



NEAR EAST UNIVERSITY

Faculty Of Engineering

Department Of Computer Engineering

**STOCK CONTROL & CUSTOMER TRACKING
PROGRAM USING VISUAL BASIC
PROGRAMMING LANGUAGE**

**Graduating Project
COM 400**

Student: Muhammed S. ABDULLAH(20020872)

Supervisor : Mr. Umit ilhan

Nicosia 2006

ACKNOWLEDGEMENTS

"firstly I would like to thank my supervisor Mr. Umit Ilhan for his greatness in his way of treatment with students, He never make us feeling shy to ask any question whenever we faced an obstacle or difficulty not just during this project but during our whole life in Near East University.

We have not to forget other Instructors, Especially Mr. Okan Donangil, he was instead of my Father during University life, whenever I cross any problem morally or educationally he helped me, I thank him very much with my all for his advices and guiding.

Secondly I would like to give my best regards to my family, especially my Mum, she deserved a lot to stay along from me, she was counting days during my university life and waiting me as an engineer, also I'm greatly indebted to my Dad, however his financial situation was not good first year but he resisted and forced me to continue my studding, he never made me need anyone. Also I thank my elder brother Aso, he encouraged me all the time to study either by calling or sending me electronically presents from my Home country (IRAQ).

Thirdly I have not to forget my dear friend Metin Ulas, he helped me in my project, either by giving thoughts or working with my.

Finally I would like to thank All my friends here in Cyprus and In my home Country Iraq, they encouraged me too, to resist and study."

ABSTRACT

Stock Control & Customer Tracking Program is a useful program to all Mobile and electronic wholesalers. By using this program they can control all the operations, sales, loan details, payments... etc. of their workplace.

I used Visual basic programming language and Access Data base in this program, the program is easy in use, everyone can use it easily without getting difficulty. The program shows you all the stock that you have, you can make sales, enter payments record loans, add customers, add users, query information's of any customer or user... etc.

Also when you add stock to your stock list you can add by special page for adding new stocks, or you can increase and decrease the quantity of the existence stock. You can categorize items in stock by type too.

I used many forms in this project, the main form is sale form, for instance when someone make a sale from a specific user in a specific date, the program records everything, we can query all the information's at any time, also when he made that sale and paid part of the money, the program store paid amount and the remaining loan amount to his/her account.

So using this kind of programs make sale operations easier and neglects future problems with customers, also enables the manager to know what is happening daily in work place. In the same time users can't trick the manager, because whatever they do is recording by the program, users can't enter everywhere of the program, they are restricted by specific pages, only the manager can control whatever he wants to see.

TABLE OF CONTENTS

ACKNOWLEDGEMENT.....	i
ABSTRACT.....	ii
TABLE OF CONTENTS.....	iii
LIST OF FIGURES.....	v
INTRODUCTION	vi
CHAPTER ONE: VISUAL BASIC.....	1
1.1.What is Visual Basic.....	01
1.2.1.The Advantages of Visual Basic.....	01
1.2.2. The Disadvantages of Visual Ba.....	02
1.3.Significant Language Features.....	02
1.4.Areas of Application.....	03
1.5.The Development Environment.....	04
1.6.The Project Explore Window.....	05
1.6.1.Properties Window.....	05
1.6.2.The Default Layout.....	05
1.6.3. Understanding The Tool Box.....	06
1.6.4. Opening an existing Visual Basic Project.....	06
1.6.5. Opening a new Visual Basic File & Inserting Source code.....	06
1.6.6. Running and Viewing the project in Detail.....	07
1.7.Understanding Events.....	08
1.7.1. What an Event is?.....	08
1.7.2. Click Event.....	09
1.7.3. Writing a Code.....	10
1.7.4. Using the Event GotFocus event.....	10
1.7.5. CheckBoxes.....	10
1.7.6. Option Buttons.....	11
1.7.7. ListBoxes.....	12
1.7.7.1 Add to a Listbox.....	13
1.7.7.2. Removing from Listboxes & Advanced Options.....	14
1.7.8. Msgboxes.....	14
1.7.9. Opening & Retrieving Information From files.....	16
1.7.10. Storing Information to a file.....	17
1.7.11. Printing Text to the printer.....	17
1.8. Control Arrays.....	18
1.9. Brief Information to the usages of Access data bases.....	18
1.10. Database Object.....	19
1.10.1. RecordSet Object.....	19
1.10.2. Accessing Records.....	19
1.10.3. Searching the RecordSet.....	20
1.10.4. Updating The database.....	21
1.10.5. Deleting And Adding records.....	21
1.10.5.1. Deleting Records.....	21
1.10.5.2. Adding Records.....	20
1.11. Managing Data Types.....	22
1.12. Introduction to VB Functions.....	24
1.13. Database Aplication.....	25
CHAPTER TWO: DATABASE STRUCTURE.....	26

2.1. Microsoft Access Description.....	26
2.2. Creating a database without using the Database Wizard.....	26
2.3. Tables.....	27
2.4. Primary Key.....	27
2.5. Tables Structure in Database.....	29
2.6. Microsoft Access Database.....	31
2.6.1. Microsoft Access Database Fundamentals.....	31
2.6.2. Creating Table.....	32
2.7. Introducing Database.....	32
2.8. Database Keys.....	33
2.9. Working With SQL.....	34
2.9.1. Data Manipulation Language.....	34
2.9.1.1. Insert.....	34
2.9.1.2. Select.....	35
CHAPTER THREE: STOCK CONTROL & CUSTOMER	
TRACKING PROGRAM USING.....	35
3.1. Entrance Page.....	35
3.2. Main Page.....	35
3.3. Menu Bar.....	41
3.3.1. Program Menu.....	41
3.3.2. Company Menu.....	41
3.3.2.1. Adding New User.....	42
3.3.2.2. Changing User Details.....	42
3.3.2.3. User & Company Query.....	43
3.3.4. Customer Menu.....	44
3.3.4.1. Adding New User.....	44
3.3.4.2. Payment Entrance Page.....	45
3.3.4.3. Customer Query Information Page.....	46
3.3.5. Stock Menu.....	46
3.3.5.1. Add Stock & Change Stock Detail Page.....	47
3.3.5.2. Refund Item Page.....	48
3.3.4.3. Customer Query Information Page.....	46
CONCLUSION.....	50
REFERENCE.....	51
APPENDIX.....	52

LIST OF FIGURES

Figure 1.1 An Empty Form.....	02
Figure 1.2: A Simple Program.....	03
Figure1.3. Development Environment.....	04
Figure 1.4. Properties Window.....	05
Figure1.5. Project of Program in Detail.....	07
Figure1.6. Form & Load.....	09
Figure 1.7. Click Event Form	09
Figure1.8. CheckBox Example.....	11
Figure 1.9. Background Changing Example.....	12
Figure1.10. A list on Form.....	13
Figure 1.10.1. Adding to List.....	13
Figure 1.11. Showing Message.....	16
Figure 2.1. Creating A database file.....	27
Figure 2.2. Customer table as an Example in Access.....	27
Figure 3.1. Entrance Page.....	36
Figure 3.2Main Menu.....	36
Figure 3.3. Adding Quantity to make sale.....	37
Figure 3.4.Show's All the customer.....	38
Figure 3.5. Main Menu When everything is ready for sale.....	38
Figure 3.6. Buttons of the main menu.....	39
Figure 3.7. The Last Sale Operation.....	40
Figure 3.8. Increase/Decrease Quantity.....	41
Figure 3.9. Program Menu in Main Page.....	41
Figure 3.10. Company menu in Main Page.....	40
Figure 3.11. Adding New User to Program.....	42
Figure 3.12. Change user Detail.....	42
Figure 3.13. Query User And Company information.....	43
Figure 3.14. Customer Menu in Main Menu.....	44
Figure 3.15. Add Customer Page.....	44
Figure 3.16. Payment Entrance page.....	45
Figure 3.17. Customer Query Information.....	46
Figure 3.18.Stock Menu in Main Page.....	46
Figure 3.19. Add Stock Page.....	47
Figure 3.20. Refund Item Page.....	48
Figure 3.21. Refund item to Customer Account.....	49

INTRODUCTION

Stock Control & Customer Tracking Program is a useful program for most mobile wholesaler, by the way of this program we can arrange everything of the market, if we use the program in a correct way nothing will be confused during any operation that take place in market for present time and future, also we can investigate past sale operations too.

By the way of this program we can calculate the net income of our products, see our capital, investigate what happened daily, list customers who make sale operations, list the users who make sales, Enter payments, calculate loans, show pictures of customers, giving every customer a special code, etc.

In the first Chapter I described Visual basic, it's properties, components and some examples, I used Visual Basic Program in my project, because I find it easy and I like it's coding system. Visual basic for applications delivers a competitive advantage for ISV seeking to provide full customization and integration capabilities to customer.

In the Second Chapter I described Database system, I used Microsoft Access Database system in my program with Visual basic, Advantages of using access is to provide exactly the same options for the problems we write as it does for the problems we selected from a database. Secondly, the process of writing or selecting problems is almost independent of page layout dictions. Also you can see more details about access and SQL.

Third Chapter is Stock Control and Customer Tracking Program, I explained it above, it is the most important Chapter of this project, because I made that program by myself alone by the help of some friends also through investigations from internet.

Finally, by the moderating of technology our program can be updated and new properties could be added in to the program in the future.

CHAPTER ONE: VISUAL BASIC

1.1. What Is Visual Basic?

VISUAL BASIC is a high level programming language evolved from the earlier DOS version called BASIC. BASIC means **B**eginners' **A**ll-purpose **S**ymbolic **I**nstruction **C**ode. It is a fairly easy programming language to learn. The codes look a bit like English Language. Different software companies produced different version of BASIC, such as Microsoft QBASIC, QUICKBASIC, GWBASIC, IBM BASICA and so on. Programmers have undergone a major change in many years of programming various machines. For example what could be created in minutes with Visual Basic could take days in other languages such: as "C" or "Pascal". Visual Basic provides many interesting sets of tools to aid you in building exciting applications. Visual Basic provides these tools to make your life far more easier because all the real hard code is already written for you.

With controls like these you can create many applications which use certain parts of windows. For example, one of the controls could be a button, which we have demonstrated in the "Hello World" program below. First create the control on the screen, then write the code which would be executed once the control button is pressed. With this sort of operation in mind, simple programs would take very little code. Why do it like the poor old "C" programmer who would have to write code to even display a window on the screen, when Visual Basic already has this part written for you.

Even though people tend to say Visual Basic's compiler is far behind the compilers of Pascal and C, it has earned itself the status of a professional programming language, and has almost freed BASIC of the reputation of a children's language. Overall you would class Visual Basic as a Graphics User Interface(GUI). Because as you draw, you write for the program. This must always be remembered in any kind of creation of a Visual Basic program. All in all, VB is the preferred language of many future program mers. If you want to start programming Windows, and don't know *how* to start, give Visual Basic a shot.

1.2.1. The advantages of Visual Basic:

- 1) It's simple language. Things that may be difficult to program with other language, Can be done in Visual Basic very easily.
- 2) Because Visual Basic is so popular, There are many good resources (Books, Web sites, News groups and more) that can help you learn the language. You can find the answers to your programming problems much more easily than other programming languages.
- 3) You can find many tools (Sharewares and Freewares) on the internet that will Spare you some programming time.

For example, if you want to ping a user over the internet in your program, Instead of writing the ping function yourself, you can download a control that does it, and use it in your program.

Compare to other languages, Visual Basic have the widest variety of tools that you can download on the internet and use in your programs.

1.2.2. The disadvantages of Visual Basic:

- 1) Visual Basic is powerful language, but it's not suit for programming really sophisticated games.
- 2) It's much more slower than other languages

1.3. Significant Language Features

Visual Basic is not only a programming language, but also a complete graphical development environment. This environment allows users with little programming experience to **quickly** develop useful Microsoft Windows applications which have the ability to use OLE (Object Linking and Embedding) objects, such as an Excel spreadsheet. Visual Basic also has the ability to develop programs that can be used as a front end application to a database system, serving as the user interface which collects user input and displays formatted output in a more appealing and useful form than many SQL versions are capable of.

Visual Basic's main selling point is the ease with which it allows the user to create nice looking, graphical programs with little coding by the programmer, unlike many other languages that may take hundreds of lines of programmer keyed code. As the programmer works in the graphical environment, much of the program code is automatically generated by the Visual Basic program. In order to understand how this happens it is necessary to understand the major concepts, objects and tools used by Visual Basic. The main object in Visual Basic is called a **form**. When you open a new project, you will start with a clear form that looks similar to this :

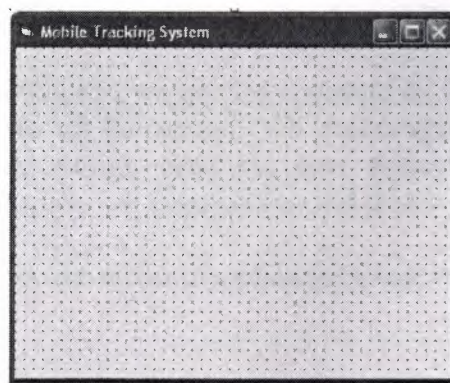


Figure 1.1. An Empty Form

This form will eventually be incorporated into our program as a window. To this form we add **controls**. Controls are things like text boxes, check boxes and command buttons...etc.

We can add **events** to our controls. Events are responses to actions performed on controls. For example, in the "Hello world" program sample on this page, when you click on the command button on our form the event that is triggered is the output of the message "Hello world" to the screen. Code must be written to create an event. we can do this in Visual Basic's **code window**.

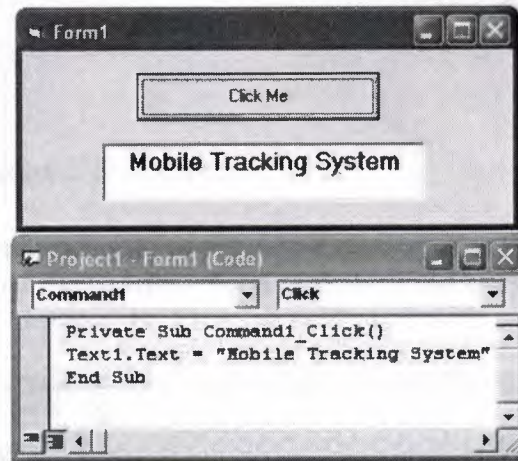


Figure 1.2. A Simple Program

Once the code box is open, you select the object to create an event for and the triggering action (such as a certain mouse action) from the drop down menus in the code box. You can open a code box for a particular form by choosing it from the **project window** and selecting the **View Code** button. The project window contains a list of objects associated with that project. Below is an example of a project window :

1.4. Areas of Application

The term "*Personal Programming*" refers to the idea that, wherever we work, whatever we do, we can expand our computer's usefulness by writing applications to use in our own job. Personal Programming is what Visual Basic is all about.

Using Visual Basic's tools, we quickly translate an abstract idea into a program design, we can actually see on the screen. VB encourages us to experiment, revise, correct, and network your design until the new project meets our requirements. However , most of all, it inspires our imagination and creativity.

Visual Basic is ideal for developing applications that run in the new Windows 95 operating system.

VB presents a 3-step approach for creating programs:

1. Design the appearance of application.
2. Assign property settings to the objects of program.
3. Write the code to direct specific tasks at runtime.

Visual Basic can be used in a number of different areas, for example:

- Education
- Research
- Medicine
- Business
- Commerce
- Marketing and Sales
- Accounting
- Consulting
- Law
- Science

1.5. The Development Environment

Learning the ins and outs of the Development Environment before we learn visual basic is somewhat like learning for a test, we must know where all the functions belong and what their purpose is. First we will start with labelling the development environment.

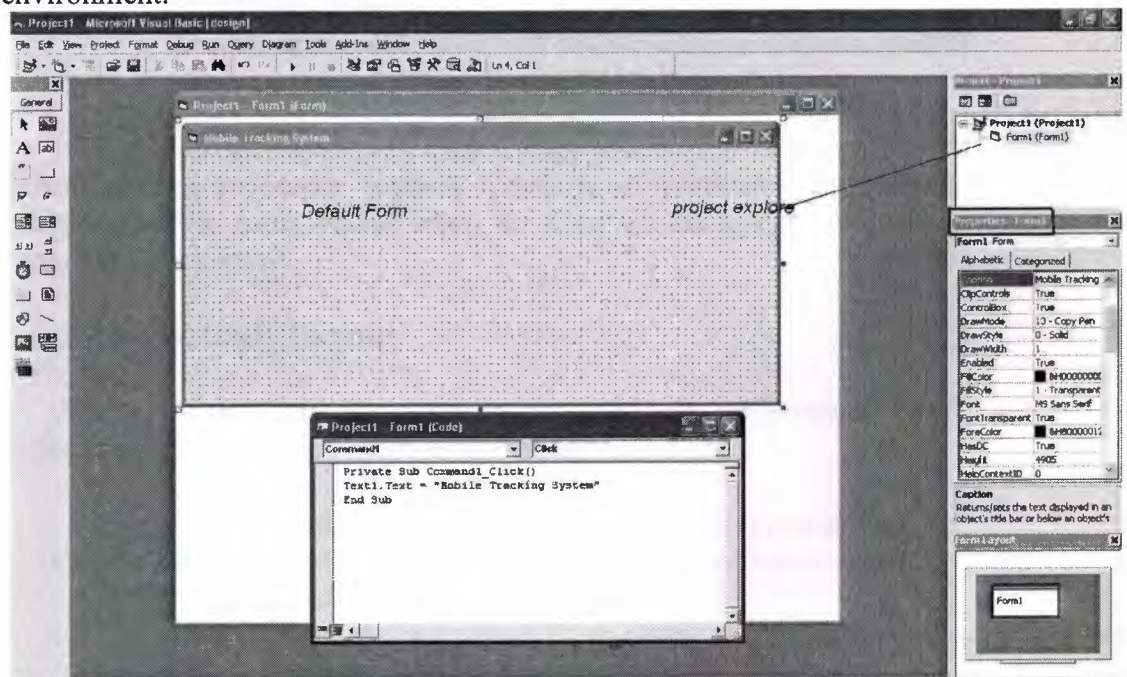


Figure 1.3. Development Environment

The above diagram shows the development environment with all the important points labelled. Many of Visual basic functions work similar to Microsoft word eg the **Tool Bar** and the tool box is similar to other products on the market which work off a single click then drag the width of the object required. The **Tool Box** contains the control we placed on the form window. All of the controls that appear on the **Tool Box** controls on the above picture never runs out of controls as soon as we place one on the form another awaits us on the **Tool Box** ready to be placed as needed.

1.6. The project explorer window

The Project explorer window gives us a tree-structured view of all the files inserted into the application. We can expand these and collapse branches of the views to get more or less detail (**Project explorer**). The project explorer window displays forms, modules or other separators which are supported by the visual basic like classes and Advanced Modules. If we want to select a form on its own simply double click on the project explorer window for a more detailed look. And it will display it where the **Default form** is located.

1.6.1.Properties Window

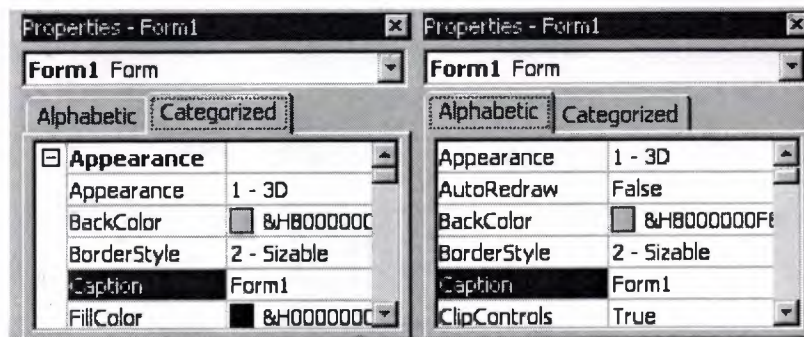


Figure 1.4. Properties Window

Some programmers prefer the Categorized view of the properties window. By defaulting, the properties window displays its properties alphabetically (with the exception of the name value) when we click on the categorized button the window changes to left picture.

1.6.2.The Default Layout

When we start Visual Basic, we are provided with a VB project. A VB project is a collection of the following modules and files.

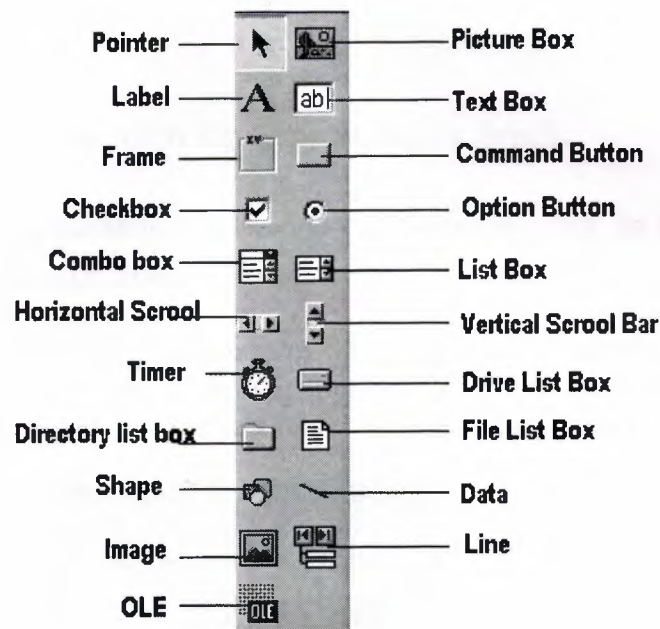
- The **global module**(*that contains declaration and procedures*)
- The **form module**(*that contains the graphic elements of the VB application along with the instruction*)
- The **general module** (*that generally contains general-purpose instructions not pertaining to anything graphic on-screen*)
- The **class module**(*that contains the defining characteristics of a class, including its properties and methods*)
- The **resource files**(*that allows you to collect all of the texts and bitmaps for an application in one place*)


On start up, Visual Basic will displays the following windows :

- The **Blank Form** window
- The **Project** window
- The **Properties** window

It also includes a **Toolbox** that consists of all the controls essential for developing a VB Application. Controls are tools such as boxes, buttons, labels and other objects draw on a form to get **input** or **display output**. They also add visual appeal.

1.6.3. Understanding the tool box.



When we click on different controls the Properties Window changes slightly this is due to different controls having different functions. Therefore more options are needed, for example if we had a picture then we want to show an image. But if we wanted to open a internet connection we would have to fill in the remote host and other such settings. When we use the command () we will find that a new set of properties come up the following will provide a description and a property.

1.6.4. Opening an existing Visual Basic project.

Microsoft have included some freebies with visual basic to show its capabilities and functions. Dismantling or modifying these sample projects is a good way to understand what is happening at runtime. These files can be located at our default directory **/SAMPLES/**

To Open these projects choose 'Open Project' from the 'File' menu. Then Double click on the samples folder to open the directory then Double click on any project to load it.

1.6.5. Opening a New Visual Basic File & Inserting Source code.

When we open the program we choose 'New Project' from the 'File' menu. We use the blank form1 to design a simple interface for an estate agents database, have some textboxes for names and other details. We insert some controls and make it look professional. Textboxes can be used to store there name and other details, we make sure that we put a picture box in for a picture of the house.




Later we insert the following source code for application.

Private Sub

```
Form_Load()Picture1.Picture=LoadPicture("C:\ProgramFiles\VB\Graphics\Icons\Misc\MISC42.ICO")
```

End Sub

1.6.6. Running and viewing the project in detail.

Once an application is loaded it can be run by clicking on the  icon from the toolbar, to pause press  and to terminate use .

Once a project is loaded, the name of the form(s) that it contains is displayed in the project window. To view a form in design mode, we select the form required by clicking with the mouse to highlight its name, then clicking on the view form button.

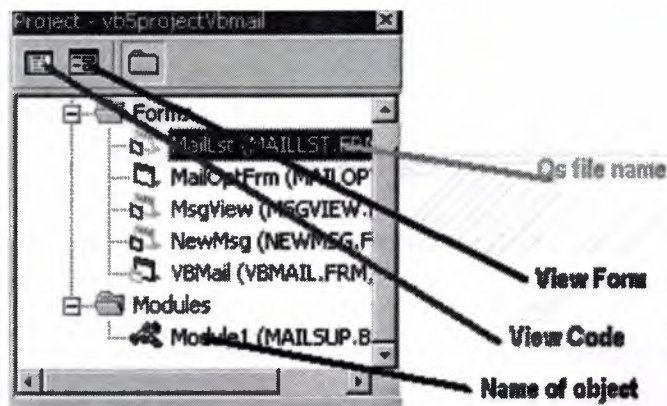


Figure 1.5. Project of the Program in Detail

Button Properties for reference

, Command Button &  labels properties

Property	Description
Name	The name of the object so you can call it at runtime
BackColor	This specifies the command button's background color. Click the <i>BackColor's</i> palette down arrow to see a list of common Windows control colours, you must change this to the style property from 0 - standard to 1 - graphical
Cancel	Determines whether the command button gets a Click event if the user presses escape
Caption	Holds the text that appears on the command button.
Default	Determines if the command button responds to an enter keypress even if


	another control has the focus
Enable	Determines whether the command button is active. Often, you'll change the enable property at runtime with code to prevent the user pressing the button
Font	Produces a Font dialog box in which you can set the caption's font name, style and size.
Height	Positions the height of the object - can be used for down
Left	Positions the left control - can be used for right
MousePointer	If selected to an icon can change the picture of the mouse pointer over that object
Picture	Hold's the name of an icon graphic image so that it appears as a picture instead of a Button for this option to work the graphical tag must be set to 1
Style	This determines if the Command Button appears as a standard windows dialog box or a graphical image
Tab index	Specifies the order of the command button in tab order
Tab Stop	Whether the object can be tabbed to (this can be used in labels which have no other function)
Tool Tip Text	If the mouse is held over the object a brief description can be displayed (for example hold your mouse over one of the above pictures to see this happening)
Visible	If you want the user to see the button/label select true other wise just press false
Width	Show the width of the object

1.7. Understanding Events

- Know what an Event is.
- Determine what Events a control can have
- Write code for one or more Events.
- Using optionbuttons to produce an event
- Using checkboxes to produce an event
- Grouping controls using a frame
- Make a simple alteration to the interface, such as changing background colour, at run time.
- Creating a listbox.
- Remove and Add listboxs functions.
- Creating Combo Boxes
- What the different types of combo boxes are.

1.7.1. What an event is

The 'look' of a Visual Basic application is determined by what controls are used, but the 'feel' is determined by the events. An event is something which can happen to a control. For example, a user can click on a button, change a text box, or resize a form. As explained in Creating a Visual Basic Application, writing a program is made up of three events: 1) select suitable controls, 2) set the properties, and 3) write the code. It is

at the code writing stage when it becomes important to choose appropriate events for each control. To do this double click on the control the event will be used for, or click on the  icon in the project window (usually top right of screen). A code window should now be displayed similar to the one shown below.

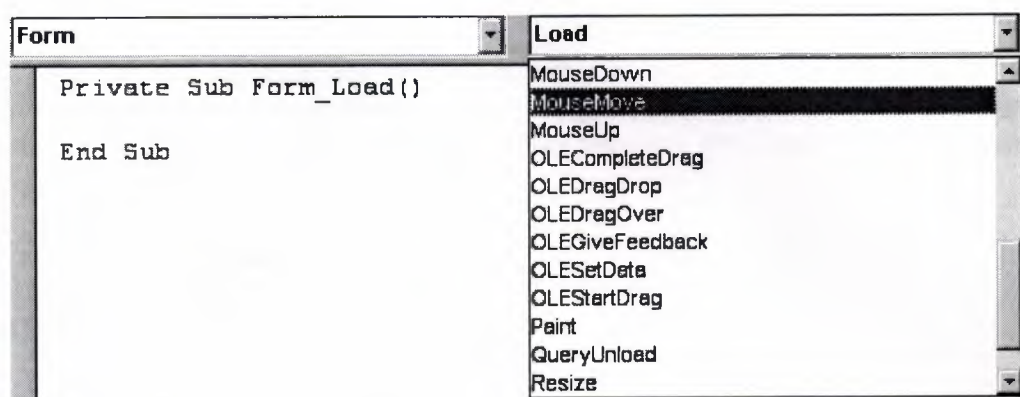


Figure 1.6. Form & Load

The left hand dropdown box provides a list of all controls used by the current form, the form itself, and a special section called General Declarations. The corresponding dropdown box on the right displays a list of all events applicable to the current control (as specified by the left hand dropdown box). Events displayed in bold signify that code has already been written for them, unbold events are unused. To demonstrate that different events can play a significant role in determining the feel of an application, a small example program will be written to add two numbers together and display the answer. The first solution to this problem will use the click event of a command button, while the second will use the change event of two text boxes.

1.7.2. Click Event

Before any events can be coded it is necessary to design the interface from suitable controls. As shown in the screen shot below use: 2 text boxes to enter the numbers, a label for the '+' sign, a command button for the '=' sign, and another label for the answer.

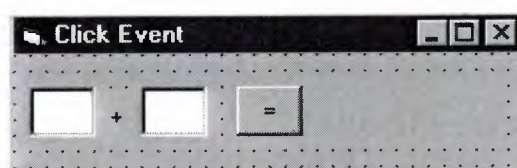
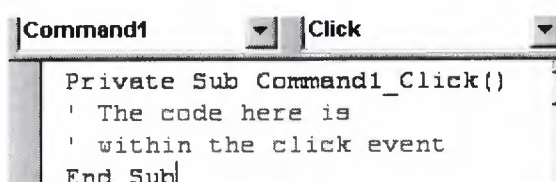


Figure 1.7. Click Event Form

Making the click event is very simple just select the button with the mouse and double click visual basic will generate



You can see on the top right there is a 'click' dropdown list this is known as a event handler.

1.7.3. Writing a Code


In the first example the user enter two numbers and then click on the equals button to produce an answer. However, the program can be changed so that the answer will be calculated every time either of the two numbers are changed without requiring an equals button.

```
Private Sub txtNumber1_Change()  
label2.Caption = Str$(Val(text1.Text) + Val(text.Text))  
End Sub
```

```
Private Sub txtNumber2_Change()  
label2.Caption = Str$(Val(text1.Text) + Val(text2.Text))  
End Sub
```

When we run the program, we enter two numbers and observe what happens. Each time a digit changes the answer is recalculated.

1.7.4. Using the event GotFocus event

So far only one event has been used per control, however this does not have to be the case! Add a StatusBar control to the bottom of the form, bring up the code window using , select the first text box (txtNumber1) from the left hand dropdown box, and then select the **GotFocus** event from the right hand dropdown box. Now some basic instructions can be written in the status bar so that when the cursor is in the text box (the text box has focus) the status bar reads "Enter the first number". After completing this change to the second text box and using the same GotFocus event change the statusbar text to "Enter a second number". The code to set the status bar can be seen below.

1.7. 5. CheckBoxes

Option bars are used quite often in the windows environment as they can only have two outputs 0 and 1 these get used to process the form. In this example it will be used to change the some text from normal to bold or to italic.

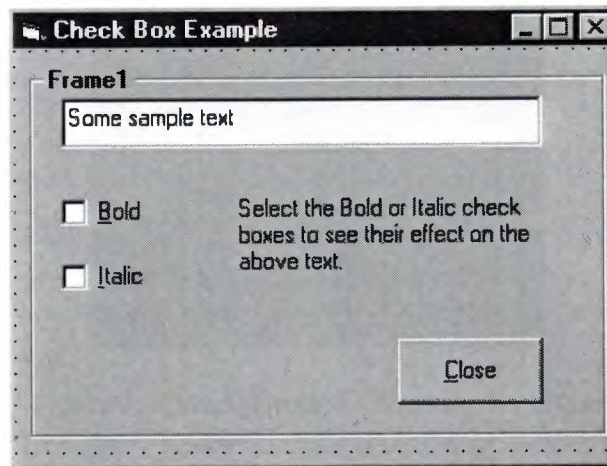


Figure 1.8. Check Box Example

Private Sub chkBold_Click()

```
If chkBold.Value = 1 Then ' If checked.
txtDisplay.FontBold = True
Else ' If not checked.
txtDisplay.FontBold = False
End If
End Sub
```

Private Sub chkItalic_Click()

```
If chkItalic.Value = 1 Then ' If checked.
txtDisplay.FontItalic = True
Else ' If not checked.
txtDisplay.FontItalic = False
End If
End Sub
```

1.7.6.Option Buttons

Changing the background colour

Changing the background colour gets used mostly by freeware, or the type of programs we generate for use with banners or adverts, anyway it might come in useful sometime. This example shows an ordinary picture of a smiley face then I put in some nice background colours to make it stand out more.

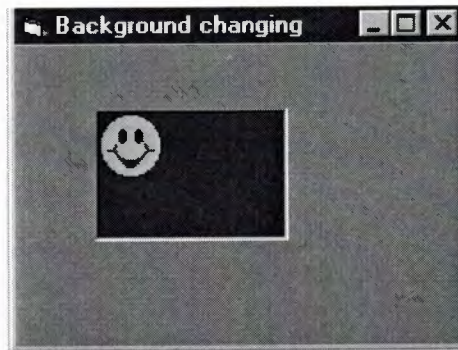




Figure 1.9. Back Ground Changing Example


```
Private Sub Form_Load()
BackColor = QBColor(Rnd * 15)
ForeColor = QBColor(Rnd * 10)
Picture1.BackColor = QBColor(Rnd * 15)
Picture1.ForeColor = QBColor(Rnd * 10)
End Sub
```

1.7.7. List boxes

List boxes and combo boxes are used to supply a list of options to the user. The toolbox icons representing these two controls are  for list box and  for combo box.

A list box is used when the user is to be presented with a fixed set of selections (i.e. a choice must be made only from the items displayed, there is no possibility of the user typing in an alternative).

Examples might be offering a list of the days in a week, the available modules in an elective catalogue, the holiday destinations available from a particular airport or the types of treatment offered by a beauty salon.

To create a list box, double click on the toolbox icon . Drag the resulting box into place on the form and size it according to the data it will hold. The left hand picture below shows a list box that has been created and sized on Form1. In the middle is the code that is required to load a selection of cars into the list. The data has been included in the procedure Sub Form_Load so that it appears when Form1 is loaded. Finally, the picture on the right shows what appears on the form when the application is run. Notice that vertical scroll bars are added automatically if the list box is not deep enough to display all the choices.

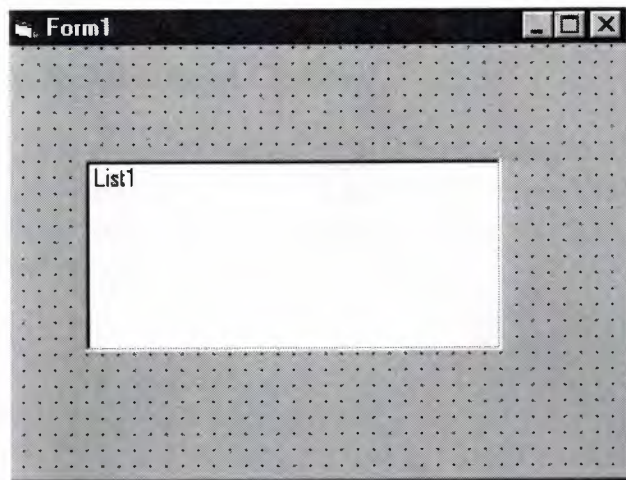


Figure 1.10. A list On Form

If we however add the following source code to this project

1.7.7.1. Add to a Lisbox

```
Private Sub Form_Load()  
List1.AddItem "Monday"  
List1.AddItem "Tuesday"  
List1.AddItem "Wednesday"  
List1.AddItem "Thursday"  
List1.AddItem "Friday"  
List1.AddItem "Saturday"  
List1.AddItem "Sunday"  
End Sub
```

The source code should look something like this :

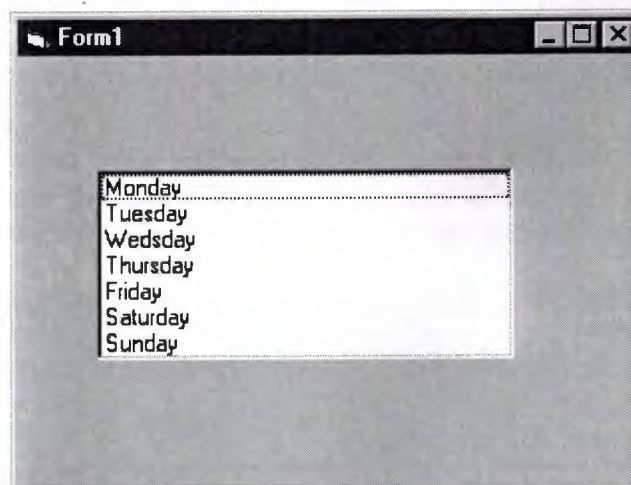


Figure 1.10.1. Adding To List

1.7.7.2.Removing & Advanced Options

They appear in the order they were typed if you changed the properties window by changing sort order = true then they will go into alphabetical order. List items can be added and deleted all the list is counted as the order they are in so for example if you wanted to delete "Tuesday" you would type

list1.RemoveItem 1

And to add to the list in a particular order just add

list1.additem "My Day", 5

This will be added after Saturday.

And finally to clear the listbox type

List1.clear This will completely clear the contents of the listbox.

The property ListCount stores the number of items in a list, so list1.ListCount can be used to determine the number of items in list box list1.

The property ListIndex gives the index of the currently selected list item. So the statement list1.RemoveItem List1.ListIndex removes the currently highlighted list item.

Adding an item can be accomplished very neatly using an input dialog box. Try this:

list1.AddItem InputBox("Enter a day", "Add a Day")

This will open a message box and prompt you for a new day to enter this will then be added to the list index at the bottom unless you specify where it should go.

1.7.8. MsgBoxes

Message boxes are used when you want to ask the user a question or display an error message(s) and advise the user. There are six types of message boxes here are their functions and what they do. Here is the listing of all the possible msgbox events

The Buttons displayed in a message here Button Layout	Value	Short Description
vbOKOnly	0	Displays the OK button.
vbOKCancel	1	Displays the ok and cancel button.
vbAbortRetryIgnore	2	Displays the Abort , Retry , Ignore
vbYesNoCancel	3	Displays Yes , No and Cancel button
vbYesNo	4	Displays the Yes / No button

vbRetryCancel	5	Displays the retry and Cancel buttons.
---------------	---	--

The Icons displayed in the message box are here

Icon on message	Value	Short Description
vbCritical	16	Displays critical message icon
vbQuestion	32	Displays question icon
vbExclamation	48	Displays exclamation icon
vbInformation	64	Displays information icon

The Default button displayed in a message form Default Button	Value	Short Description
vbDefaultButton1	0	Button 1 is default
vbDefaultButton2	256	Button 2 is default
vbDefaultButton3	512	Button 3 is default

Msgbox Return Value Return Value	Value	Short Description
vbOk	1	The User Clicked OK
vbCancel	2	The User Clicked Cancel
vbAbort	3	The User Clicked Abort
vbRetry	4	The User Clicked Retry
vbIgnore	5	The User Clicked Ignore
VbYes	6	The User Clicked Yes
VbNo	7	The User Clicked No

The syntax for use of the message box in MsgBox "TEXT", VALUE, "TITLE"
If you want to use two or more tables just add the values together. Therefore to print a message box to say "The Device was not Found!" OK & Explanation :

Source code 1

Private Sub Form_Load()

MsgBox "The Device was not Found!", 48, "Header"

End Sub

Source code 2

Private Sub Form_Load()

MsgBox "The Device was not found!", vbExclamation, "Header"

End Sub

we should get the picture shown below whatever source code you used.

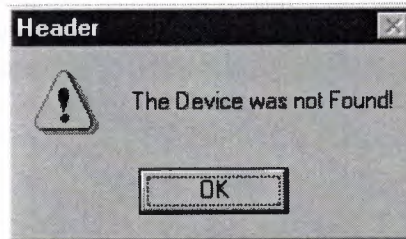


Figure 1.11. Showing message

This is a basic msgbox which in this case has not been processed in any way. The following Source code displays a msgbox that ask you for specific text. For example lets make a password program out of this message box.

Private Sub Form_Load()

lngBefore = Timer

Do

strAns = InputBox("What is the password Password is Example", "Password Required")

Loop Until Val(strAns) = Example

lngAfter = Timer

msgbox "Correct Password", vbInformation

End Sub

Once you copy and paste the source code you should get prompted for a password in a different type of msgbox as it includes text. From looking at this example you should be able to see how the loop function works to generate and then check the value. All of the Return Values work in the same way to return an object input.

1.7.9.Opening & Retriving information from files

When applications are loaded they normal get some setting out of the registry or file this section will show you how to retrieve 1 string out of a file.

Private Sub Form_Load()

Dim F As Integer, password As String

F = FreeFile

Open App.Path & "\password.txt" For Input As F

Input #F, password

Close #F

End Sub

As you can see from this source code the password is previously declared as a string. After this is done be sure to close the file otherwise next time you want to store or read the file the computer will think it is being used by another application and windows will not let you do anything with it. So as you can see it is **Very Important** to close the file

1.7.10. Storing Information to a file

FTP programs often store information to a file such as a username and password or host information in the same way. This following example will put some information into the file.

Private Sub Form_Load()

```
Dim F As Integer, pass As String
F = FreeFile
save = txtNew
Open App.Path & "\password.txt" For Output As F
Write #F, Text1.text
Close #F
```

End Sub

Although this is a bit obvious I think I should include it just incase I think differently to other people.

1.7.11. Printing text to the printer.

This is rather easy to do and it gets used in notepad etc...

Private Sub Form_Load()

```
Printer.Print " The printer will print this text "
Printer.Print ""
Printer.Print " It will leave a line here"
Printer.Print " It should add the contence of text1.text here : " & Text1.Text & " As you
can see it works"
Printer.Print ""
Printer.EndDoc "This will tell the printer it has finished.
```

End Sub

Everything that appears in position (A) will get printed by the default printer printer.print " A ". The printer enddoc is used to tell the printer the job is finished if this is not added the printer can not estimate how near it will be until it has finish and when it has finished it will think it has'nt so be sure to include this to prevent confusion.

1.8. Control Arrays

A control array is a list of controls with the same name. Therefore, instead of using five command buttons with separate five names, you can place a command button control array on the form, and that control array holds five command buttons. The control array can have a single name, and you will distinguish the control from each other with a subscript. One of the best reasons to use control array from that first control, all the elements in the control array take on the same property values, You then can change those properties that need to be changed without having to set every property for each control individually. Control arrays have a lot in common with data arrays. A control array has one array, and you distinguish all the array's controls from each other with the zero-based subscript. (The index property holds the controls subscript number). All of the control elements must be the same data type. As soon as you place a control on a form that has the same name as an existing control, Visual Basic makes sure you that you want to begin a control array by issuing the warning message to show that the control is already in use. This is used as a built in safety so that you do not over right an existing control by putting it some where else on the same form. If you answer the warning box with a no button, Visual Basic uses a default control name for the placed control.

Picture Not available at the moment!

All event procedures that use control from a control array require a special argument value passed to them that the determines which control is being worked on. For example if your application contains a single control command button named cmdtotal the click () event begins and ends as follows

If however you create a control array named the same name as before (cmdtotal) it will end up like this

1.9. Brief introduction to the usages of Access data bases

What I think is the most compelling thing about Visual Basic is it's easy way of accessing and modifying databases. This is what I think you should learn next; you will find many applications for this knowledge. I almost never make a program without using a database for data storage.

There are many ways to work with databases in Visual Basic, and I would think you have at least glanced at the Data control. I will not even mention the Data control further in this text, since it is so easy to use and too limited to be interesting for a professional developer. (Ok, there are some exceptions to this.)

What I will teach you to use in this text is DAO (Data Access Objects). You will get familiar with opening a database and retrieving/adding/deleting/updating records from tables. I will only use an Access Database (*.mdb) in my examples, since this is the most used DBMS (DataBase Management System) for smaller applications made in Visual Basic. We will at the end of this lesson have made a simple, yet functional, phone book application.

1.10. Database Object

The first thing we must do in our application is to open a database where our tables are stored. we need to declare a variable to hold our database in order to do this. This is done with:

```
Dim dbMyDB As Database
```

This gives us a variable/object that can hold a reference to our database. To open a simple Access database named "MyDatabase.mdb", do this:

```
Set dbMyDB = OpenDatabase("MyDatabase.mdb")
```

So, now we have opened a database. This won't give us any data. What we need to do is open a table in the database. We're not limited to open a single table; sometimes we have two or more tables that are related to each other and linked together with foreign keys, and there are ways to handle this to.

1.10.1. RecordSet Object

Visual Basic uses an object called RecordSet to hold our table. To declare such an object and to open the table, we do this:

```
Dim rsMyRS As RecordSet
```

```
Set rsMyRS = dbMyDB.OpenRecordSet("MyTable", dbOpenDynaset)
```

Well, I declared a RecordSet object and used the Database object's OpenRecordSet method to open a table of type Dynaset. We can open a RecordSet in several modes. VB's online help file explains the different modes and what they are for. The Dynaset mode is the mode I use mostly. It gives us a RecordSet that we can add, delete and modify records in.

1.10.2. Accessing records

Now that we have opened a table (referred to as RecordSet from now on) we want to access the records in it. The RecordSet object allows us to move in it by using the methods MoveFirst, MoveNext, MovePrevious, MoveLast (among others). I will use some of these to fill up a list box with the records of our RecordSet.

To get this example to work, we make a database (with Access) called "MyDatabase.mdb" with the table "MyTable" in it. This table should have the fields "ID" of type "Counter" that you set to be the primary key, the field "Name" of type Text and a field "Phone" of type Text. We add some records to it. We put a list box on a form and call it "lstRecords".


```
Dim dbMyDB As Database
Dim rsMyRS As RecordSet
```

Private Sub Form_Load()

```
Set dbMyDB = OpenDatabase("MyDatabase.mdb")
Set rsMyRS = dbMyDB.OpenRecordSet("MyTable", dbOpenDynaset)

If Not rsMyRS.EOF Then rsMyRS.MoveFirst
Do While Not rsMyRS.EOF
    lstRecords.AddItem rsMyRS!Name
    lstRecords.ItemData(lstRecords.NewIndex) = rsMyRS!ID
    rsMyRS.MoveNext
Loop
```

End Sub

This will make the list box fill up with our records when the form loads. I have introduced some new concepts with this example. We have all ready covered the first part where we open the table. The line that says `If Not rsMyRS.EOF Then rsMyRS.MoveFirst` tells the program to move to the first record in case there are any records at all. The `EOF` is a Boolean property that is true if the current record is the last. It is also true if there are no records in the RecordSet.

Then we make the program add the "Name" field of all records to the list box by adding the current records field "Name" and moving to the next record. You ask for a field of a RecordSet by putting a ! between the name of the RecordSet object and the name of the field. The while loop checks to see if there are more records to add.

1.10.3. Searching the RecordSet

I put the value of the field "ID" in the list box's ItemData property. I did this so that we would know the primary key for all the records in order to search for a record.

If we put a text box somewhere on the form and call it "txtPhone". Then copy the following code to the project.

Private Sub lstRecords_Click()

```
rsMyRS.FindFirst "ID=" & Str(lstRecords.ItemData(lstRecords.ListIndex))
txtPhone.Text = rsMyRS!Phone
```

End Sub

This will display the phone number of the selected person when clicking in the list box. It uses the FindFirst method of the RecordSet object. This takes a string parameter that is like what is after WHERE in a SQL expression. We state the field that we want to search in (here "ID"), then the evaluation criteria (here "=") and last the value to search for.

So what we did was to search for the record with the "ID" field value that was the same as the ItemData property of the selected item in the list box. Then we show the value of the "Phone" field in the text box.

1.10.4. Updating the Database

We want to be able to update some value of some field when doing database programming. This is done with `Edit` and `Update`. We will try to change the value of the "Phone" field by editing the text in the text box and clicking a button.

If we put a command button on the form and name it "cmdUpdate". Then copy the following code to the project.

Private Sub cmdUpdate_Click()

```
rsMyRS.Edit  
rsMyRS!Phone = txtPhone.Text  
rsMyRS.Update
```

End Sub

This changes the value of the "Phone" field of our current record. Imagine the current record being a set of boxes, with a field in each box. The `Edit` method takes the lid off all of the boxes and `Update` puts them back on. When we write `rsMyRS!Phone = txtPhone.Text` we replace the content of the "Phone" box with the content in the text box.

1.10.5. Deleting and Adding records

1.10.5.1. Deleting

Deleting records couldn't be simpler. To delete the current record we just invoke the `Delete` method of the RecordSet object. We will put this feature in our little project. If we make one more command button named "cmdDelete" and the following code will do the work of deleting our currently selected person.

Private Sub cmdDelete_Click()

```
rsMyRS.Delete  
lstRecords.RemoveItem lstRecords.ListIndex
```

End Sub

I won't even bother to explain that in greater detail =). The first statement deletes the record and the second removes the list box entry.

1.10.5.2. Adding

Adding records is much like updateing, except that we use `AddNew` instead of `Edit`. Let's add one more command button to our application.

Private Sub cmdNew_Click()

```
rsMyRS.AddNew
rsMyRS!Name = "A New Person"
lstRecords.AddItem rsMyRS!Name
lstRecords.ItemData(lstRecords.NewIndex) = rsMyRS!ID
rsMyRS!Phone = "Person's Phone Number"
rsMyRS.Update
```

End Sub

1.11. Managing Data Types

There are many types of data we come across in our daily life. For example, we need to handle data such as names, addresses, money, date, stock quotes, statistics and etc everyday. Similarly in Visual Basic, we are also going to deal with these kinds of data. However, to be more systematic, VB divides data into different types.

Numeric Data

Numeric data are data that consists of numbers, which can be computed mathematically with various standard operators such as add, minus, multiply, divide and so on. In Visual Basic, the numeric data are divided into 7 types, they are summarized in Table 1.1

Type	Storage	Range of Values
Byte	1 byte	0 to 255
Integer	2 bytes	-32,768 to 32,767
Long	4 bytes	-2,147,483,648 to 2,147,483,648
Single	4 bytes	-3.402823E+38 to -1.401298E-45 for negative values 1.401298E-45 to 3.402823E+38 for positive values.
Double	8 bytes	-1.79769313486232e+308 to -4.94065645841247E-324 for negative values 4.94065645841247E-324 to 1.79769313486232e+308 for positive values.

Table 1.1: Numeric Data Types

Non-numeric Data Types

The nonnumeric data types are summarized in Table 1.2

Data Type	Storage	Range
String(fixed length)	Length of string	1 to 65,400 characters
String(variable length)	Length + 10 bytes	0 to 2 billion characters
Date	8 bytes	January 1, 100 to December 31, 9999
Boolean	2 bytes	True or False

Table 5.2: Nonnumeric Data Types

Declaring Variables

In Visual Basic, one needs to declare the variables before using them by assigning names and data types. They are normally declared in the general section of the code windows using the **Dim** statement.

The format is as follows:

```
Dim variableName as DataType
```

Example 1.3

```
Dim password As String
Dim yourName As String
Dim firstnum As Integer
Dim secondnum As Integer
Dim total As Integer
Dim doDate As Date
```

You may also combine them in one line, separating each variable with a comma, as follows: `Dim password As String, yourName As String, firstnum As Integer,.....`

If.....Then.....Else Statements with Operators

To effectively control the VB program flow, we shall use If...Then...Else statement together with the conditional operators and logical operators. The general format for the if...then...else statement is

```
If conditions Then
    VB expressions
Else
    VB expressions
End If
```

Select Case

If you have a lot of conditional statements, using If..Then..Else could be very messy. For multiple conditional statements, it is better to use Select Case. The format is :

Select Case expression

```
Case value1
    Block of one or more VB statements
Case value2
    Block of one or more VB Statements
End Select
```

Looping

Visual Basic allows a procedure to be repeated as many times as long as the processor could support. This is generally called looping .

1. **Do**
 Block of one or more VB statements
 Loop While condition

2. **Do**
 Block of one or more VB statements

Loop Until condition

3. For....Next Loop

 For counter=startNumber to endNumber (Step increment)

 One or more VB statements

Next

1.12. Introduction to VB Functions

Creating Functions

The general format of a function is as follows:

```
Public Function functionName (Arg As dataType,.....) As dataType
```

or

```
Private Function functionName (Arg As dataType,.....) As dataType
```

*Public indicates that the function is applicable to the whole program and Private indicates that the function is only applicable to a certain module or procedure.

Declaring Array

We could use Public or Dim statement to declare an array just as the way we declare a single variable. The Public statement declares an array that can be used throughout an application while the Dim statement declare an array that could be used only in a local procedure. The general format to declare an array is as follow:

Dim arrayName(subs) as dataType

where subs indicates the last subscript in the array.

Example 13.1

Dim FindName(10) as String

1.13.Database Aplication

Finally, I want to put end point in that part, with telling about database aplication. Of course, All topics is not restricted what I emphasize in that part.

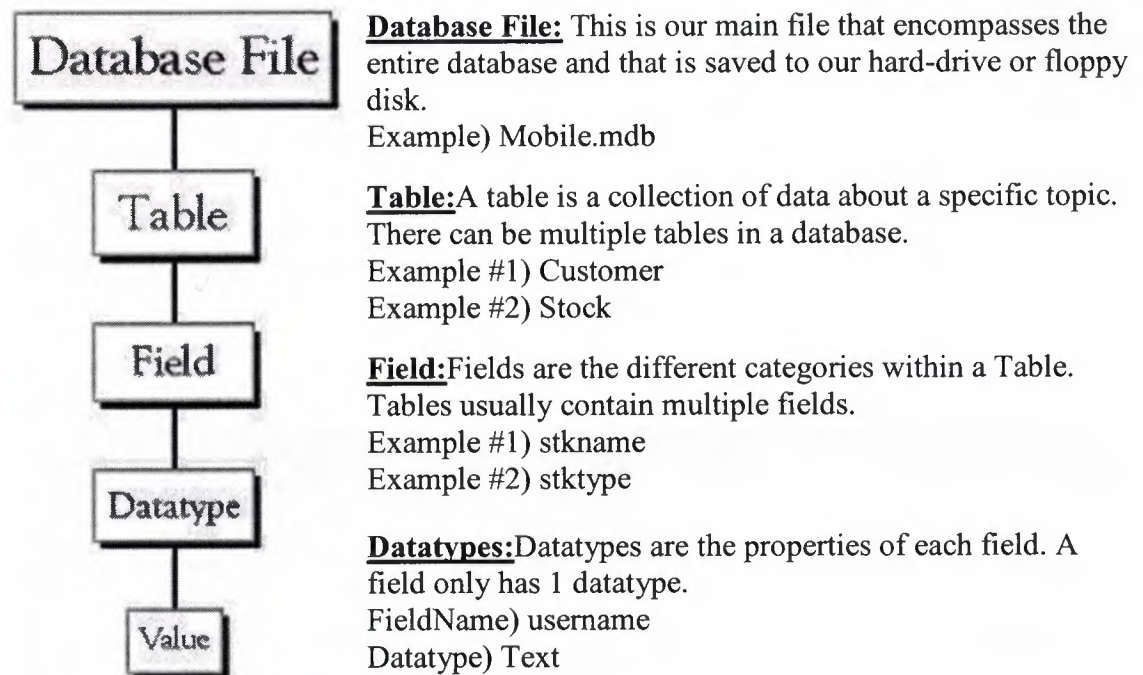
Visual basic allows us to manage databases created with different database program such as MS Access, Dbase, Paradox and etc. In this lesson, we are not dealing with how to create database files but we will see how we can access database files in the VB environment

CHAPTER TWO: DATABASE STRUCTURE

2.1. Microsoft Access Description

Microsoft Access is a powerful program to create and manage our databases. It has many built in features to assist us in constructing and viewing our information. Access is much more involved and is a more genuine database application than other programs such as Microsoft Works.

First of all we need to understand how Microsoft Access breaks down a database. Some keywords involved in this process are: *Database File, Table, Record, Field, Data-type*. Here is the Hierarchy that Microsoft Access uses in breaking down a database.



2.2. Creating a database without using the Database Wizard

1. When Microsoft Access first starts up, a dialog box is automatically displayed with options to create a new database or open an existing one. If this dialog box is displayed, we click **Blank Access Database**, and then click **OK**.

If we have already opened a database or closed the dialog box that displays when Microsoft Access starts up, we click **New Database** on the toolbar, and then double-click the **Blank Database** icon on the **General** tab.

2. We specify a name and location for the database and we click **Create**. (Below is the screen that shows up following this step)

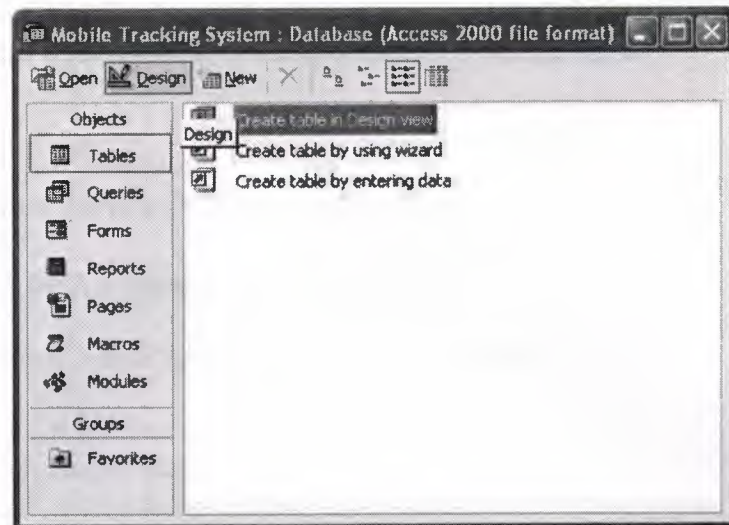


Figure 2.1. Creating A database File

2.3. Tables

A table is a collection of data about a specific topic, such as Mobiles or Customers. Using a separate table for each topic means that you store that data only once, which makes your database more efficient, and reduces data-entry errors.

Tables organize data into columns (called **fields**) and rows (called **records**).

names organize data in columns (names, mobile numbers, addresses, companies, emails)

customer : Table								
	cuscode	name	surname	mobile	tel	address	company	email
▶ 1		Muhammad	simko	05338773781	05367831198	Muhendisler Mah.-Erbil-Iraq	wow café	muhammadwow@hotmail.com
10		Muhammad	Abdulgani	+9647504450908	00447028640908	Adala-Erbil-Iraq	Furkan	hama1987@yahoo.com
11		Muhammad	Serkawt	+9647504458687	00447028648687	Iskan-Erbil-Iraq	Sarkawt Const.	hamawow@yahoo.com
12		Abdullah	Mizrakci	05338460867	05373915229	Istanbul	Mizrakci	
13		Adem	Atceken	05428738841	05363628394	Konya	Atceken	adem.atceken@hotmail.com
14		saner	Fahrettin	07504480588	0662261466	Ziraa Mah.-Erbil-Iraq	Muhiddinler	saner111@yahoo.com
15		Zubeyir	Cokakli	05428732077	03922278275	Ortakoy-Kuzey Kibris	Nurcu	batta99@yahoo.com
16		Ahmad	selman	05333333333	05334893498	Jordan	Ahmed Co.	ahmadselema@hotmail.com
17		ali	chima	05338300251	05338300251	pakistan	chima	
18		ilker	Damar	05428830353	05372500093		Damar	
19		Rdvan	Gologlu	05324262030	04322231375	Van-Merkez	Gologlu Ekmek Fab	
2		Ozer	Dastan	05338491105	05053496376	Tortum-Erzurum	Esinler	ozet dastan 1987@hotmail.com
3		Rojgar	Jemal	05336773771	05356817414	Azadi Mah-Erbil-Iraq	Peris Co.	rojgar_84@yahoo.com
4		Ali Said	Olgun	05338492631	05354566225	Istanbul	Olgunlar sirketi	ali_said_olgun@yahoo.com
5		Hasean	Karmin	05338767180	05378658884	Kilic-Konya	Sumarholding	hasean12@hotmail.com

Figure 2.2. Customer Table As an Example in Access

2.4. Primary Key

- One or more fields (columns) whose value or values uniquely identify each record in a table. A primary key does not allow Null values and must always have a unique value. A primary key is used to relate a table to foreign keys in other tables.
- We do not have to define a primary key, but it's usually a good idea. If you don't define a primary key, Microsoft Access asks us if we would like to create one when we save the table.

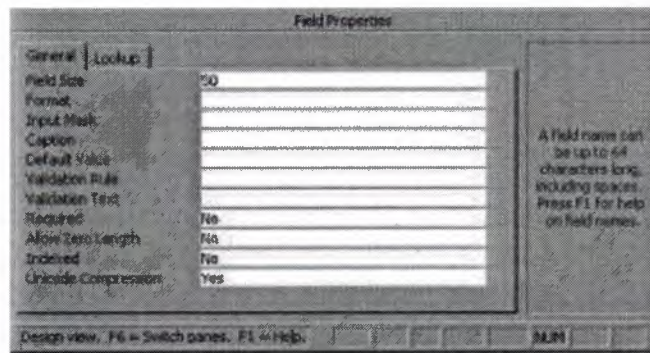
- For our tutorial, we make the **Soc Sec #** field the primary key, meaning that *every* one has a social security number and no 2 are the same.

Advanced Table Features w/Microsoft Access

- **Assigning a field a specific set of characters**

Example: Customer Name needs maximum 20 character.

1. Switch to Design View
2. Select the field you want to alter
3. At the bottom select the General Tab



4. Select **Field Size**
5. Enter the number of characters you want this field to have

- **Formatting a field to look a specific way**

Example) Formatting Phone Number w/ Area Code (xxx) xxx-xxxx

1. Switch to Design View
 2. Select the field you want to format
 3. At the bottom select the General Tab
 4. Select **Input Mask Box** and click on the ... button at the right.
- **Selecting a value from a dropdown box with a set of values that you assign to it. This saves you from typing it in each time**

Example) Choosing a city that is either Auburn, Bay City, Flint, Midland, or Saginaw

1. Switch to Design View
2. Select the field you want to alter (City)
3. At the bottom select the Lookup Tab
4. In the **Display Control** box, select **Combo Box**
5. Under **Row Source Type**, select **Value List**

2.5. Tables Structure In Database

As I mentioned before I used seven tables in Access Database. All the tables are given below.

Content.DB			
Field Name	Data Type	Field Size	Key
sellcode	Text	20	*
Stockcode	Text	20	
Total	Currency	Format:2 Decimal:2	
quantity	Number	Long Integer	

Table 2.1. Content Database Table

Customer.DB			
Field Name	Type	Size	Key
Cuscode	Text	5	*
Name	Text	30	
Surname	Text	30	
Mobile	Text	17	
Tel	Text	17	
Address	Text	50	
Company	Text	50	
Email	Text	50	

Table 2.2. Customer Database Table

Loan.DB			
Field Name	Type	Size	Key
Cuscode	Text	5	*
Sellcode	Text	20	
Total	Currency	Standart, decimal:2	

Table 2.3. Loan Database Table

Payment.DB			
Field Name	Type	Size	Key

Cuscode	Text	50	
Date	Date/Time	Short Date	
Total	Currency	Standart, Decimal: 2	
username	Text	20	

Table 2.4. Payment Database Table

Sale.DB			
Field Name	Type	Size	Key
Sellcode	Text	20	*
Username	Text	20	
Cuscode	Text	5	
Date	Date/Time	Short date	
Total	Currency	Standart, Decimal :2	

Table 2.5. Sale Database Table

Stock.DB			
Field Name	Type	Size	Key
Stockcode	Text	10	*
Stkname	Text	40	
Stktype	Text	20	
Pureprice	Currency	Standart, Decimal :2	
Transportation	Currency	Standart, Decimal :2	
Saleprice	Currency	Standart, Decimal :2	
Amount	Number	Long integer	

Table 2.6. Stock Database Table

User.DB			
Field Name	Type	Size	Key
Username	Text	20	*
Usertype	Text	20	
Password	Text	10	

Table 2.7. User Database Table

2.6. Microsoft Access Database

2.6.1. Microsoft Access Database Fundamentals

Are you overwhelmed by the large quantities of data that need to be tracked in your organization? Perhaps you're currently using a paper filing system, text documents or a spreadsheet to keep track of your critical information. If you're searching for a more flexible data management system, a database might be just the salvation you're looking for.

What is a database? Quite simply, it's an organized collection of data. A database management system (DBMS) such as Access, FileMaker Pro, Oracle or SQL Server provides you with the software tools you need to organize that data in a flexible manner. It includes facilities to add, modify or delete data from the database, ask questions (or queries) about the data stored in the database and produce reports summarizing selected contents.

Microsoft Access provides users with one of the simplest and most flexible DBMS solutions on the market today. Regular users of Microsoft products will enjoy the familiar Windows "look and feel" as well as the tight integration with other Microsoft Office family products. An abundance of wizards lessen the complexity of administrative tasks and the ever-present Microsoft Office Helper (you know... the paper clip!) is available for those who care to use it. Before purchasing Access, be sure that your system meets Microsoft's minimum system requirements. To further our discussion, let's first examine three of the major components of Access that most database users will encounter – tables, queries, forms. Once we've completed that we'll look at the added benefits of reports, web integration and SQL Server integration.

:

Payment.DB			
Field Name	Type	Size	Key
Cuscode	Text	50	
Date	Date/Time	Short Date	
Total	Currency	Standart, Decimal: 2	

Figure2.3. Sample Table

The table above contains the employee information for our organization characteristics like name, date of birth and title. Examine the construction of the table and you'll find that each column of the table corresponds to a specific employee characteristic (or **attribute** in database terms). Each row corresponds to one particular employee and contains his or her information. That's all there is to it! If it helps, think of each one of these tables as a spreadsheet-style listing of information.

In the previous section, we learned how tables allow us to create the framework for storing information in a database. Obviously, a database that only stored information would be useless -- we need methods to retrieve information as well. If you simply want to recall the information stored in a table, Microsoft Access allows you to open the

table and scroll through the records contained within it. However, the real power of a database lies in its capabilities to answer more complex requests, or **queries**. Access queries provide the capability to combine data from multiple tables and place specific conditions on the data retrieved.

2.6.2. Creating Table

Many techniques allow you to create a database, the fastest of which consists of using one of the provided examples. Microsoft Access 97 shipped with 22 sample databases while Microsoft Access 2000 ships with 10. Furthermore, the 97 version allowed to provide sample data into the database. This is not available with the 2000 release. The databases that ship with Microsoft Access can help you in two main ways: they provide a fast means of creating a database and you can learn from their structure.

To create a database using one of the samples, there is a little detail to follow depending on whether you had launched the program already or not. If Microsoft Access is not running, you can start it. When the first dialog box comes up, you can click the second radio button: Access Database Wizard, Pages

The New dialog box displays two property pages labeled General and Databases. If you want to create a database based on one of the samples, you can click the Databases property page. A list of the sample databases appears. You can then choose one and click OK.

When creating a database using one of the samples, depending on the sample you selected, the Database Wizard will display a few objects and suggest some fields for your database. Some fields are already associated with the objects and some other fields can be added. You can examine them, then add some fields you think are important for your database. You will also have the option of selecting a design layout. Some of the sample databases have been configured to require information about the company you are creating the database for.

2.7. Introducing Database

Databases are designed to offer an organized mechanism for storing, managing and retrieving information. They do so through the use of tables. If you're familiar with spreadsheets like Microsoft Excel, you're probably already accustomed to storing data in tabular form. It's not much of a stretch to make the leap from spreadsheets to databases.

Just like Excel tables, database tables consist of columns and rows. Each column contains a different type of attribute and each row corresponds to a single record. For example, imagine that we were building a database table that contained names and telephone numbers. We'd probably set up columns named "FirstName", "LastName" and "TelephoneNumber." Then we'd simply start adding rows underneath those columns that contained the data we're planning to store.

At this point, you're probably asking yourself an obvious question – if a database is so much like a spreadsheet, why can't I just use a spreadsheet? Databases are actually much more powerful than spreadsheets in the way you're able to manipulate data. Here

are just a few of the actions that you can perform on a database that would be difficult if not impossible to perform on a spreadsheet:

- Retrieve all records that match certain criteria
- Update records in bulk
- Cross-reference records in different tables
- Perform complex aggregate calculations

As we walk through this tutorial, you'll learn how you can use databases to achieve each of these objectives. Page 2 of this lesson provides you with an overview of how database keys can be used to uniquely identify records and form relationships between tables. Page 3 describes how the Structured Query Language allows you to interact with your database. On page 4, we examine the different types of databases available on the market today.

2.8. Database Keys

On the previous page of this article, you learned how databases use tables to organize data. As you probably recall, each table consists of a number of rows, each of which corresponds to a single database record. So, how do databases keep all of these records straight? It's through the use of **keys**.

The first type of key we'll discuss is the **primary key**. Every database table should have one or more columns designated as the primary key. The value this key holds should be unique for each record in the database. For example, assume we have a table called Employees that contains personnel information for every employee in our firm. We'd need to select an appropriate primary key that would uniquely identify each employee. Your first thought might be to use the employee's name.

This wouldn't work out very well because it's conceivable that you'd hire two employees with the same name. A better choice might be to use a unique employee ID number that you assign to each employee when they're hired. Some organizations choose to use Social Security Numbers (or similar government identifiers) for this task because each employee already has one and they're guaranteed to be unique. However, the use of Social Security Numbers for this purpose is highly controversial due to privacy concerns.

Once you decide upon a primary key and inform the database of this decision, it will enforce the uniqueness of the key. If you try to insert a record into a table with a primary key that duplicates an existing record, the insert will fail.

Most databases are also capable of generating their own primary keys. Microsoft Access, for example, may be configured to use the AutoNumber data type to assign a unique ID to each record in the table. While effective, this is a bad design practice because it leaves you with a meaningless value in each record in the table. Why not use that space to store something useful?

The other type of key that we'll discuss in this course is the **foreign key**. These keys are used to create relationships between tables. Natural relationships exist between tables in most database structures. Returning to our employees database, let's imagine that we

wanted to add a table containing departmental information to the database. This new table might be called Departments and would contain a large amount of information about the department as a whole. We'd also want to include information about the employees in the department, but it would be redundant to have the same information in two tables (Employees and Departments). Instead, we can create a **relationship** between the two tables.

Let's assume that the Departments table uses the Department Name column as the primary key. To create a relationship between the two tables, we add a new column to the Employees table called Department. We then fill in the name of the department to which each employee belongs. We also inform the database that the Department column in the Employees table is a **foreign key** that references the Departments table. The database will then enforce *referential integrity* by ensuring that all of the values in the Departments column of the Employees table have corresponding entries in the Departments table.

Note that there is no uniqueness constraint for a foreign key. We may (and most likely do!) have more than one employee belonging to a single department. Similarly, there's no requirement that an entry in the Departments table have *any* corresponding entry in the Employees table. It is possible that we'd have a department with no employees.

2.9. Working with SQL

SQL (pronounced "ess-que-el") stands for structured Query Language. SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems. SQL statements are used to perform tasks such as update data on a database, or retrieve data from a database. Some common relational database management systems that use SQL are: Oracle, Sybase, Microsoft SQL Server, Access, Ingress, etc. although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as "select", "insert", "delete", "create" and "drop" can be used to accomplish almost everything that one needs to do with a database.

2.9.1. Data Manipulation Language

The Data Manipulation Language (DML) is used to retrieve, insert and modify database information. These commands will be used by all database users during the routine operation of the database. Let's take a brief look at the basic DML commands:

The Data Manipulation Language (DML) is used to retrieve, insert and modify database information. These commands will be used by all database users during the routine operation of the database. Let's take a brief look at the basic DML commands:

2.9.1.1. INSERT

The INSERT command in SQL is used to add records to an existing table. Returning to the personal_info example from the previous section, let's imagine that our HR department needs to add a new employee to their database. They could use a command

similar to the one shown below:

```
INSERT INTO department  
values('c01','computer','engineering',5,6)
```

Note that there are four values specified for the record.

These correspond to the table attributes in the order they were defined: first_name, last_name, employee_id, and salary.

2.8.1.2.SELECT

The SELECT command is the most commonly used command in SQL. It allows database users to retrieve the specific information they desire from an operational database. Let's take a look at a few examples, again using the personal_info table from our employees database.

The command shown below retrieves all of the information contained within the personal_info table. Note that the asterisk is used as a wildcard in SQL. This literally means "Select everything from the personal_info table."

```
SELECT *  
FROM student
```

Alternatively, users may want to limit the attributes that are retrieved from the database. For example, the Human Resources department may require a list of the last names of all employees in the company. The following SQL command would retrieve only that information:

```
SELECT sname  
FROM student WHERE stno=20020872
```

CHAPTER THREE : STOCK CONTROL & CUSTOMER TRACKING PROGRAM USING VISUAL BASIC LANGUAGE

3.1. Entrance Page

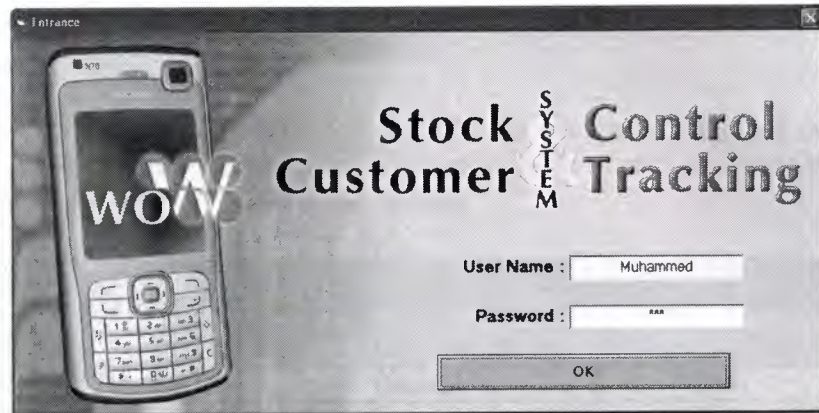


Figure 3.1 Entrance page

When we run our program main Entrance window appears and asks for user name and password, if we enter one of them wrong it gives a message that username or password is wrong.

After we write the correct username and password we pass to the second page which is the main page of our program.

3.2. Main page

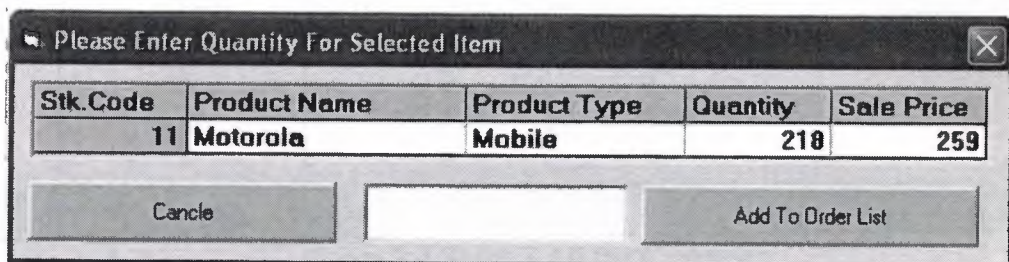


Figure 3.2 Main Menu

This page is our sale page, in order to make a sale we have to select stocks from stock list. If we know the stock code we write it directly then pressing Find Stock By Code, it gives us the stock directly, we can also categorize the stocks by type, that's by pressing Show Stock By Type, the program will classify the stocks by type, for instance: mobile, cover, battery, headphones... etc.

Also we can see all of the stocks by pressing Show All Stocks.

After we see the desired stock we double click on it and we see the above window; if we click on an empty row, the program gives us a message that you can't select an empty row and you must select a stock.



Stk.Code	Product Name	Product Type	Quantity	Sale Price
11	Motorola	Mobile	210	259

Cancel Add To Order List

Figure 3.3 Adding Quantity to Make a Sale

So we write the quantity and we press Add to Order List.

If we write a quantity which is greater than the quantity that is in stock, the program gives us another message that we don't have this quantity. If we don't write any quantity and directly press Add to Order List, the cursor focuses on text to write a quantity, also we can't write letters or other characters in the text.

After we write a quantity which is less or equal to the stock quantity and we press Add to Order List, we see that the desired item added to the order list. We do the same operations for other stocks that we want to sell.

After we select stocks we have to select a customer too.

If we know code of the customer we write code then choosing by code then find, the program will gives us the customer name, surname, company and picture of him/her.

If we don't know the code, just we know the name of the customer, we write the name of him or just the first letter of it, the program will display a window showing all the customers starting with that name or letter.

Also we can select Show All customers to see all the customers and select one of them, in this case another page will open as follows;

Cus. Code	Name	Surname	Company Name	Tel Company	Mobile	Address	E-Mail
1	Muhammed	simko	wow café	05367831198	05338773781	Muhandisler Mah. Erbil-Iraq	muhammedwow@hotmail.com
10	Muhammed	Abdulgani	Furkan	00447028640906	+9647504450906	Adale-Erbil-Iraq	hama1997@yahoo.com
11	Muhammed	Serkawit	Sarkawit Const.	00447028648987	+9647504458687	Istkan-Erbil-Iraq	hamawow@yahoo.com
12	Abdullah	Mizakci	Atceken	05373915229	05338460857	Istanbul	
13	Adem	Atceken	Atceken	05363628394	05428738841	Konya	adem_atceken@hotmail.com
14	Zubeyir	Cokakli	Nurcu	08922278275	05428732077	Ortakoy-Kuzey Kibris	battal25@yahoo.com
15	Ahmed	selman	Ahmed Co.	05334893498	05333333333	Jordan	ahmedselam@hotmail.com
16	ali	chima	chima	05338300251	05338300251	Pakistan	
17	Ozer	Dastan	Esiner	05053496376	05338491105	Tortum-Erzurum	ozet_dastan_1987@hotmail.com
18	Rogay	Jemal	Peris Co.	05356817414	05338773771	Azadi Mah-Erbil-Iraq	roger_84@yahoo.com
19	Ali Said	Olgun	Olgunler sirketi	05354566225	05338493631	Istanbul	ali_said_olgun@yahoo.com
20	Hassan	Kargin	Sumerholding	05378658894	05338767180	Kulu-Konya	hbasri42@hotmail.com
21	Fazil	Kargin	Sumerholding	05354930025	05338471301	Kulu-Konya	fazil_kargin@hotmail.com
22	Fikret	meting	Lorenito		05338668948	Van	harun_fikret@hotmail.com
23	Cofer	Yesilkaya	JAF	05364847594	05338773519	Bukdan-Denizli	coferyesilkaya@hotmail.com
24	Ilker	Damar	Damar	05372500833	05428630353	Van-Markaz	
25	Rolvan	Gologlu	Gologlu Ekmek Fabrikasi	04322231375	05324262030	Turkey-Tokat	
26	Kenan	Soylu	Soylu sirketleri		05428658309		kenan_kingdom@hotmail.com

Figure 3.4 Show's All the Customer

We select the desired customer then we press on Add To Main Menu or just double click on it. We see that the desired customer has been added to the main menu.

Now the sale page is as below and everything is ready, we select items, we choose customers, now we can make a sale and order to print.

Find Customer

Code Name Find By

Show All Customers

Add New Customer

Cus. Code: 12

Name: Ozer

Surname: Dastan

Co. Name: Esiner

Order To Print

Show Last Order List

Clear Main Page

Remove

Exit

Find Stock By Code

Show Stock By Type

Show All Stocks

Stock List

Stock Code	Product Name	Product Type	Quantity	Sale Price
18	Black colored covers	cover	492	5
19	Nokia Ncage	Mobile	98	370
20	Nec E616V	Mobile	49	450
21	Panasonic GD90	Mobile	50	200
22	Nokia 6680	Screen	100	40
23	Sony Ericsson K750	Screen	58	70
24	Nokia 3310	Battery	520	4
25	Original Nokia 3310 cover	cover	95	25
26	Memory Card 128	memory	113	40
27	Memory Card 256	memory	210	45
28	Memory Card 512	memory	200	55

Order List

Stock Code	Product Name	Product Type	Quantity	Sale Price	Total
12	Samsung	Mobile	2	163	338
15	Nokia 9110	cover	3	2	6
18	Sony Ericsson K750	Mobile	4	390	1520
3	Nokia 7610	Mobile	5	330	1650
23	Sony Ericsson K750	Screen	12	70	840
26	Memory Card 128	memory	13	28	364

Total

4718

Payment Entrance

Dollar

Dinar

100, \$

Dollar's Rate

OK

Your Loan Of This Sale

4618

wov

Figure 3.5 Main Menu When everything is Ready For Sale

We see that the total is written in the right above part of the form.

If the customer pays we enter the payment in Payment Entrance Part, there is two choices, the payment will be either Dollar or Dinar, if it is Dollar we don't have any problem because the system currency is based on Dollar, if the payment is Dinar we choose Dinar and we write Amount that will be pay and we have to write Dollars rate with Dinar.

When we press ok we see that the remaining amount of money is written in bellow part of ok, if we pay the whole money we see that nothing is written there.

In The left Part of the Form we see that there Are 5 buttons;

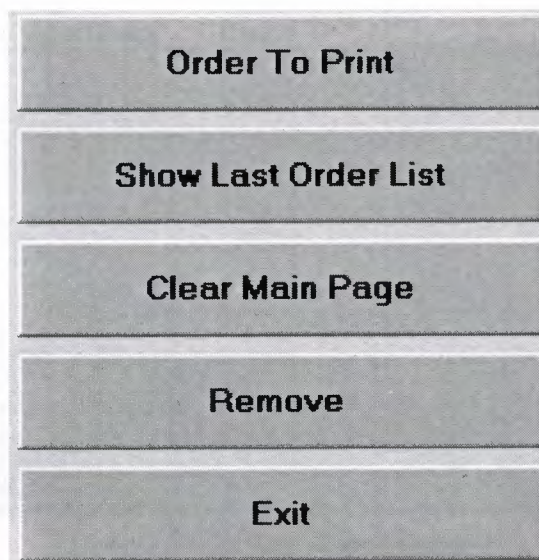


Figure 3.6 Buttons of the Main Menu

Order To print button: used to print and save all the changes after we select desired items to be sell and a customer.

Show Last Order List: When we press this button another window opens and shows us the last sale operation, that's the last customer came and make sale, the content of sale, amount of money, etc...

WOW Stock Control System Customer Tracking

Cus. Code: 14 Name: saner

Co. Name: Muhiddinler Surname: Fahrettin

Last Order List

Stock Code	Product Name	Product Type	Quantity	Sale Price	Total
12	Samsung	Mobile	2	169	338
19	Nokia Ncage	Mobile	3	370	1110
29	Nokia Original Charger	Charger	2	15	30

Refund All Print Total : 1478

OK

Figure 3.7 The Last Sale Operation

If we want to refund the last sale we can do through that form. So by pressing Refund All, the refunded items will be added to the stock and the amount of money will be decreasing from selected customers account.

If we press Print button, the last Sale operation will be printed again. By pressing Ok we can return to the main menu.

Clear Main Menu: By clicking on this button everything will be cleared in the main menu, it will be just like it is opened new, that's customer part will be removed, and selected stock part(ORDER LIST) will be removed.

Remove: we can use this button after we select items from the stock to order list, and we change our mind, to remove something from order list. So we select the desired item to be deleted and we press on Remove, we see that the desired item removed.

If we want to increase or decrease items quantity after we selected from stock to order list, we double click on it a window will appear;

Stk.Code	Product Name	Product Type	Quantity	Sale Price
11	Motorola	Mobile	218	259

Figure 3.8 Increase Or Decrease Quantity

we can write any quantity and add again to order list.

Exit : By pressing on this button we close the program and save all the changes made until that moment.

3.3. Menu Bar



Figure 3.9 Program Menu In Main Page

In Menu Bar we have Program, Company, Customer, Stock and Help

3.3.1. Program Menu

When we press on Program Menu User Change and Exit occurs.

- ❖ If we pressing on User Change, User Change window appears and we can change user.
- ❖ If we press Exit, we log out from the program.

3.3.2. Company Menu

If we press Company the below list will appear

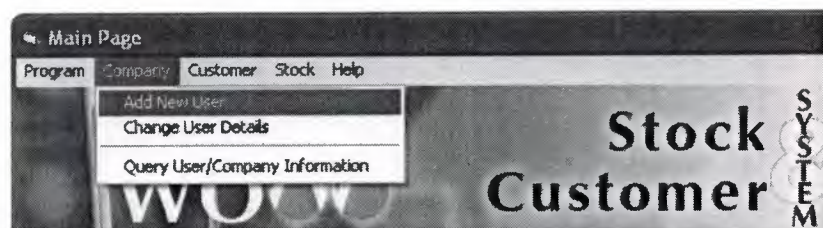


Figure 3.10 Company Menu In Main Page

- ❖ If we press on Add New User another page opens, through that page we can add new user.

3.3.2.1. Adding New User

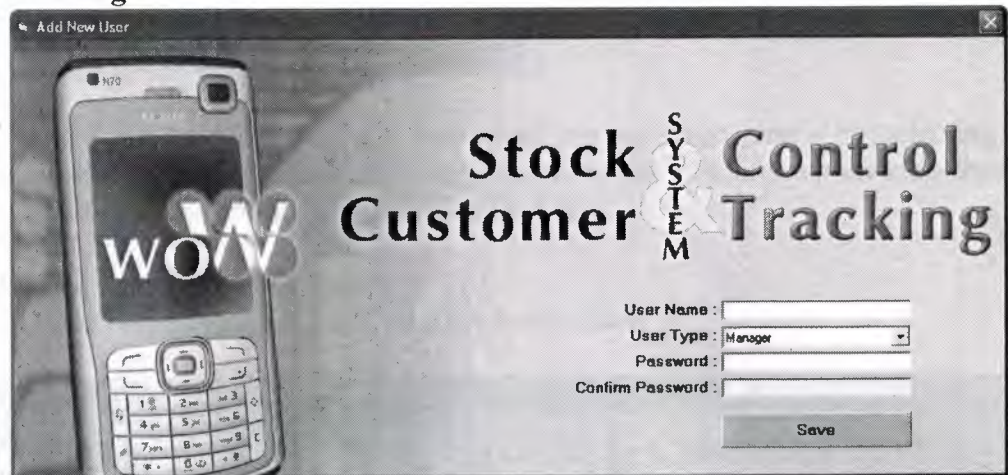


Figure 3.11 Adding New User To Program

In User Name Field we write the desired user name, In User Type field we can choose either Manager or Personal, in Password field we give a password to the selected user, in confirm password field we write the same password to make that the previous password matches the confirm one, if we write another password or a wrong one, the program gives us a message to check the password. Finally when we press Save Button, new user will be added to the program.

3.3.2.2. Changing User Details

- ❖ If we press on Change User Details, another window will appear;

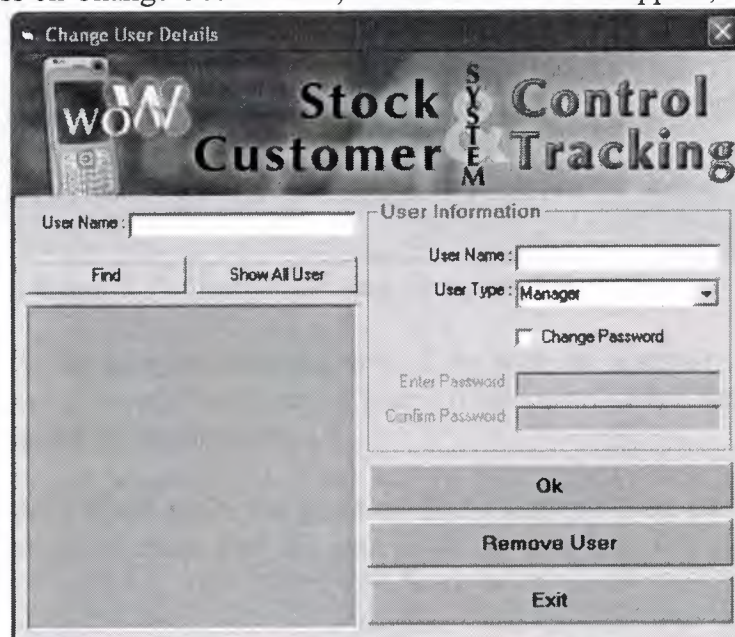


Figure 3.12 Change User Details

If we know user name, we write user name in User Name field and we press on find, if we don't have that user name, the program gives us a message that we don't have this user name, if we write just the first letter of the user, the program lists all the user starting with that letter, if we press on Show All User, the program lists all the user.

So by pressing on the desired user we can change his/her category from personal to manager or vice verse, also we can change his/her password by ticking on Change password.

Also we can Remove any user from the program, that's by selecting the desired user from the list and clicking on Remove User, Finally we save the changes by pressing on Ok, and Exit by pressing On EXIT.

3.3.2.3. User And Company Query

- ❖ If we press on Query User/Company Information's

User Query

Find User By Name

Search:

Show All User

User Names

- ahmed
- evup
- Muhammed**
- ozar

User Query

User Name:

User Type:

Show Today's Sale For This User

01.05.2006

01.06.2006

Sale's Between These Dates

Company Query

Show Today's Sales

04.05.2006

04.05.2006

Sales Between These dates

Show Total Price of all Stocks

Sale Date

Sale Code	Cus. Code	Name - Surname	Date	Total Price
1	1	Muhammed simko	21.05.2006	1443
10	15	Zubeyir Cokakli	22.05.2006	878
11	15	Zubeyir Cokakli	22.05.2006	2436
12	4	Ali Said Olgun	22.05.2006	23
13	6	Fazil Kargin	23.05.2006	1173
14	18	Ilker Damar	24.05.2006	31,4
15	3	Roger Jermal	24.05.2006	57,2
16	1	Muhammed simko	24.05.2006	200
17	1	Muhammed simko	24.05.2006	518
18	14	saner Fahrettin	24.05.2006	5,8

Total Sales : 11908

Sale Detail

Stock Code	Stock Name	quantity	price	Total
11	Motorola	2	259	518
12	Samsung	1	163	163
3	Nokia 7610	2	330	660
5	Nokia 6630	2	440	880
17	Rectangular Covers	22	4,5	99
13	Nokia 8110 Silver	21	3,2	67,2
15	Nokia 8110	21	2	42
17	Rectangular Covers	2	4,5	9

Total : 2444.20

Figure 3.13 Query User And Company Information

This page is for querying information's of the user and company, first of all we have to select a user either by Find User By Name if we know his/her name or by Show All Users, so we select the desired user.

In User Query Field when we press on Show Today's Sale Of This User, the program shows us all the sales that the user make on that date, in Sale Date field it shows Sale Code, Customer Code, Name and Surname of Customer, Date of Sale and Total price.

If we click on one of them, in Sale Detail field we see Stock code, Stock Name, quantity, price and Total price of the selected sale that we click in Sale Date field. In the same way when we press on Sales For This User Between These Dates, the program shows the sales between the selected dates.

In Company Query Field, when we press on Show Today's Sale, the program show's That days sale of All The users, when we press on Sales Between These Dates, the program show's that day's sale between selected dates.

When we press on Show Total Prices of All stocks, the program shows Total Pure price of all stocks of the company and Sale price of all stocks. So we know what is our capital any time and what will we gain if we sell all of our products.

3.3.4. Customer Menu

In Customer Menu we have;

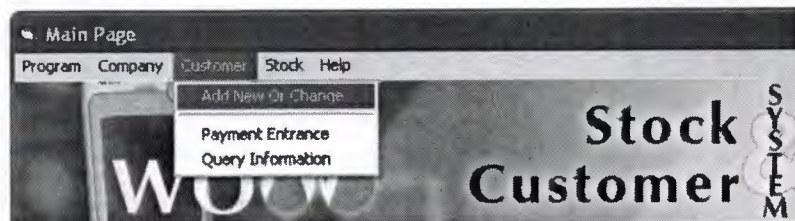


Figure 3.14 Customer Menu In Main Menu

- ❖ Add New Or Change
- ❖ Payment Entrance
- ❖ Query Information

3.3.4.1. Adding New User

By pressing on Add New Or change the below window opens;

Figure 3.15 Add Customer Page

This window is for adding new customer, by pressing on Add new Customer, the program gives Customer Code automatically, we have to fill the marked blanks at least, otherwise the program gives as a message to fill them, also we can add pictures for the new customer, so by clicking on save we save the operation of adding new customer or cancel to exit from the program without save

We can change any of the customers detail by pressing on Change Customer Detail.

3.3.4.2. Payment Entrance Page

By Pressing on Payment Entrance the below window opens;

The screenshot shows a software window titled "Payment Entrance". The window's header includes a "WOW" logo and the text "Stock Control SYSTEM Customer Tracking". The interface is split into two main panels. The left panel, labeled "Find Customer", offers two search methods: "Code" (selected) and "Name", with a "Find By" button and a "Show All Customers" button. Below these are input fields for customer details: "Cus. Code" (11), "Name" (Muhammad), "Surname" (Serkawit), and "Co. Name" (Serkawit Const.). The right panel, also labeled "Payment Entrance", allows selecting the currency: "Dollar" (selected) or "Dinar". It features a text input field containing "200 \$" and another field labeled "Dollar". At the bottom right of the window are "OK" and "Cancel" buttons.

Figure 3.16 Payment Entrance Page

This window provides the Entrance of Payment by hand, so firstly we have to select the customer either by code, or by name or by showing All the customers and selecting the desired one, later we have to decide whether the payment is Dollar or Dinar, if it is Dollar it is ok we just write the amount that will be paid in payment field, if it is Dinar we have to write Dollar's rate opposite to Dinar.

When we press on OK the program gives us a message to make us sure showing the amount of money that we entered, if we agree for the second time, the program add the payment to the selected customer payment list and return to Main Page, if we press on Cancel the program exit's without changing.

3.3.4.3. Customer Query Information Page

By pressing on Query Information the below window opens;

Find Customer

☒ Code ☐ Name
Find By

Show All Customers

Cust. Code: 1

Name: Muhammed

Surname: smko

Co. Name: wow café

Data

Sale Code	User Name	Date	Total
1	Muhammed	21.05.2006	1444.2
2	Muhammed	21.05.2006	30.5
3	Muhammed	21.05.2006	114.7
15	Muhammed	24.05.2006	200
17	Muhammed	24.05.2006	518

Payment Details

User Name	Date	Amount
ozar	21.05.2006	22
ozar	21.05.2006	23
ozar	21.05.2006	53.58
ozar	21.05.2006	343.2
ozar	23.05.2006	100.0039
ozar	01.06.2006	10
eyup	01.06.2006	20

Total Loan: 571.79

Sale Details

Payment Amount: 1000

Sale Loan: 1444.2

Total Loan: 1576.79

Details Of Sale

Stock Code	Stock Name	Price	Quantity	Total
11	Motorola	259	2	518
12	Samsung	169	1	169
3	Nokia 7610	330	2	660
5	Nokia 6530	440	2	880
17	Rectangular Covers	4.5	22	99
13	Nokia 6110 Silver	3.2	21	67.2
15	Nokia 6110	2	21	42
17	Rectangular Covers	4.5	2	9

Figure 3.17 Customer Query Information

In this page we can see all the details of any sale that has been done.

Firstly we select a customer from find customer field and we see the sale code, User name that he made sale from, Date and Total amount of that sale in Date field, when we press any of the sale we can see stock code, stock name, price, quantity and total price of that quantity in Details of Sale field, also the payments that he/she made to whom user in which date is showing in Payment Details.

When we click on any sale done in Date field the program shows us if the customer paid for that sale or not and that is written in Payment Amount and Sale loan fields for that sale only.

The total Loan of the selected user is shown in Left bottom part of the window, if it is zero there will be a smile face, if there is a loan there will be a sad face.

3.3.5. Stock Menu

In Stock Menu we have;

- ❖ Add Stock/Change Details
- ❖ Refund Items

Figure 3.18 Stock Menu in Main Page

3.3.5.1. Add Stock And Change Stock Detail Page

By Pressing on Add Stock/Change Details

Stock Code	Product Name	Product Type	Quantity	Sale Price
21	Panasonic GD90	Mobile	50	200
22	Nokia 6680	Screen	100	40
23	Sony Ericsson K750	Screen	58	70
24	Nokia 3310	Battery	520	4
25	Original Nokia 3310 cover	cover	95	25
26	Memory Card 128	memory	200	28
27	Memory Card 256	memory	210	45
28	Memory Card 512	memory	200	55
29	Nokia Original Charger	Charger	198	15
3	Nokia 7610	Mobile	92	330
30	Nokia Green Charger	Charger	202	7
31	Sony Ericsson Charger	Charger	202	10
32	Samsung Charger	Charger	302	8
33	Nokia Original Headphones	Headphone	200	6
34	Nokia single headphones	Headphone	300	4
35	Sony Ericsson Original Headp	Headphone	300	7
36	Sonic Bluetooth	Bluetooth	110	20
37	Memory Sony Ericsson 128	memory	100	30
38	Memory Reader	memory	50	30
39	Connection Cable	Cables	50	10
4	Nokia 6600	Mobile	122	300
40	Nokia Interior cable	Cables	100	6
41	Samsung Interior cable	Cables	200	7
5	Nokia 6630	Mobile	118	440
6	Nokia 8800	Mobile	49	700
7	Nokia N70	Mobile	129	520

Figure 3.19 Add Stock Page

Through this window we can add quantities of the existence stock, in the same time we can add new items to the stock.

If we want to add quantity of the existence item, we find it first then just by clicking on it we see that it's properties will be written in the above right part of the window, by pressing on Change Details we can Add quantity to the selected item through Adding Amount field, if we don't press on Change details we can't add anything or change any detail of the stock, finally we press on Save The Changes to save the changes done on selected items property.

When we add an item to our stock for the first time, we Press on Add new Stock to give an automatic Code to the new item that will be adding to the stock and filling other blanks later pressing on Add button.

3.3.5.2. Refund Item Page

By pressing on Refund items

Refund Items

Stock Control Tracking

Find Stock By Code: Show All Stocks

Show Stock By Type:

Stock Cod	Product Name	Product Type	Quantity	Price
26	Memory Card 128	memory	200	28
27	Memory Card 256	memory	210	45
28	Memory Card 512	memory	200	55
29	Nokia Original Charger	Charger	198	15
3	Nokia 7610	Mobile	92	330
30	Nokia Green Charger	Charger	202	7
31	Sony Ericsson Charger	Charger	202	10
32	Samsung Charger	Charger	302	8
33	Nokia Original Headphones	Headphone	200	6
34	Nokia single headphones	Headphone	300	4
35	Sony Ericsson Original Head	Headphone	300	7
36	Sonic Bluetooth	Bluetooth	110	20

Refund Items

Stock Cod	Product Name	Product Type	Ref.Amou	Price
12	Samsung	Mobile	2	169
14	Nokia 8110 Cristal	cover	4	2,9
18	Black colored covers	cover	5	5
27	Memory Card 256	memory	3	45
29	Nokia Original Charger	Charger	4	15
33	Nokia Original Headphones	Headphone	4	6
35	Sony Ericsson Original Head	Headphone	3	7

Refund From Stock

Refund From A Customer Account

Remove

OK

Cancel

Figure 3.20 Refund Item Page

We see that Refund From Stock option is in press situation, if we leave it as it is and we find the refunded stock either by type or by code or looking to the whole stock so we double click on it we find another window opens to write the quantity of the refunded item, later the refunded items will be added to the refunded items list. If we press Ok the Program adds the refunded items to the stock and gives us a message that the total price of all refunded items is that much.

But if we press on Refund From A Customer Account we see that the window changes like below;

Stock Cod	Product Name	Product Type	Quantity	Price
26	Memory Card 128	memory	200	28
27	Memory Card 256	memory	210	45
28	Memory Card 512	memory	200	55
29	Nokia Original Charger	Charger	198	15
3	Nokia 7610	Mobile	92	330
30	Nokia Green Charger	Charger	202	7
31	Sony Ericsson Charger	Charger	202	10
32	Samsung Charger	Charger	302	8
33	Nokia Original Headphones	Headphone	200	6
34	Nokia single headphones	Headphone	300	4
35	Sony Ericsson Original Head	Headphone	300	7
36	Sennheiser Bluetooth	Bluetooth	110	20

Stock Cod	Product Name	Product Type	Ref.Amount	Price
12	Samsung	Mobile	2	169
14	Nokia 8110 Cristal	cover	4	2.9
18	Black colored covers	cover	5	5
27	Memory Card 256	memory	3	45
29	Nokia Original Charger	Charger	4	15
33	Nokia Original Headphones	Headphone	4	6
35	Sony Ericsson Original Head	Headphone	3	7

Figure 3.21 Refund Item To Customer Account

By Pressing On Refund From A Customer Account, the refunded items total price will be added to the selected customers account as a payment.

We have Remove Button in the right bottom part of the form, that's to remove the selected Refund items from Refund Items List when we change our decision to refund that item or not, also we can change the quantity of refunded items after we decide and add items to the Refund Items List, that's by double clicking on the selected item and change the quantity that will be Refunded.

By pressing on Ok the program save the changes that made.

By pressing on Cancel the program exits from Refund Item page and returns to the main menu.

Finally we have Help Menu

It contains information About the Programmer.

CONCLUSION

Through building of this project I learned many things, because until now I was just learning programs literary or I was doing small programs, but by the way of this program I investigated a lot in internet, I read many books, I tried many examples in order o improve my knowledge about Visual basic to make my program a useful thing.

There are advantages and disadvantages of Visual basic, the advantage of it is easy in use and gives message when a wrong code is written and some other advantages, the disadvantage of Visual basic is slow and don't have every characteristics, especially we can not change colors of writing on buttons.

The Program can be updated in the future, we can add other properties to it, and we can use it through internet by adding extra codes, while doing the project wonderful imaginations come to my mind but if I applied them I couldn't be able to finish this project in time, but I'll do my imaginations project in the future if God Wishes.

My Project program is a tracking system program, I used Stock Control & Customer Tracking Program, because we are already a wholesaler of Mobile and it's accessories in North of Iraq, we'll get benefit from this program too.

REFERENCES

1. H.M.Deitel, P.J.Deitel and T.R.Nieto, Visual Basic 6: How To Program, Prentice Hall,Inc. Upper Saddle River., New Jersey, 1999
2. Ihsan Karagülle and Zeydin Pala, Microsoft Visual Basic 6.0 Pro, Türkmen Printing House, Istanbul ,2001.
3. A research for finding Visual Basic code, Finded November 01, 2005 from the World Wide Web "<http://www.vbtürk.com/allcodes/capture.htm>"
4. A guide research for writing program. Retrieved October 10, 2005 from the World Wide Web "<http://www.programlama.com>
5. A guide for writin about Visual Basic description, Retrieved December 05, 2005 from the World Wide Web <http://www.vbtutor.net/lesson.html>."

APPENDICES

Form1

```
Public db As Database
Public content As Recordset
Public customer As Recordset
Public loan As Recordset
Public sale As Recordset
Public stock As Recordset
Public payment As Recordset
Public user As Recordset
Public programyer As String
Public username As String
Public userpermission As Boolean

Public Sub open_database()
Set db = OpenDatabase(programyer + "\wow.mdb")
Set loan = db.OpenRecordset("loan")
Set sale = db.OpenRecordset("sale")
Set content = db.OpenRecordset("content")
Set stock = db.OpenRecordset("stock")
Set customer = db.OpenRecordset("customer")
Set payment = db.OpenRecordset("payment")
Set user = db.OpenRecordset("user")
End Sub

Public Sub close_database()
loan.Close
sale.Close
content.Close
customer.Close
payment.Close
stock.Close
user.Close
db.Close
End Sub

Private Sub Command1_Click()
Form1.user.Index = "primarykey"
Form1.user.Seek "=", Text1.Text
If Form1.user.NoMatch = 0 Then
If Form1.user.Fields("password") = Text2.Text Then
username = Text1
If Form1.user.Fields("usertype") = "Manager" Then
userpermission = True
Else
userpermission = False
End If
Form2.Show
Form1.Hide
Else
MsgBox ("password is wrong!")
```

```

End If
Else
MsgBox ("Username or password is not correct")
Text1.Text = ""
Text2.Text = ""
End If
End Sub

```

```

Private Sub Form_Load()
programyer = CurDir
Text1.Text = ""
Text2.Text = ""
Call Form1.open_database
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Call close_database
End
End Sub

```

Form2

```

Public Sub writetotal()
Dim total
total = 0
For i = 1 To MSFlexGrid2.Rows - 1
total = total + CDBl(MSFlexGrid2.TextMatrix(i, 5))
Next
Form2.Text6 = total
End Sub
Private Sub Check2_Click()
Text14.Text = ""
Dim al_sql As String
Dim sql1 As Recordset
Combo1.clear
If Check2.value = 1 Then
Combo1.Locked = False
Combo1.Enabled = True
al_sql = "select stktype from stock group by stktype"
Set sql1 = Form1.db.OpenRecordset(al_sql)
If sql1.RecordCount > 0 Then
sql1.MoveFirst
While Not sql1.EOF
Combo1.AddItem sql1.Fields("stktype")
sql1.MoveNext
Wend
End If
Combo1.ListIndex = 0
Else
Combo1.Locked = True
Combo1.Enabled = False
Combo1.clear
End If

```



```

End Sub
Private Sub Combo1_Click()
Dim d As Long
Call clear1
If Combo1.ListCount > 0 Then
d = 1
Form1.stock.MoveFirst
While Not Form1.stock.EOF
If Form1.stock.Fields("stktype") = Combo1.Text Then
MSFlexGrid1.AddItem ""
MSFlexGrid1.TextMatrix(d, 0) = (Form1.stock.Fields("stockcode"))
MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
d = d + 1
End If
Form1.stock.MoveNext
Wend
End If
End Sub
Private Sub Command1_Click()
Form13.comingform = 2
On Error Resume Next
If Option1.value = True Then
If Text9.Text = "" Then
MsgBox ("please write a code")
Else
Form1.customer.Index = "primarykey"
Form1.customer.Seek "=", Text9.Text
Text1 = Form1.customer.Fields("cuscode")
Text2 = Form1.customer.Fields("name")
Text3 = Form1.customer.Fields("surname")
Text10 = Form1.customer.Fields("company")
End If
End If
If Option2.value = True Then
If Text9.Text = "" Then
MsgBox ("please write a name")
Else
Form1.customer.MoveFirst
While Not Form1.customer.EOF
If LCase(Text9.Text) = LCase(Left(Form1.customer.Fields("name"),
Len(Text9.Text))) Then
Form2.Enabled = False
Form13.Show
Form13.List1.AddItem Form1.customer.Fields("cuscode")
Form13.List2.AddItem Form1.customer.Fields("name")
Form13.List3.AddItem Form1.customer.Fields("surname")
Form13.List4.AddItem Form1.customer.Fields("company")

```

```

        If IsNull(Form1.customer.Fields("tel")) = False Then
            Form13.List5.AddItem (Form1.customer.Fields("tel"))
        Else
            Form13.List5.AddItem " "
        End If
        Form13.List6.AddItem Form1.customer.Fields("mobile")
        If IsNull(Form1.customer.Fields("address")) = False Then
            Form13.List7.AddItem Form1.customer.Fields("address")
        Else
            Form13.List7.AddItem " "
        End If
        If IsNull(Form1.customer.Fields("email")) = False Then
            Form13.List8.AddItem Form1.customer.Fields("email")
        Else
            Form13.List8.AddItem " "
        End If
    End If
    Form1.customer.MoveNext
Wend
End If
End If
Text9.Text = ""
End Sub

```



```

Private Sub Command10_Click()
    Unload Me
End Sub
Private Sub Command12_Click()
    Dim msj As Integer
    Dim result As Double
    If Text1.Text = "" Then
        MsgBox ("You have to Select a Customer")
    Else
        If Text6.Text = "" Then
            MsgBox ("You have to make a sale")
        Else
            If Text4.Text = "" Then
                MsgBox ("You Have To Enter Payment")
            Else
                If Option3.value = True Then
                    msj = MsgBox("You Entered " + Format(Text4.Text, "#,##0"), 36, "Are you sure")
                    If msj = 6 Then
                        Label3.Visible = True
                        Text7 = (Val(Text6 * 100) - Val(Left(Text4, Len(Text4) - 2) * 100)) / 100
                    End If
                End If
                If Option4.value = True Then
                    If Text5.Text <> "" And IsNumeric(Text5) = True Then
                        result = Val(Text4.Text) / Val(Text5.Text)
                        msj = MsgBox("You Entered " + Text4.Text + " Dinar", 36, "Are you sure")
                        If msj = 6 Then

```



```

    Label3.Visible = True
    Text7 = Format(Text6 - result, "#.##")
    End If
    If msj = 7 Then
        Exit Sub
    End If
Else
    Text5.SetFocus
    MsgBox ("Write Today's Rate of Dollar")
    End If
End If
End If
End Sub

```

```

Private Sub Command2_Click()
    Form7.Show
    Form7.Text1.Text = ""
    Form7.Text2.Text = ""
    Form7.Text3.Text = ""
    Form7.Text4.Text = ""
    Form7.Text5.Text = ""
    Form7.Text6.Text = ""
    Form7.Text7.Text = ""
    Form7.Text8.Text = ""
End Sub

```

```

Private Sub Command3_Click()
    If MSFlexGrid2.Rows > 2 Then
        MSFlexGrid2.RemoveItem (MSFlexGrid2.Row)
    Else
        Call clear2
    End If
    Call writetotal
End Sub

```

```

Private Sub Command4_Click()
    Dim d As Long
    Check2.value = 0
    Call clear1
    Form1.stock.Index = "primarykey"
    Form1.stock.Seek "=", Text14.Text
    If Text14.Text = "" Then
        MsgBox ("please Write a Stock Code")
        Call clear1
    Else
        If Form1.stock.NoMatch <> 0 Then
            MsgBox ("Database don't have this Stock Code")
            Text14.Text = ""
        Else
            d = 1
            MSFlexGrid1.AddItem ""

```

```

MSFlexGrid1.TextMatrix(d, 0) = Text14.Text
MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
d = d + 1
End If
End If
End Sub

```

```

Private Sub Command5_Click()
Dim satiskodu As String
Dim msj As Integer
If MSFlexGrid2.Rows = 1 Then
MsgBox ("Select item to make a sale..!")
Exit Sub
End If
msj = MsgBox("Are you sure you want to print", 36, "PRINT")
If msj = 6 Then
If Text1 <> "" Then
If MSFlexGrid2.Rows > 1 Then
If Form1.sale.RecordCount > 0 Then
Form1.sale.MoveFirst
While Not Form1.sale.EOF
If Val(Form1.sale.Fields("sellcode")) > Val(satiskodu) Then
satiskodu = Form1.sale.Fields("sellcode")
End If
Form1.sale.MoveNext
Wend
Else
satiskod = "1"
End If
total = 0
For i = 1 To MSFlexGrid2.Rows - 1
total = total + Val(MSFlexGrid2.TextMatrix(i, 5) * 100)
Next
Form1.sale.AddNew
Form1.sale.Fields("sellcode") = Val(satiskodu) + 1
Form1.sale.Fields("cuscode") = Text1
Form1.sale.Fields("date") = Date
Form1.sale.Fields("total") = total / 100
Form1.sale.Fields("username") = Form1.username
Form1.sale.Update
For i = 1 To MSFlexGrid2.Rows - 1
Form1.content.AddNew
Form1.content.Fields("sellcode") = Val(satiskodu) + 1
Form1.content.Fields("stockcode") = MSFlexGrid2.TextMatrix(i, 0)
Form1.content.Fields("total") = MSFlexGrid2.TextMatrix(i, 5)
Form1.content.Fields("quantity") = MSFlexGrid2.TextMatrix(i, 3)
Form1.content.Update
Form1.stock.Index = "primarykey"

```



```

Form1.stock.Seek "=", MSFlexGrid2.TextMatrix(i, 0)
If Form1.stock.NoMatch = False Then
    Form1.stock.Edit
    Form1.stock.Fields("amount") = Form1.stock.Fields("amount") -
MSFlexGrid2.TextMatrix(i, 3)
    Form1.stock.Update
End If
Next
'add to customer account
If Text7 = "" Then
    Form1.loan.AddNew
    Form1.loan.Fields("cuscode") = Text1
    Form1.loan.Fields("sellcode") = Val(satiskodu) + 1
    Form1.loan.Fields("total") = Text6
    Form1.loan.Update
Else
    If Text7 <> "0" Then
        Form1.loan.AddNew
        Form1.loan.Fields("cuscode") = Text1
        Form1.loan.Fields("sellcode") = Val(satiskodu) + 1
        Form1.loan.Fields("total") = Text7
        Form1.loan.Update
    End If
End If
'add to customer account
MsgBox ("Sale Operation has done successfully")
Call clear2
Call clear1
Call textclear
Else
    MsgBox ("Select Stock From list")
End If
Else
    MsgBox ("Please Select a Customer")
End If
End If
If msj = 7 Then
    Form2.Show
End If
End Sub
Private Sub Command6_Click()
Dim d As Long
Combo1.clear
Check2.value = 0
d = 1
Form1.stock.MoveFirst
While Not Form1.stock.EOF
    MSFlexGrid1.AddItem ""
    MSFlexGrid1.TextMatrix(d, 0) = (Form1.stock.Fields("stockcode"))
    MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))

```

```

MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
d = d + 1
Form1.stock.MoveNext
Wend
End Sub

```

```

Private Sub Command7_Click()
Form13.comingform = 2
Form13.List1.clear
Form13.List2.clear
Form13.List3.clear
Form13.List4.clear
Form13.List5.clear
Form13.List6.clear
Form13.List7.clear
Form13.List8.clear
Form1.customer.MoveFirst
While Not Form1.customer.EOF
Form13.List1.AddItem (Form1.customer.Fields("cuscode"))
Form13.List2.AddItem (Form1.customer.Fields("name"))
Form13.List3.AddItem (Form1.customer.Fields("surname"))
Form13.List4.AddItem (Form1.customer.Fields("company"))
If IsNull(Form1.customer.Fields("tel")) = False Then
Form13.List5.AddItem (Form1.customer.Fields("tel"))
Else
Form13.List5.AddItem " "
End If
Form13.List6.AddItem (Form1.customer.Fields("mobile"))
If IsNull(Form1.customer.Fields("address")) = False Then
Form13.List7.AddItem (Form1.customer.Fields("address"))
Else
Form13.List7.AddItem " "
End If
If IsNull(Form1.customer.Fields("email")) = False Then
Form13.List8.AddItem (Form1.customer.Fields("email"))
Else
Form13.List8.AddItem " "
End If
Form1.customer.MoveNext
Wend
Form2.Enabled = False
Form13.Show
End Sub

```

```

Public Sub clear2()
MSFlexGrid2.clear
MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Product Name"
MSFlexGrid2.TextMatrix(0, 2) = "Product Type"
MSFlexGrid2.TextMatrix(0, 3) = "Quantity"

```



```

MSFlexGrid2.TextMatrix(0, 4) = "Sale Price"
MSFlexGrid2.TextMatrix(0, 5) = "Total"
MSFlexGrid2.ColWidth(0) = 950
MSFlexGrid2.ColWidth(1) = 2040
MSFlexGrid2.ColWidth(2) = 1300
MSFlexGrid2.ColWidth(3) = 800
MSFlexGrid2.ColWidth(4) = 850
MSFlexGrid2.ColWidth(5) = 800
MSFlexGrid2.Rows = 1
End Sub

Public Sub clear1()
MSFlexGrid1.clear
MSFlexGrid1.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid1.TextMatrix(0, 1) = "Product Name"
MSFlexGrid1.TextMatrix(0, 2) = "Product Type"
MSFlexGrid1.TextMatrix(0, 3) = "Quantity"
MSFlexGrid1.TextMatrix(0, 4) = "Sale Price"
MSFlexGrid1.ColWidth(0) = 950
MSFlexGrid1.ColWidth(1) = 2290
MSFlexGrid1.ColWidth(2) = 1700
MSFlexGrid1.ColWidth(3) = 900
MSFlexGrid1.ColWidth(4) = 900
End Sub

```

```

Private Sub Command8_Click()
Call clear2
Dim maxsalecode As String
Dim r
Dim toplam
If Form1.sale.RecordCount > 0 Then
maxsalecode = 0
Form1.sale.MoveFirst
While Not Form1.sale.EOF
If Val(Form1.sale.Fields("sellcode")) > Val(maxsalecode) Then
maxsalecode = Form1.sale.Fields("sellcode")
End If
Form1.sale.MoveNext
Wend
Form1.content.MoveFirst
While Not Form1.content.EOF
If Form1.content.Fields("sellcode") = maxsalecode Then
Form1.stock.Index = "primarykey"
Form1.stock.Seek "=", Form1.content.Fields("stockcode")
With Form11.MSFlexGrid2
.AddItem ""
r = .Rows - 1
.TextMatrix(r, 0) = Form1.content.Fields("stockcode")
If Form1.stock.NoMatch = False Then
.TextMatrix(r, 1) = Form1.stock.Fields("stkname")
.TextMatrix(r, 2) = Form1.stock.Fields("stktype")
End If

```

```

        .TextMatrix(r, 3) = Form1.content.Fields("quantity")
        .TextMatrix(r, 4) = Form1.content.Fields("total") /
Form1.content.Fields("quantity")
        .TextMatrix(r, 5) = Form1.content.Fields("total")
    End With
    End If
Form1.content.MoveNext
Wend
Form1.sale.Index = "primarykey"
Form1.sale.Seek "=", maxsalecode
Form1.customer.Index = "primarykey"
Form1.customer.Seek "=", Form1.sale.Fields("cuscode")
    If Form1.customer.NoMatch = False Then
        Form11.Text1 = Form1.customer.Fields("cuscode")
        Form11.Text2 = Form1.customer.Fields("name")
        Form11.Text3 = Form1.customer.Fields("surname")
        Form11.Text10 = Form1.customer.Fields("company")
    End If
    toplam = 0
    With Form11.MSFlexGrid2
        For r = 1 To .Rows - 1
            toplam = toplam + .TextMatrix(r, 5)
        Next
    End With
    Form11.Text4 = toplam
End If
Form11.Show
Form2.Hide
End Sub

```

```

Private Sub Command9_Click()
    Call clear1
    Call clear2
    MSFlexGrid2.Rows = 1
    Text1.Text = ""
    Text2.Text = ""
    Text3.Text = ""
    Text9.Text = ""
    Text10.Text = ""
    Text14.Text = ""
    Text4.Text = ""
    Text5.Text = ""
    Text6.Text = ""
    Text7.Text = ""
End Sub

```

```

Private Sub MSFlexGrid1_Click()
    MSFlexGrid1.Col = 0
    MSFlexGrid1.ColSel = 4
End Sub

```

```

Private Sub Form_Load()
    Call clear1

```



```

Call clear2
MSFlexGrid1.Rows = 13
MSFlexGrid1.ColWidth(0) = 950
MSFlexGrid1.ColWidth(1) = 2290
MSFlexGrid1.ColWidth(2) = 1700
MSFlexGrid1.ColWidth(3) = 900
MSFlexGrid1.ColWidth(4) = 900
Call textclear
Text1.Locked = True
Text2.Locked = True
Text3.Locked = True
Text10.Locked = True
Combo1.Enabled = False
Option1.value = True
Text6.Locked = True
Text7.Locked = True
Combo1.clear
Check2.value = 0
Option3.value = True
Label3.Visible = False
End Sub
Private Sub textclear()
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text6.Text = ""
Text7.Text = ""
Text9.Text = ""
Text10.Text = ""
Text14.Text = ""
End Sub


---


Private Sub Form_Unload(Cancel As Integer)
Call Form1.close_database
End
End Sub


---


Private Sub List1_Click()
List2.ListIndex = List1.ListIndex
List2.TopIndex = List1.TopIndex
List3.ListIndex = List1.ListIndex
List3.TopIndex = List1.TopIndex
List4.ListIndex = List1.ListIndex
List4.TopIndex = List1.TopIndex
List9.ListIndex = List1.ListIndex
List9.TopIndex = List1.TopIndex
End Sub


---


Private Sub List1_DblClick()
Text4.Text = List1.Text
Text5.Text = List2.Text

```

```

Text6.Text = List3.Text
Text7.Text = List9.Text
End Sub

```

```

Private Sub List2_Click()
List1.ListIndex = List2.ListIndex
End Sub

```

```

Private Sub List2_DblClick()
Text4.Text = List1.Text
Text5.Text = List2.Text
Text6.Text = List3.Text
Text7.Text = List9.Text
End Sub

```

```

Private Sub List3_Click()
List1.ListIndex = List3.ListIndex
End Sub

```

```

Private Sub List3_DblClick()
Text4.Text = List1.Text
Text5.Text = List2.Text
Text6.Text = List3.Text
Text7.Text = List9.Text
End Sub

```

```

Private Sub List4_Click()
List1.ListIndex = List4.ListIndex
End Sub

```

```

Private Sub List4_DblClick()
Text4.Text = List1.Text
Text5.Text = List2.Text
Text6.Text = List3.Text
Text7.Text = List9.Text
End Sub

```

```

Private Sub List9_Click()
List1.ListIndex = List9.ListIndex
End Sub

```

```

Private Sub List9_DblClick()
Text4.Text = List1.Text
Text5.Text = List2.Text
Text6.Text = List3.Text
Text7.Text = List9.Text
End Sub

```

```

Private Sub mnadditem_Click()
If Form1.userpermission = True Then
Form10.Show
Form2.Hide
Else
MsgBox ("you cannot enter this menu")
End If
End Sub

```

```

Private Sub mnaddnewuser_Click()
If Form1.userpermission = True Then

```



```

Form3.Show
Form2.Hide
Else
MsgBox ("you cannot enter this menu")
End If
End Sub

```

```

Private Sub mnchangeprice_Click()
Form14.Show
Form2.Hide
End Sub

```

```

Private Sub mnchangeuserdetails_Click()
If Form1.userpermission = True Then
Form4.Show
Form2.Hide
Else
MsgBox ("you cannot enter this menu")
End If
End Sub

```

```

Private Sub mnexit_Click()
End
End Sub

```

```

Private Sub mnpaymnetentrance_Click()
Form8.Show
Form2.Hide
End Sub

```

```

Private Sub mnquerrycompanyinformation_Click()
Form2.Hide
Form6.Show
End Sub

```

```

Private Sub mnquerryinformation_Click()
Form9.Show
Form2.Hide
End Sub

```

```

Private Sub mnquerryuserinformation_Click()
Form2.Hide
Form5.Show
End Sub

```

```

Private Sub mnuserchange_Click()
Form1.Text1.Text = ""
Form1.Text2.Text = ""
Form1.Show
Form2.Hide
End Sub
Private Sub MSFlexGrid1_DblClick()
If MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 0) <> "" Then
Form6.MSFlexGrid1.TextMatrix(1, 0) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
0)
Form6.MSFlexGrid1.TextMatrix(1, 1) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
1)

```

```

Form6.MSFlexGrid1.TextMatrix(1, 2) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
2)
Form6.MSFlexGrid1.TextMatrix(1, 3) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
3)
Form6.MSFlexGrid1.TextMatrix(1, 4) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
4)
Form6.form6gelenform = 2
Form6.Show
Else
MsgBox ("You Can't Make a Blank Row Selection")
End If
End Sub

```

```

Private Sub MSFlexGrid2_Click()
On Error Resume Next
MSFlexGrid2.Col = 0
MSFlexGrid2.ColSel = 5
End Sub

```

```

Private Sub MSFlexGrid2_DblClick()
If MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 0) <> "" Then
Form6.MSFlexGrid1.TextMatrix(1, 0) = MSFlexGrid2.TextMatrix(MSFlexGrid2.Row,
0)
Form6.MSFlexGrid1.TextMatrix(1, 1) = MSFlexGrid2.TextMatrix(MSFlexGrid2.Row,
1)
Form6.MSFlexGrid1.TextMatrix(1, 2) = MSFlexGrid2.TextMatrix(MSFlexGrid2.Row,
2)
Form6.MSFlexGrid1.TextMatrix(1, 3) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
3)
Form6.MSFlexGrid1.TextMatrix(1, 4) = MSFlexGrid2.TextMatrix(MSFlexGrid2.Row,
4)
Form6.form6gelenform = 22
Form6.Caption = "Increase Or Decrease Quantity"
Form6.Show
Else
MsgBox ("You Can't Make a Blank Row Selection")
End If
End Sub

```

```

Private Sub Option3_Click()
If Option3.value = True Then
Text4 = "0,00 $"
End If
End Sub
Private Sub Option4_Click()
If Option4.value = True Then
Text4 = "0 Dinar"
End If
End Sub

```

```

Private Sub Text1_Change()
If Text1.Text <> "" Then
On Error Resume Next
Image1.Picture = LoadPicture(Form1.programyer + "\picture\" + Text1.Text + ".jpg")

```



```

End If
End Sub


---


Private Sub Text4_GotFocus()
If Option3.value = True Then
    If Len(Text4) > 2 Then
        Text4 = Format(Left(Text4, Len(Text4) - 2), "#.##")
    End If
Else
    If Len(Text4) > 6 Then
        Text4 = Format(Left(Text4, Len(Text4) - 6), "#")
    End If
End If
Text4.SelLength = 0
Text4.SelLength = Len(Text4)
End Sub


---


Private Sub Text4_LostFocus()
Dim control As String
Dim result As Boolean
Dim i As Integer
Dim one As String
result = False
If Option3.value = True Then
    If Text4 <> "" And Text4 <> "," And Text4 <> "0, $" Then
        If IsNumeric(Text4) = True Then
            Text4 = Format(Text4, "#,##0.## $")
        Else
            Text4 = "0,00"
            Text4.SetFocus
        End If
    Else
        Text4 = "0,00 $"
    End If
End If
End Sub


---


Private Sub Text5_GotFocus()
Text5.SelStart = 0
Text5.SelLength = Len(Text5)
End Sub


---



```

Form3

```

Private Sub Command1_Click()
If Text1.Text <> "" And Text2.Text <> "" And Text3.Text <> "" Then
If Text2.Text = Text3.Text Then
Form1.user.Index = "primarykey"
Form1.user.Seek "=", Text1.Text
If Form1.user.NoMatch = 0 Then
MsgBox ("Database has this user..!")
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""

```

```

Text1.SetFocus
Else
Form1.user.AddNew
Form1.user.Fields("username") = Text1.Text
Form1.user.Fields("usertype") = Combo1.Text
Form1.user.Fields("password") = Text2.Text
Form1.user.Update
MsgBox ("New Account has been saved..!")
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text1.SetFocus
Unload Me
Form2.Show
End If
Else
MsgBox ("Check the password")
Text2.Text = ""
Text3.Text = ""
Text2.SetFocus
End If
Else
MsgBox ("You must enter all informations..!")
End If
End Sub

```

```

Private Sub Form_Load()
Combo1.clear
Combo1.AddItem "Manager"
Combo1.AddItem "Personel"
Combo1.ListIndex = 0
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Combo1.ListIndex = 0
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Form2.Show
End Sub

```

Form4

```

Private Sub Command1_Click()
Form1.user.Index = "primarykey"
Form1.user.Seek "=", Text1.Text
If Text1.Text = "" Then
MsgBox ("please Write a user name")
Else

```

```

If Form1.user.NoMatch <> 0 Then
MsgBox ("Database don't have this user")
Text1.Text = ""
List1.clear
Else
List1.clear
List1.AddItem (Text1.Text)
End If
End If
End Sub

```

```

Private Sub Command2_Click()
If Text3.Text <> Text4.Text Then
    MsgBox ("please Check the password")
    Text3.Text = ""
    Text4.Text = ""
Else
    If Text2.Text <> "" Then
        Form1.user.Index = "primarykey"
        Form1.user.Seek "=", Text2.Text
        Form1.user.Edit
        Form1.user.Fields("usertype") = Combo1.Text
        If Check1.value = 1 Then
            Form1.user.Fields("password") = Text3.Text
        End If
        Form1.user.Update
        MsgBox ("Changes has been saved successfully")
        Unload Me
        Form2.Show
    End If
End If
End Sub

```

```

Private Sub Command3_Click()
Text1.Text = ""
List1.clear
List1.clear
Form1.user.MoveFirst
While Not Form1.user.EOF
List1.AddItem (Form1.user.Fields("username"))
Form1.user.MoveNext
Wend
End Sub

```

```

Private Sub Command5_Click()
If List1.ListIndex >= 0 Then
Form1.user.Index = "primarykey"
Form1.user.Seek "=", List1.Text
Form1.user.Delete
List1.RemoveItem (List1.ListIndex)
Text2 = ""
End If
End Sub

```

```
Private Sub Command4_Click()
```

```
Unload Me
```

```
Form2.Enabled = True
```

```
Form2.Show
```

```
End Sub
```

```
Private Sub Form_Load()
```

```
Text2.Locked = True
```

```
Check1.value = 0
```

```
Combo1.clear
```

```
Combo1.AddItem "Manager"
```

```
Combo1.AddItem "Personal"
```

```
Combo1.ListIndex = 0
```

```
List1.clear
```

```
Text1.Text = ""
```

```
Text2.Text = ""
```

```
Text3.Text = ""
```

```
Text4.Text = ""
```

```
End Sub
```

```
Private Sub Form_Unload(Cancel As Integer)
```

```
Form2.Show
```

```
End Sub
```

```
Private Sub Label4_Click()
```

```
wow
```

```
End Sub
```

```
Private Sub List1_Click()
```

```
Text2.Text = List1.Text
```

```
Form1.user.Index = "primarykey"
```

```
Form1.user.Seek "=", Text2.Text
```

```
Combo1.Text = Form1.user.Fields("usertype")
```

```
End Sub
```

```
Private Sub Text3_Change()
```

```
wow
```

```
End Sub
```

```
Private Sub Text4_Change()
```

```
wow
```

```
End Sub
```

```
Sub wow()
```

```
If Check1.value = 0 Then
```

```
Text3.Enabled = False
```

```
Text4.Enabled = False
```

```
Text3.ForeColor = &H80000010
```

```
Text4.ForeColor = &H80000010
```

```
Text3.BackColor = &H8000000B
```

```
Text4.BackColor = &H8000000B
```

```
Label4.ForeColor = &H80000010
```

```
Label5.ForeColor = &H80000010
```

```
Text3.Text = ""
```

```
Text4.Text = ""
```

```
Else
```

```

Text3.ForeColor = &H80000008
Text4.ForeColor = &H80000008
Text3.BackColor = &H80000005
Text4.BackColor = &H80000005
Label4.ForeColor = &H80000008
Label5.ForeColor = &H80000008
Text3.Enabled = True
Text4.Enabled = True
End If
End Sub

```

Form5

```

Private Sub Command1_Click()
List1.clear
If Text1.Text = "" Then
    MsgBox ("please write a name")
Else
    Form1.user.MoveFirst
    While Not Form1.user.EOF
        If LCase(Text1.Text) = LCase(Left(Form1.user.Fields("username"),
Len(Text1.Text))) Then
            Form5.List1.AddItem Form1.user.Fields("username")
            End If
        Form1.user.MoveNext
    Wend
    End If
Text1.Text = ""
End Sub

```

```

Private Sub Command2_Click()
Text1.Text = ""
List1.clear
List1.clear
Form1.user.MoveFirst
While Not Form1.user.EOF
List1.AddItem (Form1.user.Fields("username"))
Form1.user.MoveNext
Wend
End Sub

```

```

Private Sub Command3_Click()
Dim d As Long
Dim toplam As Double
Dim gridcontrol As Integer
Text4.Text = ""
Text5.Text = ""
gridkontrol = 1
MSFlexGrid2.clear
MSFlexGrid1.clear
MSFlexGrid1.Rows = 1
MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Stock Name"

```

```

MSFlexGrid2.TextMatrix(0, 2) = "quantity"
MSFlexGrid2.TextMatrix(0, 3) = "price"
MSFlexGrid2.TextMatrix(0, 4) = "Total"
MSFlexGrid1.TextMatrix(0, 0) = "Sale Code"
MSFlexGrid1.TextMatrix(0, 1) = "Cus. Code"
MSFlexGrid1.TextMatrix(0, 2) = "Name - Surname"
MSFlexGrid1.TextMatrix(0, 3) = "Date"
MSFlexGrid1.TextMatrix(0, 4) = "Total Price"
Form1.sale.MoveFirst
d = 1
toplam = 0
While Not Form1.sale.EOF
    If CDate(Form1.sale.Fields("date")) = Date And Text2 <> "" And Text2 =
Form1.sale.Fields("username") Then
        MSFlexGrid1.AddItem ""
        MSFlexGrid1.TextMatrix(d, 0) = Form1.sale.Fields("sellcode")
        MSFlexGrid1.TextMatrix(d, 1) = Form1.sale.Fields("cuscode")
        Form1.customer.Index = "primarykey"
        Form1.customer.Seek "=", Form1.sale.Fields("cuscode")
        If Form1.customer.NoMatch = 0 Then
            MSFlexGrid1.TextMatrix(d, 2) = Form1.customer.Fields("name") + " " +
Form1.customer.Fields("surname")
        End If
        MSFlexGrid1.TextMatrix(d, 3) = Form1.sale.Fields("date")
        MSFlexGrid1.TextMatrix(d, 4) = Form1.sale.Fields("total")
        toplam = toplam + CDBl(Form1.sale.Fields("total"))
        d = d + 1
    End If
    If gridkontrol > 12 Then
        MSFlexGrid1.AddItem ""
    Else
        gridkotrol = gridkontrol + 1
    End If
    Form1.sale.MoveNext
Wend
Text4.Text = Format(Trim(Str(toplam)), "#.##")
End Sub

```

```

Private Sub Command4_Click()
Dim d As Long
Dim toplam As Double
Dim gridkontrol As Integer
Text4.Text = ""
Text5.Text = ""
MSFlexGrid2.clear
MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Stock Name"
MSFlexGrid2.TextMatrix(0, 2) = "quantity"
MSFlexGrid2.TextMatrix(0, 3) = "price"
MSFlexGrid2.TextMatrix(0, 4) = "Total"
MSFlexGrid1.clear

```



```

MSFlexGrid1.TextMatrix(0, 0) = "Sale Code"
MSFlexGrid1.TextMatrix(0, 1) = "Cus. Code"
MSFlexGrid1.TextMatrix(0, 2) = "Name - Surname"
MSFlexGrid1.TextMatrix(0, 3) = "Date"
MSFlexGrid1.TextMatrix(0, 4) = "Total Price"
gridkontrol = 1
MSFlexGrid1.Rows = 12
If DTPicker1.value <= DTPicker2.value Then
Form1.sale.MoveFirst
d = 1
toplam = 0
While Not Form1.sale.EOF
    If CDate(Form1.sale.Fields("date")) >= DTPicker1.value And
CDate(Form1.sale.Fields("date")) <= DTPicker2.value And
Form1.sale.Fields("username") = Text2 And Text2 <> "" Then
        MSFlexGrid1.AddItem ""
        MSFlexGrid1.TextMatrix(d, 0) = Form1.sale.Fields("sellcode")
        MSFlexGrid1.TextMatrix(d, 1) = Form1.sale.Fields("cuscode")
        Form1.customer.Index = "primarykey"
        Form1.customer.Seek "=", Form1.sale.Fields("cuscode")
        If Form1.customer.NoMatch = 0 Then
            MSFlexGrid1.TextMatrix(d, 2) = Form1.customer.Fields("name") + " " +
Form1.customer.Fields("surname")
            End If
            MSFlexGrid1.TextMatrix(d, 3) = Form1.sale.Fields("date")
            MSFlexGrid1.TextMatrix(d, 4) = Form1.sale.Fields("total")
            toplam = toplam + Val(Form1.sale.Fields("total"))
            d = d + 1
            If gridkontrol > 12 Then
                MSFlexGrid1.AddItem ""
            Else
                gridkontrol = gridkontrol + 1
            End If
        End If
        Form1.sale.MoveNext
    Wend
Text4.Text = Trim(Str(toplam))
Else
MsgBox ("Enter Dates Correctly")
End If
End Sub

```

```

Private Sub Command5_Click()
Dim d As Long
Dim toplam As Double
Dim gridcontrol As Integer
Text4.Text = ""
Text5.Text = ""
gridkontrol = 1
MSFlexGrid1.Rows = 12
MSFlexGrid2.clear

```

```

MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Stock Name"
MSFlexGrid2.TextMatrix(0, 2) = "quantity"
MSFlexGrid2.TextMatrix(0, 3) = "price"
MSFlexGrid2.TextMatrix(0, 4) = "Total"
MSFlexGrid1.clear
MSFlexGrid1.TextMatrix(0, 0) = "Sale Code"
MSFlexGrid1.TextMatrix(0, 1) = "Cus. Code"
MSFlexGrid1.TextMatrix(0, 2) = "Name - Surname"
MSFlexGrid1.TextMatrix(0, 3) = "Date"
MSFlexGrid1.TextMatrix(0, 4) = "Total Price"
Form1.sale.MoveFirst
d = 1
toplam = 0
While Not Form1.sale.EOF
    If CDate(Form1.sale.Fields("date")) = Date Then
        MSFlexGrid1.AddItem ""
        MSFlexGrid1.TextMatrix(d, 0) = Form1.sale.Fields("sellcode")
        MSFlexGrid1.TextMatrix(d, 1) = Form1.sale.Fields("cuscode")
        Form1.customer.Index = "primarykey"
        Form1.customer.Seek "=", Form1.sale.Fields("cuscode")
        If Form1.customer.NoMatch = 0 Then
            MSFlexGrid1.TextMatrix(d, 2) = Form1.customer.Fields("name") + " " +
Form1.customer.Fields("surname")
            End If
            MSFlexGrid1.TextMatrix(d, 3) = Form1.sale.Fields("date")
            MSFlexGrid1.TextMatrix(d, 4) = Form1.sale.Fields("total")
            toplam = toplam + Val(Form1.sale.Fields("total"))
            End If
        d = d + 1
        If gridkontrol > 12 Then
            MSFlexGrid1.AddItem ""
        Else
            gridkotrol = gridkontrol + 1
        End If
        Form1.sale.MoveNext
    Wend
Text4.Text = Trim(Str(toplam))
End Sub

Private Sub Command6_Click()
    Dim d As Long
    Dim toplam As Double
    Dim gridkontrol As Integer
    Text4.Text = ""
    Text5.Text = ""
    MSFlexGrid2.clear
    MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
    MSFlexGrid2.TextMatrix(0, 1) = "Stock Name"
    MSFlexGrid2.TextMatrix(0, 2) = "quantity"
    MSFlexGrid2.TextMatrix(0, 3) = "price"

```

```

MSFlexGrid2.TextMatrix(0, 4) = "Total"
MSFlexGrid1.clear
MSFlexGrid1.TextMatrix(0, 0) = "Sale Code"
MSFlexGrid1.TextMatrix(0, 1) = "Cus. Code"
MSFlexGrid1.TextMatrix(0, 2) = "Name - Surname"
MSFlexGrid1.TextMatrix(0, 3) = "Date"
MSFlexGrid1.TextMatrix(0, 4) = "Total Price"
gridkontrol = 1
MSFlexGrid1.Rows = 12
If DTPicker3.value <= DTPicker4.value Then
Form1.sale.MoveFirst
d = 1
toplam = 0
While Not Form1.sale.EOF
    If CDate(Form1.sale.Fields("date")) >= DTPicker3.value And
CDate(Form1.sale.Fields("date")) <= DTPicker4.value Then
        MSFlexGrid1.AddItem ""
        MSFlexGrid1.TextMatrix(d, 0) = Form1.sale.Fields("sellcode")
        MSFlexGrid1.TextMatrix(d, 1) = Form1.sale.Fields("cuscode")
        Form1.customer.Index = "primarykey"
        Form1.customer.Seek "=", Form1.sale.Fields("cuscode")
        If Form1.customer.NoMatch = 0 Then
            MSFlexGrid1.TextMatrix(d, 2) = Form1.customer.Fields("name") + " " +
Form1.customer.Fields("surname")
            End If
            MSFlexGrid1.TextMatrix(d, 3) = Form1.sale.Fields("date")
            MSFlexGrid1.TextMatrix(d, 4) = Form1.sale.Fields("total")
            toplam = toplam + Val(Form1.sale.Fields("total"))
            d = d + 1
        If gridkontrol > 12 Then
            MSFlexGrid1.AddItem ""
        Else
            gridkontrol = gridkontrol + 1
        End If
        End If
        Form1.sale.MoveNext
    Wend
Text4.Text = Trim(Str(toplam))
Else
MsgBox ("Enter Dates Correctly")
End If
End Sub

Private Sub Command7_Click()
Image1.Visible = False
Label1.Visible = True
Label7.Visible = True
Dim total As Double
Dim total2 As Double
total = 0
total2 = 0

```



```

Form1.stock.MoveFirst
While Not Form1.stock.EOF
total = total + (Form1.stock.Fields("pureprice")) * (Form1.stock.Fields("amount"))
total2 = total2 + (Form1.stock.Fields("saleprice")) * (Form1.stock.Fields("amount"))
Form1.stock.MoveNext
Wend
Label1.Caption = "Total Pure Price " + Format(Str(total), "#.##")
Label7.Caption = "Total sale Price " + Format(Str(total2), "#.##")
End Sub

```

```

Private Sub Form_Load()
Text2.Locked = True
Text3.Locked = True
Text4.Locked = True
Text5.Locked = True
Label1.Visible = False
Label7.Visible = False
Text4.Text = ""
Text5.Text = ""
MSFlexGrid1.clear
MSFlexGrid2.clear
MSFlexGrid1.Rows = 12
MSFlexGrid2.Rows = 12
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
List1.clear
Text2.Locked = True
Text3.Locked = True
MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Stock Name"
MSFlexGrid2.TextMatrix(0, 2) = "quantity"
MSFlexGrid2.TextMatrix(0, 3) = "price"
MSFlexGrid2.TextMatrix(0, 4) = "Total"
MSFlexGrid2.ColWidth(0) = 1000
MSFlexGrid2.ColWidth(1) = 2400
MSFlexGrid2.ColWidth(2) = 800
MSFlexGrid2.ColWidth(3) = 840
MSFlexGrid2.ColWidth(4) = 840
MSFlexGrid1.ColWidth(0) = 880
MSFlexGrid1.ColWidth(1) = 880
MSFlexGrid1.ColWidth(2) = 2400
MSFlexGrid1.ColWidth(3) = 950
MSFlexGrid1.ColWidth(4) = 900
MSFlexGrid1.TextMatrix(0, 0) = "Sale Code"
MSFlexGrid1.TextMatrix(0, 1) = "Cus. Code"
MSFlexGrid1.TextMatrix(0, 2) = "Name - Surname"
MSFlexGrid1.TextMatrix(0, 3) = "Date"
MSFlexGrid1.TextMatrix(0, 4) = "Total Price"
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)

```

```

Form2.Show
End Sub
Private Sub List1_Click()
Text2.Text = List1.Text
Form1.user.Index = "primarykey"
Form1.user.Seek "=", Text2.Text
Text3 = Form1.user.Fields("usertype")
End Sub


---


Private Sub Option1_Click()
If Option1.value = True Then
Label1.Caption = "By Code: "
End If
End Sub


---


Private Sub Option2_Click()
If Option2.value = True Then
Label1.Caption = "By Name: "
End If
End Sub


---


Private Sub MSFlexGrid1_Click()
MSFlexGrid1.Col = 0
MSFlexGrid1.ColSel = 4
Dim selkod As String
Dim d As Long
d = 1
selkod = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 0)
Form1.content.MoveFirst
While Not Form1.content.EOF
    If Form1.content.Fields("sellcode") = selkod Then
        MSFlexGrid2.TextMatrix(d, 0) = Form1.content.Fields("stockcode")
        Form1.stock.Index = "primarykey"
        Form1.stock.Seek "=", Form1.content.Fields("stockcode")
        If Form1.stock.NoMatch = 0 Then
            MSFlexGrid2.TextMatrix(d, 1) = Form1.stock.Fields("stkname")
        End If
        MSFlexGrid2.TextMatrix(d, 2) = Form1.content.Fields("quantity")
        MSFlexGrid2.TextMatrix(d, 3) = Form1.stock.Fields("saleprice")
        MSFlexGrid2.TextMatrix(d, 4) = Form1.content.Fields("total")
        MSFlexGrid2.AddItem ""
        d = d + 1
    End If
    Form1.content.MoveNext
Wend
Text5.Text = Format(MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 4), "#.00")
End Sub


---


Private Sub MSFlexGrid1_RowColChange()
MSFlexGrid1.Col = 0
MSFlexGrid1.ColSel = 4
End Sub


---


Private Sub MSFlexGrid2_Click()
MSFlexGrid2.Col = 0

```

```
MSFlexGrid2.ColSel = 4
End Sub
```

Form6

```
Public form6gelenform As Integer
Private Sub Command1_Click()
    Dim d As Long
    Dim total As Double
    Dim i As Integer
    If Form6.form6gelenform = 14 Then
        If Text1.Text = "" Or Text1.Text = "Fill" Then
            Form6.Show
            Text1.Text = "Fill"
            Text1.SelStart = 0
            Text1.SelLength = Len(Text1.Text)
            Text1.SetFocus
        End If
        d = Form14.MSFlexGrid2.Rows
        Form14.MSFlexGrid2.AddItem ""
        Form14.MSFlexGrid2.TextMatrix(d, 0) = MSFlexGrid1.TextMatrix(1, 0)
        Form14.MSFlexGrid2.TextMatrix(d, 1) = MSFlexGrid1.TextMatrix(1, 1)
        Form14.MSFlexGrid2.TextMatrix(d, 2) = MSFlexGrid1.TextMatrix(1, 2)
        Form14.MSFlexGrid2.TextMatrix(d, 3) = Text1.Text
        Form14.MSFlexGrid2.TextMatrix(d, 4) = MSFlexGrid1.TextMatrix(1, 4)
        Unload Me
        Form14.Show
    End If
    If Form6.form6gelenform = 2 Then
        If Text1.Text = "" Or Text1.Text = "Fill" Then
            Form6.Show
            Text1.Text = "Fill"
            Text1.SelStart = 0
            Text1.SelLength = Len(Text1.Text)
            Text1.SetFocus
        Else
            If Val(Form6.MSFlexGrid1.TextMatrix(1, 3)) < Val(Form6.Text1.Text) Then
                MsgBox ("we don't have this Quantity")
            Else
                d = Form2.MSFlexGrid2.Rows
                Form2.MSFlexGrid2.AddItem ""
                Form2.MSFlexGrid2.TextMatrix(d, 0) = MSFlexGrid1.TextMatrix(1, 0)
                Form2.MSFlexGrid2.TextMatrix(d, 1) = MSFlexGrid1.TextMatrix(1, 1)
                Form2.MSFlexGrid2.TextMatrix(d, 2) = MSFlexGrid1.TextMatrix(1, 2)
                Form2.MSFlexGrid2.TextMatrix(d, 3) = Text1.Text
                Form2.MSFlexGrid2.TextMatrix(d, 4) = MSFlexGrid1.TextMatrix(1, 4)
                Form2.MSFlexGrid2.TextMatrix(d, 5) = MSFlexGrid1.TextMatrix(1, 4) *
                Val(Text1.Text)
                Call Form2.writetotal
            End If
        End If
        Form2.Show
        Unload Me
    End If
End Sub
```



```

End If
End If
End If
If Form6.form6gelenform = 22 Then
If Val(Form6.MSFlexGrid1.TextMatrix(1, 3)) < Val(Form6.Text1.Text) Then
MsgBox ("we don't have this Quantity")
Text1.SetFocus
Else
Form2.MSFlexGrid2.TextMatrix(Form2.MSFlexGrid2.Row, 3) = Form6.Text1.Text
Unload Me
Form2.Show
End If
End If
End Sub

```

```

Private Sub Command2_Click()
Unload Me
End Sub

```

```

Private Sub Form_Load()
MSFlexGrid1.clear
MSFlexGrid1.TextMatrix(0, 0) = "Stk.Code"
MSFlexGrid1.TextMatrix(0, 1) = "Product Name"
MSFlexGrid1.TextMatrix(0, 2) = "Product Type"
MSFlexGrid1.TextMatrix(0, 3) = "Quantity"
MSFlexGrid1.TextMatrix(0, 4) = "Sale Price"
MSFlexGrid1.ColWidth(0) = 1260
MSFlexGrid1.ColWidth(1) = 2300
MSFlexGrid1.ColWidth(2) = 1800
MSFlexGrid1.ColWidth(3) = 1200
MSFlexGrid1.ColWidth(4) = 1280
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Unload Me
End Sub

```

```

Private Sub Text1_GotFocus()
Text1.SelStart = 0
Text1.SelLength = Len(Text1)
End Sub

```

```

Private Sub Text1_LostFocus()
Dim metin As String
Dim resultchar As Boolean
For i = 1 To Len(Text1)
metin = Mid(Text1, i, 1)
If Asc(metin) < 48 Or Asc(metin) > 57 Then
resultchar = True
End If
Next
If resultchar = True Then
Text1.SetFocus
End If

```

End Sub

Form7

```
Private Sub temizle()
```

```
Text1.Text = ""
```

```
Text2.Text = ""
```

```
Text3.Text = ""
```

```
Text4.Text = ""
```

```
Text5.Text = ""
```

```
Text6.Text = ""
```

```
Text7.Text = ""
```

```
Text8.Text = ""
```

```
End Sub
```

```
Private Sub Command1_Click()
```

```
Dim max As Long
```

```
max = 0
```

```
Call temizle
```

```
Text2.Locked = False
```

```
Text3.Locked = False
```

```
Text4.Locked = False
```

```
Text5.Locked = False
```

```
Text6.Locked = False
```

```
Text7.Locked = False
```

```
Text8.Locked = False
```

```
If Form1.customer.RecordCount > 0 Then
```

```
Form1.customer.MoveFirst
```

```
While Not Form1.customer.EOF
```

```
If Form1.customer.Fields("cuscode") > max Then
```

```
max = Val(Form1.customer.Fields("cuscode"))
```

```
End If
```

```
Form1.customer.MoveNext
```

```
Wend
```

```
End If
```

```
Text1.Text = Trim(Str(max + 1))
```

```
End Sub
```

```
Private Sub Command2_Click()
```

```
Dim mesaj As String
```

```
If Text2.Text = "" Or Text3.Text = "" Or Text4.Text = "" Or Text6.Text = "" Then  
MsgBox ("At least you have to fill the marked blanks")
```

```
Else
```

```
Form1.customer.AddNew
```

```
Form1.customer.Fields("cuscode") = Text1.Text
```

```
Form1.customer.Fields("name") = Text2.Text
```

```
Form1.customer.Fields("surname") = Text3.Text
```

```
Form1.customer.Fields("company") = Text4.Text
```

```
Form1.customer.Fields("mobile") = Text6.Text
```

```
Form1.customer.Fields("tel") = Text5.Text
```

```
Form1.customer.Fields("address") = Text7.Text
```

```
Form1.customer.Fields("email") = Text8.Text
```

```
Form1.customer.Update
```

```

FileCopy File1.Path + "\" + File1.FileName, Form1.programyer + "\picture\" +
Text1.Text + ".jpg"
Text2.Locked = True
Text3.Locked = True
Text4.Locked = True
Text5.Locked = True
Text6.Locked = True
Text7.Locked = True
Text8.Locked = True
mesaj = "New Customer has been saved with this customer code=" + Text1.Text
MsgBox (mesaj)
Form7.Hide
Form2.Show
End If
End Sub

```

```

Private Sub Command3_Click()
Form13.comingform = 7
Form13.List1.clear
Form13.List2.clear
Form13.List3.clear
Form13.List4.clear
Form13.List5.clear
Form13.List6.clear
Form13.List7.clear
Form13.List8.clear
Form1.customer.MoveFirst
While Not Form1.customer.EOF
Form13.List1.AddItem (Form1.customer.Fields("cuscode"))
Form13.List2.AddItem (Form1.customer.Fields("name"))
Form13.List3.AddItem (Form1.customer.Fields("surname"))
Form13.List4.AddItem (Form1.customer.Fields("company"))
If IsNull(Form1.customer.Fields("tel")) = False Then
Form13.List5.AddItem (Form1.customer.Fields("tel"))
Else
Form13.List5.AddItem ""
End If
Form13.List6.AddItem (Form1.customer.Fields("mobile"))
If IsNull(Form1.customer.Fields("address")) = False Then
Form13.List7.AddItem (Form1.customer.Fields("address"))
Else
Form13.List7.AddItem " "
End If
If IsNull(Form1.customer.Fields("email")) = False Then
Form13.List8.AddItem (Form1.customer.Fields("email"))
Else
Form13.List8.AddItem " "
End If
Form1.customer.MoveNext
Wend
Form2.Enabled = False

```



```

Form13.Show
End Sub


---


Private Sub Dir1_Change()
File1.Path = Dir1.Path
End Sub


---


Private Sub Drive1_Change()
On Error Resume Next
Dir1.Path = Drive1.Drive
End Sub


---


Private Sub File1_Click()
Image1.Picture = LoadPicture(File1.Path + "\" + File1.FileName)
End Sub


---


Private Sub Form_Load()
Call temizle
File1.Pattern = "*.jpg"
End Sub


---


Private Sub Form_Unload(Cancel As Integer)
Form2.Show
End Sub


---



```

Form8

```

Private Sub Command2_Click()
Dim msg As Integer
Dim result As Double
If Text1.Text = "" Then
MsgBox ("You have to Select a Customer")
Else
If Text4.Text = "" Then
MsgBox ("You Have To Enter Payment")
Else
If Option3.value = True Then
msg = MsgBox("You Entered " + Format(Text4.Text, "#,##0"), 36, "Are you sure")
If msg = 6 Then
Form1.payment.AddNew
Form1.payment.Fields("cuscode") = Text1.Text
Form1.payment.Fields("date") = Date
Form1.payment.Fields("total") = Text4.Text
Form1.payment.Fields("username") = Form1.Text1.Text
Form1.payment.Update
MsgBox ("You Payied : " + Format(Text4, "#.##") + "dolar")
Unload Me
End If
End If
If Option4.value = True Then
If Text5.Text <> "" And IsNumeric(Text5) = True Then
result = Val(Text4.Text) / Val(Text5.Text)
msg = MsgBox("You Entered " + Text4.Text + " Dinar", 36, "Are you sure")
If msg = 6 Then
Form1.payment.AddNew

```

```

Form1.payment.Fields("cuscode") = Text1.Text
Form1.payment.Fields("date") = Date
Form1.payment.Fields("total") = Format(result, "#.##")
Form1.payment.Fields("username") = Form1.Text1.Text
Form1.payment.Update
MsgBox ("You Payied : " + Format(result, "#.##") + " dolar")
Unload Me
End If
If msj = 7 Then
Exit Sub
End If
Else
Text5.SetFocus
MsgBox ("Write Today's Rate of Dollar")
End If
End If
End If
End Sub

```

```

Private Sub Command3_Click()
Unload Me
Form2.Show
End Sub
Private Sub Command6_Click()
Form13.comingform = 8
On Error Resume Next
If Option1.value = True Then
If Text9.Text = "" Then
MsgBox ("please write a code")
Else
Form1.customer.Index = "primarykey"
Form1.customer.Seek "=", Text9.Text
Text1 = Form1.customer.Fields("cuscode")
Text2 = Form1.customer.Fields("name")
Text3 = Form1.customer.Fields("surname")
Text10 = Form1.customer.Fields("company")
End If
End If
If Option2.value = True Then
If Text9.Text = "" Then
MsgBox ("please write a name")
Else
Form1.customer.MoveFirst
While Not Form1.customer.EOF
If LCase(Text9.Text) = LCase(Left(Form1.customer.Fields("name"),
Len(Text9.Text))) Then
Form2.Enabled = False
Form13.Show
Form13.List1.AddItem Form1.customer.Fields("cuscode")
Form13.List2.AddItem Form1.customer.Fields("name")

```

```

Form13.List3.AddItem Form1.customer.Fields("surname")
Form13.List4.AddItem Form1.customer.Fields("company")
    If IsNull(Form1.customer.Fields("tel")) = False Then
        Form13.List5.AddItem (Form1.customer.Fields("tel"))
    Else
        Form13.List5.AddItem " "
    End If
Form13.List6.AddItem Form1.customer.Fields("mobile")
    If IsNull(Form1.customer.Fields("address")) = False Then
        Form13.List7.AddItem Form1.customer.Fields("address")
    Else
        Form13.List7.AddItem " "
    End If
    If IsNull(Form1.customer.Fields("email")) = False Then
        Form13.List8.AddItem Form1.customer.Fields("email")
    Else
        Form13.List8.AddItem " "
    End If
End If
Form1.customer.MoveNext
Wend
End If
End If
End Sub

```

```

Private Sub Command7_Click()
Form13.comingform = 8
Form13.List1.clear
Form13.List2.clear
Form13.List3.clear
Form13.List4.clear
Form13.List5.clear
Form13.List6.clear
Form13.List7.clear
Form13.List8.clear
Form1.customer.MoveFirst
While Not Form1.customer.EOF
Form13.List1.AddItem (Form1.customer.Fields("cuscode"))
Form13.List2.AddItem (Form1.customer.Fields("name"))
Form13.List3.AddItem (Form1.customer.Fields("surname"))
Form13.List4.AddItem (Form1.customer.Fields("company"))
    If IsNull(Form1.customer.Fields("tel")) = False Then
        Form13.List5.AddItem (Form1.customer.Fields("tel"))
    Else
        Form13.List5.AddItem ""
    End If
Form13.List6.AddItem (Form1.customer.Fields("mobile"))
    If IsNull(Form1.customer.Fields("address")) = False Then
        Form13.List7.AddItem (Form1.customer.Fields("address"))
    Else
        Form13.List7.AddItem " "
    End If
End While
End Sub

```



```

End If
If IsNull(Form1.customer.Fields("email")) = False Then
Form13.List8.AddItem (Form1.customer.Fields("email"))
Else
Form13.List8.AddItem " "
End If
Form1.customer.MoveNext
Wend
Form2.Enabled = False
Form13.Show
End Sub

```

```

Private Sub Form_Load()
Option1.value = True
Option3.value = True
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text9.Text = ""
Text10.Text = ""
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Form2.Show
End Sub

```

```

Private Sub Text1_Change()
If Text1.Text <> "" Then
On Error Resume Next
Image1.Picture = LoadPicture(Form1.programyer + "\picture\" + Text1.Text + ".jpg")
End If
End Sub

```

```

Private Sub Text4_LostFocus()
Dim metin As String
Dim resultchar As Boolean
For i = 1 To Len(Text1)
metin = Mid(Text4, i, 1)
If Asc(metin) < 48 Or Asc(metin) > 57 Then
resultchar = True
End If
Next
If resultchar = True Then
Text4.SetFocus
End If
If IsNumeric(Text4.Text) = False And Text <> "" Then
MsgBox ("Enter Currency Please")
Text4.SetFocus
End If
End Sub

```

Form9

```
Private Sub Command6_Click()
Form13.comingform = 9
If Option1.value = True Then
    If Text9.Text = "" Then
        MsgBox ("please write a code")
    Else
        Form1.customer.Index = "primarykey"
        Form1.customer.Seek "=", Text9.Text
        If Form1.customer.NoMatch = False Then
            Text2 = Form1.customer.Fields("name")
            Text3 = Form1.customer.Fields("surname")
            Text10 = Form1.customer.Fields("company")
            Text1 = Form1.customer.Fields("cuscode")
        Else
            MsgBox ("Not Found")
        End If
    End If
End If
If Option2.value = True Then
    If Text9.Text = "" Then
        MsgBox ("please write a name")
    Else
        Form1.customer.MoveFirst
        While Not Form1.customer.EOF
            If LCase(Text9.Text) = LCase(Left(Form1.customer.Fields("name"),
Len(Text9.Text))) Then
                Form2.Enabled = False
                Form13.Show
                Form13.List1.AddItem Form1.customer.Fields("cuscode")
                Form13.List2.AddItem Form1.customer.Fields("name")
                Form13.List3.AddItem Form1.customer.Fields("surname")
                Form13.List4.AddItem Form1.customer.Fields("company")
                If IsNull(Form1.customer.Fields("tel")) = False Then
                    Form13.List5.AddItem (Form1.customer.Fields("tel"))
                Else
                    Form13.List5.AddItem " "
                End If
                Form13.List6.AddItem Form1.customer.Fields("mobile")
                If IsNull(Form1.customer.Fields("address")) = False Then
                    Form13.List7.AddItem Form1.customer.Fields("address")
                Else
                    Form13.List7.AddItem " "
                End If
                If IsNull(Form1.customer.Fields("email")) = False Then
                    Form13.List8.AddItem Form1.customer.Fields("email")
                Else
                    Form13.List8.AddItem " "
                End If
            End If
        End If
    End If
End If
```

```

        Form1.customer.MoveNext
    Wend
End If
End If
End Sub

```

```

Private Sub Command7_Click()
Form13.comingform = 9
Form13.List1.clear
Form13.List2.clear
Form13.List3.clear
Form13.List4.clear
Form13.List5.clear
Form13.List6.clear
Form13.List7.clear
Form13.List8.clear
Form1.customer.MoveFirst
While Not Form1.customer.EOF
Form13.List1.AddItem (Form1.customer.Fields("cuscode"))
Form13.List2.AddItem (Form1.customer.Fields("name"))
Form13.List3.AddItem (Form1.customer.Fields("surname"))
Form13.List4.AddItem (Form1.customer.Fields("company"))
If IsNull(Form1.customer.Fields("tel")) = False Then
Form13.List5.AddItem (Form1.customer.Fields("tel"))
Else
Form13.List5.AddItem ""
End If
Form13.List6.AddItem (Form1.customer.Fields("mobile"))
If IsNull(Form1.customer.Fields("address")) = False Then
Form13.List7.AddItem (Form1.customer.Fields("address"))
Else
Form13.List7.AddItem " "
End If
If IsNull(Form1.customer.Fields("email")) = False Then
Form13.List8.AddItem (Form1.customer.Fields("email"))
Else
Form13.List8.AddItem " "
End If
Form1.customer.MoveNext
Wend
Form2.Enabled = False
Form13.Show
End Sub
Private Sub clear1()
MSFlexGrid1.clear
MSFlexGrid1.Rows = 1
MSFlexGrid1.TextMatrix(0, 0) = "Sale Code"
MSFlexGrid1.TextMatrix(0, 1) = "User Name"
MSFlexGrid1.TextMatrix(0, 2) = "Date"
MSFlexGrid1.TextMatrix(0, 3) = "Total "
MSFlexGrid1.ColWidth(0) = 950

```



```

MSFlexGrid1.ColWidth(1) = 2150
MSFlexGrid1.ColWidth(2) = 1150
MSFlexGrid1.ColWidth(3) = 1100
End Sub

```

```

Private Sub clear2()
MSFlexGrid2.clear
MSFlexGrid2.Rows = 1
MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Stock Name"
MSFlexGrid2.TextMatrix(0, 2) = "Price"
MSFlexGrid2.TextMatrix(0, 3) = "Quantity"
MSFlexGrid2.TextMatrix(0, 4) = "Total"
MSFlexGrid2.ColWidth(0) = 950
MSFlexGrid2.ColWidth(1) = 1800
MSFlexGrid2.ColWidth(2) = 900
MSFlexGrid2.ColWidth(3) = 900
MSFlexGrid2.ColWidth(4) = 900
End Sub

```

```

Private Sub clear3()
MSFlexGrid3.clear
MSFlexGrid3.Rows = 1
MSFlexGrid3.TextMatrix(0, 0) = "User Name"
MSFlexGrid3.TextMatrix(0, 1) = "Date"
MSFlexGrid3.TextMatrix(0, 2) = "Amount"
MSFlexGrid3.ColWidth(0) = 1500
MSFlexGrid3.ColWidth(1) = 890
MSFlexGrid3.ColWidth(2) = 860
End Sub

```

```

Private Sub Form_Load()
Call clear1
Call clear2
Call clear3
Option1.value = True
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Text6.Text = ""
Text7.Text = ""
Text8.Text = ""
Text9.Text = ""
Text10.Text = ""
Text1.Locked = True
Text2.Locked = True
Text3.Locked = True
Text4.Locked = True
Text6.Locked = True
Text7.Locked = True
Text8.Locked = True
Text10.Locked = True

```

```

End Sub
Private Sub Form_Unload(Cancel As Integer)
Form2.Show
End Sub
Private Sub MSFlexGrid1_Click()
On Error Resume Next
MSFlexGrid1.Col = 0
MSFlexGrid1.ColSel = 3
Dim d As Long
Call clear2
If MSFlexGrid1.Rows > 1 Then
    If Form1.content.RecordCount > 0 Then
        d = 1
        Form1.content.MoveFirst
        While Not Form1.content.EOF
            If Form1.content.Fields("sellcode") =
MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 0) Then
                MSFlexGrid2.AddItem ""
                MSFlexGrid2.TextMatrix(d, 0) = Form1.content.Fields("stockcode")
                Form1.stock.Index = "primarykey"
                Form1.stock.Seek "=", Form1.content.Fields("stockcode")
                If Form1.stock.NoMatch = False Then
                    MSFlexGrid2.TextMatrix(d, 1) = Form1.stock.Fields("stkname")
                    MSFlexGrid2.TextMatrix(d, 2) = Form1.stock.Fields("saleprice")
                End If
                MSFlexGrid2.TextMatrix(d, 3) = Form1.content.Fields("quantity")
                MSFlexGrid2.TextMatrix(d, 4) = Form1.content.Fields("total")
                d = d + 1
            End If
            Form1.content.MoveNext
        Wend
    End If
    If Form1.loan.RecordCount > 0 Then
        Form1.loan.MoveFirst
        While Not Form1.loan.EOF
            If Form1.loan.Fields("sellcode") =
MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 0) Then
                Text7 = Form1.loan.Fields("total")
                Text8 = ((MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 3) * 100) - (Text7 *
100)) / 100
            End If
            Form1.loan.MoveNext
        Wend
    Else
        Text7 = "0"
        Text8 = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 3)
    End If
End If
End Sub
Private Sub MSFlexGrid2_Click()

```

```

On Error Resume Next
MSFlexGrid2.Col = 0
MSFlexGrid2.ColSel = 4
End Sub

```

```

Private Sub MSFlexGrid3_Click()
On Error Resume Next
MSFlexGrid3.Col = 0
MSFlexGrid3.ColSel = 2
End Sub

```

```

Private Sub Text1_Change()
Dim d As Long
Dim d2 As Long
Dim total As Double
Call clear1
Call clear2
Call clear3
Text7.Text = ""
Text8.Text = ""
If Text1.Text <> "" Then
On Error Resume Next
Image1.Picture = LoadPicture(Form1.programyer + "\picture\" + Text1.Text + ".jpg")
End If
If Text1 <> "" Then
If Form1.sale.RecordCount > 0 Then
d = 1
Form1.sale.MoveFirst
While Not Form1.sale.EOF
If Form1.sale.Fields("cuscode") = Text1 Then
MSFlexGrid1.AddItem ""
MSFlexGrid1.TextMatrix(d, 0) = Form1.sale.Fields("sellcode")
MSFlexGrid1.TextMatrix(d, 1) = Form1.sale.Fields("username")
MSFlexGrid1.TextMatrix(d, 2) = Form1.sale.Fields("date")
MSFlexGrid1.TextMatrix(d, 3) = Form1.sale.Fields("total")
d = d + 1
End If
Form1.sale.MoveNext
Wend
End If
If Form1.payment.RecordCount > 0 Then
d2 = 1
Form1.payment.MoveFirst
While Not Form1.payment.EOF
If Form1.payment.Fields("cuscode") = Text1 Then
MSFlexGrid3.AddItem ""
MSFlexGrid3.TextMatrix(d2, 0) = Form1.payment.Fields("username")
MSFlexGrid3.TextMatrix(d2, 1) = Form1.payment.Fields("date")
MSFlexGrid3.TextMatrix(d2, 2) = Form1.payment.Fields("total")
d2 = d2 + 1
End If
Form1.payment.MoveNext

```



```

Wend
For i = 1 To MSFlexGrid3.Rows - 1
    total = total + MSFlexGrid3.TextMatrix(i, 2)
Next
Text6 = Format(total, "#.00")
End If
If Form1.loan.RecordCount > 0 Then
    total = 0
    Form1.loan.MoveFirst
    While Not Form1.loan.EOF
        If Form1.loan.Fields("cuscode") = Text1 Then
            total = total + Form1.loan.Fields("total")
        End If
        Form1.loan.MoveNext
    Wend
    Text4 = ((total * 100) - (Text6 * 100)) / 100
End If
End If
End Sub

```

```

Private Sub Text4_Change()
If Text4.Text <> "" Then
If Val(Text4) <= 0 Then
Image2.Picture = LoadPicture(Form1.programyer + "\picture\" + "gulmek.jpg")
Else
Image2.Picture = LoadPicture(Form1.programyer + "\picture\" + "aglamak.jpg")
End If
Else
End If
End Sub

```

Form10

```

Private Sub Check1_Click()
If Check1.value = 1 Then
Text1.Locked = True
Text2.Locked = False
Text3.Locked = False
Text4.Locked = False
Text5.Locked = False
Text6.Locked = False
Else
Text1.Locked = True
Text2.Locked = True
Text3.Locked = True
Text4.Locked = True
Text5.Locked = True
Text6.Locked = True
End If
End Sub

```

```

Private Sub Check2_Click()
Dim al_sql As String

```

```

Dim sql1 As Recordset
Call clearmsflexgrid
Text14.Text = ""
Call cleartext
Check1.value = 0
Combo1.clear
If Check2.value = 1 Then
Combo1.Locked = False
Combo1.Enabled = True
al_sql = "select stktype from stock group by stktype"
Set sql1 = Form1.db.OpenRecordset(al_sql)
    If sql1.RecordCount > 0 Then
        sql1.MoveFirst
        While Not sql1.EOF
            Combo1.AddItem sql1.Fields("stktype")
            sql1.MoveNext
        Wend
    End If
Combo1.ListIndex = 0
Else
Combo1.Locked = True
Combo1.Enabled = False
Combo1.clear
End If
End Sub
Private Sub Combo1_Click()
Dim d As Long
Call clearmsflexgrid
If Combo1.ListCount > 0 Then
d = 1
Form1.stock.MoveFirst
    While Not Form1.stock.EOF
        If Form1.stock.Fields("stktype") = Combo1.Text Then
            MSFlexGrid1.AddItem ""
            MSFlexGrid1.TextMatrix(d, 0) = (Form1.stock.Fields("stockcode"))
            MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
            MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
            MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
            MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
            d = d + 1
        End If
        Form1.stock.MoveNext
    Wend
End If
End Sub


---


Private Sub Command1_Click()
Dim d As Long
Check2.value = 0
Call clearmsflexgrid
Call cleartext

```

```

Check1.value = 0
Form1.stock.Index = "primarykey"
Form1.stock.Seek "=", Text14.Text
If Text14.Text = "" Then
MsgBox ("please Write a Stock Code")
Call clearmsflexgrid
Else
If Form1.stock.NoMatch <> 0 Then
MsgBox ("Database don't have this Stock Code")
Text14.Text = ""
Else
d = 1
MSFlexGrid1.AddItem ""
MSFlexGrid1.TextMatrix(d, 0) = Text14.Text
MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
d = d + 1
End If
End If
End Sub

```

```

Private Sub temizle()
Text7.Text = ""
Text8.Text = ""
Text9.Text = ""
Text10.Text = ""
Text11.Text = ""
Text12.Text = ""
Text13.Text = ""
End Sub

```

```

Private Sub Command2_Click()
Dim al_sql As String
Dim sql1 As Recordset
If Text7.Text = "" Or Text8.Text = "" Or Text9.Text = "" Or Text10.Text = "" Or
Text11.Text = "" Or Text12.Text = "" Or Text13.Text = "" Then
MsgBox ("please fill all the blanks")
Else
Command3.Enabled = True
Command2.Enabled = False
Form1.stock.AddNew
Form1.stock.Fields("stockcode") = Text7.Text
Form1.stock.Fields("stkname") = Text8.Text
Form1.stock.Fields("stktype") = Text9.Text
Form1.stock.Fields("pureprice") = Text10.Text
Form1.stock.Fields("transportation") = Text11.Text
Form1.stock.Fields("saleprice") = Text12.Text
Form1.stock.Fields("amount") = Text13.Text
Form1.stock.Update
Text7.Text = ""

```



```

Text8.Text = ""
Text9.Text = ""
Text10.Text = ""
Text11.Text = ""
Text12.Text = ""
Text13.Text = ""
Command3.SetFocus
Combo1.clear
al_sql = "select stktype from stock group by stktype"
Set sql1 = Form1.db.OpenRecordset(al_sql)
    If sql1.RecordCount > 0 Then
        sql1.MoveFirst
        While Not sql1.EOF
            Combo1.AddItem sql1.Fields("stktype")
            sql1.MoveNext
        Wend
    End If
MsgBox ("New Product has benn Added")
End If
End Sub

```

```

Private Sub Command3_Click()
Dim max As Long
max = 0
Command2.Enabled = True
Command3.Enabled = False
Call temizle
Text8.Locked = False
Text9.Locked = False
Text10.Locked = False
Text11.Locked = False
Text12.Locked = False
Text13.Locked = False
If Form1.stock.RecordCount > 0 Then
Form1.stock.MoveFirst
While Not Form1.stock.EOF
If Form1.stock.Fields("stockcode") > max Then
max = Val(Form1.stock.Fields("stockcode"))
End If
Form1.stock.MoveNext
Wend
End If
Text7.Text = Trim(Str(max + 1))
End Sub

```

```

Private Sub Command4_Click()
Dim value As String
If Check1.value = 1 Then
If Text1.Text <> "" And Text2.Text <> "" And Text3.Text <> "" And Text4.Text <> ""
And Text5.Text <> "" And Text6.Text <> "" Then
Form1.stock.Index = "primarykey"
Form1.stock.Seek "=", Text1.Text

```

```

Form1.stock.Edit
Form1.stock.Fields("stkname") = Text2.Text
Form1.stock.Fields("stktype") = Text3.Text
Form1.stock.Fields("saleprice") = Text4.Text
value = Val(Text5) + Val(Text6)
Form1.stock.Fields("amount") = value
Form1.stock.Update
MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 3) = Val(Text5) + Val(Text6)
Call cleartext
MsgBox ("Changes has been saved successfully")
Else
MsgBox ("Fill All the blanks please")
End If
End If
End Sub

```

```

Private Sub Command6_Click()
Dim d As Long
Call clearmsflexgrid
Combo1.clear
Check2.value = 0
d = 1
Form1.stock.MoveFirst
While Not Form1.stock.EOF
MSFlexGrid1.AddItem ""
MSFlexGrid1.TextMatrix(d, 0) = (Form1.stock.Fields("stockcode"))
MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
d = d + 1
Form1.stock.MoveNext
Wend
End Sub

```

```

Private Sub Form_Load()
Call clearmsflexgrid
Command2.Enabled = False
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text6.Text = ""
Text7.Text = ""
Text8.Text = ""
Text9.Text = ""
Text10.Text = ""
Text11.Text = ""
Text12.Text = ""
Text13.Text = ""
Text14.Text = ""

```

```

Combo1.clear
Text1.Locked = True
Text2.Locked = True
Text3.Locked = True
Text4.Locked = True
Text5.Locked = True
Text6.Locked = True
Text7.Locked = True
Combo1.Enabled = False
End Sub
Private Sub cleartext()
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text4.Text = ""
Text5.Text = ""
Text6.Text = ""
End Sub
Private Sub clearmsflexgrid()
MSFlexGrid1.Rows = 27
MSFlexGrid1.clear
MSFlexGrid1.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid1.TextMatrix(0, 1) = "Product Name"
MSFlexGrid1.TextMatrix(0, 2) = "Product Type"
MSFlexGrid1.TextMatrix(0, 3) = "Quantity"
MSFlexGrid1.TextMatrix(0, 4) = "Sale Price"
MSFlexGrid1.ColWidth(0) = 950
MSFlexGrid1.ColWidth(1) = 2290
MSFlexGrid1.ColWidth(2) = 1600
MSFlexGrid1.ColWidth(3) = 880
MSFlexGrid1.ColWidth(4) = 880
End Sub


---


Private Sub Option1_Click()
If Option1 = True Then
Label6.Caption = "Stock Code :"
End If
End Sub


---


Private Sub Option2_Click()
If Option2 = True Then
Label6.Caption = "Product Name :"
End If
End Sub


---


Private Sub Form_Unload(Cancel As Integer)
Form2.Show
End Sub


---


Private Sub List1_Click()
List2.ListIndex = List1.ListIndex
List2.TopIndex = List1.TopIndex
List3.ListIndex = List1.ListIndex
List3.TopIndex = List1.TopIndex

```



```
List4.ListIndex = List1.ListIndex
List4.TopIndex = List1.TopIndex
List9.ListIndex = List1.ListIndex
List9.TopIndex = List1.TopIndex
Text1.Text = List1.Text
Text2.Text = List2.Text
Text3.Text = List3.Text
Text4.Text = List9.Text
Text5.Text = List4.Text
End Sub
```

```
Private Sub List2_Click()
List1.ListIndex = List2.ListIndex
End Sub
```

```
Private Sub List3_Click()
List1.ListIndex = List3.ListIndex
End Sub
```

```
Private Sub List4_Click()
List1.ListIndex = List4.ListIndex
End Sub
```

```
Private Sub List9_Click()
List1.ListIndex = List9.ListIndex
End Sub
```

```
Private Sub MSFlexGrid1_Click()
On Error Resume Next
MSFlexGrid1.Col = 0
MSFlexGrid1.ColSel = 4
With MSFlexGrid1
Text1.Text = .TextMatrix(MSFlexGrid1.Row, 0)
Text2.Text = .TextMatrix(MSFlexGrid1.Row, 1)
Text3.Text = .TextMatrix(MSFlexGrid1.Row, 2)
Text5.Text = .TextMatrix(MSFlexGrid1.Row, 3)
Text4.Text = .TextMatrix(MSFlexGrid1.Row, 4)
End With
End Sub
```

Form11

```
Private Sub Command1_Click()
Dim totalrefund
totalrefund = 0
Dim smj As String
msj = MsgBox("Are you sure you want to Refund All items to the selected Customer",
36, "Refund Items")
If msj = 6 Then
With MSFlexGrid2
If Text1.Text <> "" Then
For i = 1 To .Rows - 1
totalrefund = totalrefund + .TextMatrix(i, 5)
Form1.stock.Index = "primarykey"
```

```

Form1.stock.Seek "=", .TextMatrix(i, 0)
If Form1.stock.NoMatch = False Then
Form1.stock.Edit
Form1.stock.Fields("amount") = Form1.stock.Fields("amount") + .TextMatrix(i,
3)
Form1.stock.Update
End If
Next
Form1.payment.AddNew
Form1.payment.Fields("cuscode") = Text1.Text
Form1.payment.Fields("date") = Date
Form1.payment.Fields("total") = totalrefund
Form1.payment.Fields("username") = Form1.username
Form1.payment.Update
MsgBox ("Total refunds : " + Format(totalrefund, "#.##") + " added to customer
payment...")
End If
End With
Form2.Show
End If
If msj = 7 Then
Exit Sub
End If
End Sub
Private Sub Command3_Click()
Unload Me
End Sub
Private Sub Command4_Click()
Unload Me
End Sub
Private Sub Form_Load()
Call textclear
Call textlock
Call clear
End Sub
Private Sub textclear()
Text1.Text = ""
Text2.Text = ""
Text3.Text = ""
Text10.Text = ""
End Sub
Private Sub textlock()
Text1.Locked = True
Text2.Locked = True
Text3.Locked = True
Text10.Locked = True
End Sub
Private Sub clear()
MSFlexGrid2.clear

```

```

MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Product Name"
MSFlexGrid2.TextMatrix(0, 2) = "Product Type"
MSFlexGrid2.TextMatrix(0, 3) = "Quantity"
MSFlexGrid2.TextMatrix(0, 4) = "Sale Price"
MSFlexGrid2.TextMatrix(0, 5) = "Total"
MSFlexGrid2.ColWidth(0) = 950
MSFlexGrid2.ColWidth(1) = 2040
MSFlexGrid2.ColWidth(2) = 1300
MSFlexGrid2.ColWidth(3) = 800
MSFlexGrid2.ColWidth(4) = 850
MSFlexGrid2.ColWidth(5) = 800
MSFlexGrid2.Rows = 1
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Form2.Show
End Sub

```

```

Private Sub MSFlexGrid2_Click()
On Error Resume Next
MSFlexGrid1.Col = 0
MSFlexGrid1.ColSel = 5
End Sub

```

Form12

```

Private Sub Command1_Click()
Command2.Visible = False
Form11.Show
End Sub

```

```

Private Sub Form_Load()

End Sub

```

Form13

```

Public comingform As Integer
Private Sub Command1_Click()
If comingform = 7 Then
    Form7.Text1 = List1.Text
    Form7.Text2 = List2.Text
    Form7.Text3 = List3.Text
    Form7.Text4 = List4.Text
    Form7.Text5 = List5.Text
    Form7.Text6 = List6.Text
    Form7.Text7 = List7.Text
    Form7.Text8 = List8.Text
    Form7.Image1.Picture = LoadPicture(Form1.programyer + "\picture\" +
Form2.Text1.Text + ".jpg")
    Unload Me
    Form7.Show
End If
If comingform = 14 Then

```



```

Form14.Label1 = List1.Text
Form14.Label3 = List2.Text + " " + List3.Text
Form14.Label4 = List4.Text
Form14.customercode = List1.Text
Unload Me
Form14.Show
End If
If comingform = 2 Then
Form2.Text1.Text = List1.Text
Form2.Text2.Text = List2.Text
Form2.Text3.Text = List3.Text
Form2.Text10.Text = List4.Text
Unload Me
Form2.Enabled = True
Form2.Show
End If
If comingform = 9 Then
Form9.Text1.Text = List1.Text
Form9.Text2.Text = List2.Text
Form9.Text3.Text = List3.Text
Form9.Text10.Text = List4.Text
Unload Me
Form2.Hide
Form9.Show
End If
If comingform = 8 Then
Form8.Text1.Text = List1.Text
Form8.Text2.Text = List2.Text
Form8.Text3.Text = List3.Text
Form8.Text10.Text = List4.Text
Unload Me
Form2.Hide
Form8.Show
End If
End Sub


---


Private Sub Command2_Click()
Unload Me
Form2.Enabled = True
Form2.Show
End Sub


---


Private Sub Form_Load()
List1.clear
List2.clear
List3.clear
List4.clear
List5.clear
List6.clear
List7.clear
List8.clear
End Sub


---



```

```
Private Sub Form_Unload(Cancel As Integer)
Form2.Enabled = True
Form2.Show
End Sub
```

```
Private Sub List1_Click()
List2.ListIndex = List1.ListIndex
List2.TopIndex = List1.TopIndex
List3.ListIndex = List1.ListIndex
List3.TopIndex = List1.TopIndex
List4.ListIndex = List1.ListIndex
List4.TopIndex = List1.TopIndex
List5.ListIndex = List1.ListIndex
List5.TopIndex = List1.TopIndex
List6.ListIndex = List1.ListIndex
List6.TopIndex = List1.TopIndex
List7.ListIndex = List1.ListIndex
List7.TopIndex = List1.TopIndex
List8.ListIndex = List1.ListIndex
List8.TopIndex = List1.TopIndex
End Sub
```

```
Private Sub List1_DblClick()
Call Command1_Click
End Sub
```

```
Private Sub List2_Click()
List1.ListIndex = List2.ListIndex
End Sub
```

```
Private Sub List2_DblClick()
Call Command1_Click
End Sub
```

```
Private Sub List3_Click()
List1.ListIndex = List3.ListIndex
End Sub
```

```
Private Sub List3_DblClick()
Call Command1_Click
End Sub
```

```
Private Sub List4_Click()
List1.ListIndex = List4.ListIndex
End Sub
```

```
Private Sub List4_DblClick()
Call Command1_Click
End Sub
```

```
Private Sub List5_Click()
List1.ListIndex = List5.ListIndex
End Sub
```

```
Private Sub List5_DblClick()
Call Command1_Click
End Sub
```

```
Private Sub List6_Click()
List1.ListIndex = List6.ListIndex
```

```

End Sub
Private Sub List6_DblClick()
Call Command1_Click
End Sub
Private Sub List7_Click()
List1.ListIndex = List7.ListIndex
End Sub
Private Sub List7_DblClick()
Call Command1_Click
End Sub
Private Sub List8_Click()
List1.ListIndex = List8.ListIndex
End Sub
Private Sub List8_DblClick()
Call Command1_Click
End Sub

```

Form14

```

Public customercode As String
Private Sub clear2()
MSFlexGrid2.clear
MSFlexGrid2.Rows = 1
MSFlexGrid2.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid2.TextMatrix(0, 1) = "Product Name"
MSFlexGrid2.TextMatrix(0, 2) = "Product Type"
MSFlexGrid2.TextMatrix(0, 3) = "Ref.Amount"
MSFlexGrid2.TextMatrix(0, 4) = "Price"
MSFlexGrid2.ColWidth(0) = 850
MSFlexGrid2.ColWidth(1) = 2080
MSFlexGrid2.ColWidth(2) = 1050
MSFlexGrid2.ColWidth(3) = 800
MSFlexGrid2.ColWidth(4) = 750
End Sub
Private Sub clear1()
MSFlexGrid1.clear
MSFlexGrid1.Rows = 15
MSFlexGrid1.TextMatrix(0, 0) = "Stock Code"
MSFlexGrid1.TextMatrix(0, 1) = "Product Name"
MSFlexGrid1.TextMatrix(0, 2) = "Product Type"
MSFlexGrid1.TextMatrix(0, 3) = "Quantity"
MSFlexGrid1.TextMatrix(0, 4) = "Price"
MSFlexGrid1.ColWidth(0) = 850
MSFlexGrid1.ColWidth(1) = 2080
MSFlexGrid1.ColWidth(2) = 1050
MSFlexGrid1.ColWidth(3) = 800
MSFlexGrid1.ColWidth(4) = 750
End Sub
Private Sub Check2_Click()
Text14.Text = ""

```



```

Dim al_sql As String
Dim sql1 As Recordset
Combo1.clear
If Check2.value = 1 Then
    Combo1.Locked = False
    Combo1.Enabled = True
    al_sql = "select stktype from stock group by stktype"
    Set sql1 = Form1.db.OpenRecordset(al_sql)
    If sql1.RecordCount > 0 Then
        sql1.MoveFirst
        While Not sql1.EOF
            Combo1.AddItem sql1.Fields("stktype")
            sql1.MoveNext
        Wend
    End If
    Combo1.ListIndex = 0
Else
    Combo1.Locked = True
    Combo1.Enabled = False
    Combo1.clear
End If
End Sub

```

```

Private Sub Combo1_Click()
Dim d As Long
Call clear1
If Combo1.ListCount > 0 Then
    d = 1
    Form1.stock.MoveFirst
    While Not Form1.stock.EOF
        If Form1.stock.Fields("stktype") = Combo1.Text Then
            MSFlexGrid1.AddItem ""
            MSFlexGrid1.TextMatrix(d, 0) = (Form1.stock.Fields("stockcode"))
            MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
            MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
            MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
            MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
            d = d + 1
        End If
        Form1.stock.MoveNext
    Wend
End If
End Sub

```

```

Private Sub Command1_Click()
Form13.comingform = 14
Form13.List1.clear
Form13.List2.clear
Form13.List3.clear
Form13.List4.clear
Form13.List5.clear
Form13.List6.clear

```

```

Form13.List7.clear
Form13.List8.clear
Form1.customer.MoveFirst
While Not Form1.customer.EOF
Form13.List1.AddItem (Form1.customer.Fields("cuscode"))
Form13.List2.AddItem (Form1.customer.Fields("name"))
Form13.List3.AddItem (Form1.customer.Fields("surname"))
Form13.List4.AddItem (Form1.customer.Fields("company"))
If IsNull(Form1.customer.Fields("tel")) = False Then
Form13.List5.AddItem (Form1.customer.Fields("tel"))
Else
Form13.List5.AddItem ""
End If
Form13.List6.AddItem (Form1.customer.Fields("mobile"))
If IsNull(Form1.customer.Fields("address")) = False Then
Form13.List7.AddItem (Form1.customer.Fields("address"))
Else
Form13.List7.AddItem " "
End If
If IsNull(Form1.customer.Fields("email")) = False Then
Form13.List8.AddItem (Form1.customer.Fields("email"))
Else
Form13.List8.AddItem " "
End If
Form1.customer.MoveNext
Wend
Form2.Enabled = False
Form13.Show
End Sub

```

```

Private Sub Command2_Click()
Form13.comingform = 14
On Error Resume Next
If Option4.value = True Then
If Text1.Text = "" Then
MsgBox ("please write a code")
Else
Form1.customer.Index = "primarykey"
Form1.customer.Seek "=", Text1.Text
If Form1.customer.NoMatch = False Then
Label1 = Form1.customer.Fields("cuscode")
Label3 = Form1.customer.Fields("name") + " " + Form1.customer.Fields("surname")
Label4 = Form1.customer.Fields("company")
customercode = Text1
End If
End If
End If
If Option3.value = True Then
If Text1.Text = "" Then
MsgBox ("please write a name")
Else

```

```

Form1.customer.MoveFirst
While Not Form1.customer.EOF
    If LCase(Text1.Text) = LCase(Left(Form1.customer.Fields("name"),
Len(Text1.Text))) Then
        Form2.Enabled = False
        Form13.Show
        Form13.List1.AddItem Form1.customer.Fields("cuscode")
        Form13.List2.AddItem Form1.customer.Fields("name")
        Form13.List3.AddItem Form1.customer.Fields("surname")
        Form13.List4.AddItem Form1.customer.Fields("company")
        If IsNull(Form1.customer.Fields("tel")) = False Then
            Form13.List5.AddItem (Form1.customer.Fields("tel"))
        Else
            Form13.List5.AddItem " "
        End If
        Form13.List6.AddItem Form1.customer.Fields("mobile")
        If IsNull(Form1.customer.Fields("address")) = False Then
            Form13.List7.AddItem Form1.customer.Fields("address")
        Else
            Form13.List7.AddItem " "
        End If
        If IsNull(Form1.customer.Fields("email")) = False Then
            Form13.List8.AddItem Form1.customer.Fields("email")
        Else
            Form13.List8.AddItem " "
        End If
    End If
    Form1.customer.MoveNext
Wend
End If
End If
Text1.Text = ""
End Sub

```

```

Private Sub Command3_Click()
Dim totalrefund
totalrefund = 0
With MSFlexGrid2
If .Rows > 1 Then
    If Option2.value = True Then
        For i = 1 To .Rows - 1
            Form1.stock.Index = "primarykey"
            Form1.stock.Seek "=", .TextMatrix(i, 0)
            If Form1.stock.NoMatch = False Then
                Form1.stock.Edit
                Form1.stock.Fields("amount") = Form1.stock.Fields("amount") + .TextMatrix(i,
3)
                Form1.stock.Update
            End If
            totalrefund = totalrefund + .TextMatrix(i, 4)
        Next
    End If
End With
End Sub

```



```

MsgBox ("Total Price to refund : " + Format(totalrefund, "#.##"))
Form2.Show
Unload Me
End If
If Option1.value = True Then
    If Label1.Caption <> "" Then
        For i = 1 To .Rows - 1
            totalrefund = totalrefund + .TextMatrix(i, 4)
            Form1.stock.Index = "primarykey"
            Form1.stock.Seek "=", .TextMatrix(i, 0)
            If Form1.stock.NoMatch = False Then
                Form1.stock.Edit
                Form1.stock.Fields("amount") = Form1.stock.Fields("amount") +
.TextMatrix(i, 3)
                Form1.stock.Update
            End If
        Next
        Form1.payment.AddNew
        Form1.payment.Fields("cuscode") = customercode
        Form1.payment.Fields("date") = Date
        Form1.payment.Fields("total") = totalrefund
        Form1.payment.Fields("username") = Form1.username
        Form1.payment.Update
        MsgBox ("Total refunds : " + Format(totalrefund, "#.##") + " added to customer
payment...")
        Form2.Show
        Unload Me
    Else
        MsgBox ("Find customer")
    End If
End If
Else
MsgBox ("Select refund items..!")
End If
End With
End Sub

```

```

Private Sub Command4_Click()
Dim d As Long
Check2.value = 0
Call clear1
Form1.stock.Index = "primarykey"
Form1.stock.Seek "=", Text14.Text
If Text14.Text = "" Then
MsgBox ("please Write a Stock Code")
Call clear1
Else
If Form1.stock.NoMatch <> 0 Then
MsgBox ("Database don't have this Stock Code")
Text14.Text = ""
Else

```

```

d = 1
MSFlexGrid1.AddItem ""
MSFlexGrid1.TextMatrix(d, 0) = Text14.Text
MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
d = d + 1
End If
End If
End Sub

```

```

Private Sub Command6_Click()
Dim d As Long
Combo1.clear
Check2.value = 0
d = 1
Form1.stock.MoveFirst
While Not Form1.stock.EOF
MSFlexGrid1.AddItem ""
MSFlexGrid1.TextMatrix(d, 0) = (Form1.stock.Fields("stockcode"))
MSFlexGrid1.TextMatrix(d, 1) = (Form1.stock.Fields("stkname"))
MSFlexGrid1.TextMatrix(d, 2) = (Form1.stock.Fields("stktype"))
MSFlexGrid1.TextMatrix(d, 3) = (Form1.stock.Fields("amount"))
MSFlexGrid1.TextMatrix(d, 4) = (Form1.stock.Fields("saleprice"))
d = d + 1
Form1.stock.MoveNext
Wend
End Sub

```

```

Private Sub Form_Load()
Option4.value = True
Label1.Caption = ""
Label2.Caption = ""
Label3.Caption = ""
Label4.Caption = ""
Form13.comingform = 14
Option2.value = True
If Option2.value = True Then
Frame1.Visible = False
Image3.Visible = True
End If
Call clear1
Call clear2
Text1.Text = ""
Text14.Text = ""
End Sub

```

```

Private Sub Form_Unload(Cancel As Integer)
Form2.Show
End Sub

```

```

Private Sub Label1_Change()
If Label1.Caption <> "" Then

```

```

On Error Resume Next
Image1.Picture = LoadPicture(Form1.programyer + "\picture\" + Label1.Caption +
".jpg")
End If
End Sub

```

```

Private Sub MSFlexGrid1_Click()
On Error Resume Next
MSFlexGrid1.Col = 0
MSFlexGrid1.ColSel = 4
End Sub

```

```

Private Sub MSFlexGrid1_DblClick()
If MSFlexGrid1.TextMatrix(MSFlexGrid1.Row, 0) <> "" Then
Form6.MSFlexGrid1.TextMatrix(1, 0) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
0)
Form6.MSFlexGrid1.TextMatrix(1, 1) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
1)
Form6.MSFlexGrid1.TextMatrix(1, 2) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
2)
Form6.MSFlexGrid1.TextMatrix(1, 3) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
3)
Form6.MSFlexGrid1.TextMatrix(1, 4) = MSFlexGrid1.TextMatrix(MSFlexGrid1.Row,
4)
Form6.form6gelenform = 14
Form6.Show
Else
MsgBox ("You Can't Make a Blank Row Selection")
End If
End Sub

```

```

Private Sub MSFlexGrid2_Click()
On Error Resume Next
MSFlexGrid2.Col = 0
MSFlexGrid2.ColSel = 4
End Sub

```

```

Private Sub MSFlexGrid2_DblClick()
Form6.Show
End Sub

```

```

Private Sub Option1_Click()
If Option1.value = True Then
Frame1.Visible = True
Image3.Visible = False
End If
End Sub

```

```

Private Sub Option2_Click()
If Option2.value = True Then
Frame1.Visible = False
Image3.Visible = True
End If
End Sub

```