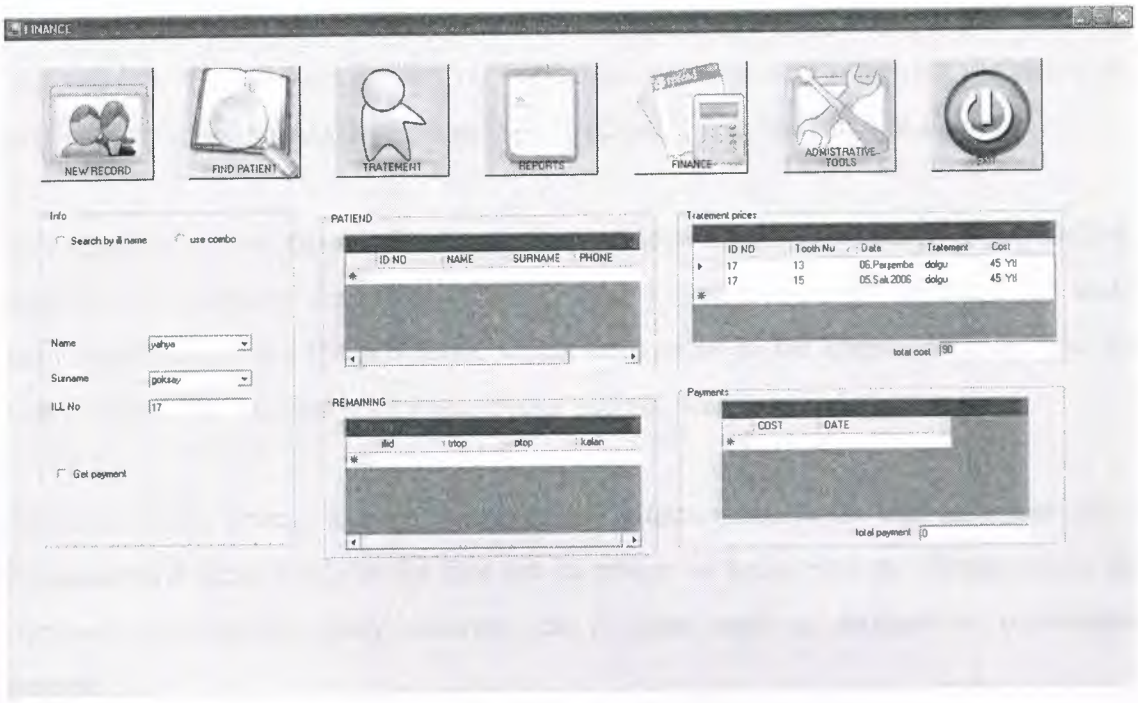

Fig 3.5.

This form show financial detail of patient

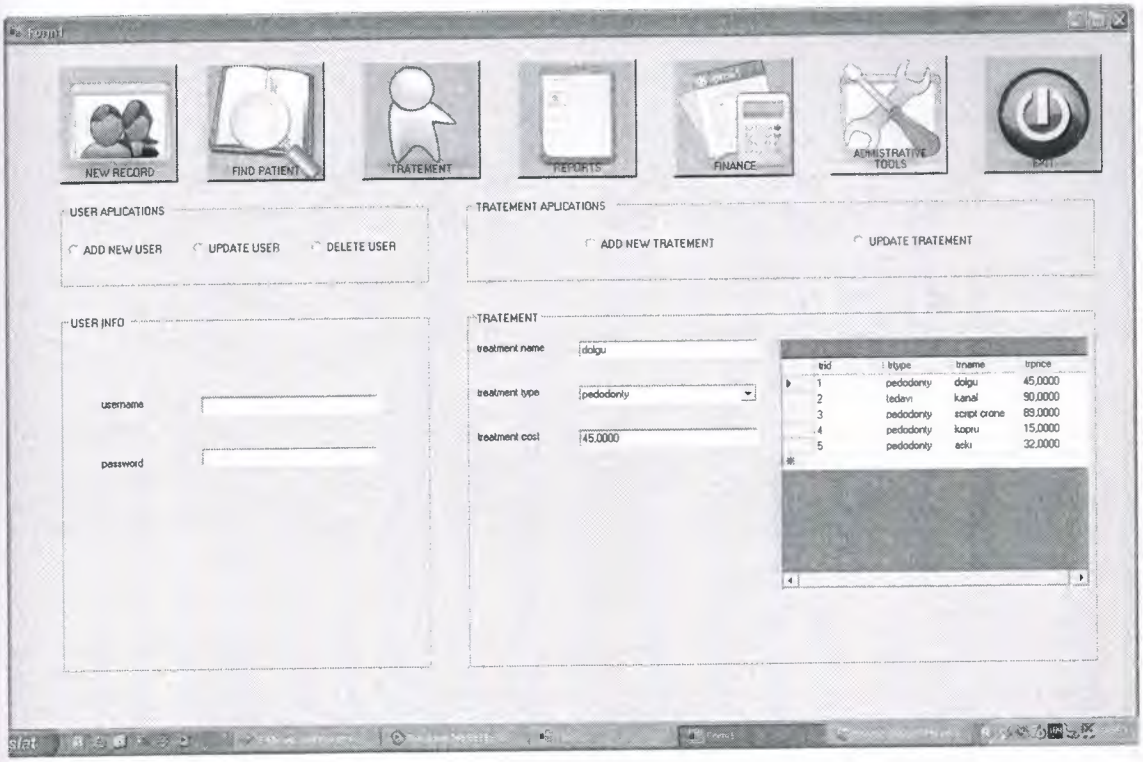

Fig 3.6.

This form for admistrative applications add,delete, update user and treatment.

CONCLUSION

The validity of a software depents on the facility of usege and supplying the needs. Also the productivity of a database depents and spedy of query and recording in a safety.

Practically implementation of software for business though it is related to any field needs a devoted and complete life cycle. In this project I contact a dentist so that I can understand their requirements and the problems, which may occur in the implementation. The software was created after analysing all requirement and get necessary informations

The aim of this project is have to supply the informations to be kept in a data base in the computer in a dental clinic so the data can be preserved longer and the loosing of the data will decrease, accessibility, query, ordering, can be done easily so productivity of foundation will increase.

REFERENCES

The Visual Studio .NET online Help

The World Wide Web Consortium Web site (http://www.w3c.org)

The Microsoft ASP.NET public newsgroup (news://msnews.microsoft.com/microsoft.public.dotnet.framework.aspnet)

http://www.microsoft.com/sql

http://msdn.microsoft.com/sqlserver/

http://microsoft.com/technet/sql/

http://www.swynk.com

microsoft.public.sqlserver.server

microsoft.public.sqlserver.tools

APPENDIX : PROGRAM CODES

```
Public Class Form1
    Inherits System.Windows.Forms.Form
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        Dim con1 As New SqlClient.SqlConnection
        Dim com1 As New SqlClient.SqlCommand
        con1.ConnectionString = "data source=OEM;initial
CATALOG=kamil;integrated security=true"
        con1.Open()
        com1.Connection = con1
        Dim dr As SqlClient.SqlDataReader
        Dim s As Integer = 0
        If TextBox1.Text = "" Or TextBox2.Text = "" Or
TextBox3.Text = "" Or TextBox4.Text = "" Then
            MsgBox("enter values")
            Exit Sub

        End If
        com1.CommandText = " update userk set username='" &
TextBox3.Text & "' , userpassword='" & TextBox4.Text & "'
where username='" & TextBox1.Text & "' and userpassword='" &
TextBox2.Text & "' "
        s = com1.ExecuteNonQuery
        If s > 0 Then
            MsgBox("changed")
        End If
        con1.Close()
        TextBox1.Text = ""
        TextBox2.Text = ""
        TextBox3.Text = ""
        TextBox4.Text = ""
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        Dim con1 As New SqlClient.SqlConnection
        Dim com1 As New SqlClient.SqlCommand
        Dim com2 As New SqlClient.SqlCommand
        con1.ConnectionString = "data source=OEM;initial
CATALOG=KAMİL;integrated security=true"
        Dim dr As SqlClient.SqlDataReader
        Dim s1 As Integer = 0
        If TextBox1.Text = "" Or TextBox2.Text = "" Then
            MsgBox("USER PASSWORD OR NAME EMPTY")
            Exit Sub
        End If
        Try
            com1.CommandText = "select * from userk"
```

```
            com2.CommandText = " insert into
userk(username,userpassword) values('" & TextBox1.Text & "',
'" & TextBox2.Text & "')"
            com1.Connection = con1
            com2.Connection = con1
            con1.Open()
            dr = com1.ExecuteReader
            Do While dr.Read
                If TextBox1.Text = dr("username") Then
                    MsgBox("farklı bir kullanıcı ismi
giriniz")
                    dr.Close()
                    con1.Close()
                    Exit Sub
                End If
            Loop
            dr.Close()
            s1 = com2.ExecuteNonQuery
            Dim a As String
            a = TextBox1.Text & "---" & TextBox2.Text
            If s1 > 0 Then MsgBox(a & " recorded")
        Catch ex As Exception
            MsgBox(ex.Message)

        End Try
        con1.Close()
        dr.Close()
        TextBox1.Text = ""
        TextBox2.Text = ""
    End Sub
    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        Dim con2 As New SqlClient.SqlConnection
        Dim com1 As New SqlClient.SqlCommand
        con2.ConnectionString = "data source=OEM;initial
CATALOG=kamil;integrated security=true"
        Dim s1 As Integer = 0
        If TextBox1.Text = "" Or TextBox2.Text = "" Then
            MsgBox("username and userpassword empty")
            Exit Sub

        End If
        Try

            com1.CommandText = " delete from userk where
username='" & TextBox1.Text & "' and  userpassword='" &
TextBox2.Text & "'"
            com1.Connection = con2
            con2.Open()
            s1 = com1.ExecuteNonQuery
            Dim a As String
```

```
            a = TextBox1.Text & "---" & TextBox2.Text
            If s1 > 0 Then
                MsgBox(a & "deleted")
            End If
            If s1 = 0 Then
                MsgBox("not deleted")
            End If
        Catch ex As Exception
            MsgBox(ex.Message)
        Finally
            con2.Close()
        End Try
    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
        trda.Fill(Trds1.tratement)
        Try
            Dim co As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
            co.CommandText = "select * from tratement"
            co.Connection = con
            dr = co.ExecuteReader
            ComboBox1.Items.Clear()
            ComboBox2.Items.Clear()

            Do While dr.Read
                If ComboBox1.Items.Contains(dr("trtype")) =
False Then
                    ComboBox1.Items.Add(dr("trtype"))
                End If
                If ComboBox2.Items.Contains(dr("trtype")) =
False Then
                    ComboBox2.Items.Add(dr("trtype"))
                End If
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            ComboBox1.Focus()



        End Try
    End Sub

    Private Sub RadioButton1_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RadioButton1.CheckedChanged
```

```
        Button3.Visible = True
        Button1.Visible = False
        Button2.Visible = False
        TextBox3.Visible = False
        TextBox4.Visible = False
        Label3.Visible = False
        Label4.Visible = False
    End Sub

    Private Sub RadioButton2_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RadioButton2.CheckedChanged
        Button3.Visible = False
        Button2.Visible = False
        Button1.Visible = True
        TextBox3.Visible = True
        TextBox4.Visible = True
        Label3.Visible = True
        Label4.Visible = True

    End Sub

    Private Sub RadioButton3_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RadioButton3.CheckedChanged
        Button2.Visible = True
        Button3.Visible = False
        Button1.Visible = False
        TextBox3.Visible = False
        TextBox4.Visible = False
        Label3.Visible = False
        Label4.Visible = False

    End Sub

    Private Sub GroupBox1_Enter(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles GroupBox1.Enter

    End Sub


 Private Sub Button7_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button7.Click

        f1.Hide()
        f2.Show()

    End Sub
```

```
    Private Sub Button6_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button6.Click

        f1.Hide()
        f7.Show()
    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button5.Click

        f1.Hide()
        f4.Show()
    End Sub

    Private Sub Button8_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button8.Click

        f1.Hide()
        f6.Show()
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button4.Click

        f1.Hide()
        f5.Show()
    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button9.Click
        Application.Exit()

    End Sub

    Private Sub Button12_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button12.Click
        Dim co As New SqlClient.SqlCommand
        Dim co2 As New SqlClient.SqlCommand
        Dim dr As SqlClient.SqlDataReader
        Dim i As Integer = 0
        Trds1.tratement.Clear()
        If TextBox8.Text = "" And TextBox6.Text = "" Then
            MessageBox.Show("please fill tratement and price",
"fill empty spaces", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
            Exit Sub
        End If

        Try

            co2.CommandText = "select * from tratement"
```

```
            co.CommandText = "insert into
tratement(trname,trtype,trprice) values('" & TextBox8.Text &
"','" & ComboBox1.Text & "'," & TextBox6.Text & ")"

            co.Connection = con
            co2.Connection = con
            con.Open()
            dr = co2.ExecuteReader
            Do While dr.Read
                If TextBox8.Text = dr("trname") Then
                    MessageBox.Show(TextBox1.Text & " this
tratement already exist", "please change tratement name",
MessageBoxButtons.OK, MessageBoxIcon.Warning)
                    dr.Close()
                    con.Close()
                    Exit Sub
                End If
            Loop
            dr.Close()

            i = co.ExecuteNonQuery()
            Dim a As String
            a = "tratement= " & TextBox8.Text & vbCrLf &
"price= " & TextBox6.Text
            If i > 0 Then MessageBox.Show(a & vbCrLf &
"Succesfully added to database", "Succesfully Added",
MessageBoxButtons.OK, MessageBoxIcon.Information)

        Catch ex As SqlClient.SqlException


            MessageBox.Show(ex.Message)
        Finally
            trda.Fill(Trds1.tratement)


        End Try
        TextBox8.Text = ""
        TextBox6.Text = ""

        con.Close()
        dr.Close()

    End Sub

    Private Sub Button11_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button11.Click
        Dim co As New SqlClient.SqlCommand
        Dim co2 As New SqlClient.SqlCommand
        Dim dr As SqlClient.SqlDataReader
        Dim i As Integer = 0
```

```
        Trds1.tratement.Clear()
        If TextBox8.Text = "" Then
            MessageBox.Show("please fill tratement name",
"fill empty spaces", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
            Exit Sub
        End If


        Try
            If TextBox8.Text = "" Then
                MessageBox.Show("please fill tratement name",
"fill empty spaces", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
                Exit Sub
            End If

            co.CommandText = "update tratement set trprice=" &
TextBox12.Text & ", trtype='" & ComboBox2.Text & "' where
trname='" & TextBox8.Text & "'"

            con.Open()
            co.Connection = con
            i = co.ExecuteNonQuery()
            Dim a As String
            a = "New tratement= " & TextBox8.Text & vbCrLf &
"New cost= " & TextBox12.Text
            If i > 0 Then MessageBox.Show(a & vbCrLf &
"Succesfully Updated", "Succesfully Updated",
MessageBoxButtons.OK, MessageBoxIcon.Information)

        Catch ex As SqlClient.SqlException
            MessageBox.Show(ex.Message)
        Finally
            trda.Fill(Trds1.tratement)
            con.Close()
        End Try
        TextBox12.Text = ""
        TextBox6.Text = ""
    End Sub
Private Sub RadioButton6_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RadioButton6.CheckedChanged
        Button12.Visible = True
        TextBox12.Visible = False
        Label11.Visible = False
        Label9.Visible = False
        ComboBox2.Visible = False
    End Sub
```

```
    Private Sub RadioButton4_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RadioButton4.CheckedChanged
        Button12.Visible = False
        Button11.Visible = True
        ComboBox2.Visible = True
        TextBox12.Visible = True
        Label11.Visible = True
        Label9.Visible = True

    End Sub
```

```
Public Class Form2
    Inherits System.Windows.Forms.Form
Public tar As DateTime
    Public rdate As String
    Public c As String
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        Dim con1 As New SqlClient.SqlConnection
        Dim com1 As New SqlClient.SqlCommand
        Dim com2 As New SqlClient.SqlCommand
        con1.ConnectionString = "data source=OEM;initial
CATALOG=KAMİL;integrated security=true"
        Dim dr As SqlClient.SqlDataReader
        Dim s1 As Integer = 0
        Dim a As String
        Dim s As String
        tar = DateTimePicker1.Value
        rdate = tar.Month & "." & tar.Day & "." & tar.Year
        If TextBox22.Text = "" Or TextBox21.Text = "" Then
            MsgBox("please fill name surname")
            Exit Sub
        End If
        If CheckBox11.Checked = True Then
            a = a & ", " & CheckBox11.Text
        End If
        If CheckBox12.Checked = True Then
            a = a & ", " & CheckBox12.Text
        End If
        If CheckBox13.Checked = True Then
            a = a & ", " & CheckBox13.Text
        End If
```

```
        If CheckBox1.Checked = True Then
            a = a & ", " & CheckBox1.Text
        End If
        If CheckBox2.Checked = True Then
            a = a & ", " & CheckBox2.Text
        End If
        If CheckBox3.Checked = True Then
            a = a & ", " & CheckBox3.Text
        End If
        If CheckBox4.Checked = True Then
            a = a & ", " & CheckBox4.Text
        End If
        If CheckBox5.Checked = True Then
            a = a & ", " & CheckBox6.Text
        End If
        If CheckBox7.Checked = True Then
            a = a & ", " & CheckBox7.Text
        End If
        If CheckBox8.Checked = True Then
            a = a & ", " & CheckBox8.Text
        End If
        If CheckBox9.Checked = True Then
            a = a & ", " & CheckBox9.Text
        End If
        If CheckBox10.Checked = True Then
            a = a & ", " & CheckBox10.Text
        End If
        If RadioButton4.Checked = True Then
            s = RadioButton4.Text
        ElseIf RadioButton3.Checked = True Then
            s = RadioButton3.Text
        End If


        Try
            con1.Open()
            com1.Connection = con1
            com2.Connection = con1
            com1.CommandText = "select * from id"
            com2.CommandText = " insert into
id(illname,illsurname,hphone,wphone,mphone,address,birtplace,m
ail,age,notes,anemnes,sex,blood,pic,regdate) values('" &
TextBox22.Text & "', '" & TextBox21.Text & "','" &
TextBox20.Text & "', '" & TextBox18.Text & "','" &
TextBox19.Text & "', '" & TextBox17.Text & "','" &
TextBox24.Text & "', '" & TextBox23.Text & "', '" &
TextBox16.Text & "','" & TextBox1.Text & "','" & a & "','" & s
& "','" & ComboBox1.Text & "','" & c & "','" & rdate & "')"


            dr = com1.ExecuteReader
```

```
            Do While dr.Read
                If TextBox22.Text = dr("illname") And
TextBox21.Text = dr("illsurname") Then
                    MsgBox("already exist record")
                    dr.Close()
                    con1.Close()
                    Exit Sub
                End If
            Loop
            dr.Close()
            s1 = com2.ExecuteNonQuery
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con1.Close()
            dr.Close()
        End Try
    End Sub

Private Sub Button2_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button2.Click
        Dim f As String
        OpenFileDialog1.InitialDirectory = "c:\"
        OpenFileDialog1.Filter = "Jpg files (*.jpg)|*.jpg"
        OpenFileDialog1.ShowDialog()
        If OpenFileDialog1.ShowDialog = DialogResult.OK Then
            f = OpenFileDialog1.FileName.ToString
            Me.Text = f
            Dim dosya As New FileInfo(f)
            c = "C:\dental\img\ill\" & TextBox22.Text &
TextBox21.Text & "." & "jpg"

            dosya.CopyTo(c, True)
            PictureBox1.Image = Image.FromFile(c)
            MessageBox.Show("Your logo saved this " & c & "
directory", " Succesfully added to database",
MessageBoxButtons.OK, MessageBoxIcon.Information)
        Else
            MessageBox.Show("Your logo not saved database
succesfully", "Not successful", MessageBoxButtons.OK,
MessageBoxIcon.Warning)
        End If
    End Sub

Public Class Form3
    Inherits System.Windows.Forms.Form
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        Dim c As New SqlClient.SqlConnection
        Dim co As New SqlClient.SqlCommand
        Dim dr As SqlClient.SqlDataReader
```

```
        Try
            c.ConnectionString = "data source=OEM;initial
CATALOG=kamil;integrated security=true"
            c.Open()
            co.Connection = c
            co.CommandText = " select * from userk"
            dr = co.ExecuteReader
            Dim s As Integer = 0
            Do While dr.Read
                If dr("username") = TextBox1.Text And
dr("userpassword") = TextBox2.Text Then
                    s = 1
                    Exit Do
                End If
            Loop
            If s = 0 Then
                MsgBox("user or password not correct")
                Exit Sub

            End If
            f3.Dispose()
            f2.Show()
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)

        End Try
        c.Close()
        dr.Close()

    End Sub
Public Class Form4
    Inherits System.Windows.Forms.Form
Public a As Integer
    Public t(32) As Integer
    Public pic As String
    Public id As Integer


    Private Sub Form4_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
        Try
            Dim co As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
            co.CommandText = "select * from tratement"
            co.Connection = con
            dr = co.ExecuteReader
            ComboBox1.Items.Clear()
            Do While dr.Read
                If ComboBox1.Items.Contains(dr("trname")) =
False Then
```

```
                    ComboBox1.Items.Add(dr("trname"))
                End If
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            ComboBox1.Focus()
        End Try
        Try
            Dim com As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
            com.CommandText = "select * from id order by
illname"
            com.Connection = con
            dr = com.ExecuteReader
            ComboBox2.Items.Clear()
            Do While dr.Read
                ComboBox2.Items.Add(dr("illname"))
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
        End Try
    End Sub


    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox1.SelectedIndexChanged

        Try
            TextBox1.Text = "selected operation" + "   " +
ComboBox1.Text
            Dim com As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
            com.CommandText = "select trprice from tratement
where trname='" & ComboBox1.Text & "'"
            com.Connection = con
            dr = com.ExecuteReader
            Do While dr.Read
                a = dr("trprice")
                Exit Sub
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
```

```
            con.Close()
        End Try
    End Sub

Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click


        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values(" & TextBox2.Text & ",'" &
ComboBox1.Text & "'," & a & "," & 18 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
            con.Close()
        End Try
    End Sub

    Private Sub ComboBox2_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox2.SelectedIndexChanged
        Try
            ComboBox3.Items.Clear()
            Dim com As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
```

```
            com.CommandText = "select * from id"
            com.Connection = con
            dr = com.ExecuteReader
            Do While dr.Read
                If ComboBox3.Items.Contains(dr("illsurname"))
= False Then
                    If ComboBox2.Text = dr("illname") Then
                        ComboBox3.Items.Add(dr("illsurname"))

                    End If
                End If

            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()



        End Try
    End Sub

    Private Sub ComboBox3_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox3.SelectedIndexChanged
        Try

            Dim com As New SqlClient.SqlCommand
            Dim com2 As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            Dim dr2 As SqlClient.SqlDataReader
            con.Open()
            com.CommandText = "select ilid,pic,anemnes,notes
from id where illname='" & ComboBox2.Text & "' and
illsurname='" & ComboBox3.Text & "' "

            com.Connection = con
            dr = com.ExecuteReader
            Do While dr.Read
                TextBox2.Text = dr("ilid")
                id = dr("ilid")
                pic = dr("pic")
                TextBox4.Text = dr("anemnes")
                TextBox5.Text = dr("notes")
            Loop
            PictureBox2.Image = Image.FromFile(pic)
            dr.Close()
            com2.CommandText = "select cost from "
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
```

```
        Finally
            con.Close()
        End Try
    End Sub


    Private Sub Button8_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button8.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 48 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button4.Click

        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
```

```
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0
            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 17 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button10_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button10.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 16 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
```

```
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button9.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 15 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
```

```
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button16_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button16.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 14 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button17_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button17.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0
```

```
            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 13 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""



        End Try
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 12 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
```

```
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""



        End Try
    End Sub

    Private Sub Button15_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button15.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 11 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If
```

```
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button14_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button14.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 21 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub
```

```
    Private Sub Button6_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button6.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 22 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button22_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button22.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
```

```
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 23 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button23_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button23.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 24 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
```

```
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button21_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button21.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 25 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
```

```
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button30_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button30.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 26 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""



        End Try
    End Sub

    Private Sub Button31_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button31.Click
```

```
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 27 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button32_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button32.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
```

```
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 28 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button11_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button11.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 47 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
```

```
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""



        End Try
    End Sub

    Private Sub Button3_Click_1(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0
            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 46 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
```

```
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button5.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 45 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button19_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button19.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
```

```
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 44 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""

        End Try
    End Sub

    Private Sub Button18_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button18.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 43 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
```

```
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button20_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button20.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 42 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
```

```
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""

        End Try
    End Sub

    Private Sub Button12_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button12.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 41 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""

        End Try
    End Sub
```

```
    Private Sub Button13_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button13.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 31 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""

        End Try
    End Sub

    Private Sub Button7_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button7.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
```

```
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 32 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button25_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button25.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0
            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 33 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
```

```
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""



        End Try
    End Sub

    Private Sub Button24_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button24.Click
        Try
            ds.Clear()
            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0
            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 34 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
```

```
            con.Close()
            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""

        End Try
    End Sub

    Private Sub Button26_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button26.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 35 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub

    Private Sub Button27_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button27.Click
        Try
```

```
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 36 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""



        End Try
    End Sub

    Private Sub Button29_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button29.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0
```

```
            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 37 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""



        End Try
    End Sub

    Private Sub Button28_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button28.Click
        Try
            ds.Clear()

            Dim con As New SqlClient.SqlConnection
            con.ConnectionString = "data source=oem;initial
catalog=kamil;integrated security=true"
            Dim co As New SqlClient.SqlCommand
            Dim i As Integer = 0

            con.Open()
            co.CommandText = "insert into
illtr(illid,tr,cost,tno) values('" & TextBox2.Text & "','" &
ComboBox1.Text & "','" & CType(a, Integer) & "'," & 38 & ")"
            co.Connection = con
            i = co.ExecuteNonQuery()
```

```
            If i > 0 Then
                MessageBox.Show("Inserted succesfully",
"Inserted Succesfully", MessageBoxButtons.OK,
MessageBoxIcon.Information)
                Exit Sub
            ElseIf i = 0 Then
                MessageBox.Show("Can't make Insert operation",
"Not Succesfull", MessageBoxButtons.OK,
MessageBoxIcon.Exclamation)
            End If

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally

            con.Close()

            dailltr.Fill(ds.illtr)
            DataView1.RowFilter = "illid=" & TextBox2.Text &
""
        End Try
    End Sub



    Private Sub Button37_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button37.Click

        f4.Hide()
        f2.Show()
    End Sub

    Private Sub Button36_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button36.Click

        f4.Hide()
        f7.Show()
    End Sub

    Private Sub Button38_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button38.Click

        f4.Hide()
        f6.Show()
    End Sub

    Private Sub Button34_Click_1(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button34.Click

        f4.Hide()
```

```
        f5.Show()
    End Sub

    Private Sub Button40_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button40.Click

        f4.Hide()
        f1.Show()
    End Sub

    Private Sub Button39_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button39.Click
        Application.Exit()

    End Sub

End Class

public Class Form5
    Inherits System.Windows.Forms.Form
Private Sub CheckBox1_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
CheckBox1.CheckedChanged
        If CheckBox1.Checked Then
            TextBox1.Visible = True
            Label1.Visible = True
            Button1.Visible = True
        End If
    End Sub

    Private Sub ComboBox2_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox2.SelectedIndexChanged
        tot = 0
        ptot = 0
        Dim com As New SqlClient.SqlCommand
        Dim com2 As New SqlClient.SqlCommand
        Dim com3 As New SqlClient.SqlCommand
        Dim dr As SqlClient.SqlDataReader
        Dim dr2 As SqlClient.SqlDataReader
        Dim dr3 As SqlClient.SqlDataReader
        Dim a As String
        If a = illid Then
            DataGrid3.IsSelected(a)

        End If


        Try
            con.Open()
            com.CommandText = "select * from id"
```

```
            com2.CommandText = "select illid, cost from illtr"
            com3.CommandText = "select illno,payment from
payment"

            com.Connection = con
            com2.Connection = con
            com3.Connection = con

            dr = com.ExecuteReader
            Do While dr.Read
                If ComboBox1.Text = dr("illname") And
ComboBox2.Text = dr("illsurname") Then
                    illid = dr("ilid")
                    TextBox4.Text = illid
                    Exit Do

                End If
            Loop
            dr.Close()
            dr2 = com2.ExecuteReader

            Do While dr2.Read
                If illid = dr2("illid") Then
                    tot = tot + dr2("cost")

                End If
            Loop



            TextBox2.Text = tot
            dr2.Close()
            dr3 = com3.ExecuteReader
            Do While dr3.Read
                If illid = dr3("illno") Then
                    ptot = ptot + dr3("payment")

                End If
            Loop
            TextBox3.Text = ptot
            dr3.Close()
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            DataView1.RowFilter = "illid=" & illid & ""
            DataView2.RowFilter = "illno=" & illid & ""
            con.Close()
        End Try
    End Sub
```

```
    Private Sub Form5_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
        illtrds.Clear()
        payds.Clear()
        illtr.Fill(illtrds.illtr)
        pay.Fill(payds.payment)
        Dim com As New SqlClient.SqlCommand
        Dim dr As SqlClient.SqlDataReader
        Try
            con.Open()
            com.CommandText = "select * from id"
            com.Connection = con
            dr = com.ExecuteReader
            ComboBox1.Items.Clear()
            Do While dr.Read
                If ComboBox1.Items.Contains(dr("illname")) =
False Then
                    ComboBox1.Items.Add(dr("illname"))
                End If
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dr.Close()
            darem.Fill(dsrem.remainig)




        End Try
    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox1.SelectedIndexChanged
        Dim com As New SqlClient.SqlCommand
        Dim dr As SqlClient.SqlDataReader
        Try
            con.Open()
            com.CommandText = "select * from id"
            com.Connection = con
            dr = com.ExecuteReader
            ComboBox2.Items.Clear()
            Do While dr.Read
                If ComboBox1.Text = dr("illname") Then
                    If
ComboBox2.Items.Contains(dr("illsurname")) = False Then
                        ComboBox2.Items.Add(dr("illsurname"))
                    End If
```

```
                End If
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            dr.Close()
        End Try
    End Sub


    Private Sub Button7_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button7.Click
        Application.Exit()
    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button5.Click

        f5.Hide()
        f2.Show()
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click

        f5.Hide()
        f7.Show()
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click

        f5.Hide()
        f4.Show()
    End Sub

    Private Sub Button6_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button6.Click

        f5.Hide()
        f6.Show()
    End Sub

    Private Sub Button8_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button8.Click

        f5.Hide()
        f1.Show()
    End Sub
```

```
    Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
        ptot = 0
        Dim com2 As New SqlClient.SqlCommand
        Dim dr3 As SqlClient.SqlDataReader

        pd = Now.Month & "." & Now.Day & "." & Now.Year
        Dim i As String = 0

        Dim com3 As New SqlClient.SqlCommand

        Try
            con.Open()

            com2.CommandText = " insert into
payment(illno,payment,paydate) values(" & illid & ", " &
TextBox1.Text & ",'" & pd & "')"
            com2.Connection = con

            i = com2.ExecuteNonQuery
            If i > 0 Then
                MsgBox("kaydedildi")
            Else
                MsgBox("olmadı")
            End If

            com3.CommandText = "select illno,payment from
payment"
            com3.Connection = con
            dr3 = com3.ExecuteReader
            Do While dr3.Read
                If illid = dr3("illno") Then
                    ptot = ptot + dr3("payment")
                End If
            Loop
            TextBox3.Text = ptot
            dr3.Close()
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
            pay.Fill(payds.payment)

        End Try
    End Sub


    Private Sub ComboBox1_KeyPress(ByVal sender As Object,
ByVal e As System.Windows.Forms.KeyPressEventArgs) Handles
ComboBox1.KeyPress
        If e.KeyChar = ChrW(13) Then
```

```
            Dim com As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            Try
                con.Open()
                com.CommandText = "select * from id"
                com.Connection = con
                dr = com.ExecuteReader
                ComboBox2.Items.Clear()
                Do While dr.Read
                    If ComboBox1.Text = dr("illname") Then
                        If
ComboBox2.Items.Contains(dr("illsurname")) = False Then

ComboBox2.Items.Add(dr("illsurname"))
                        End If
                    End If
                Loop
            Catch ex As SqlClient.SqlException
                MsgBox(ex.Message)
            Finally
                con.Close()
                dr.Close()




            End Try
        End If
    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button9.Click

        TextBox5.Focus()
        Dim com As New SqlClient.SqlCommand
        Dim com2 As New SqlClient.SqlCommand
        Dim com3 As New SqlClient.SqlCommand
        Dim dr As SqlClient.SqlDataReader
        Dim dr2 As SqlClient.SqlDataReader
        Dim dr3 As SqlClient.SqlDataReader
        Dim a As String
        If a = illid Then
            DataGrid3.IsSelected(a)
        End If


        Try
            con.Open()
            com.CommandText = "select * from id"
            com2.CommandText = "select illid, cost from illtr"
```

```
            com3.CommandText = "select illno,payment from
payment"
            com.Connection = con
            com2.Connection = con
            com3.Connection = con
            dr = com.ExecuteReader
            Do While dr.Read
                If TextBox5.Text = dr("illname") And
TextBox6.Text = dr("illsurname") Then
                    illid = dr("ilid")
                    TextBox4.Text = illid
                    Exit Do
                End If
            Loop
            dr.Close()
            dr2 = com2.ExecuteReader
            Do While dr2.Read
                If illid = dr2("illid") Then
                    tot = tot + dr2("cost")
                End If
            Loop
            TextBox2.Text = tot
            dr2.Close()
            dr3 = com3.ExecuteReader


            Do While dr3.Read
                If illid = dr3("illno") Then
                    ptot = ptot + dr3("payment")
                End If
            Loop

            TextBox3.Text = ptot
            dr3.Close()

        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            DataView1.RowFilter = "illid=" & illid & ""
            DataView2.RowFilter = "illno=" & illid & ""
            con.Close()
        End Try

    End Sub


    Private Sub RadioButton1_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RadioButton1.CheckedChanged
        Panel1.Visible = True
    End Sub
```

```
    Private Sub RadioButton2_CheckedChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
RadioButton2.CheckedChanged
        Panel1.Visible = False

    End Sub
End Class

Public Class Form7
    Inherits System.Windows.Forms.Form
Private Sub Form7_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
        idda.Fill(ds.id)
        cm = CType(Me.BindingContext(DataView1),
CurrencyManager)

        Try
            Dim com As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
            com.CommandText = "select * from id"
            com.Connection = con
            dr = com.ExecuteReader
            ComboBox1.Items.Clear()
            Do While dr.Read
                ComboBox1.Items.Add(dr("illname"))
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
        End Try
    End Sub

    Private Sub ComboBox2_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox2.SelectedIndexChanged
        Try
            Dim pic As String
            Dim com As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
            com.CommandText = "select * from id where
illname='" & ComboBox1.Text & "' and illsurname='" &
ComboBox2.Text & "' "
            com.Connection = con
            dr = com.ExecuteReader
            Do While dr.Read
                TextBox1.Text = dr("notes")
                TextBox2.Text = dr("anemnes")
```

```
                TextBox5.Text = dr("hphone")
                TextBox6.Text = dr("mphone")
                TextBox7.Text = dr("wphone")
                TextBox8.Text = dr("address")
                TextBox9.Text = dr("birtplace")
                TextBox10.Text = dr("mail")
                TextBox11.Text = dr("age")
                TextBox12.Text = dr("regdate")
                TextBox13.Text = dr("blood")
                TextBox14.Text = dr("sex")
                PictureBox1.Image = Image.FromFile(dr("pic"))
            Loop

            dr.Close()


        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)


        Finally
        End Try
        con.Close()
    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
ComboBox1.SelectedIndexChanged
        Try
            ComboBox2.Items.Clear()


            Dim com As New SqlClient.SqlCommand
            Dim dr As SqlClient.SqlDataReader
            con.Open()
            com.CommandText = "select * from id"
            com.Connection = con
            dr = com.ExecuteReader
            Do While dr.Read
                If ComboBox2.Items.Contains(dr("illsurname"))
= False Then

                    If ComboBox1.Text = dr("illname") Then
                        ComboBox2.Items.Add(dr("illsurname"))

                    End If
                End If
```

```
            Loop
        Catch ex As SqlClient.SqlException
            MsgBox(ex.Message)
        Finally
            con.Close()
        End Try

    End Sub


    Friend WithEvents Button6 As System.Windows.Forms.Button

    Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click

        f7.Hide()
        f2.Show()
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click

        f7.Hide()
        f4.Show()
    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button5.Click

        f7.Hide()
        f6.Show()
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button4.Click

        f7.Hide()
        f5.Show()
    End Sub

    Private Sub Button8_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button8.Click

        f7.Hide()
        f1.Show()
    End Sub
```

```
    Private Sub Button7_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button7.Click

        Try


        If cm.Position = 0 Then
            MsgBox("You are on first record",
MsgBoxStyle.Information, "First Record")
        Else
            cm.Position = cm.Position - 1

            End If
        Catch ex As Exception
            MsgBox(ex.Message)

        End Try
    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button9.Click
        If cm.Position = 0 Then
            MsgBox("You are already first record")
        Else
            cm.Position = 0
        End If

    End Sub

    Private Sub Button10_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button10.Click
        If cm.Position = cm.Count - 1 Then
            MsgBox("You are already last record")
        Else
            cm.Position = cm.Count - 1

        End If
    End Sub

    Private Sub Button11_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button11.Click
        If cm.Position = cm.Count - 1 Then
            MsgBox("You are on last record",
MsgBoxStyle.Information, "Last Record")
        Else
            cm.Position = cm.Position + 1


        End If
    End Sub
End Class
```

```
Public Class Form8
    Inherits System.Windows.Forms.Form

Public xx As New kamil1

    Private Sub TabControl1_SelectedIndexChanged(ByVal sender
As System.Object, ByVal e As System.EventArgs) Handles
TabControl1.SelectedIndexChanged
        Try


        If TabPage1.Focus = True Then
            Dim xx As New kamil1
            CrystalReportViewer1.ReportSource = xx
            Dim z As New
CrystalDecisions.Shared.ParameterValues
            Dim z1 As New
CrystalDecisions.Shared.ParameterDiscreteValue
            Dim urun = InputBox("Fill the illid ", "Fill the
ill id")

            z1.Value = urun
            z.Add(z1)

xx.DataDefinition.ParameterFields("@illid").ApplyCurrentValues
(z)
            CrystalReportViewer1.ReportSource = xx

        End If
        If TabPage2.Focus = True Then
            Dim xx As New kamil2
            CrystalReportViewer2.ReportSource = xx
            Dim z As New
CrystalDecisions.Shared.ParameterValues
            Dim z1 As New
CrystalDecisions.Shared.ParameterDiscreteValue
            Dim urun = InputBox("Fill the illid ", "Fill the
ill id")

            z1.Value = urun
            z.Add(z1)

xx.DataDefinition.ParameterFields("@illno").ApplyCurrentValues
(z)
            CrystalReportViewer2.ReportSource = xx

        End If

        Catch ex As Exception
```

```
        End Try

    End Sub

    Private Sub Form8_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
        CrystalReportViewer1.ReportSource = xx
        CrystalReportViewer2.ReportSource = xx
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
        f6.Hide()
        f2.Show()
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        f6.Hide()
        f7.Show()
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        f6.Hide()
        f4.Show()
    End Sub



    Private Sub Button4_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button4.Click
        f6.Hide()
        f5.Show()
    End Sub

    Private Sub Button8_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button8.Click
        f6.Hide()
        f1.Show()
    End Sub


    End Sub

End Class

Module Module1
    Public f1 As New Form1
    Public f2 As New Form2
    Public f3 As New Form3
```

```
    Public f4 As New Form4
    Public f5 As New Form5
    Public f6 As New Form8
    Public f7 As New Form7
    Public illid As Integer
    Public d1, d2, d3 As DateTime
    Public sss, pd As String
    Public sss2 As String
    Public sss3 As String
    Public tot As Integer
    Public ptot As Integer
End Module
```