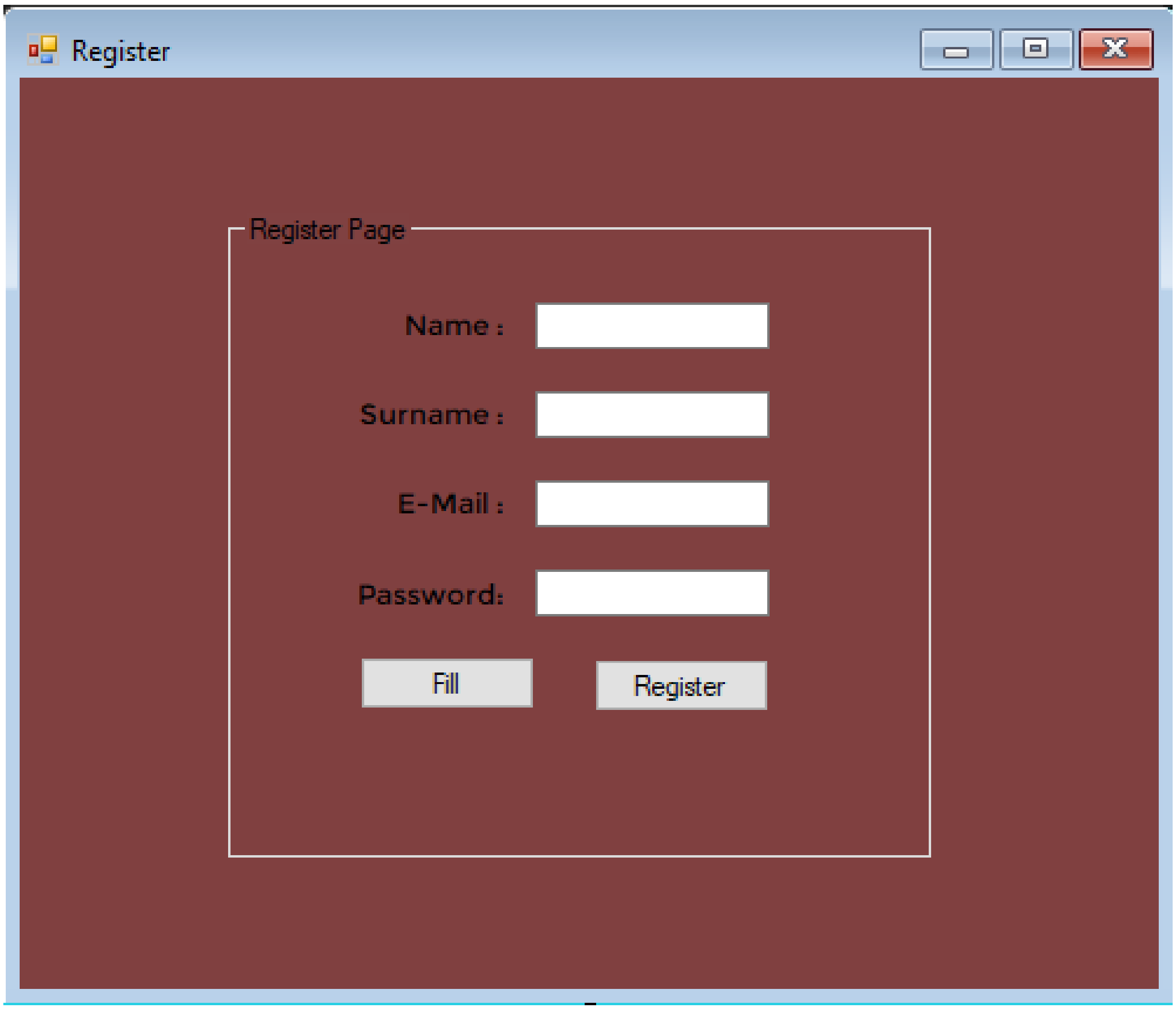
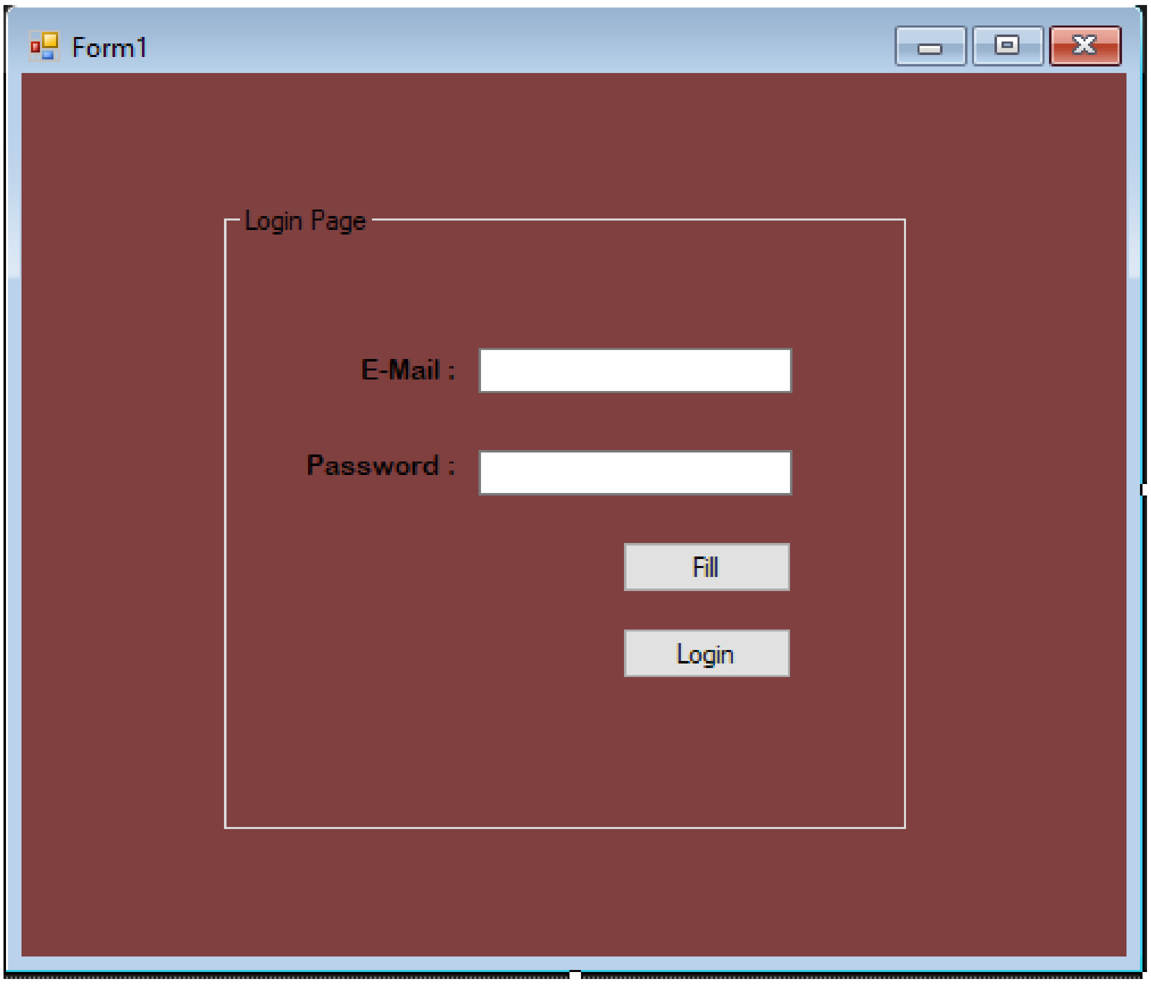
Yeni uygulamamızda bir görüşme için form sayfası dolduruyoruz.
Ancak bu sayfayı doldurabilmemiz için kayıt olmamız gerekiyor. Kayıt olduktan sonra e-mail ve şifre ile giriş yapıyoruz. Ardından form sayfamıza ulaşıyoruz. Burada bilgileri giriyoruz. 1 adet kayıt butonumuz, yazdığımız alanları temizleme butonumuz ve kaydedilen listeyi gösterme butonumuz olacak.
Bugünlük Register sayfası ve Login sayfasının arayüzünü yaptım ve Register sayfası ile login sayfasının kodlarını yazdım.Kolaylık olması adına bu alanları otomatik dolduran butonlar koydum.
Register Sayfası :



## Kodu:

```
namespace Interview
{
    public partial class Register : Form
    {
        People Reg_people1;
        public Register()
        {
            InitializeComponent();
        }

        private void Register_Load(object sender, EventArgs e)
        {
            txtBxPasswrd.PasswordChar = '*';
        }

        private void btnRegister_Click(object sender, EventArgs e)
        {
            Reg_people1 = new People();
            Reg_people1.Name = txtBxName.Text;
            Reg_people1.Surname = txtBxSurname.Text;
            Reg_people1.EMail = txtBxMail.Text;
            Reg_people1.Password = txtBxPasswrd.Text;

            this.Hide();
            Form1 form1 = new Form1();
            form1.RegisteredPeople(Reg_people1.Name, Reg_people1.Surname,
Reg_people1.EMail, Reg_people1.Password);
            form1.ShowDialog();

        }

        private void bttnFill_Click(object sender, EventArgs e)
        {
            txtBxName.Text = "Funda";
            txtBxSurname.Text = "Aydın";
            txtBxMail.Text = "funda@gmail.com";
            txtBxPasswrd.Text = "123";
        }
    }
}
```

Login sayfası :

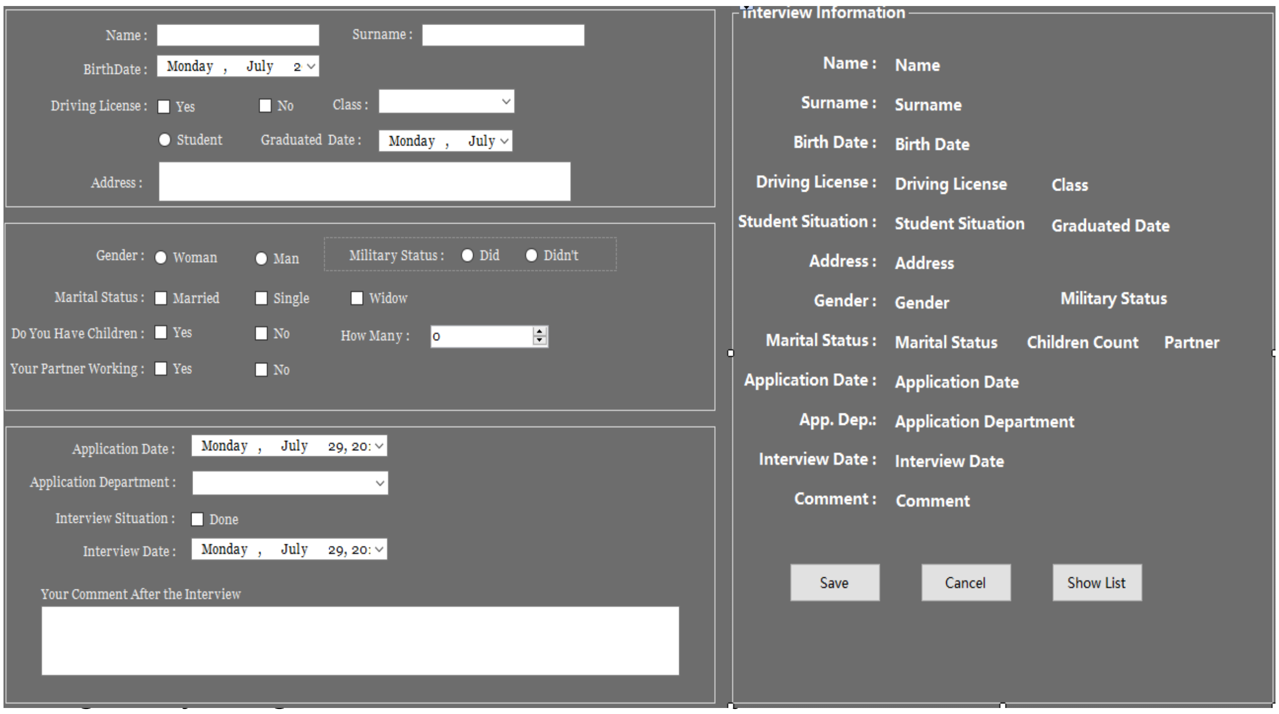


Kodu:

```
namespace Interview
```

```
{
    public partial class Form1 : Form
    {
        People person;
        public void RegisteredPeople(string RName,string RSurname,string REmail,string
RPassword)
        {
            person = new People();
            person.Name = RName;
            person.Surname = RSurname;
            person.EMail = REmail;
            person.Password = RPassword;
        }
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
            txtBxPasswrd.PasswordChar = '*';

        }

        private void bttnLogin_Click(object sender, EventArgs e)
        {
           if(txtBxEMail.Text==person.EMail&& txtBxPasswrd.Text==person.Password)
            {
                this.Hide();
                RegistrationForm registrationform = new RegistrationForm();
                registrationform.ShowDialog();

            }
           else
            {
                MessageBox.Show("Email or password is wrong!!");
            }



        }

        private void bttnFFill_Click(object sender, EventArgs e)
        {
            txtBxEMail.Text = "funda@gmail.com";
            txtBxPasswrd.Text = "123";
        }


    }
}
```

Arayüz:



Kod için yazdığımız sınıflar:

Bilgileri almak için yazdığımız sınıf ContactInformation sınıfı:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Interview
{
    public class ContactInformation
    {
        public string Name { get; set; }

        public string Surname { get; set; }

        public DateTime BirthDate { get; set; }

        public string DrivingLicense { get; set; }

        public string Class { get; set; }

        public string StudentSituation { get; set; }

        public DateTime Graduated { get; set; }

        public string Address { get; set; }
        public string Gender { get; set; }

        public string MilitaryStatus { get; set; }

        public string MaritalStatus { get; set; }

        public int ChildrenCount { get; set; }

        public string PartnerWorking { get; set; }

        public DateTime App_Date { get; set; }

        public string Department { get; set; }

        public DateTime InterviewDate { get; set; }

        public string Comment { get; set; }

        public override string ToString()
        {
            return
string.Format("{0},{1},{2},{3},{4},{5},{6},{7},{8},{9},{10},{11},{12},{13},{14},
{15},{16}",Name,Surname,BirthDate,DrivingLicense,Class,StudentSituation,Graduated,Addr
ess,Gender,
MilitaryStatus,MaritalStatus,ChildrenCount,PartnerWorking,App_Date,Department,Intervie
wDate,Comment);
        }
    }
}
```

Bir comboboxın içini doldurmak için yazdığım department sınıfı:

```
namespace Interview
{
    enum Department
    {
        Software=1,
        Hardware=2,
        Network=3,
        System=4
    };
}
```

Sürücü belgesini doldurmak için yazdığım enum:

```
namespace Interview
{
    enum DrivingLicense
    {
        A=1,
        B=2,
        C=3,
        D=4,
        F=5
    };
}
```

Medeni durum için yazdığım enum:
```
namespace Interview
{
    enum Marital
    {
        Single=1,
        Married=2,
        Widow=3
    };
}
```

Register ve Login sayfasında tutulan bilgiler için yazdığım sınıf:
```
namespace Interview
{
    class People
    {
        public string Name { get; set; }
        public string Surname { get; set; }
        public string EMail { get; set; }
        public string Password { get; set; }
        public override string ToString()
        {
            return string.Format("{0},{1},{2},{3}", Name, Surname, EMail, Password);
        }
    }
}
```

Gender comboboxı için yazdığım enum:
```
namespace Interview
{
    enum WomanManENum
    {
        Woman=1,
        Man=2
    };
}
```

Yes No comboboxı için yazdığım enum:
```
namespace Interview
{
    enum YesNoEnum
    {
        Yes=2,
        No=1
    };
}
```

Kod:

Clear butonu için metod , Load metodu ve her alan label arası ilişki için olan metodları yazdık.

Kod:

```
   public partial class RegistrationForm : Form
    {
        ContactInformation person;
        List<ContactInformation> listContact = new List<ContactInformation>();
        ListForm formlist;

        public void Clear()
        {
            txtBxName.Clear();
            lblInfName.Text = "Name";
            txtBxSurname.Clear();
            lblInfSurname.Text = "Surname";

            cmbBxdrilic.SelectedIndex = 0;
            chckBxStudent.Checked = false;
            cmbBxClass.SelectedIndex = 0;
            dttmPckrGraduate.Value = DateTime.Now;
            txtBxAddress.Clear();
            cmbBxWmoanMan.SelectedIndex = 0;
            rdbttnDid.Checked = false;
            rdbttnDidnt.Checked = false;
            rdBttnChildNo.Checked = false;
            rdBttnChildYEs.Checked = false;
            nmrcUDChild.Value = 0;
            rdBttnPartnerNo.Checked = false;
            rdBttnYesPartner.Checked = false;
            dttmPckrAppDate.Value = DateTime.Now;
            cmbBxDepartment.SelectedIndex = 0;
            chckBxDone.Checked = false;
            dttmPckrInterwDate.Value = DateTime.Now;
            txtBxComment.Clear();

            cmbBxClass.Visible = false;
            lblInfClass.Visible = false;
            lblClass.Visible = false;
            lblGraduate.Visible = false;
            dttmPckrGraduate.Visible = false;
            lblMilitary.Visible = false;
            rdbttnDid.Visible = false;
            rdbttnDidnt.Visible = false;
            lblChild.Visible = false;
            rdBttnChildYEs.Visible = false;
            rdBttnChildNo.Visible = false;
            lblCountChild.Visible = false;
            nmrcUDChild.Visible = false;
            lblpartnerWorking.Visible = false;
            rdBttnYesPartner.Visible = false;
            rdBttnPartnerNo.Visible = false;
            lblInfMilitSt.Visible = false;
            lblInfPartner.Visible = false;
            lblInfChildren.Visible = false;
            lblInfGraduated.Visible = false;
            lblIntervDate.Visible = false;
            dttmPckrInterwDate.Visible = false;
            lblComment.Visible = false;
            txtBxComment.Visible = false;
            lblInfInterwDate.Visible = false;
            lblInfComment.Visible = false;
            lblInterwInf.Visible = false;
            lblCommentInf.Visible = false;


            lblInfBirthDate.Text = "Birth Date";
            lblInfDrivingL.Text = "Driving License";
            lblInfClass.Text = "Class";
            lblInfStutendSt.Text = "Student Situation";
            lblInfGraduated.Text = "Graduated Date";
            lblInfAddress.Text = "Address";
            lblInfGender.Text = "Gender";
            lblInfMilitSt.Text = "Military Status";
            lblInfMarital.Text = "Marital Status";
            lblInfChildren.Text = "Children Count";
            lblInfPartner.Text = "Partner";
            lblInfApplDate.Text = "Application Date";
            lblInfApllDepart.Text = "Application Department";
            lblInfInterwDate.Text = "Interview Date";
            lblInfComment.Text = "Comment";

        }
        People PersonClone;
        public void loginInformation(string IName,string ISurnmae,string IEmail,
string IPassword)
        {
            PersonClone = new People();
            PersonClone.Name = IName;
            PersonClone.Surname = ISurnmae;
            PersonClone.EMail = IEmail;
            PersonClone.Password = IPassword;

        }
        public RegistrationForm()
        {
            InitializeComponent();
        }

        private void RegistrationForm_Load(object sender, EventArgs e)
        {
            formlist = new ListForm();
            cmbBxClass.DataSource = Enum.GetValues(typeof(DrivingLicense));
            cmbBxDepartment.DataSource = Enum.GetValues(typeof(Department));
            cmbBxdrilic.DataSource= Enum.GetValues(typeof(YesNoEnum));
            cmbBxWmoanMan.DataSource = Enum.GetValues(typeof(WomanManENum));
            cmbBxMarital.DataSource = Enum.GetValues(typeof(Marital));



            cmbBxClass.Visible = false;
            lblInfClass.Visible = false;
            lblClass.Visible = false;
            lblGraduate.Visible = false;
            dttmPckrGraduate.Visible = false;
            lblMilitary.Visible = false;
            rdbttnDid.Visible = false;
            rdbttnDidnt.Visible = false;
            lblChild.Visible = false;
            rdBttnChildYEs.Visible = false;
```

```
    rdBttnChildNo.Visible = false;
    lblCountChild.Visible = false;
    nmrcUDChild.Visible = false;
    lblpartnerWorking.Visible = false;
    rdBttnPartnerNo.Visible = false;
    rdBttnYesPartner.Visible = false;
    lblInfMilitSt.Visible = false;
    lblInfPartner.Visible = false;
    lblInfChildren.Visible = false;
    lblInfGraduated.Visible = false;
    lblIntervDate.Visible = false;
    dttmPckrInterwDate.Visible = false;
    lblComment.Visible = false;
    txtBxComment.Visible = false;
    lblInfInterwDate.Visible = false;
    lblInfComment.Visible = false;
}

private void txtBxName_TextChanged(object sender, EventArgs e)
{
    lblInfName.Text = txtBxName.Text;
}

private void dtTmPckrBirthDt_ValueChanged(object sender, EventArgs e)
{

    if(DateTime.Now.Year-dtTmPckrBirthDt.Value.Year<18)
    {
        MessageBox.Show("Please enter a value bigger than 18!");
    }
    else if(dtTmPckrBirthDt.Value>DateTime.Now)
    {
        MessageBox.Show("You have selected a future date!!");
    }
    else
    {
    lblInfBirthDate.Text = dtTmPckrBirthDt.Text;
    }
}
```

Kod:

```
private void txtBxSurname_TextChanged(object sender, EventArgs e)
{
    lblInfSurname.Text = txtBxSurname.Text;
}

private void txtBxAddress_TextChanged(object sender, EventArgs e)
{
    lblInfAddress.Text = txtBxAddress.Text;
}
private void cmbBxClass_SelectedIndexChanged(object sender, EventArgs e)
{
    lblInfClass.Text = cmbBxClass.Text;
}

private void dttmPckrGraduate_ValueChanged(object sender, EventArgs e)
{
    lblInfGraduated.Text = dttmPckrGraduate.Text;
    if(dttmPckrGraduate.Value.Month-DateTime.Now.Month>3)
```

```
            {
                MessageBox.Show("Graduation max can be after 3 months!");
            }
        }
        private void rdbttnDid_CheckedChanged(object sender, EventArgs e)
        {
            lblInfMilitSt.Text = rdbttnDid.Text;
        }

        private void rdbttnDidnt_CheckedChanged(object sender, EventArgs e)
        {
            lblInfMilitSt.Text = rdbttnDidnt.Text;
        }

        private void chckbxChilYes_CheckedChanged(object sender, EventArgs e)
        {
            lblInfChildren.Text = nmrcUDChild.Value.ToString() + "Children";
            lblCountChild.Visible = true;
            nmrcUDChild.Visible = true;
         }

        private void chckBxChildNo_CheckedChanged(object sender, EventArgs e)
        {
            lblInfChildren.Visible = false;
        }

        private void chckBxPartnerYes_CheckedChanged(object sender, EventArgs e)
        {
            lblInfPartner.Text = "Partner is working";
        }

        private void chckBxPartnerNo_CheckedChanged(object sender, EventArgs e)
        {
            lblInfPartner.Text = "Partner don't working";
        }

        private void dttmPckrAppDate_ValueChanged(object sender, EventArgs e)
        {
            lblInfApplDate.Text = dttmPckrAppDate.Text;
        }

        private void cmbBxDepartment_SelectedIndexChanged(object sender, EventArgs e)
        {
            lblInfApllDepart.Text = cmbBxDepartment.Text;
        }



private void dttmPckrInterwDate_ValueChanged(object sender, EventArgs e)
        {
            lblInfInterwDate.Text = dttmPckrInterwDate.Text;
            if(dttmPckrAppDate.Value>dttmPckrInterwDate.Value)
            {
                MessageBox.Show("Please select a date after the application date!");
            }
            else if((dttmPckrInterwDate.Value.Date-DateTime.Today).Days>5)
            {
                MessageBox.Show("Date have to be max 5 day later");
            }
            else if((DateTime.Today-dttmPckrInterwDate.Value.Date ).Days> 5)
            {
                MessageBox.Show("The date cannot be before the last 5 days!");
            }
```

```
}

private void textBox2_TextChanged(object sender, EventArgs e)
{
    lblInfComment.Text = txtBxComment.Text;
}

private void nmrcUDChild_ValueChanged(object sender, EventArgs e)
{
    lblInfChildren.Text = nmrcUDChild.Value.ToString();
}

private void chckBxDone_CheckedChanged(object sender, EventArgs e)
{
    if(chckBxDone.Checked==true)
    {
        lblIntervDate.Visible = true;
        dttmPckrInterwDate.Visible = true;
        lblComment.Visible = true;
        txtBxComment.Visible = true;
        lblInfInterwDate.Visible = true;
        lblInfComment.Visible = true;
        lblInterwInf.Visible = true;
        lblCommentInf.Visible = true;
    }
    else
    {
        lblIntervDate.Visible = false;
        dttmPckrInterwDate.Visible = false;
        lblComment.Visible = false;
        txtBxComment.Visible = false;
        lblInfInterwDate.Visible = false;
        lblInfComment.Visible = false;
        lblInterwInf.Visible = false;
        lblCommentInf.Visible = false;
    }

}

private void rdbttnDid_CheckedChanged_1(object sender, EventArgs e)
{
    lblInfMilitSt.Text = rdbttnDid.Text;
}

private void rdbttnDidnt_CheckedChanged_1(object sender, EventArgs e)
{
    lblInfMilitSt.Text = rdbttnDidnt.Text;
}

private void bttnSave_Click(object sender, EventArgs e)
{
    person = new ContactInformation();
    person.Name = lblInfName.Text;
    person.Surname = lblInfSurname.Text;
    try
    {
        person.BirthDate = Convert.ToDateTime(lblInfBirthDate.Text);

    }
    catch (Exception)
    {
```

```
    person.BirthDate = DateTime.Now;
}

person.DrivingLicense = lblInfDrivingL.Text;
if(lblInfClass.Visible==true)
{
    person.Class = lblInfClass.Text;
}
person.StudentSituation = lblInfStutendSt.Text;
if(lblInfGraduated.Visible==true)
{
    try
    {
        person.Graduated = Convert.ToDateTime(lblInfGraduated.Text);
    }
    catch (Exception)
    {
        person.Graduated = DateTime.Now;
    }
}
person.Address = lblInfAddress.Text;
person.Gender = lblInfGender.Text;
if(lblInfMilitSt.Visible==true)
{
    person.MilitaryStatus = lblInfMilitSt.Text;
}
person.MaritalStatus = lblInfMarital.Text;
if(lblInfChildren.Visible==true)
{
    try
    {
        person.ChildrenCount = Convert.ToInt32(lblInfChildren.Text);
    }
    catch (Exception)
    {
        person.ChildrenCount = 0;
    }

}
if(lblInfPartner.Visible==true)
{
    person.PartnerWorking = lblInfPartner.Text;
}
try
{
    person.App_Date = Convert.ToDateTime(lblInfApplDate.Text);
}
catch (Exception)
{

    person.App_Date = DateTime.Now;
}

person.Department = lblInfApllDepart.Text;
if(lblInfInterwDate.Visible==true)
{
    try
    {
        person.InterviewDate = Convert.ToDateTime(lblInfInterwDate.Text);
    }
    catch (Exception)
    {
```

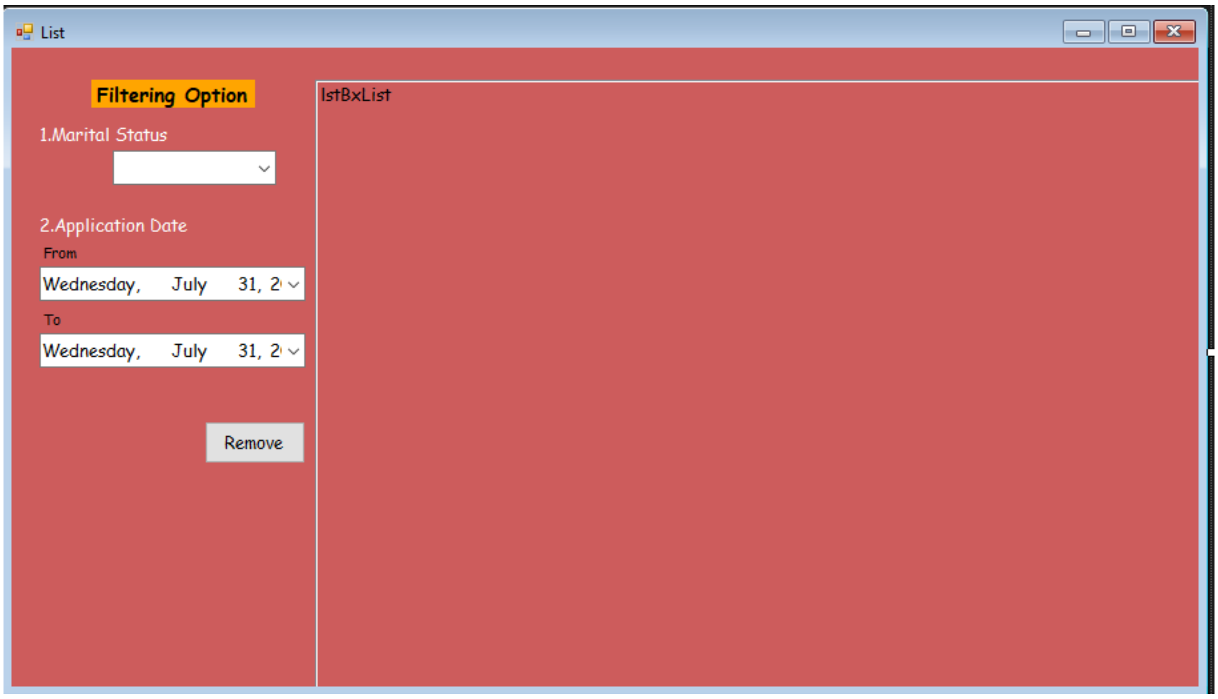
```
                person.InterviewDate = DateTime.Now;
            }
        }
        if(lblInfComment.Visible==true)
        {
            person.Comment = lblInfComment.Text;
        }

        listContact.Add(person);

        Clear();
        MessageBox.Show("Register is succesfull!!");
    }
```

Arayüzde koşullu olan diğer butonları, comboboxları vs. yazdım.
Son kodlar:

```
    private void bttnCancel_Click(object sender, EventArgs e)
    {
        Clear();
    }

    private void bttnShowTheList_Click(object sender, EventArgs e)
    {
        ListForm lstForm = new ListForm();

        lstForm.func_contact(listContact);
        lstForm.ShowDialog();

    }

    private void cmbBxdrilic_SelectedIndexChanged(object sender, EventArgs e)
    {
        if(cmbBxdrilic.Text=="Yes")
        {
            lblInfDrivingL.Text = cmbBxdrilic.Text;
            cmbBxClass.Visible = true;
            lblInfClass.Visible = true;
            lblClass.Visible = true;
        }
        else
        {
            lblInfDrivingL.Text = cmbBxdrilic.Text;
            cmbBxClass.Visible = false;
            lblInfClass.Visible = false;
            lblClass.Visible = false;

        }
    }

    private void cmbBxWmoanMan_SelectedIndexChanged(object sender, EventArgs e)
    {
        if(cmbBxWmoanMan.Text=="Woman")
        {
            lblInfGender.Text = cmbBxWmoanMan.Text;
            lblMilitary.Visible = false;
            rdbttnDid.Visible = false;
```

```
                rdbttnDidnt.Visible = false;
                lblInfMilitSt.Visible = false;
            }
            else
            {
                lblInfGender.Text = cmbBxWmoanMan.Text;
                lblMilitary.Visible = true;
                rdbttnDid.Visible = true;
                rdbttnDidnt.Visible = true;
                lblInfMilitSt.Visible = true;
            }


        }

        private void cmbBxMarital_SelectedIndexChanged(object sender, EventArgs e)
        {
            if(cmbBxMarital.Text=="Married")
            {
                lblInfMarital.Text = cmbBxMarital.Text;
                lblChild.Visible = true;
                rdBttnChildYEs.Visible = true;
                rdBttnChildNo.Visible = true;
                lblInfChildren.Visible = false;
                nmrcUDChild.Visible = false;
                lblpartnerWorking.Visible = true;
                rdBttnYesPartner.Visible = true;
                rdBttnPartnerNo.Visible = true;
                lblInfPartner.Visible = true;
            }
            else if(cmbBxMarital.Text=="Single")
            {
                lblInfMarital.Text = cmbBxMarital.Text;
                lblChild.Visible = false;
                rdBttnChildYEs.Visible = false;
                rdBttnChildNo.Visible = false;
                lblInfChildren.Visible = false;
                nmrcUDChild.Visible = false;
                lblpartnerWorking.Visible = false;
                rdBttnYesPartner.Visible = false;
                rdBttnPartnerNo.Visible = false;
                lblInfPartner.Visible = false;
            }
            else
            {
                lblInfMarital.Text = cmbBxMarital.Text;
                lblChild.Visible = true;
                rdBttnChildYEs.Visible = true;
                rdBttnChildNo.Visible = true;
                lblCountChild.Visible = true;
                nmrcUDChild.Visible = true;
                lblpartnerWorking.Visible = false;
                rdBttnYesPartner.Visible = false;
                rdBttnPartnerNo.Visible = false;
            }




        }
        private void rdBttnChildYEs_CheckedChanged(object sender, EventArgs e)
        {
            lblInfChildren.Text = nmrcUDChild.Value.ToString();
            nmrcUDChild.Visible = true;
            lblCountChild.Visible = true;
            lblInfChildren.Visible = true;
        }



        private void rdBttnChildNo_CheckedChanged(object sender, EventArgs e)
        {
            nmrcUDChild.Visible = false;
            lblCountChild.Visible = false;
            lblInfChildren.Visible = true;
            lblInfChildren.Text = "0";
        }

        private void rdBttnYesPartner_CheckedChanged(object sender, EventArgs e)
        {
            lblInfPartner.Text = rdBttnYesPartner.Text;

        }

        private void rdBttnPartnerNo_CheckedChanged(object sender, EventArgs e)
        {
            lblInfPartner.Text = rdBttnPartnerNo.Text;

        }

        private void chckBxStudent_CheckedChanged(object sender, EventArgs e)
        {
            if(chckBxStudent.Checked==true)
            {
                lblInfGraduated.Visible = true;
                lblInfStutendSt.Text = chckBxStudent.Text;
                dttmPckrGraduate.Visible = true;
                lblGraduate.Visible = true;
            }
            else
            {
                lblInfGraduated.Visible = false;
                lblInfStutendSt.Text = chckBxStudent.Text;
                dttmPckrGraduate.Visible = false;
                lblGraduate.Visible = false;
            }
        }
    }
}
```

Uygulamanın Liste kısmını yaptık. Formda Show List butonuna tıklayınca gelen formun arayüzünü ve kodunu yazdım.



Arayüzde iki filtreleme seçeneğimiz var.Bunlardan ilki medeni duruma göre filtre ikincisi ise başvuru tarihine göre filtreleme. Ve bir de kaydı silen bir buton ekledik.

Kod:

```
namespace Interview
{
    public partial class ListForm : Form
    {
      public   List<ContactInformation>  contactList;
        List<ContactInformation> listRemove;
        DialogResult dialogResult;
        public void func_contact( List<ContactInformation> list)
        {
            contactList = new List<ContactInformation>();
            contactList = list;

            lstBxList.DataSource = contactList;
            lstBxList.Refresh();

        }
        public ListForm()
        {
            InitializeComponent();
        }

        private void List_Load(object sender, EventArgs e)
        {
            //contactList = new List<ContactInformation>();
            cmbBxMaritalFilt.DataSource = Enum.GetValues(typeof(Marital));
            listRemove = new List<ContactInformation>();

            lstBxList.DataSource = null;
            lstBxList.DataSource = contactList;
            lstBxList.Refresh();
        }

        private void cmbBxMaritalFilt_SelectedIndexChanged(object sender, EventArgs e)
        {

            if (cmbBxMaritalFilt.Text=="Married")
            {
                lstBxList.DataSource = null;
                lstBxList.DataSource = contactList.Where(x => x.MaritalStatus == 
"Married").ToList();
                lstBxList.Refresh();

            }
            else if(cmbBxMaritalFilt.Text=="Single")
            {
                lstBxList.DataSource = null;
                lstBxList.DataSource = contactList.Where(x => x.MaritalStatus == 
"Single").ToList();

                lstBxList.Refresh();
            }
            else
            {
                lstBxList.DataSource = null;

                lstBxList.DataSource = contactList.Where(x => x.MaritalStatus == 
"Widow").ToList();
                lstBxList.Refresh();
            }
        }

        private void dttmPckrfrom_ValueChanged(object sender, EventArgs e)
        {
            lstBxList.DataSource = null;

            lstBxList.DataSource = contactList.Where(x => (x.App_Date > 
dttmPckrfrom.Value) && (x.App_Date < dttmPckrTo.Value)).ToList();
            lstBxList.Refresh();

        }

        private void bttnRemove_Click(object sender, EventArgs e)
        {
            listRemove = contactList;
            dialogResult = MessageBox.Show(lstBxList.SelectedItem.ToString()+"\nDo you 
want to remove this register?","Remove Panel",MessageBoxButtons.YesNo, 
MessageBoxIcon.Information);
            if(dialogResult==DialogResult.Yes)
            {
                listRemove.Remove((ContactInformation)lstBxList.SelectedItem);
                lstBxList.DataSource = null;
                lstBxList.DataSource = listRemove;
                lstBxList.Refresh();
            }
        }

    }
}
```

## Main class:

```
namespace Interview
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Register());
        }
    }
}
```