NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

HOSPITAL AUTOMATION PROGRAM

Graduation Project
COM-400

Student : Recep KÜTÜK

Supervisor : Mr.Ümit İLHAN

Nicosia-2002

ACKNOWLEDGEMENTS

I am very greatfull to those who have given me the courge to finish the project that my the teacher which I have looked upon gave me to complete.

My wonderful teacher Mr **Ümit İLHAN** I can not explain how much this project has tought me and how much usefull it will be in my life. I thank you very much for assigning me for the project.

I can not thank enough my family for the things they had to put up with until they bought me to the age that I am. Especially during the time of time of my life in the faculty.

To all my friends I don't know how much and how to say thank you for making me what I am.

Mean while I am a very forgetful person I would like to send a special thanks to everyone which has helped me with the project and in life.

ABSTRACT

The hospital automation program will supply the user efficient usimg of the time time and patient information. After very simple installation of the program and later starting form is opened as the splash form. Later the main menu is opened. One screen is displayed. There are time and date components are on the main menu.

The user record the new patient to the new patient registiration form. If there are a patient in the hospital then user go to patient search menu and search the patient according to patient's surname and name. User find the patient's protocol no and search the protocol no and other information of the patient. Program is included very much details of the hospital and patient's details. Also when the user bored, the user go to the tools menu and listen the music or draw a figure on the paintbrush.

TABLE OF CONTENTS

INTRODUCTION

What is Delphi?

Delphi5 uses a 'visual' programming paradigm with ObjectPascal [a derivate of Pascal, with support for object oriented programming] as its underlying language. This means you use the mouse to design the look of your application - arranging 'components' (buttons, scroll bars, menus etc) on skeleton windows ('forms'). This leaves you with a series of files (ObjectPascal source code, forms definitions and resources) to which you add the code which makes your application carry out its function. The attributes of each component (size, position, font of lettering etc) are specified interactively at design time, or can be modified dynamically by the program.

There is no need to master the complexities of Windows programming, as the components act as an interface between your program and the operating system.

Borland Delphi is a sophisticated Windows programming environment, suitable for beginners and professional programmers alike. Using Delphi you can easily create self-contained, user friendly, highly efficient Windows applications in a very short time - with a minimum of manual coding.

Delphi provides all the tools you need to develop, test and deploy Windows applications, including a large number of so-called reusable components. Borland Delphi, in it's latest version, provides a cross platform solution when used with Borland Kylix - Borland's RAD tool for the Linux platform.

Delphi's roots lie in Borland's Turbo Pascal, introduced in the mid-1980s. Object Pascal, the object-oriented extensions to Pascal, is the underlying language of Delphi. The Visual Component Library, or VCL, is a hierarchy of Object Pascal objects that allow you to design applications. A better way of describing Delphi is an Object Pascal-based visual development environment

Most business applications need to store information in a database, and software development therefore has two elements: constructing a database to support the business process, and developing an application to access the database. Choice of database should be determined on a project by project basis, as should choice of interface. Therefore, if a Windows client application is the preferred user interface, we call in our Delphi Centre of Excellence. We believe that Delphi is the best development tool for Windows currently available, and have emphasised our skills and belief in the product by the creation of our Centre of Excellence.

Delphi is an incredibly versatile and easy to use development tool. It is more than just a programming language, because it has additional features that reduce development costs and improve productivity. For example, the user interface is so good that it makes developers much more productive than some others.

Delphi's versatilitity means that it genuinely can be used for most types of project: from real time control to Enterprise-wide multi-tier database systems. It is designed from the ground up to provide database functionality; we prefer to use it with Microsoft Access for low end projects, Microsoft SQL Server for middle range tasks and Oracle for high end projects. We are not tied to these, however, because any ODBC or OLE-DB database can be used. This means that we can work with clients' existing database infrastructure, protecting your investment, while developing first class front ends.

Delphi is fully object-oriented, and we tend to design most projects using components to improve reusability and maintainability. We avoid supplying bug-ridden software through thorough testing but prefer our development environment to support us: Delphi uses Object Pascal which helps our developers to write robust code, because Pascal is strongly-typed for example.

The main alternatives to Delphi are Visual Basic and Powerbuilder. Powerbuilder suffers from poor take-up, while VB suffers because it is *still* not fully object-oriented, even version 6, whereas Delphi has been from version 1. Delphi does not require that any DLLs are distributed with compiled applications, unlike VB.

CHAPTER-1

INSTALLATION OF THE PROGRAM

When the hospital program setup the run,first step is to insert the CD into the CDROM. If the computer is set tostart automatically the CD will start to install. If not start it,then the user must open hospital from the CD manually. Once this process is done the installation will begin. Installation procedures are completed in five steps and these are as follows.

1.1 First Step

The screen below gives the details of the licence for the program and tell us about warning of licence concerning(as **Figure 1-1).**

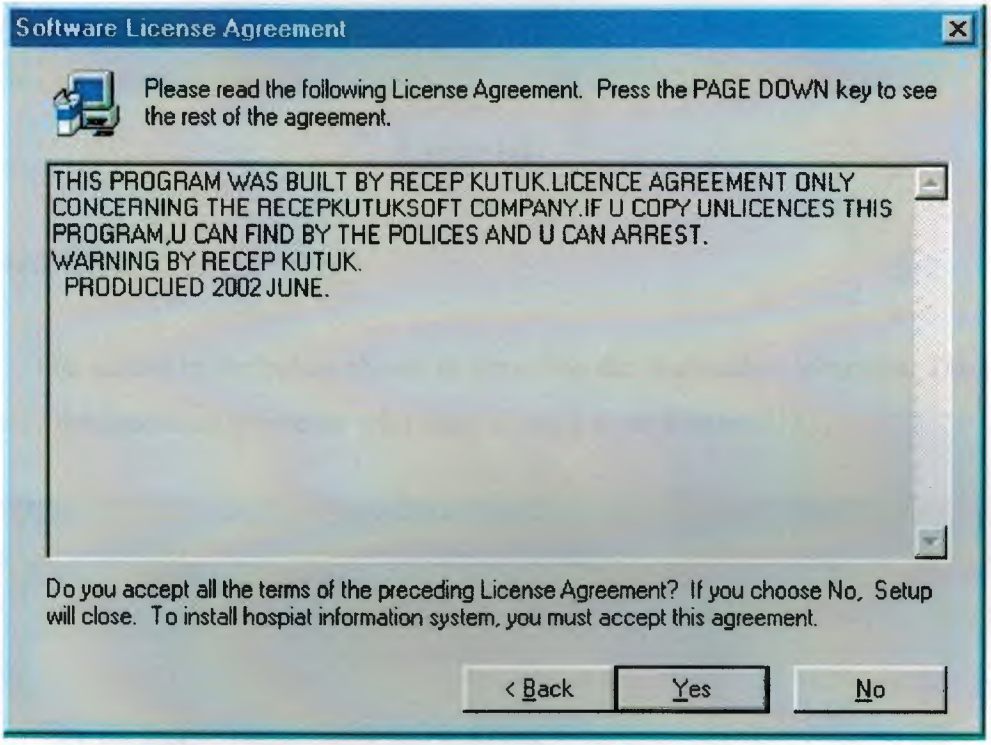

Figure 1-1

1.2 Second Step

The registration of the new user is completed in the screen shown below. You enter the user name in the first edit box, you enter the your company's name to the second edit box(as **Figure 1-2)..**

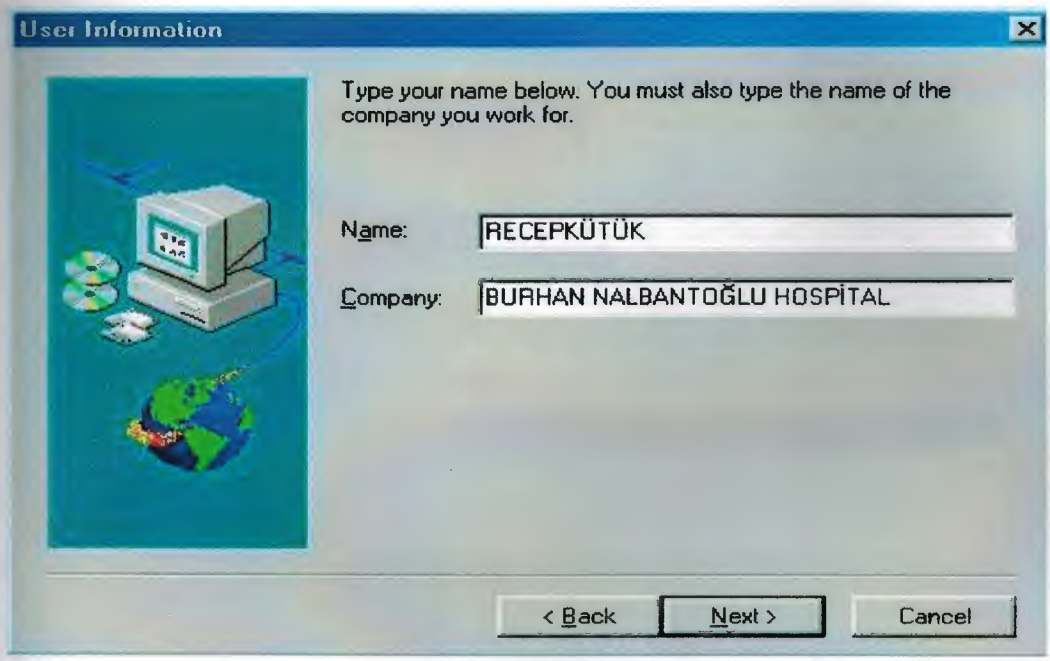

Figure 1-2.

1.3 Third Step

The screen in the below shows us choosing the destination locations. The user can specify the locatioun wherever who want to copy it (as **Figure 1-3).**.

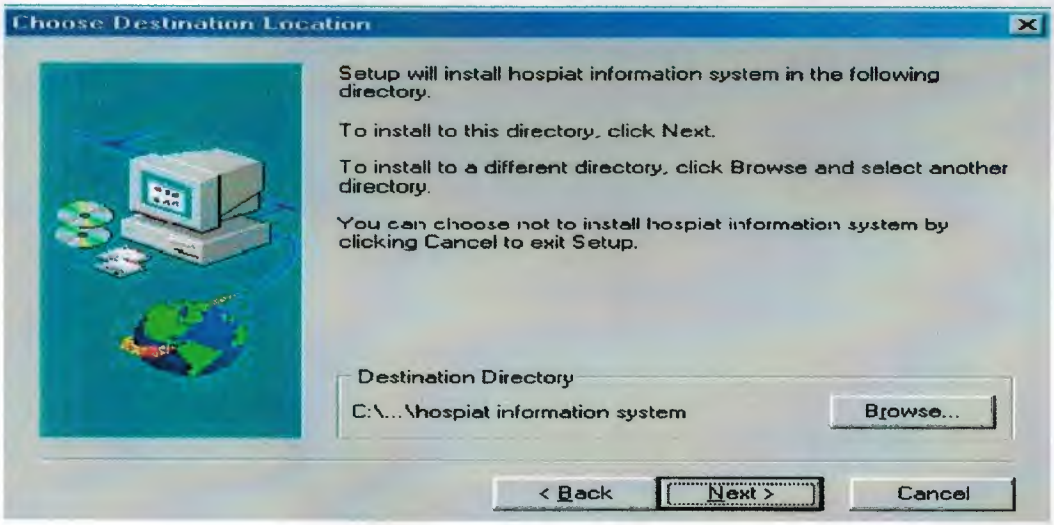

Figure 1-3

1.4 Fourth Step

This steps the setup is select the setup folder in the staring space(as **Fig 1-4).**

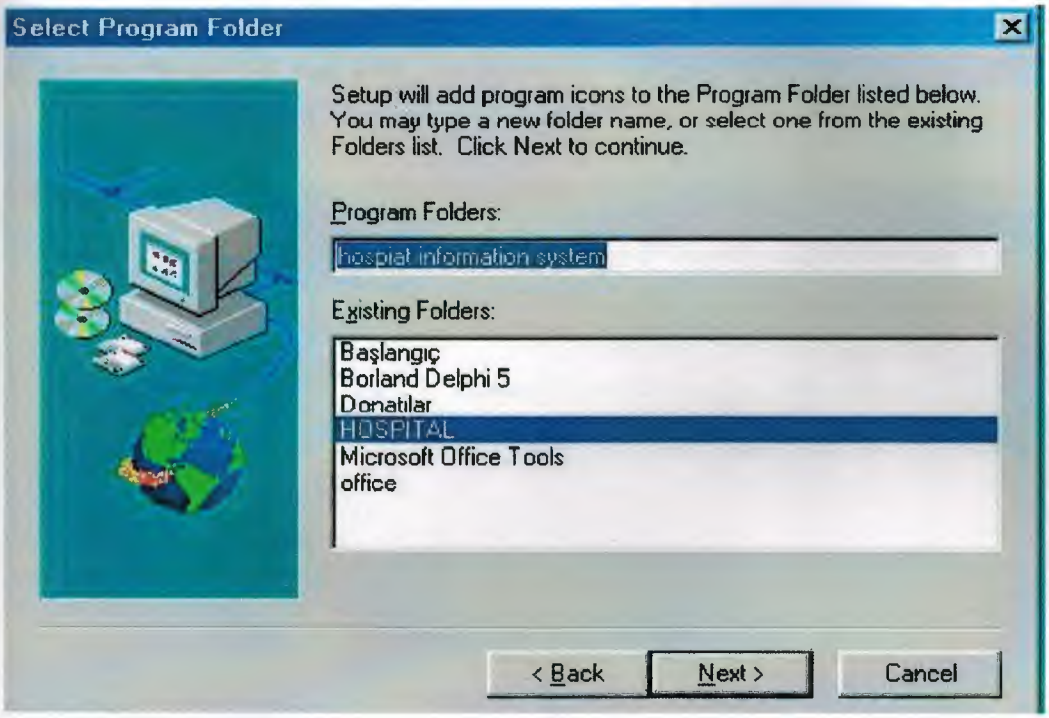

Figure1-4

1.5 Fifth Step

The setup is preparing the files the copying to harddiscks(as **Figure 1-5**).

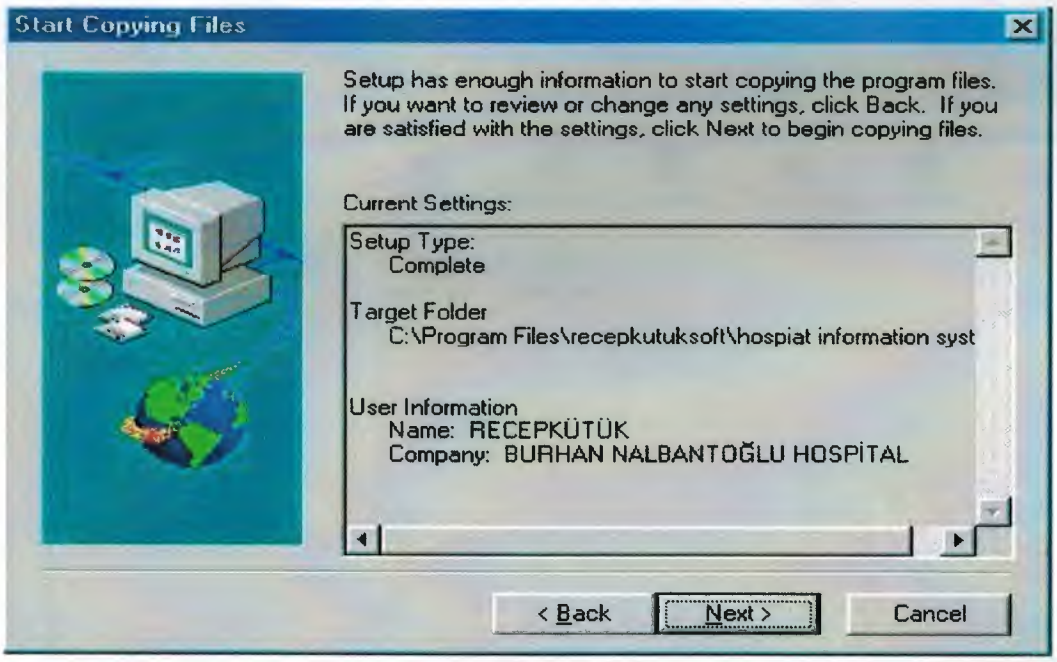

Figure 1-5

1.6 Sixth Step

Now setup is finished and when u clicked finish you can open the hospital folders(as **Figure 1-6 and 1-7).**

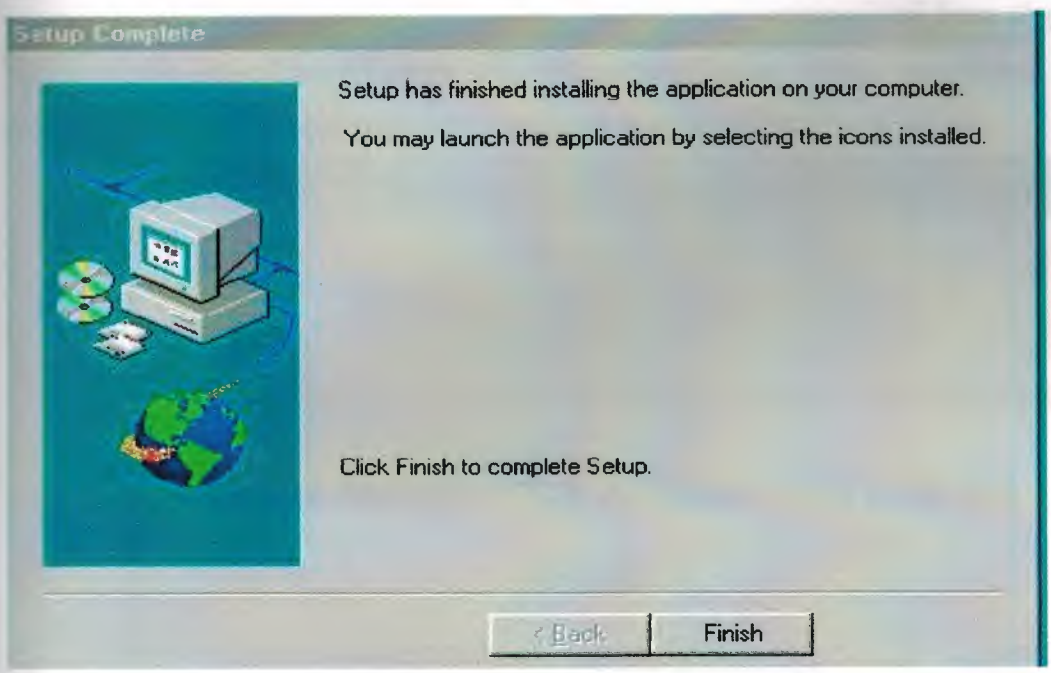

Figure 1-6.

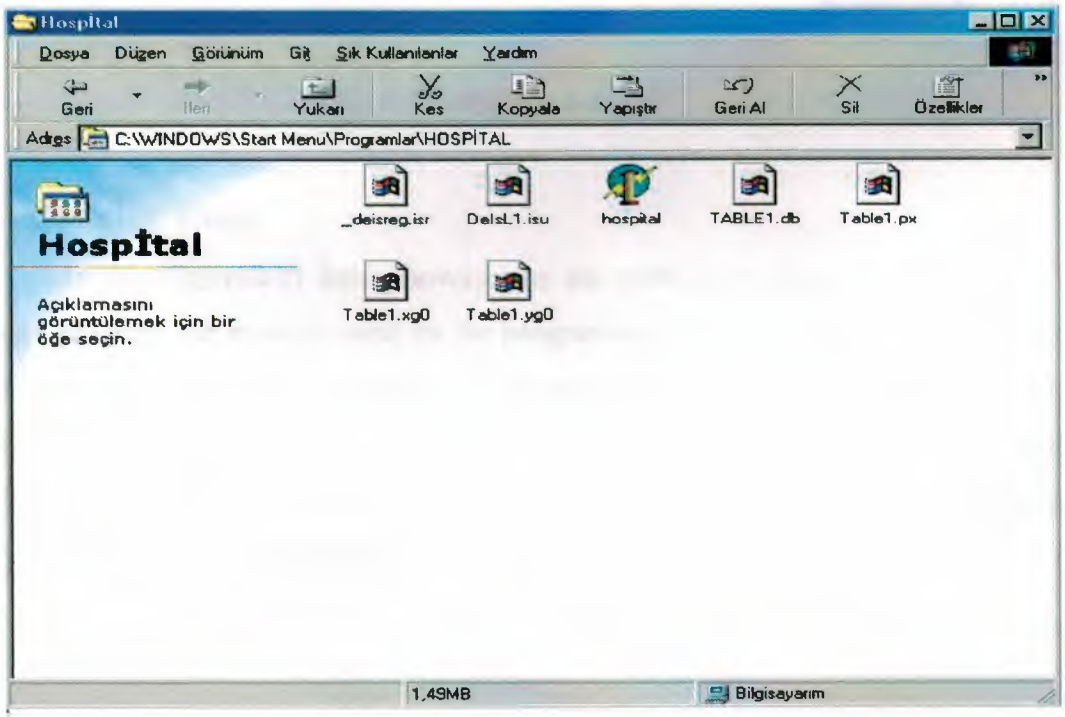

Figure1-7

CHAPTER-2

SCREENS

2.1 Starting Part

In this form(**Figure2-1**) is starting form of the program.Thif form display us name of the project,lecture name and programmer name. When we clicked the go buttons,we will go to next page. The next page is splash form.

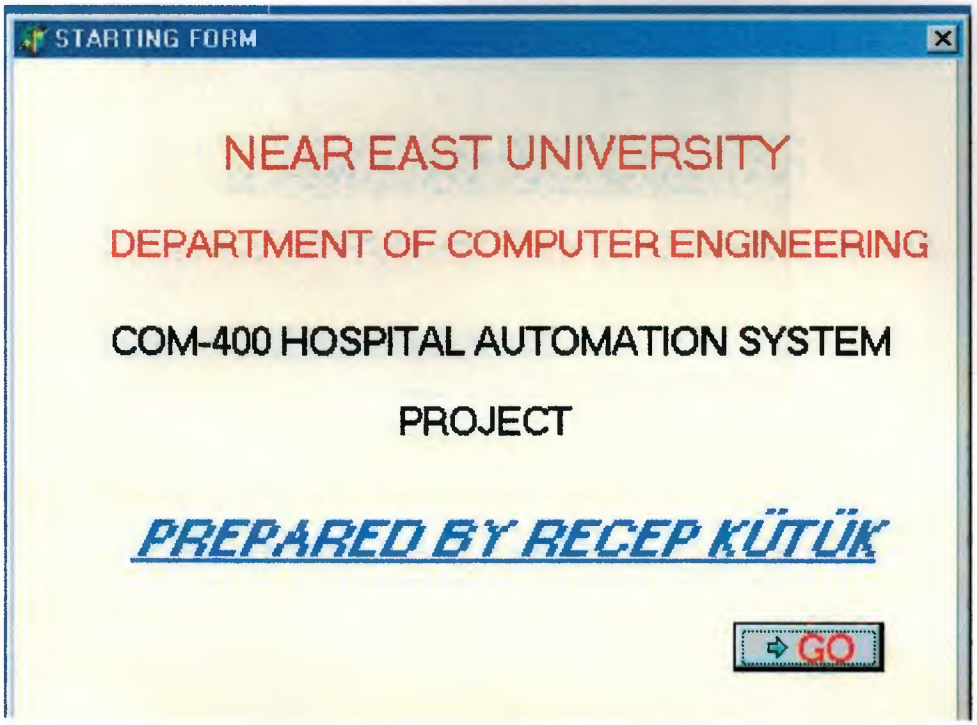

Figure 2-1

2-2 Opening menu

In this(**Figure2-2**) form shows us as the splah form opening of program. Tha gauge progress bar menu is used by the programmer. This splash ramdom is designed randomize. When the gauge egual to %100 splash is closed and mainform is opened. I used the gauge progress bar component in this program. Because, when we first looked the program,program is displaying us nicely and most important is shows us opening of the program. The form(**Figure2-2**) is in the other page.

Figure2-2

2.2.1 Delphi Opening Menu form Source codes

```
type
Form2 = class(TForm)
Panel1: TPanel;
Label1: TLabel;
Label2: TLabel;
Timer1: TTimer;
Gauge1: TGauge;
Label3: TLabel;
Image1: TImage;
Label4: TLabel;
procedure Timer1Timer(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure FormCreate(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
```

```
end;
var
Form2: TForm2;
implementation
uses Unit3;
{$R *.DFM}
procedure TForm2.Timer1Timer(Sender: TObject);
begin
randomize;
gauge1.progress:=gauge1.progress + random(23);
if gauge1.progress >= gauge1.MaxValue then
begin
mainform.show;
form2.close;
end;
end;
procedure TForm2.FormClose(Sender: TObject; var Action: TCloseAction);
begin
free;
end;
```

2.3 Main Form

The main form of the program includes four main menu files. The first one is patient information. The patient information has two submenus. First one is new patient. When the newpatient is clicked,it goes the for a new patient registeration form. The other submenu of the patient information is change the patientid. The change the patientrecords menu is clicked by the user and come to patientid form.

The secondpart of the main menu is doctor parts. Doctor part is includes three submenu. The first one is daily medicine. The daily medicine shows the necessary drinking medicine for a patient. The second submenu is patient analyzing. This shows the information about patient analysis,labarotory test and etc.... The third submenu of the docotor menu is patient report. The patient report is shows the report documentation of the patient.

The third menu is the search menu. It has one submenu patientsearch. The patient search is the find the registering patient.

The fourth menu is the exit. The aim of the exit menu is the exiting of the programs. When the user clicked he exit menu. Program is automatically will be closed.

There are a figure of the KKTC MINISTIRY OF THE HEALTH on the main form. There are time and date box on the mainform.

The mainform is shows the below the menu at **Figure2-3**

Figure2-3

2.4 New Patient Registiration

The **Figure 2.4** below shows the details of a new patient records. First user clicked the patient information and later enter the patient identity and later click new patient and show the below the **Figure2.4**

Figure2-4

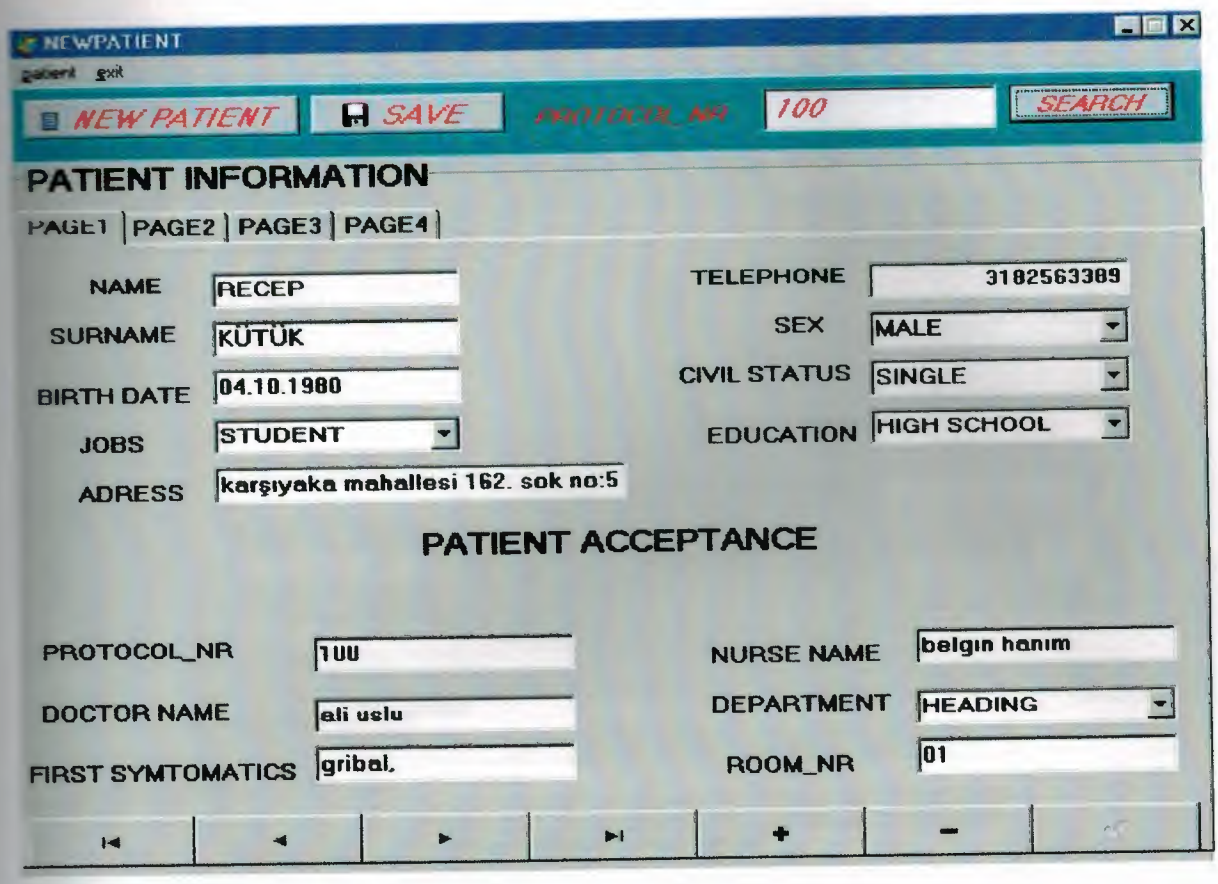

Figure2-5

The leads the user to the above form which details of the new patients records are entered or old records can deleted. New patient form is also search the patient protocolno. The tabsheets are used the new patient form. There are four pages on the tab sheets. First page is includes the patient name,surname,birthdate,sex,job,education,civil status,address,new protocolno,doctor name,first sysmtomatics,nurse name,patient department and room nr. The program user is entered the according to their particulary value.

There are save, new patient buttons and patient protocolno searching area on the top of the page. The save button is saved the patient records according to patient's protocol no. When the user want to add new patient then user can click the newpatient button and can add the new patient information. The same protocol no never used to same protocol no for the patient. And so the user could search the patient protocol no.

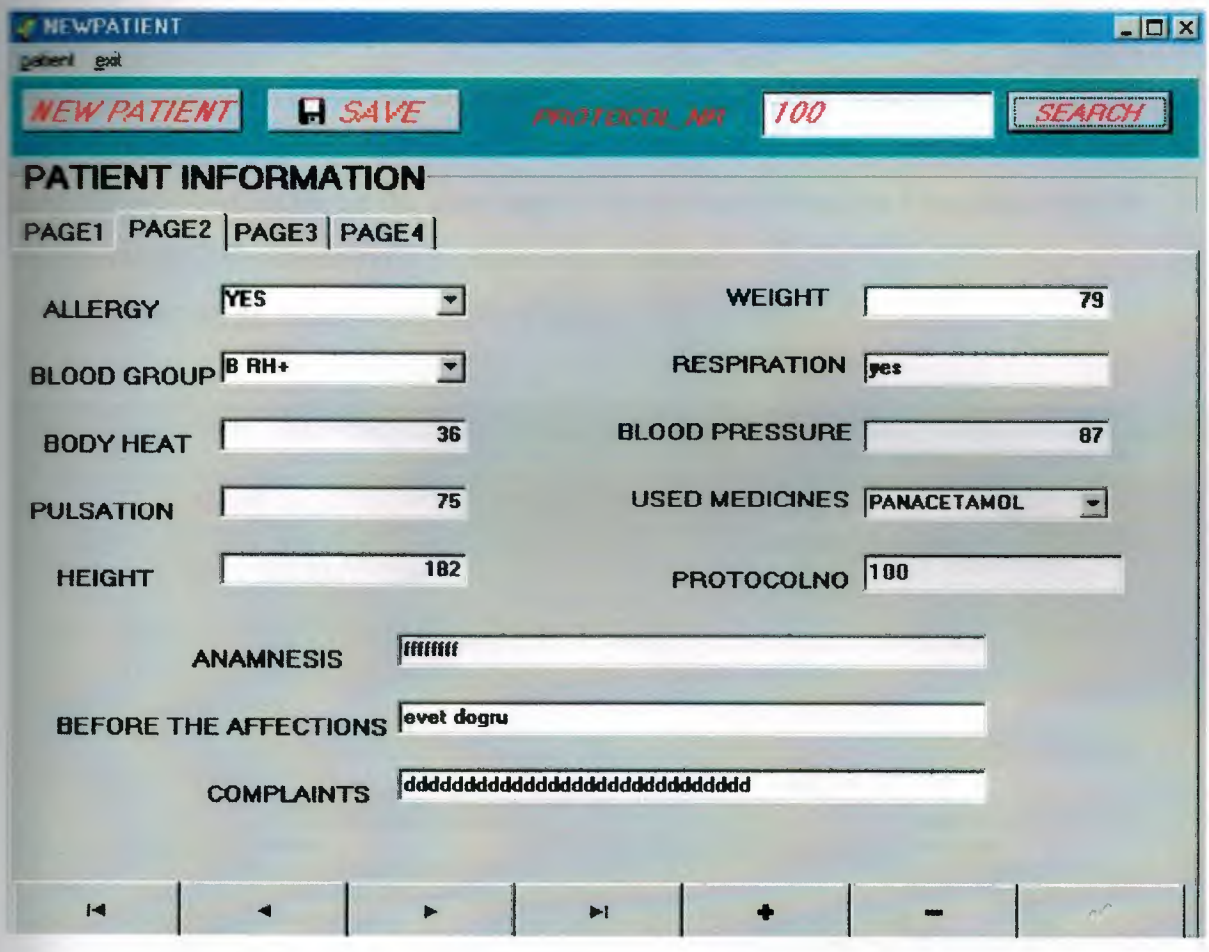

Figure2-6

Later go to the 2nd page of the tabsheets. The second page of the tabshhets menu are includes details of the patients. The page 2 is shows the above the form. Allergy part is add to database patient allery as yes or no.

The user can add the blood group, can add the body heat,pulsation,height,patient weight,patient blood pressure,medicines,anamnesis(it means that patient whose have the history of health or operations),include before the affections and includes complaints parts.

The user or doctor can change the this records the over their clinical tests results. Of course these data are add to patient whose have same protocol no. I demonstrated it according to protocol no is on the all pages.

Then we go to third page.

The below the figure.... are includes the healthly details of the patient. Add the prosthesis contolleable,hearing tools,reactions,add the mother tongue of the patient, ache,acheplace, cougher,heard pulse,last job declaration,smoking,drug,hiv,alchol and surgicial intervention are on the third page of the tabsheets form.The user add,control or change the patient records to their clinical specification.

There are all includes below the **Figure 2.7**

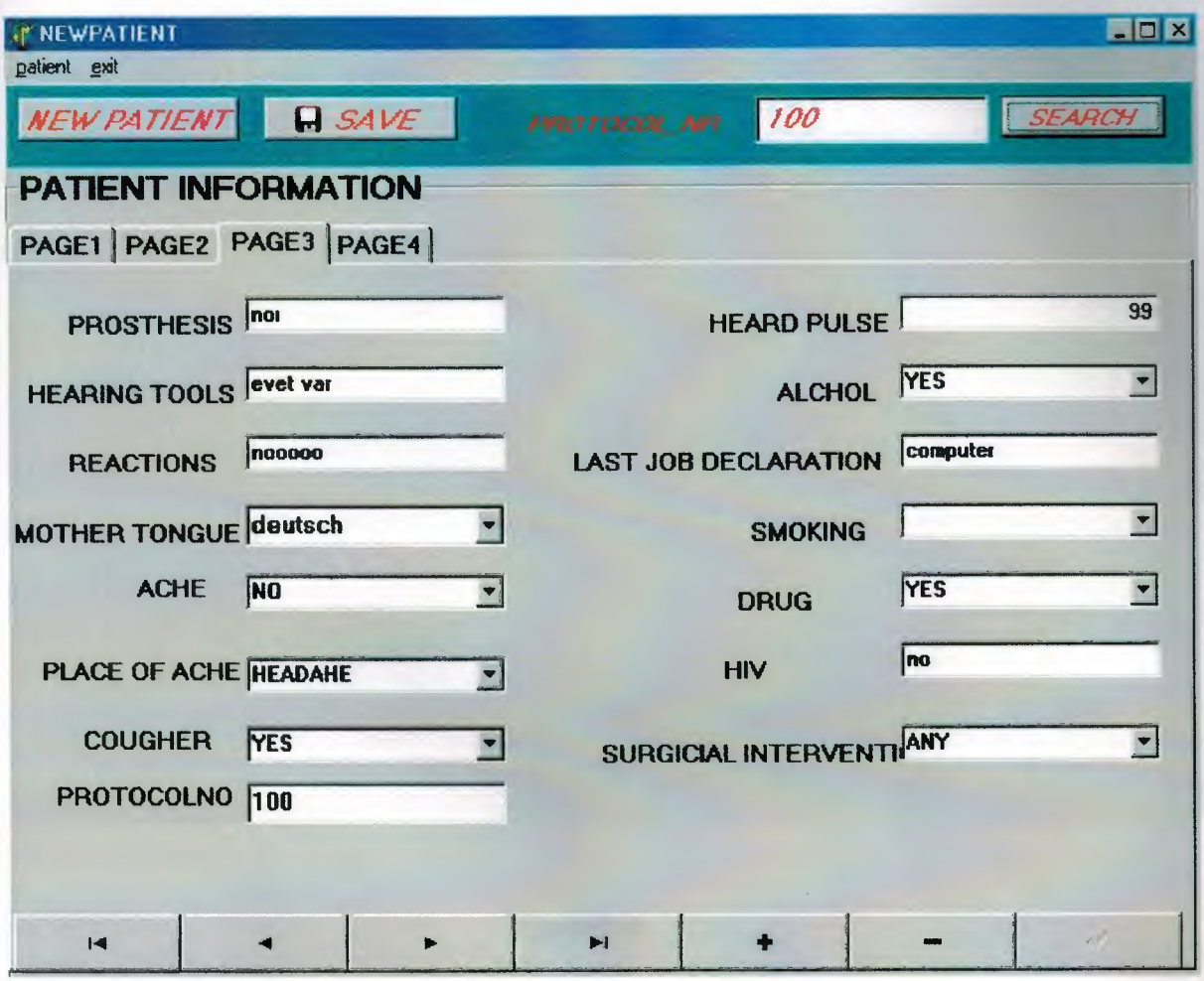

Figure2-7

The page four is include the patient room recors. Must the patient room have the Tv-vcd,smoking,caller service,telephone and 24 hrs caller center. The below the figure is control the these criterias.

And later user can save the all recors the patient.

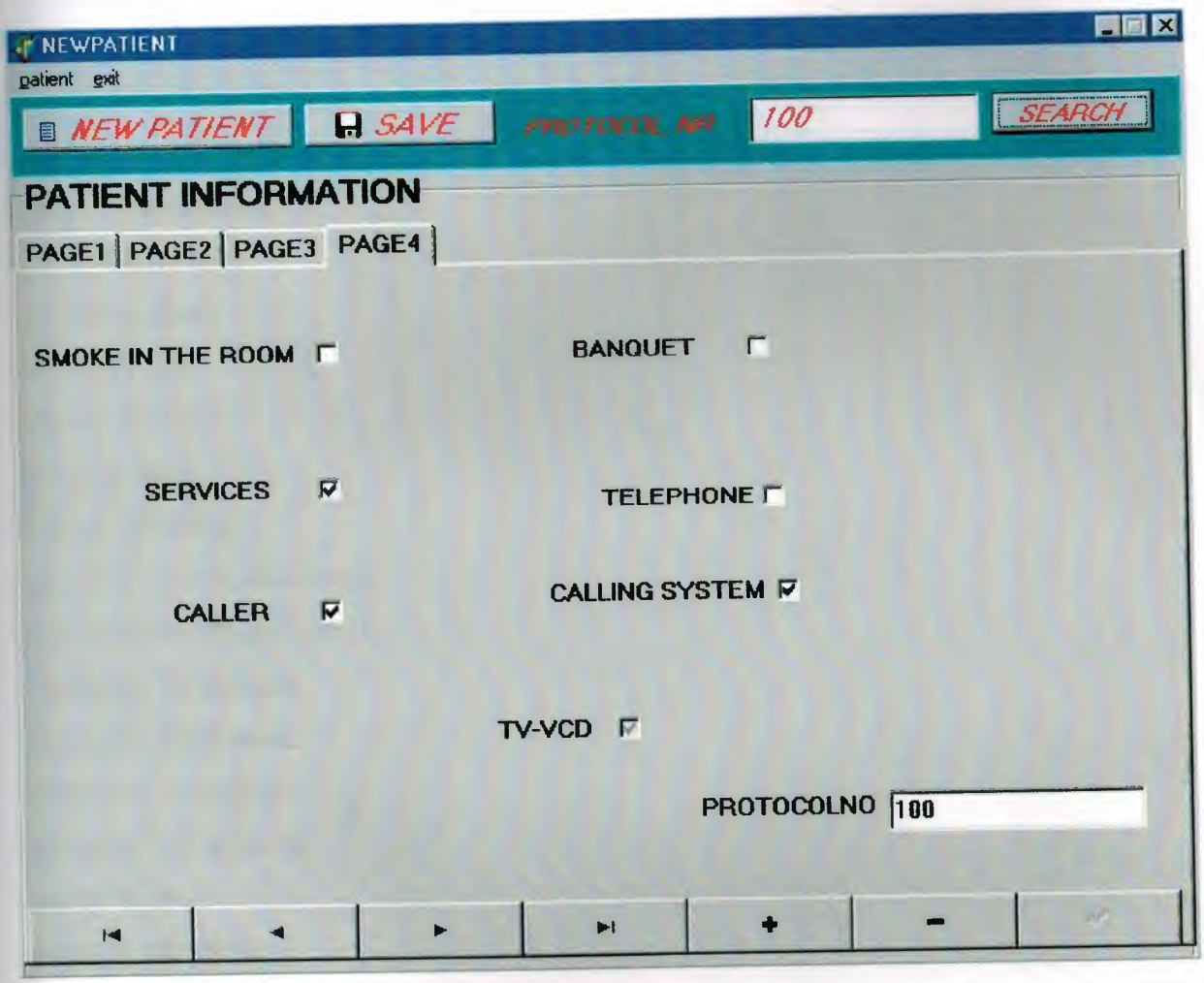

Figure2-8

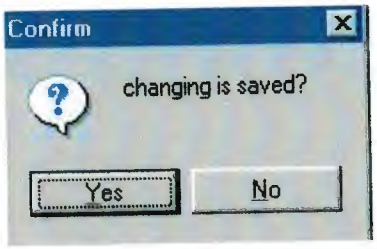

Figure2-9

2.4.1 Delphi Source Codes of New Patient Registiration

```
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, DBCtrls, ComCtrls, ExtCtrls, Mask, Buttons, Menus, Db, DBTables;
type
Tnewpatient = class(TForm)
MainMenu1: TMainMenu;
patient1: TMenuItem;
save1: TMenuItem;
exit: TMenuItem;
Panel1: TPanel;
BitBtn1: TBitBtn;
Label1: TLabel;
BitBtn5: TBitBtn;
RadioGroup1: TRadioGroup;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
TabSheet2: TTabSheet;
TabSheet3: TTabSheet;
TabSheet4: TTabSheet;
Label2: TLabel;
DBEdit2: TDBEdit;
Label3: TLabel;
DBEdit3: TDBEdit;
Label4: TLabel;
DBEdit4: TDBEdit;
Label5: TLabel;
DBComboBox1: TDBComboBox;
Label6: TLabel;
DBEdit5: TDBEdit;
Label7: TLabel;
DBEdit6: TDBEdit;
```

```
Label8: TLabel;
DBComboBox2: TDBComboBox;
Label9: TLabel;
DBComboBox3: TDBComboBox;
Label10: TLabel;
DBComboBox4: TDBComboBox;
Label11: TLabel;
Label15: TLabel;
Label16: TLabel;
Label21: TLabel;
DBEdit7: TDBEdit;
DBEdit8: TDBEdit;
DBEdit10: TDBEdit;
DBEdit9: TDBEdit;
DBComboBox5: TDBComboBox;
DBEdit11: TDBEdit;
Label13: TLabel;
Label23: TLabel;
Label24: TLabel;
Label12: TLabel;
Label14: TLabel;
Label17: TLabel;
Label18: TLabel;
Label19: TLabel;
DBComboBox6: TDBComboBox;
Label20: TLabel;
DBComboBox7: TDBComboBox;
Label22: TLabel;
DBEdit12: TDBEdit;
Label25: TLabel;
DBEdit13: TDBEdit;
DBEdit14: TDBEdit;
DBEdit15: TDBEdit;
DBEdit17: TDBEdit;
```

DBEdit18: TDBEdit;

DBEdit19: TDBEdit;

DBComboBox9: TDBComboBox;

Label26: TLabel;

Label27: TLabel;

Label28: TLabel;

Label29: TLabel;

Label30: TLabel;

Label31: TLabel;

Label32: TLabel;

DBEdit20: TDBEdit;

Label33: TLabel;

DBEdit21: TDBEdit;

DBEdit22: TDBEdit;

DBEdit23: TDBEdit;

DBComboBox10: TDBComboBox;

DBComboBox11: TDBComboBox;

DBComboBox12: TDBComboBox;

Label34: TLabel;

Label35: TLabel;

Label36: TLabel;

Label37: TLabel;

Label39: TLabel;

Label40: TLabel;

Label41: TLabel;

DBEdit25: TDBEdit;

Label42: TLabel;

DBComboBox13: TDBComboBox;

Label43: TLabel;

Label44: TLabel;

Label45: TLabel;

Label46: TLabel;

Label47: TLabel;

DBEdit26: TDBEdit;

```
DBComboBox14: TDBComboBox;
DBComboBox15: TDBComboBox;
DBEdit27: TDBEdit;
DBComboBox16: TDBComboBox;
Label48: TLabel;
Label49: TLabel;
Label50: TLabel;
Label51: TLabel;
Label52: TLabel;
Label53: TLabel;
Label54: TLabel;
Label55: TLabel;
DBNavigator2: TDBNavigator;
DataSource1: TDataSource;
Table1: TTable;
Table2: TTable;
DataSource2: TDataSource;
Table3: TTable;
DataSource3: TDataSource;
Table1SURNAME: TStringField;
Table1NAME: TStringField;
Table1TELEPHONE: TFloatField;
Table1JOB: TStringField;
Table1SEX: TStringField;
Table1ADRESS: TStringField;
Table1CITY: TStringField;
Table1BIRTHPLACE: TStringField;
Table1BIRTHDATE: TDateField;
Table1CIVILSTATUS: TStringField;
Table1EDUCATION: TStringField;
Table1DOCTORNAME: TStringField;
Table1NURSENAME: TStringField;
Table2SYMTOMATICS: TStringField;
Table2DEPARTMENT: TStringField;
```

```
Table1ROOMNR: TStringField;
Table1PROTOCOLNO: TStringField;
Table2PROTOCOLNO: TStringField;
Table2HELPERTOOLS: TStringField;
Table2BLOODGROUP: TStringField;
Table2HEARINGTOOLS: TStringField;
Table2ALLERGY: TStringField;
Table2HEIGHT: TFloatField;
Table2WEIGHT: TFloatField;
Table2BODYHEAT: TFloatField;
Table2ESTIMATION: TStringField;
Table2PULSATION: TFloatField;
Table2BLOODPRESSURE: TFloatField;
Table2OTHERILLNESS: TStringField;
Table2BEFOREMEDICINE: TStringField;
Table2ACHE: TStringField;
Table2ACHEPLACE: TStringField;
Table2SMOKE: TStringField;
Table2ALCHOL: TStringField;
Table2HEARDPULSATION: TFloatField;
Table3PROTOCOLNO: TStringField;
Table3SMOKE: TBooleanField;
Table3TELEPHONE: TBooleanField;
Table3SERVICES: TBooleanField;
Table3EATING: TBooleanField;
Table3TV_VCD: TBooleanField;
Table3CALLINGSYSTEM: TBooleanField;
Table3SPECIALROOM: TBooleanField;
Table2RESPIRATION: TStringField;
Table2ANAMNESIS: TStringField;
Table2COMPLAINT: TStringField;
Table2BEFOREAFFECTION: TStringField;
Table2PROSTHESIS: TStringField;
Table2REACTIONS: TStringField;
```

```
Table2MOTHERTONGUE: TStringField;
Table2COUGHER: TStringField;
Table2LASTJOBDECLARATION: TStringField;
Table2DRUG: TStringField;
Table2HIV: TStringField;
Table2SURGICIALINTERVENTION: TStringField;
DBCheckBox1: TDBCheckBox;
DBCheckBox2: TDBCheckBox;
DBCheckBox3: TDBCheckBox;
DBCheckBox4: TDBCheckBox;
DBCheckBox5: TDBCheckBox;
DBCheckBox6: TDBCheckBox;
DBCheckBox7: TDBCheckBox;
newpatient1: TMenuItem;
BitBtn6: TBitBtn;
DBComboBox17: TDBComboBox;
Edit2: TEdit;
DBEdit1: TDBEdit;
Label38: TLabel;
DBEdit24: TDBEdit;
Label56: TLabel;
DBEdit28: TDBEdit;
Label57: TLabel;
Table1LABARATORY: TStringField;
Table1CASE: TStringField;
Table1CERTAINTY: TStringField;
Table1CLINICALTEST: TStringField;
Table1COMMITTING: TStringField;
DBComboBox8: TDBComboBox;
procedure exitClick(Sender: TObject);
procedure DBNavigator2Click(Sender: TObject; Button: TNavigateBtn);
procedure BitBtn1Click(Sender: TObject);
procedure BitBtn5Click(Sender: TObject);
procedure BitBtn6Click(Sender: TObject);
```

```
procedure newpatient1Click(Sender: TObject);
procedure save1Click(Sender: TObject);
procedure TabSheet4ContextPopup(Sender: TObject; MousePos: TPoint;
var Handled: Boolean);
private
{ Private declarations }
public
{ Public declarations }
end;
var
newpatient: Tnewpatient;
implementation
{$R *.DFM}
procedure Tnewpatient.exitClick(Sender: TObject);
begin
NEWPATIENT.CLOSE;
end;
procedure Tnewpatient.DBNavigator2Click(Sender: TObject;
Button: TNavigateBtn);
begin
table2protocolno:=table1protocolno;
table3protocolno:=table1protocolno;
dbedit7.Text:=Table1PROTOCOLNO.AsString;
end;
procedure Tnewpatient.BitBtn1Click(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
then begin
```

```
table1.post;
end else
abort;
end;
end;
procedure Tnewpatient.BitBtn5Click(Sender: TObject);
begin
if (edit2.text<>'') then begin
table1.open;
table2.open;
table3.open;
if not table1.FindKey([edit2.text]) then begin
table1.close;
table2.close;
table3.close;
messagedlg('PATIENT WAS NOT FOUND',mterror,[mbok],0);
end;
end
else begin
showmessage('ENTER THE PROTOCOL NO');
abort;
end;
end;
procedure Tnewpatient.BitBtn6Click(Sender: TObject);
begin
TABLE1.APPEND;
TABLE2.APPEND;
TABLE3.APPEND;
end;
procedure Tnewpatient.newpatient1Click(Sender: TObject);
begin
TABLE1.APPEND;
TABLE2.APPEND;
TABLE3.APPEND;
```

```
end;
procedure Tnewpatient.save1Click(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
then begin
table1.post;
end else
abort;
end;
end;
```

2.5 Change the patientid records

The below the **Figure 2.10** is show the recording patientidentity records. Patient identity forms is shows the patient name,surname,adress,city,phonenr,education and civil status. If the user wants to change the concerning the patient identity records the user can change from this screen.

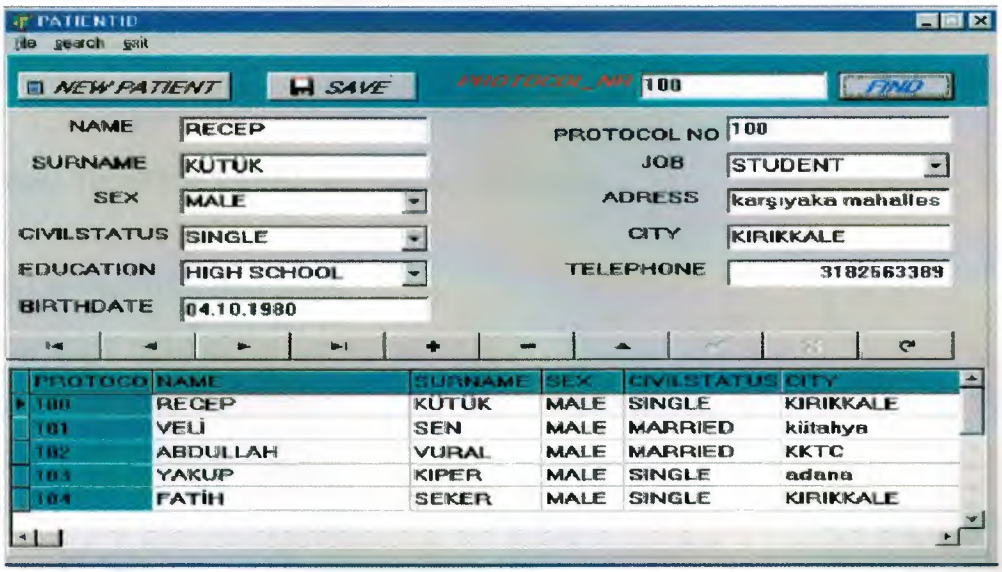

Figure2-10

2.5.1 Delphi a part of the source codes of the patientid

```
procedure Tpatientid.FINDClick(Sender: TObject);
begin
if (edit2.text<>'') then begin
table1.open;
if not table1.FindKey([edit2.text]) then begin
table1.close;
messagedlg('PATIENT WAS NOT FOUND',mterror,[mbok],0);
end;
end
else begin
showmessage('ENTER THE PROTOCO NO');
abort;
end;
END;
procedure Tpatientid.patient1Click(Sender: TObject);
begin
form9.table1.open;
FORM9.SHOW;
end;
procedure Tpatientid.BitBtn6Click(Sender: TObject);
begin
TABLE1.APPEND;
end;
procedure Tpatientid.SAVEClick(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
table1.post;
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
```

```
then begin
end else
abort;
end;
end;
procedure Tpatientid.save1Click(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
then begin
end else
abort;
end;
end;
procedure Tpatientid.exit1Click(Sender: TObject);
begin
patientid.close;
end;
procedure Tpatientid.newpatient1Click(Sender: TObject);
begin
TABLE1.APPEND;
end;
```

2.6 Daily medicine

In now, the user goto doctor menu and later click the daily medicine sub menu and then enter the daily medicine. The daily medicine is includes to must taken the medicine lists and information about registeration patient record. Name,surname etc…

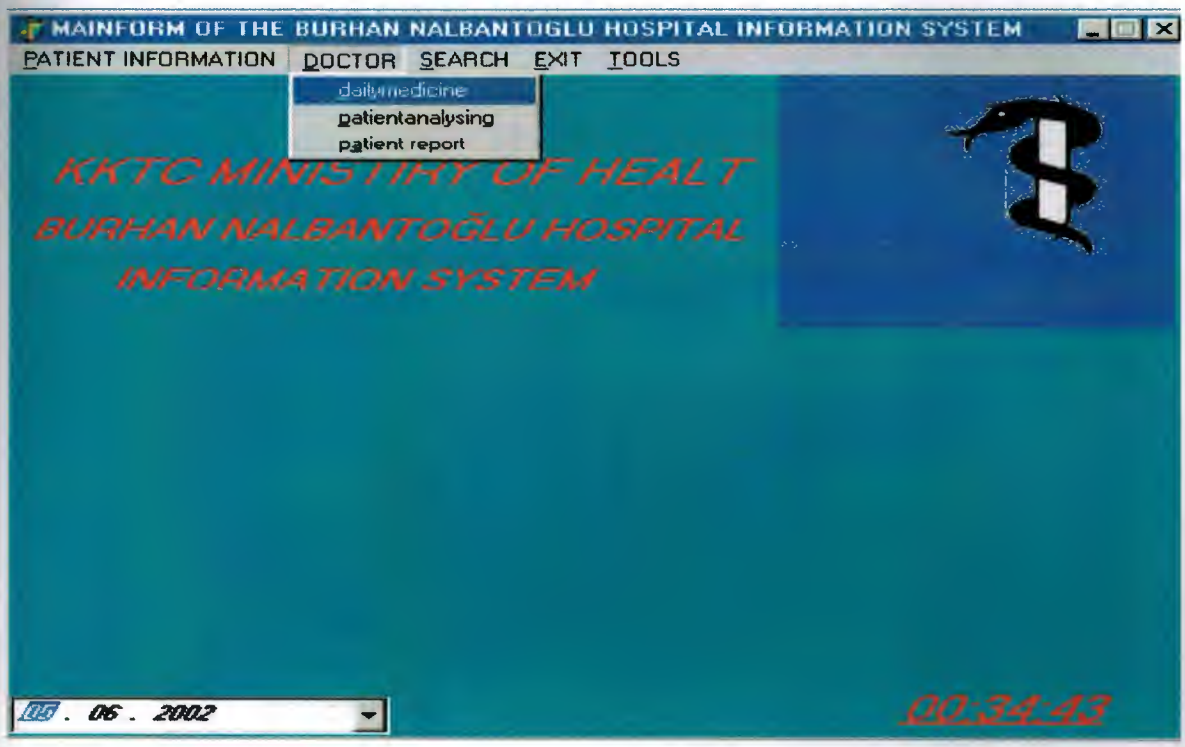

Figure2.11

Figure2.12

In this form we observe the patient medicine patient medicine records. When we click the preview button, then patient report will be open before the print

out(as the **Figure2.13**). First the user search the patient according to patient's protocolno then the patient's name,patient's surname and protocol no is shows on the menu. Later the user can enter the patient's medicine name, medicine drinking ways,medicine drink time interval and medicine drink total dose are included on the daily medicines menu.

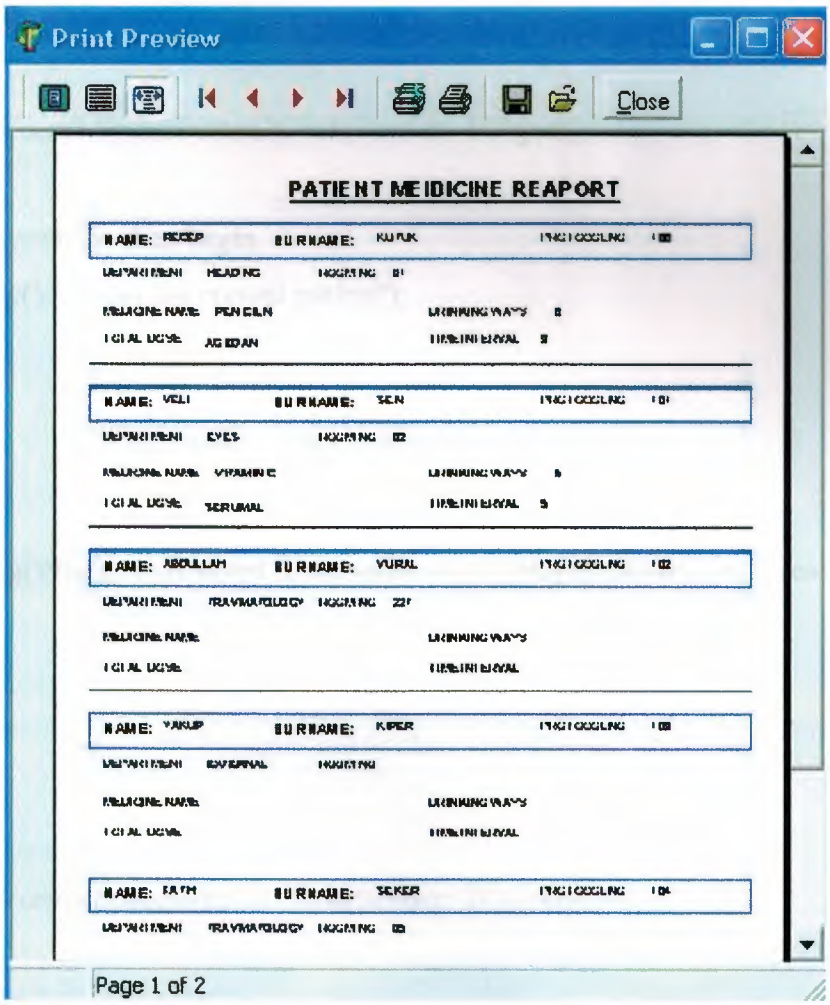

Figure2.13

2.6.1 Delphi Source Codes of the Daily Medicine Menu

```
procedure TForm8.Button1Click(Sender: TObject);
begin
if (edit2.text<>'') then begin
table1.open;
```

```
if not table1.FindKey([edit2.text]) then begin
table1.close;
messagedlg('Aranan Kişi Bulunamadı',mterror,[mbok],0);
end;
end
else begin
showmessage('protokol no giriniz');
abort;
end;
end;
procedure TForm8.SpeedButton3Click(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
table1.post;
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
then begin
end else
abort;
end;
end;
procedure TForm8.SpeedButton1Click(Sender: TObject);
begin
table1.append;
end;
procedure TForm8.SpeedButton6Click(Sender: TObject);
begin
form7.table1.open;
FORM7.QuickRep1.Preview;
form7.table1.close;
end;
```

```
procedure TForm8.SpeedButton7Click(Sender: TObject);
begin
form7.table1.open;
FORM7.QuickRep1.Print;
form7.table1.Close;
end;
procedure TForm8.Table1BeforePost(DataSet: TDataSet);
begin
if (DBEdit5.text='') or (DBedit6.Text='') or (DBedit7.Text='')
or (DBComboBox2.Text='') then begin
MessageDlg('Boş ilaç alanlarını giriniz!', mtWarning, [mbOK], 0);
abort;
end;
end;
procedure TForm8.k1Click(Sender: TObject);
begin
form8.close;
end;
procedure TForm8.HastaArama1Click(Sender: TObject);
begin
form9.table1.open;
FORM9.SHOW;
end;
procedure TForm8.nizleme1Click(Sender: TObject);
begin
form7.table1.open;
FORM7.QuickRep1.Preview;
form7.table1.close;
end;
procedure TForm8.Yazdr1Click(Sender: TObject);
begin
form7.table1.open;
FORM7.QuickRep1.Print;
form7.table1.Close;
```

```
end;
procedure TForm8.YeniKayt1Click(Sender: TObject);
begin
table1.append;
end;
procedure TForm8.Kaydet1Click(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
table1.post;
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
then begin
end else
abort;
end;
end;
```

2.7 Patient Analysing Menu

In this form is included the patient name,surname,protocolno,patient birthdate,department,clinical test result,rontgen result ana labarotory tests results. The user is search the patient according to patient's protocolno and later user can change or add the patient analysing the results as the technical health results. This form is shows below the **Figure 2.14.**

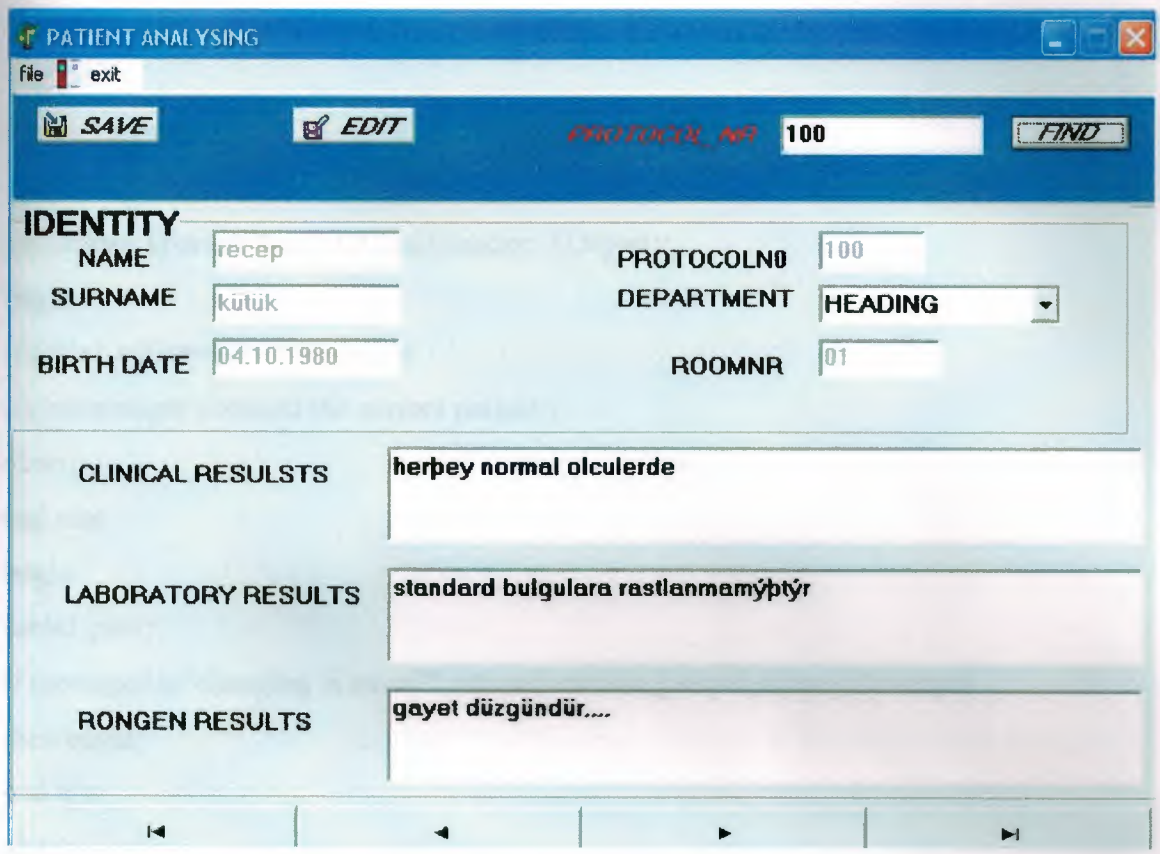

Figure 2.14.

2.7.1 Delphi Source Codes of the Patient Analysing Form

```
procedure TForm6.SpeedButton3Click(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
table1.post;
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
then begin
end else
abort;
END;
end;
```

```
procedure TForm6.SpeedButton5Click(Sender: TObject);
begin
table1.append;
end;
procedure TForm6.Kaydet1Click(Sender: TObject);
begin
if table1.active=false then begin
showmessage('choosed the correct patient');
abort;
end else
begin
table1.post;
if messagedlg('changing is saved?',mtconfirmation,[mbyes,mbno],0)=mryes
then begin
end else
abort;
end;
end;
procedure TForm6.KaytDzelt1Click(Sender: TObject);
begin
table1.append;
end;
procedure TForm6.Button1Click(Sender: TObject);
begin
if (edit1.text<>'') then begin
table1.open;
if not table1.FindKey([edit1.text]) then begin
table1.close;
messagedlg('Aranan Kişi Bulunamadı',mterror,[mbok],0);
end;
end
else begin
showmessage('protokol no giriniz');
abort;
```

```
end;
end;
procedure TForm6.DBNavigator1Click(Sender: TObject; Button: TNavigateBtn);
begin
dbedit3.Text:=Table1PROTOCOLNO.AsString;
end;
procedure TForm6.k1Click(Sender: TObject);
begin
form6.close;
end;
```

2.8 The Patient Search Form

The patient search form is included the searching part according to patient surname or name. When user enter the surname or name of the patient automatically,if the patient is in the recording in the hospital will be show. As the **Figure2.15.**

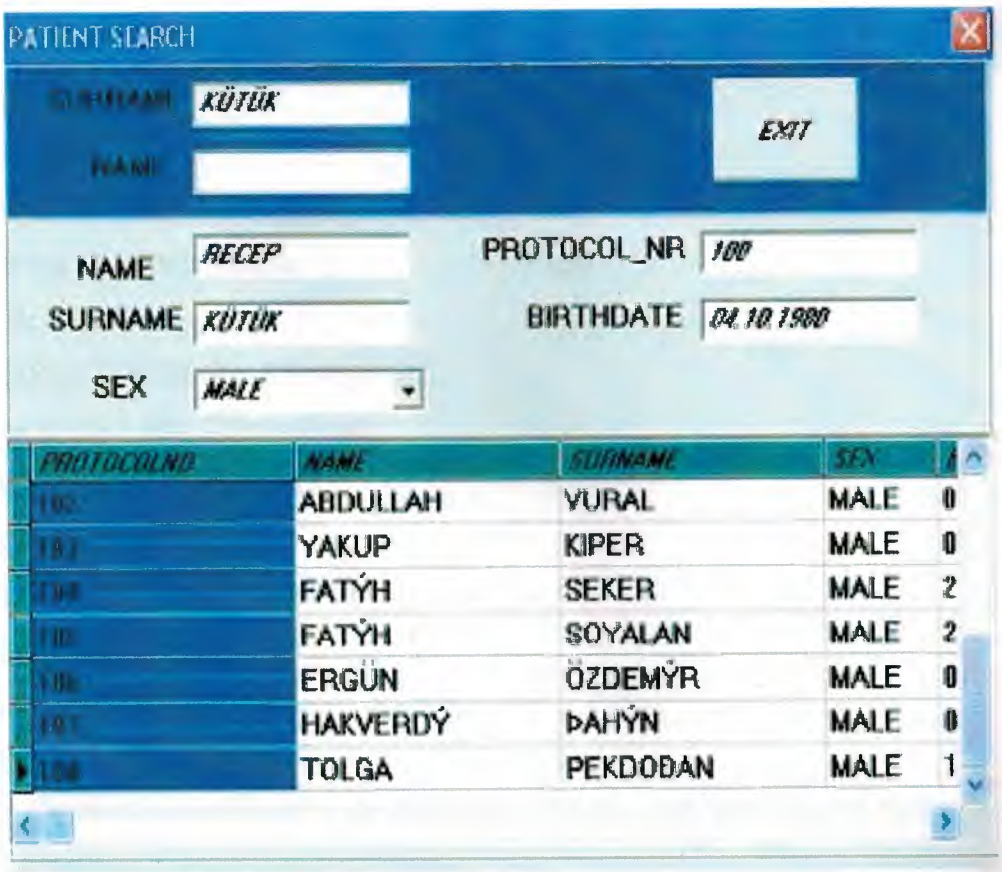

Figure2.15

2.8.1 Delphi Source Codes of the Patient Search Menu

```
unit Unit9;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
Grids, DBGrids, StdCtrls, Buttons, ExtCtrls, DBTables, Db, Menus,
ComCtrls, DBCtrls, Mask;
type
TForm9 = class(TForm)
Panel3: TPanel;
SpeedButton1: TSpeedButton;
Panel2: TPanel;
Label1: TLabel;
Label2: TLabel;
Label4: TLabel;
Label6: TLabel;
Label8: TLabel;
Panel1: TPanel;
DBGrid1: TDBGrid;
DataSource1: TDataSource;
Table1: TTable;
Table1PROTOCOLNO: TStringField;
Table1SURNAME: TStringField;
Table1NAME: TStringField;
Table1SEX: TStringField;
Table1BIRTHDATE: TDateField;
Edit1: TEdit;
Label3: TLabel;
Label5: TLabel;
Edit2: TEdit;
Edit4: TEdit;
Edit5: TEdit;
ComboBox1: TComboBox;
```

```
Edit6: TEdit;
Edit7: TEdit;
procedure exit1Click(Sender: TObject);
procedure Edit1Change(Sender: TObject);
procedure Edit2Change(Sender: TObject);
procedure SpeedButton1Click(Sender: TObject);
procedure Panel3Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form9: TForm9;
implementation
{$R *.DFM}
procedure TForm9.exit1Click(Sender: TObject);
begin
form9.close;
end;
procedure TForm9.Edit1Change(Sender: TObject);
begin
Table1.First;
While not Table1.Eof Do
Begin
If (Table1name.Text=Edit1.Text) then
Begin
Edit4.Text:=Table1name.Text;
edit5.text:=table1surname.text;
edit6.text:=table1protocolno.text;
edit7.text:=table1birthdate.text;
combobox1.text:=table1sex.text;
end;
table1.next;
```

```
end;
end;
procedure TForm9.Edit2Change(Sender: TObject);
begin
Table1.First;
While not Table1.Eof Do
Begin
If (Table1surname.Text=Edit2.Text) then
Begin
Edit4.Text:=Table1name.Text;
edit5.text:=table1surname.text;
edit6.text:=table1protocolno.text;
edit7.text:=table1birthdate.text;
combobox1.text:=table1sex.text;
end;
table1.next;
end;
end;
procedure TForm9.SpeedButton1Click(Sender: TObject);
begin
FORM9.CLOSE;
end;
```

CHAPTER-3

DATABASES

3.1 What is the Paradox?

The Paradox standard table format was introduced in Paradox for DOS version 4. Other products that use the standard format include Paradox for DOS version 4.5, ObjectVision 2.1, and Paradox for Windows versions 1.0 and 4.5.

Earlier versions of the Paradox table type are referred to as the Compatible table type. In the BDE Configuration Utility, the level option for the Paradox driver dictates what default table type is created by Paradox for Windows. Use 3 for Compatible tables, 4 for Standard tables (the default). Following are the specifications for standard Paradox tables.

- 256MB file size limit if the table is in Paradox format and using a 4K block size.
- Up to 255 fields per record.
- Up to 64 validity checks per table.
- A primary index can have up to 16 fields.
- Tables can have up to 127 secondary indexes.
- Up to two billion records per file. Because of the 256MB file size limit and other factors such as block size, however, the limit is much smaller. Tables of 190,000 records are easily achievable (and you can have more if you don't use up the 1,350-bytes-per-record limit for a keyed table). Tables with close to a million records are common.
- Block size can be 1024, 2048, 3072, or 4096. Paradox stores data in fixed records. Even if part or all of the record is empty, the space is claimed. Knowing the interworkings can save you disk space. Paradox stores records in fixed blocks of1024 2048, 3072, 4096 in size.

  After a block size is set for a table, that size is fixed, and all blocks in the table will be of that size. To conserve disk space, you want to try to get your record size as close to a multiple of block size as possible (minus 6 bytes, which are used by Paradox to manage the table).
- Record size. 1,350 for keyed tables and 4,000 for unkeyed tables. When figuring out the size (the number of bytes or characters) of a table, remember that Alpha fields take up their size (for example, an A10 = 10

bytes), numeric field types take up 8 bytes, short number field types take up 2 bytes, money takes up 8, and dates take up 4 bytes.

- Memos, BLOBs, and so on take 10 bytes plus however much of the memo is stored in the .DB. For example, M15 takes 25 bytes.
-

3.2 Paradox5 Table Specifications

The Paradox 5 table format was introduced in Paradox for Windows version 5. Following are the specifications for Paradox 5 tables.

- Up to two billion records per file.
- File size is limited to two gigabytes.
- Up to 255 fields per record.
- Record size: Up to 10,800 bytes per record for indexed tables and 32,750 bytes per record for nonindexed tables. When figuring out the size (the number of bytes or characters) of a table, remember that Alpha fields take up their size (for example, an A10 = 10 bytes), numeric field types take up 8 bytes, short number field types take up 2 bytes, money takes up 8, and dates take up 4 bytes.
- Memos, BLOBs, and so on take 10 bytes plus however much of the memo is stored in the .DB. For example, M15 takes 25 bytes.
- Up to 64 validity checks per table for Paradox for Windows tables.
- A primary index can have up to 16 fields.
- Tables can have up to 127 secondary indexes.
- Block size can be from 1K to 32K in steps of 1K. For example, 1024, 2048, 3072, 4096, 5120...32768.

3.3 Paradox 7 and above Table Specifications

The Paradox 7 table format was introduced in Paradox version 7 for Windows 95/NT. The Paradox 7 table format has all the same specifications as the Paradox 5 table format with two additions. Following are the specification additions for the Paradox 7 table format.

- Added descending secondary indexes.
- Added unique secondary indexes

3.4 Paradox Field Types

3.4.1Alpha (A)

Paradox 3.5, 4, 5, and 7 field type that can contain up to 255 letters and numbers. This field type was called Alphanumeric in versions of Paradox before version 5. It is similar to the Character field type in dBASE.

3.4.2 Autoincrement (+)

Field type introduced in the Paradox 5 table format that adds one to the highest number in the table whenever a record is inserted. The starting range can from -2,147,483,647 to 2,147,483,647. Deleting a record does not change the field values of other records.

3.4.3 BCD (#)

Paradox 5 and 7 field type which is provided only for compatibility with other applications that use BCD data. Paradox correctly interprets BCD data from other applications that use the BCD type. When Paradox performs calculations on BCD data, it converts the data to the numeric float type, then converts the result back to BCD. When this field type is fully supported, it will support up to 32 significant digits.

3.4.4 Binary (B)

Paradox 1, 5, and 7 field type that can store binary data up to 256MB per field.

3.4.5 Bytes (Y)

Paradox 5 and 7 field type for storing binary data up to 255 bytes. Unlike binary fields, bytes fields are stored in the Paradox table (rather than in the separate .MB file), allowing for faster access.

3.4.6 Date (D)

Paradox 3.5, 4, 5, and 7 as well as dBASE III+, IV, and V. dBASE tables can store dates from January 1, 100, to December 31, 9999. Paradox 5 tables can store from 12/31/9999 B.C. to 12/31/9999 A.D.

3.4.7 Formatted Memo (F)

Paradox 1, 4.5, 5, and 7 field type is like a memo field except that you can format the text. You can alter and store the text attributes of typeface, style, color, and size. This rich text document has a variable-length up to 256MB per field.

3.4.8 Graphic (G)

Paradox 1, 5, and 7 field type can contain pictures in .BMP (up to 24 bit), .TIF (up to 256 color), .GIF (up to 256 color), .PCX, and .EPS file formats. Not all graphic variations are available. For example, currently you cannot store a 24-bit .TIF graphic. When you paste a graphic into a graphic field, Paradox converts the graphic into the .BMP format.

3.4.9 Logical (L)

Paradox 5 and 7 and dBASE III+, IV, and V field type can store values representing True or False (yes or no). By default, valid entries include T and F (case is not important).

3.4.10 Memo (M)

Paradox 4, 5, and 7 as well as dBASE III+, IV, and V field. A Paradox field type is an Alpha variable-length field up to 256MB per field. dBASE Memo fields can contain binary as well as memo data.

For Paradox tables, the file is divided into blocks of 512 characters. Each block is referenced by a sequential number, beginning at zero. Block 0 begins with a 4-byte number in hexadecimal format, in which the least significant byte comes first. This number specifies the number of the next available block. It is, in effect, a pointer to the end of the memo file. The remainder of Block 0 isn't used.

3.4.11 Money ($)

Paradox 3.5, 4, 5, and 7 field type, like number fields, can contain only numbers. They can hold positive or negative values. Paradox recognizes up to six decimal places when performing internal calculations on money fields. This field type was called Currency in previous versions of Paradox.

3.4.12 OLE (O)

Paradox 1, 5, and 7 as well as dBASE V field type that can store OLE data.

3.4.13 Number (N)

Paradox 3.5, 4, 5, and 7 as well as dBASE III+, IV, and V field type can store up to 15 significant digits -10307 to + 10308 with up to 15 significant digits.

dBASE number fields contain numeric data in a Binary Coded Decimal (BCD) format. Use number fields when you need to perform precise calculations on the field data. Calculations on number fields are performed more slowly but with

greater precision than are calculations on float number fields. The size of a dBASE number field can be from 1 to 20. Remember, however, that BCD is in Paradox 5 and 7 only for compatibility and is mapped directly to the Number field type.

3.4.14 Short (S)

Paradox 3.5, 4, 5, and 7 field type that can contain integers from -- 32,767 through 32,767 (no decimal).

3.4.15 Time (T)

Paradox 5 and 7 field type that can contain time times of day, stored in milliseconds since midnight and limited to 24 hours.

This field type does not store duration which is the difference between two times. For example, if you need to store the duration of a song, use an Alpha field. Whenever you need to store time, make a distinction between clock time and duration. The Time field type is perfect for clock time. Duration can be stored in an Alpha field and manipulated with code.

3.4.16 TimeStamp (@)

Paradox 5 field type comprised of both date and time values. Rules for this field type are the same as those for date fields and time fields.

3.5 Tables

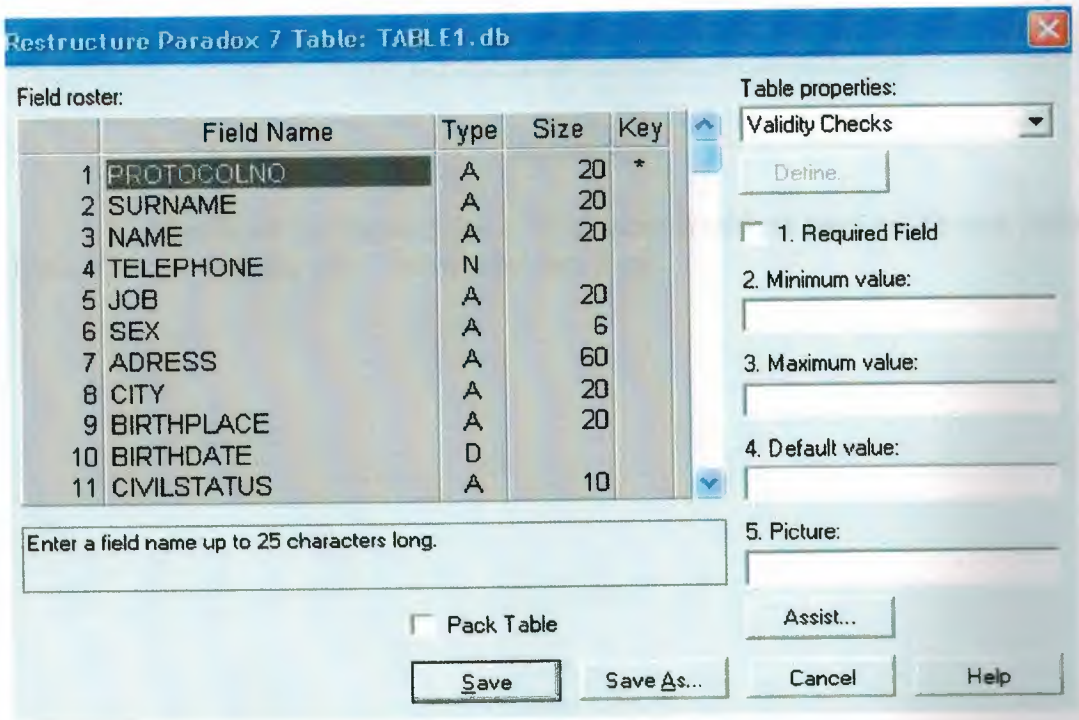

Figure 3.1

Above the **Figure 3.1**,table1 is used the newapatient,patientid and some menus will call the data from table 1. I used the unique according to protocl no and so the same protocol never used to other patient's protocolno. I used to for table 1 datasource1. Because I used to usually dbedit component,dbmemo on the program and I call from datasource.

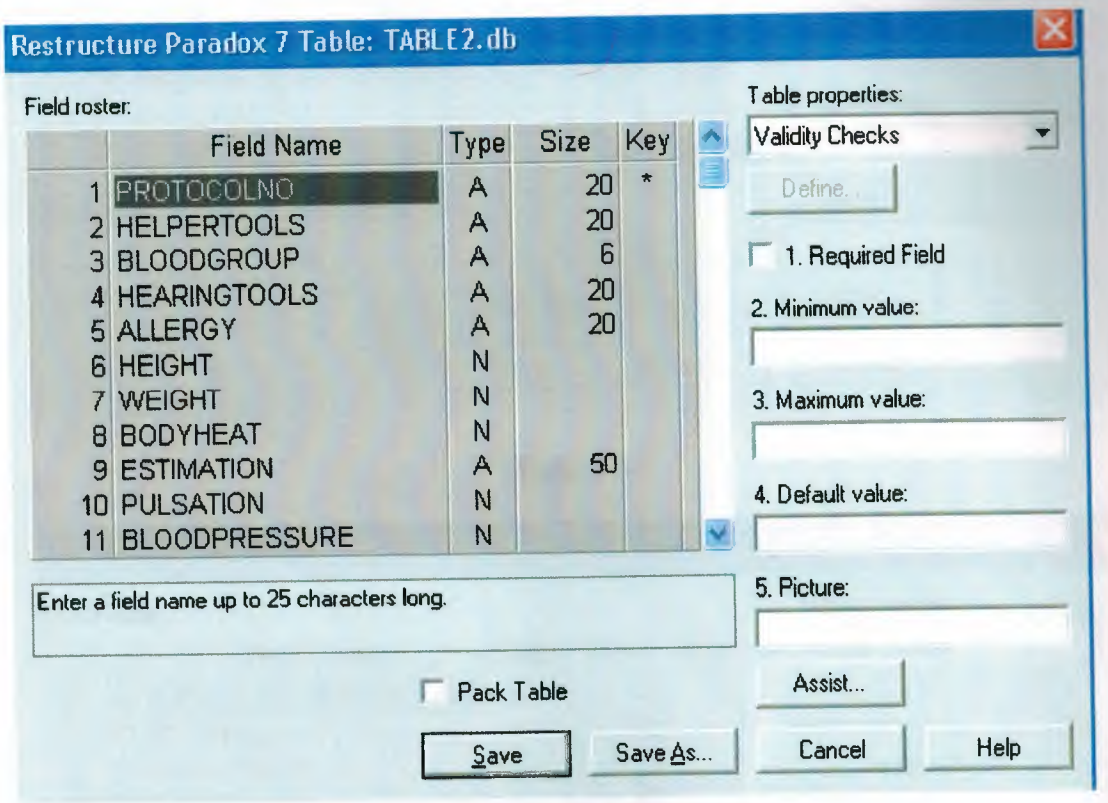

Figure 3.2

The above the **Figure3.2** is used to other tabshhets page on the new patient form. The user call the table2 other tabsheets form.

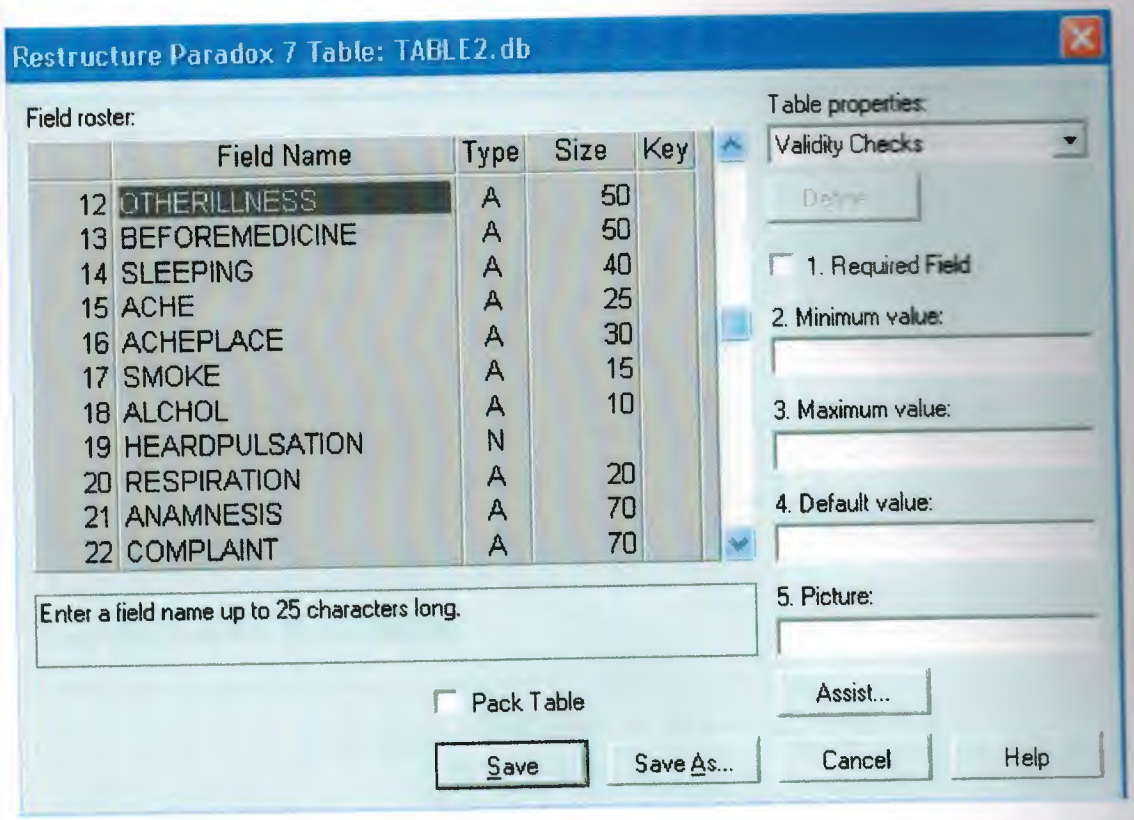

Figure 3.3

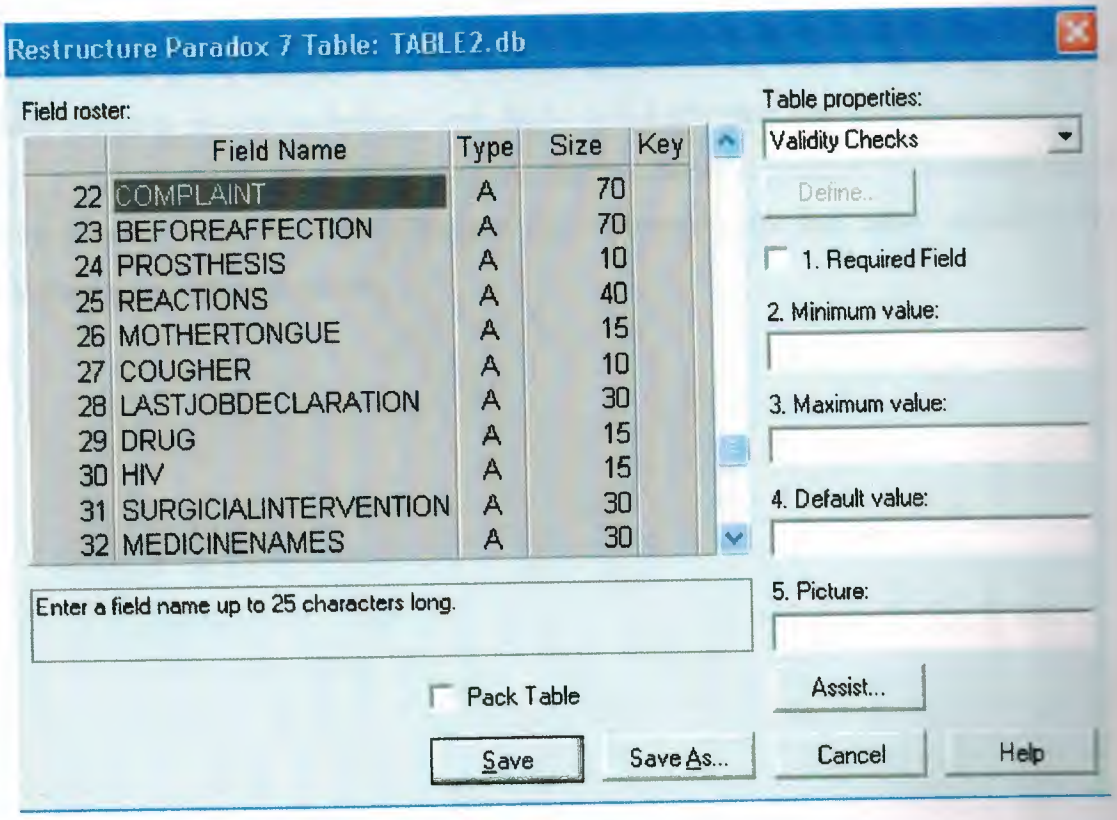

Figure 3.4

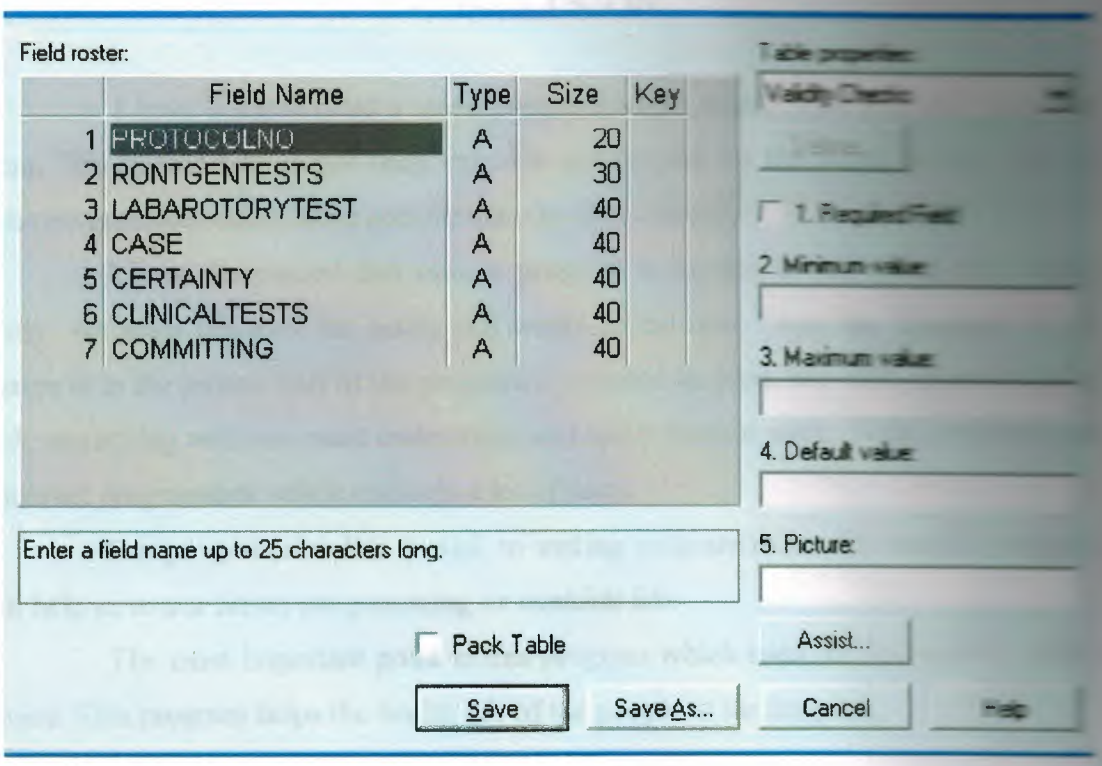

Figure 3.5

I used the four databases tables in the databases system in the programs. The table is used to patient analysing form. And I called this table from the datasource.

I used a lot of datasource on the prgram. Because I connected database table1 with datasources components. Because this is more easily and more usefull in the database programming system.

CONCLUSION

I have demonstrated a small program which details with data and processes them. This demonstration has been valuable and helpful for the future so that we can write programmes much more acurate than the first attempt.

I have discovered that once a program is needed to be written and design firstly we must discover the needs and wants of the user. Once the flowchart of the system is in the picture half of the program is counted as done. We have discovered that to do something well you must understand and apply to your work. With Delphi we can construct programmes which can help a lot of users.

This program also has started to writing program in our life and this program will help us to our future programming or workink life.

The most important point in this program which used the hospital or healthy clinics. This program helps the health life of the people in the hospital.

REFERENCES

[1] Memik Yanık, Borland Delphi 5 ile Görsel Programlama Teknikleri, Beta Yayım dağıtım San. Ve Tic. Ltd Şti

.

[2] Marcu Cantu, Delphi 5(pratic to develop the guide), Alfa Basım Yayım dağıtım A.Ş. İstanbul/ Turkey, January 2000.