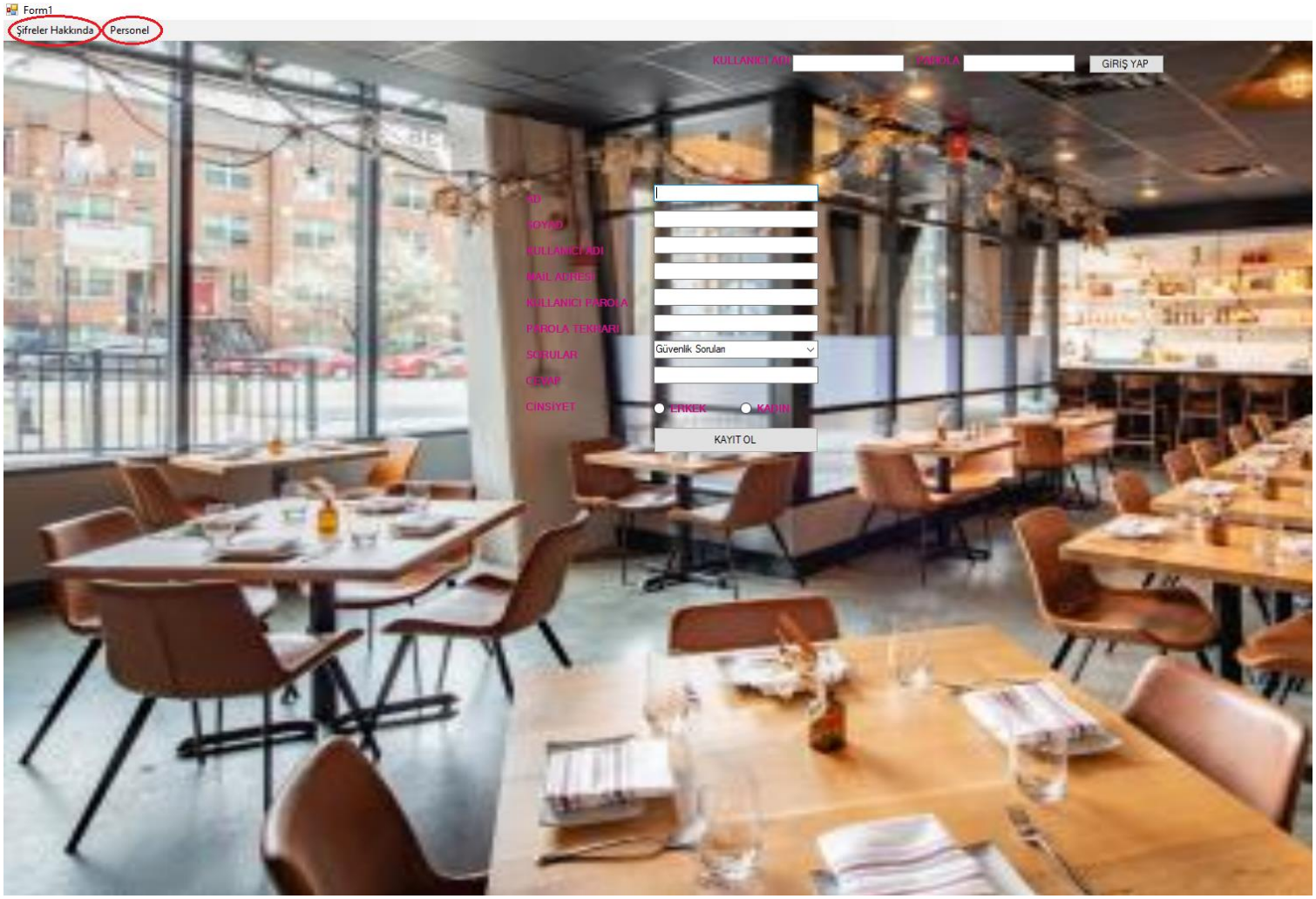
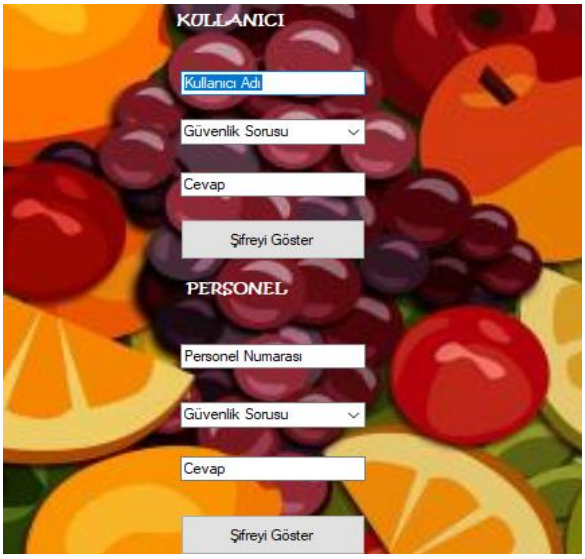
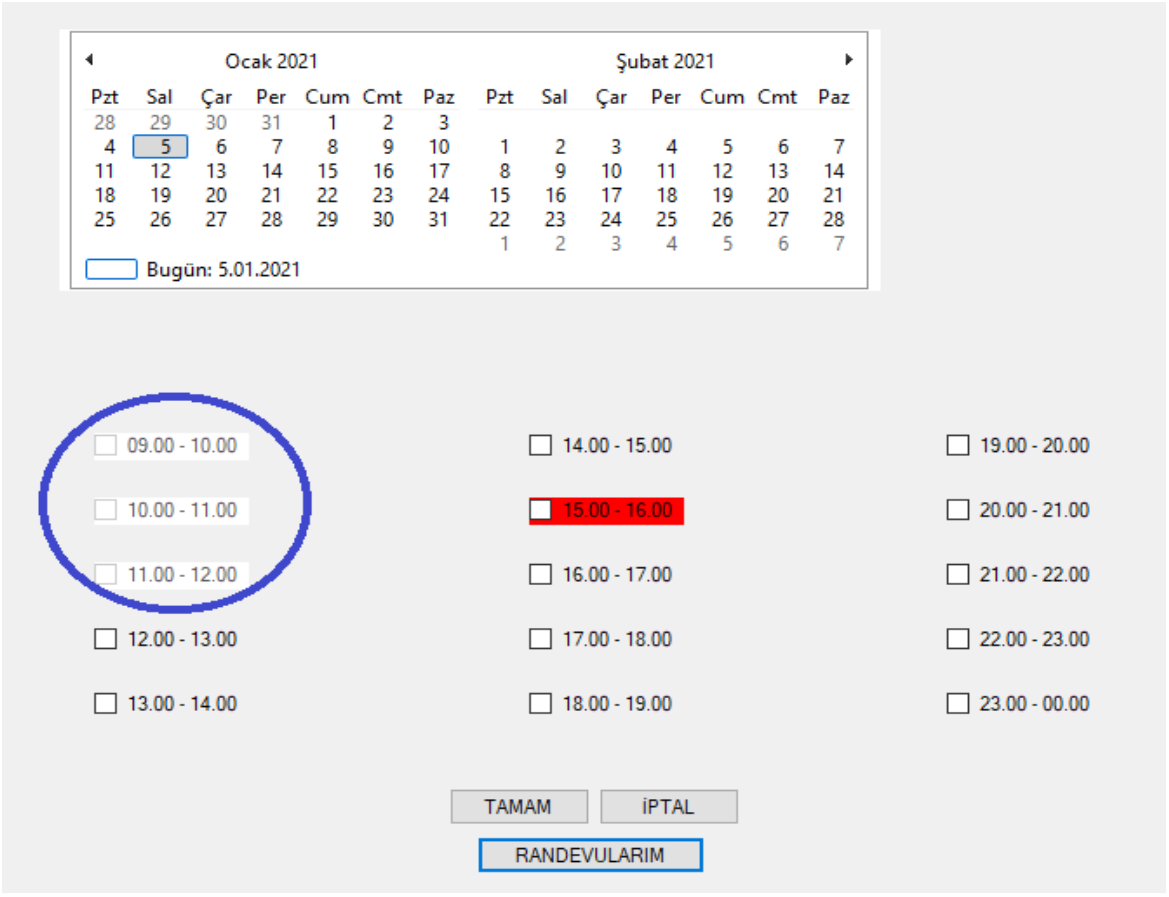


It is the homepage of our project. You can create a new user on this page. After logging in, you can go to the selection page. You can go to the staff page. You can see your forgotten password or change your password by clicking the **Sifreler Hakkinda** button.

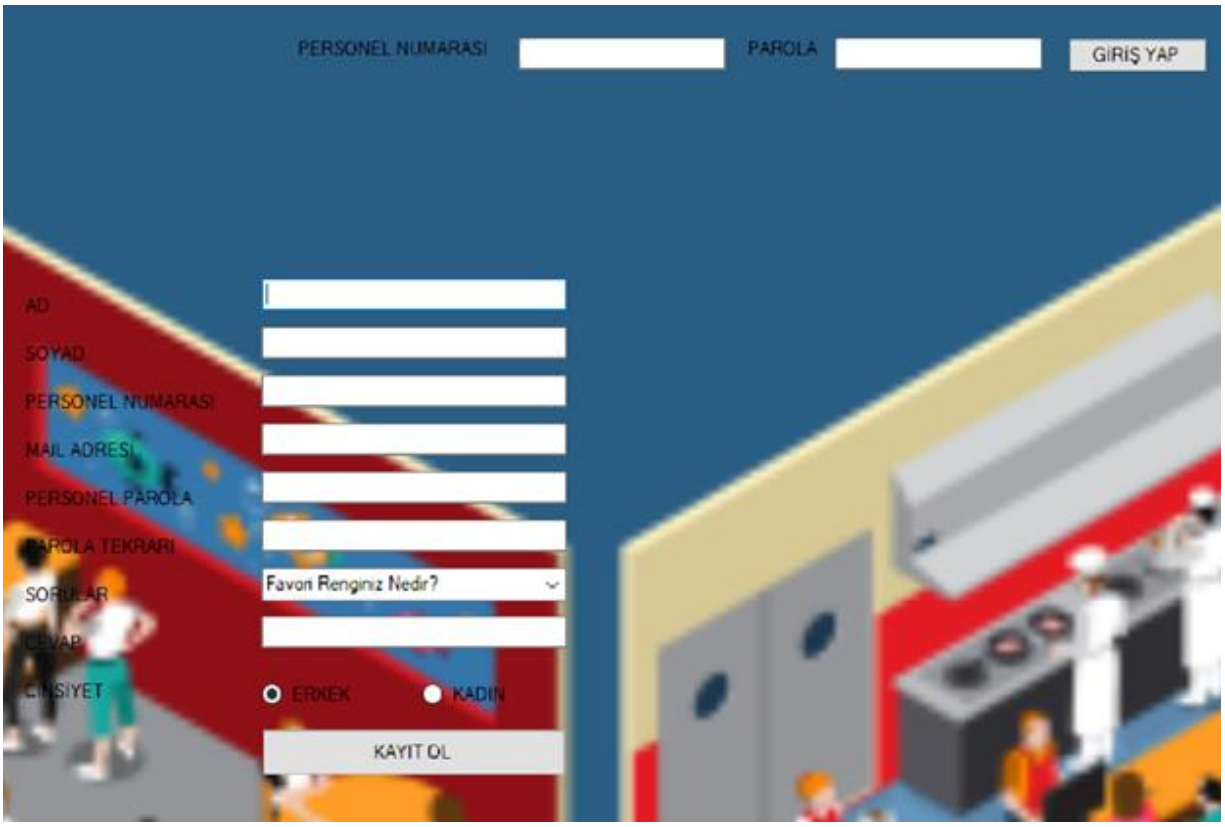


Password change and Forgot Password Screens

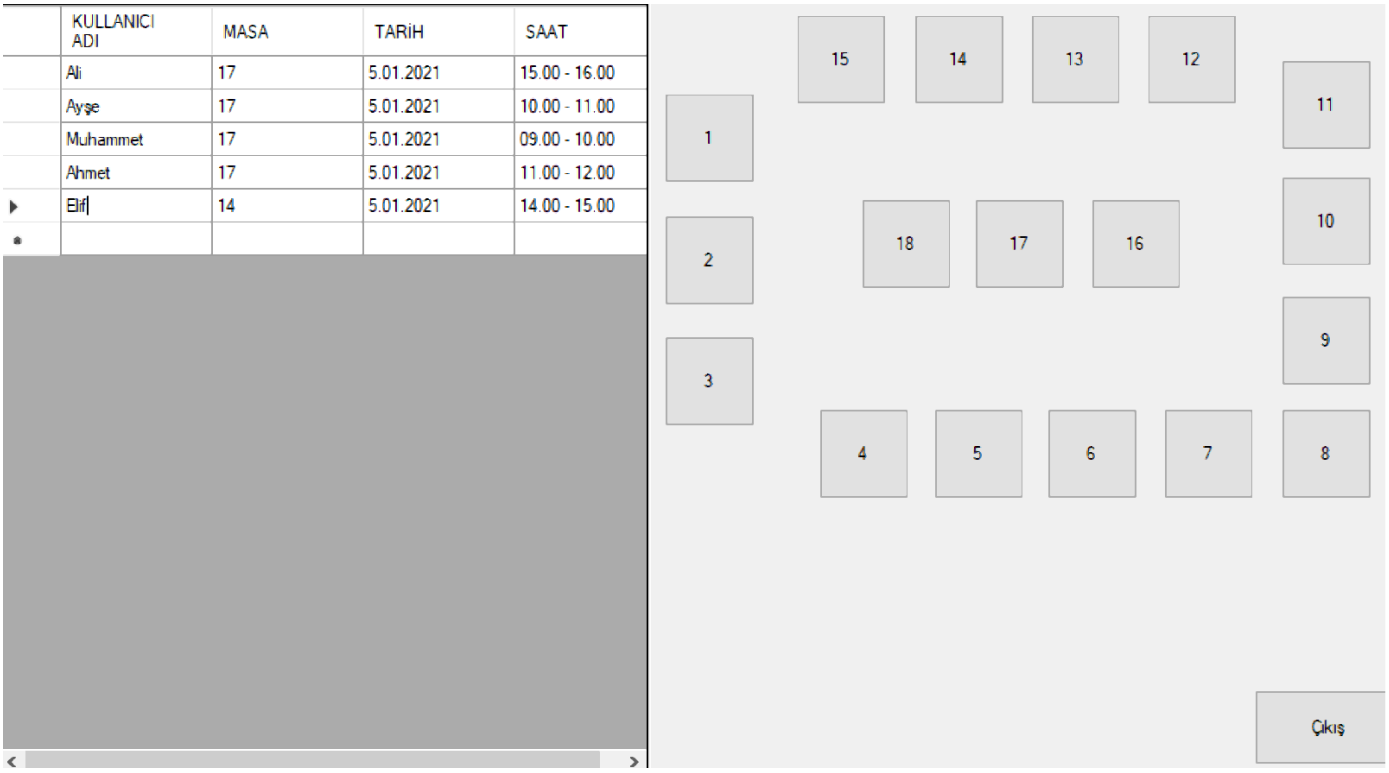
Table selection screen. You can reserve any table.



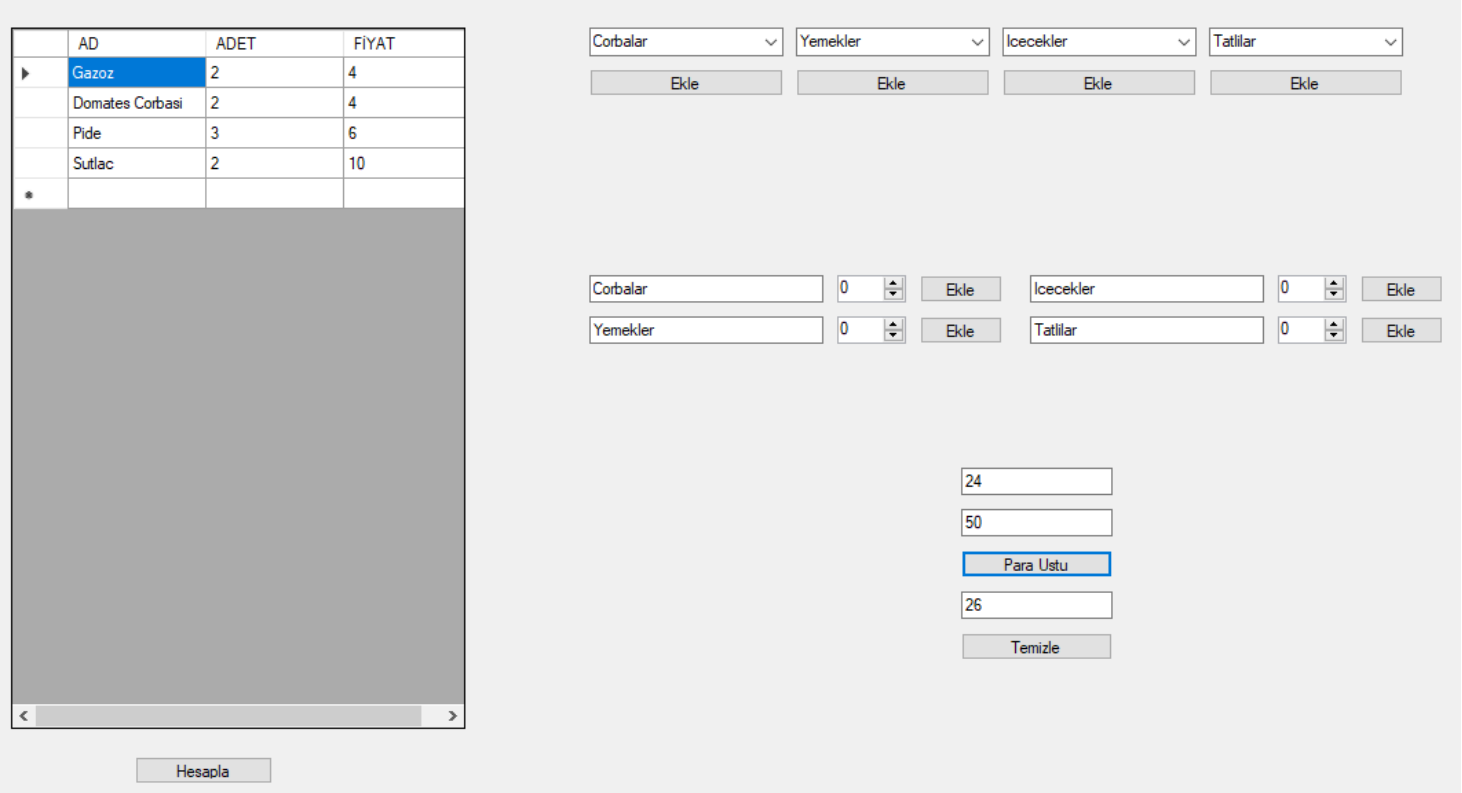
After our table is selected, you can see the reserved transactions of that table in days and hours. The watches selected with the blue circle cannot be selected because they were previously purchased by other users. It is displayed in red in the appointments you have selected. You can cancel your appointments selected from the Cancellation section.



Personal Login and New Personal Registration Page



On this screen, you can see the existing appointments and look at the orders on the tables.



The selected desktops are listed on the left after the refreshed and drinking ones are entered into the system. It is calculated with the Calculate button and the user shows the reminder of money.

The constructor method and data members of the Birey class. Since the data in this class is also in the Kullanici and Personel classes, it inherits in both classes.

```
public class Birey
{
    static public string ad;
    static public string soyad;
    static public string mail;
    static public string parola;
    static public string guvenlikSoru;
    static public string cevap;
    static public string cinsiyet;
    2 başvuru
public Birey(string isim, string soyisim, string email, string parolasi, string soru, string cevabi, string cinsiyeti)
    {
        ad = isim;
        soyad = soyisim;
        parola = parolasi;
        mail = email;
        guvenlikSoru = soru;
        cevap = cevabi;
        cinsiyet = cinsiyeti;

    }
    0 başvuru
    public string Isim
    {
        get
        {
            return ad;
        }
        set
        {
            ad = value;
        }
    }
    public string soyIsim
    {
        get
        {
            return soyad;
        }
        set
        {
            soyad = value;
        }
    }
    0 başvuru
    public string eMail
    {
        get
        {
            return mail;
        }
        set
        {
            mail = value;
        }
    }
    0 başvuru
    public string birParola
    {
        get
        {
            return parola;
        }
        set
        {
            parola = value;
        }
    }
    public string birSoru
    {
        get
        {
            return guvenlikSoru;
        }
        set
        {
            guvenlikSoru = value;
        }
    }
    0 başvuru
    public string birCinsiyet
    {
        get
        {
            return cinsiyet;
        }
        set
        {
            cinsiyet = value;
        }
    }
    0 başvuru
    public string birCevabi
    {
        get
        {
            return cevap;
        }
        set
        {
            cevap = value;
        }
    }
    6 başvuru
    public virtual void box()
    {

    }
}
```

Class Kullanici and Class Birey

```
public class Kullanici : Birey
{
    static public string kullanici_adi;

    2 başvuru
    public Kullanici(string isim, string soyisim, string email, string kullanciAdi, string parolasi, string guvenlikSorusu,
    string cevabi, string cinsiyeti) : base(isim, soyisim, email, parolasi, guvenlikSorusu, cevabi, cinsiyeti)
    {
        kullanici_adi = kullanciAdi;

    }
    0 başvuru
    public string kullaniciadi
    {
        get
        {
            return kullanici_adi;
        }
        set
        {
            kullanici_adi = value;
        }
    }
    6 başvuru
    public override void box()
    {
        MessageBox.Show("Kullanici Olusturuldu.");
    }
}

public class Personel : Birey
{
    static public string no;

    2 başvuru
    public Personel(string personelNo, string isim, string soyisim, string email, string parola, string guvenlikSorusu,
    string cevabi, string cinsiyeti) : base(isim, soyisim, email, parola, guvenlikSorusu, cevabi, cinsiyeti)
    {
        no = personelNo;
    }
    0 başvuru
    public string personelNo
    {
        get
        {
            return no;
        }
        set
        {
            no = value;
        }
    }
    6 başvuru
    public override void box()
    {
        MessageBox.Show("Personel Olusturuldu.");
    }
}
```

Base is used in the constructor methods of the Kullanici and Personel classes. Registering with Abstract Class; Contains the functions in the member class. In our member class, data members and the status of the member are checked. Thanks to the registration function, a new member is registered with the values entered on the home page and the values entered on the home page are assigned.

```
namespace Proje
{
    2 başvuru
    abstract class kayitOlmak
    {
        5 başvuru
        public abstract void kayit(string isim, string soyisim, string email, string kullanciAdi, string parolasi, string guvenlikSorusu, string cevabi, string cinsiyeti);
        4 başvuru
        public abstract bool giris(string kullanici_adi, string parola);
        2 başvuru
        public abstract void cikis();
        4 başvuru
        public abstract bool sifreDegis(string kullanici_adi, string parola, string guvenlikSoru, string cevap, string yeni);
        4 başvuru
        public abstract void sifremiUnuttum(string kullanici_adi, string guvenlikSoru, string cevap);
    }
    11 başvuru
    class uye : kayitOlmak
    {
        public static int sayac=0;
        List<string> uyeler = new List<string>();
        public static string suanki_kullanici;
        public static bool online;
        public static bool kontrol;
        public static bool dogru;
        5 başvuru
        public override void kayit(string isim, string soyisim, string email, string kullaniciAdi, string parolasi, string guvenlikSorusu, string cevabi, string cinsiyeti)
        {
            uyeler.Add(isim);
            uyeler.Add(soyisim);
            uyeler.Add(email);
            uyeler.Add(kullaniciAdi);
            uyeler.Add(parolasi);
            uyeler.Add(guvenlikSorusu);
            uyeler.Add(cevabi);
            uyeler.Add(cinsiyeti);
            sayac++;

        }
        2 başvuru
        public override void cikis()
        {
            online = false;
            suanki_kullanici = null;
        }
        4 başvuru
        public override bool giris(string kullanici_adi, string parola)
        {
            try
            {
                for (int i = 3; i < sayac * 8; i += 8)
                {
                    if (uyeler[i] == kullanici_adi && uyeler[i + 1] == parola)
                    {
                        suanki_kullanici = kullanici_adi;
                        online = true;
                        dogru = true;
                        break;
                    }
                }
                if (online == true && dogru == true)
                    return true;
                else
                    return false;
            }
            catch(Exception)
            {
                return false;
            }

        }
```

Function cikis and Function giris.

```
        public override bool sifreDegis(string kullanici_adi,string parola,string guvenlikSoru,string cevap,string yeni)
        {
            try
            {
                for (int i = 3; i < sayac * 8; i += 8)
                {
                    if (uyeler[i] == kullanici_adi && uyeler[i + 1] == parola && uyeler[i+2] == guvenlikSoru && uyeler[i+3] == cevap)
                    {
                        uyeler[i + 1] = yeni;
                        kontrol = true;
                    }
                }
                if (kontrol == true)
                    return true;
                else
                    return false;
            }
            catch (Exception)
            {
                return false;
            }
        }
        4 başvuru
        public override void sifremiUnuttum(string kullanici_adi,string guvenlikSoru, string cevap)
        {
            try
            {
                for (int i = 3; i < sayac * 8; i += 8)
                {
                    if (uyeler[i] == kullanici_adi && uyeler[i + 2] == guvenlikSoru && uyeler[i + 3] == cevap)
                    {
                        MessageBox.Show(uyeler[i + 1]);
                    }
                }
            }
            catch (Exception)
            {

            }
    }
    }
```

If the user name, password and security question in the data we assign to the array in the sifreDegis function are correct, they are changed. Otherwise it returns false.

If the username and security question are correct in the Forgot Password function, it will show the password, otherwise it will not.

In our personnel class, data members and the status of the Personnel are checked. Thanks to the registration function, new Personnel are registered with the values entered in the personnel page and the values entered in the personnel page are assigned. With the exit function, the Personnel exits the application.

```
class personeler : kayitOlmak
{
    public static int persayac=0;
    List<string> personeller = new List<string>();
    public bool online;
    public bool dogru;
    public bool kontrol;
    public static string suanki_personel;
    5 başvuru
    public override void kayit(string isim, string soyisim, string email, string personelNo, string parolasi, string guvenlikSorusu, string cevabi, string cinsiyeti)
    {
        personeller.Add(isim);
        personeller.Add(soyisim);
        personeller.Add(email);
        personeller.Add(personelNo);
        personeller.Add(parolasi);
        personeller.Add(guvenlikSorusu);
        personeller.Add(cevabi);
        personeller.Add(cinsiyeti);
        persayac++;

    }
    2 başvuru
    public override void cikis()
    {
        online = false;
        suanki_personel = null;
    }
```

```
public override bool giris(string personel_no, string parola)
{
    try
    {
        for (int i = 3; i < persayac * 8; i += 8)
        {
            if (personeller[i] == personel_no && personeller[i + 1] == parola)
            {
                online = true;
                dogru = true;
                break;
            }
        }
        if (online == true && dogru == true)
            return true;
        else
            return false;
    }
    catch (Exception)
    {
        return false;
    }

}
4 başvuru
public override bool sifreDegis(string personelNo, string parola, string guvenlikSoru, string cevap, string yeni)
{
    try
    {
        for (int i = 3; i < persayac * 8; i += 8)
        {
            if (personeller[i] == personelNo && personeller[i + 1] == parola && personeller[i + 2] == guvenlikSoru && personeller[i + 3] == cevap)
            {
                personeller[i + 1] = yeni;
                kontrol = true;
            }
        }
        if (kontrol == true)
            return true;
        else
            return false;
    }
    catch (Exception)
    {
        return false;
    }
}
4 başvuru
public override void sifremiUnuttum(string personelNo, string guvenlikSoru, string cevap)
{
    try
    {
        for (int i = 3; i < persayac * 8; i += 8)
        {
            if (personeller[i] == personelNo && personeller[i + 2] == guvenlikSoru && personeller[i + 3] == cevap)
            {
                MessageBox.Show(personeller[i + 1]);
            }
        }
    }
    catch (Exception)
    {
        MessageBox.Show("Hatalı Bilgiler");
    }
}
```

```
namespace Proje
{
    17 başvuru
    public static class taneFiyati
    {
        public static double mercimek = 1;
        public static double domates = 2;
        public static double iskembe = 3;
        public static double tarhana = 4;
        public static double adana = 2;
        public static double lahmacun = 1;
        public static double pide = 2;
        public static double tavuk = 3;
        public static double pizza = 4;
        public static double cigkofte = 5;
        public static double ayran = 6;
        public static double kola = 1;
        public static double gazoz = 2;
        public static double cay = 3;
        public static double kunefe = 4;
        public static double sutlac = 5;
        public static double kemal = 6;
    }
}
```

In the static class of taneFiyati, the price of the soups, meals, drinks and desserts on the menu is specified.

```
    2 başvuru
    interface Hesaplama
    {
        3 başvuru
        double fiyat
        {
            get;
            set;
        }
        3 başvuru
        double alinan
        {
            get;
            set;
        }
        2 başvuru
        double paraustu
        {
            get;
            get;
        }
    }
    2 başvuru
    class Hesap : Hesaplama
    {
        1 başvuru
        public Hesap(double x, double y)
        {
            fiyat = x;
            alinan = y;
        }
        3 başvuru
        public double fiyat { get; set; }
        3 başvuru
        public double alinan { get; set; }
        2 başvuru
        public double paraustu =>
          (alinan - fiyat);
    }
}
```

```
namespace Proje
{
    1 başvuru
    interface Sonuc
    {
        3 başvuru
        double fiyat
        {
            get;
            set;
        }
        3 başvuru
        double adet
        {
            get;
            set;
        }
        18 başvuru
        double toplamFiyat
        {
            get;
        }
    }
    35 başvuru
    class ToplamSonuc : Sonuc
    {
        17 başvuru
        public ToplamSonuc(double x, double y)
        {
            fiyat = x;
            adet = y;
        }
        3 başvuru
        public double fiyat { get; set; }
        3 başvuru
        public double adet { get; set; }
        18 başvuru
        public double toplamFiyat =>
          (fiyat * adet);
    }
}
```

Here are two Interfaces named Hesaplama and Sonuc. Hesap and Toplam sonuç have inherited from their respective interfaces. As a result of the operations in the classes, the menu account and cash values were found.