



**CELAL BAYAR ÜNİVERSİTESİ**  
**HASAN FERDİ TURGUTLU TEKNOLOJİ FAKÜLTESİ**  
**YAZILIM MÜHENDİSLİĞİ**

**YZM 2118**  
**YAZILIM MİMARİSİ ve TASARIMI**  
**2020 – 2021 BAHAR DÖNEMİ**  
**Diyetisyen Uygulaması**

**182805003 – Alper ÜLGER – II. Öğretim**  
**182805011 – Muratcan ŞEN – II. Öğretim**  
**182805013 – Yakup Hamit HANCI – II. Öğretim**

**Öğretim Görevlisi**  
**Dr. Mansur Alp TOÇOĞLU**



<https://github.com/yakuphanci/diyetisyenUygulaması>



<https://www.youtube.com/watch?v=Rg4MAL5-ThI>

**HAZİRAN 2021**  
**MANİSA**

## İÇİNDEKİLER

Proje Amacı.....	2
Proje Kapsamı.....	2
Proje UML Diyagramı.....	3
Hastalık UML Diyagramı.....	4
IHastalık Sınıfı.....	4
Obez Hastalık Sınıfı.....	4
Çölyak Hastalık Sınıfı.....	5
Şeker Hastalık Sınıfı.....	5
HastalıkCesidi Sınıfı.....	5
Diyet UML Diyagramı.....	6
IDiyet Sınıfı.....	5
Akdeniz Diyeti Sınıfı.....	7
Deniz Ürünleri Diyeti Sınıfı.....	7
GlutenFree Diyeti Sınıfı.....	8
Yeşillikler Dünyası Diyeti Sınıfı.....	8
DiyetYontemi Sınıfı.....	9
Rapor UML Diyagramı.....	9
RaporImplementor Soyut Sınıfı.....	10
HTML Rapor Somut Sınıfı.....	11
JSON Rapor Somut Sınıfı.....	11
RaporAbstraction Sınıfı.....	13
RefinedRaporAbstraction Sınıfı.....	13
RaporBilgi Sınıfı.....	13
Veriler Sınıfı.....	14
Kullanıcı Sınıfı.....	15
Hasta Sınıfı.....	15
Diyetisyen Sınıfı.....	16
Admin Sınıfı.....	16
Ekran Görüntüleri.....	17

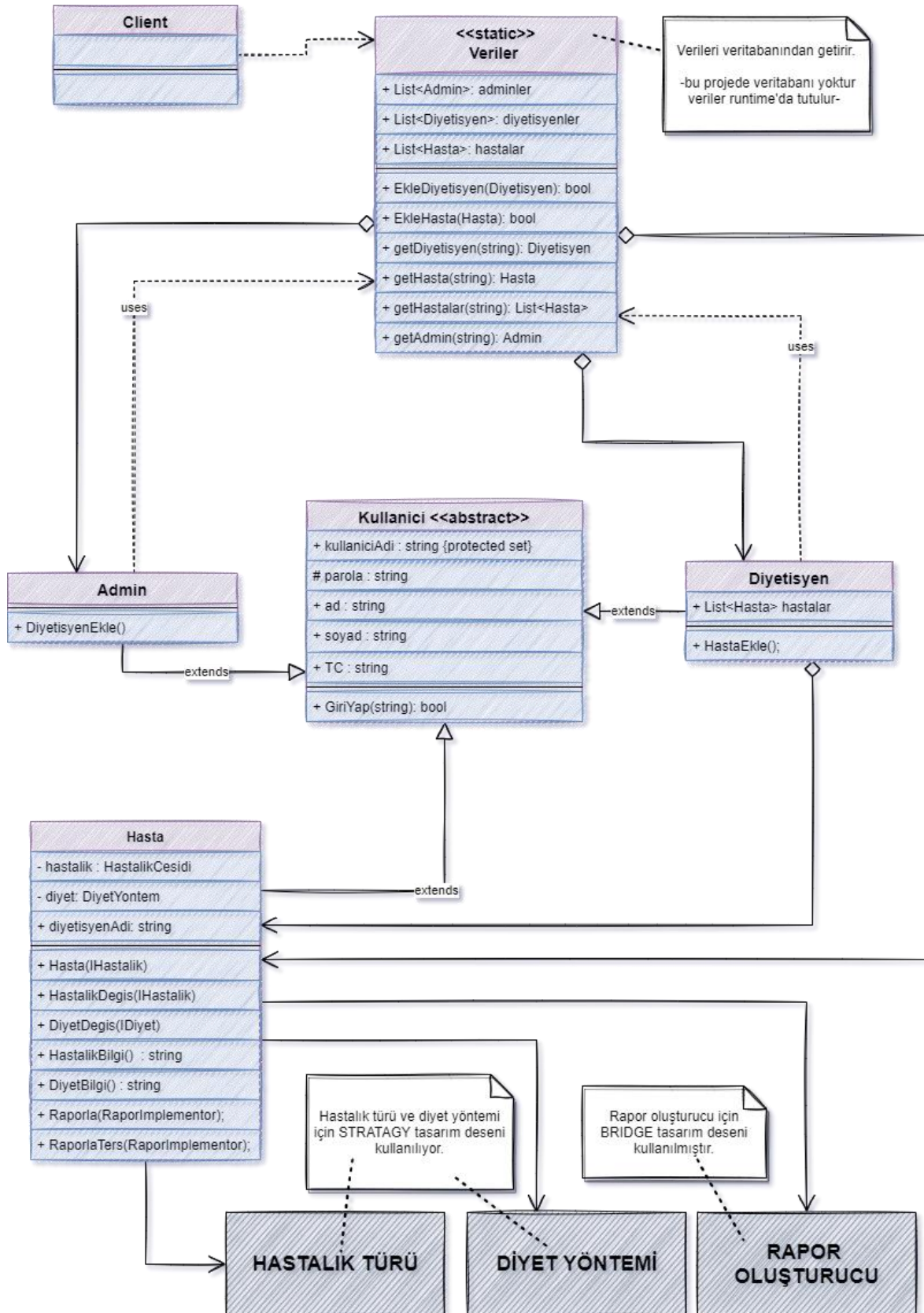
## **Projenin Amacı**

Projenin amacı diyetisyenlere ve hastalarına yönelik kullanıcı dostu bir uygulama geliştirmekle birlikte, diyetisyenin temelde üç tip hastalığa göre diyet takviminin oluşturulabildiği, ilgili diyetisyenin diyet verdiği hastasına yine temelde dört farklı diyet yöntemi uygulayabildiği, bir hastanın sadece tek bir diyet alabildiği, diyet ve hastalık çeşitlerinin de arttırılabildiği, hasta için raporlar alınabilen bir yapının oluşturulmasıdır.

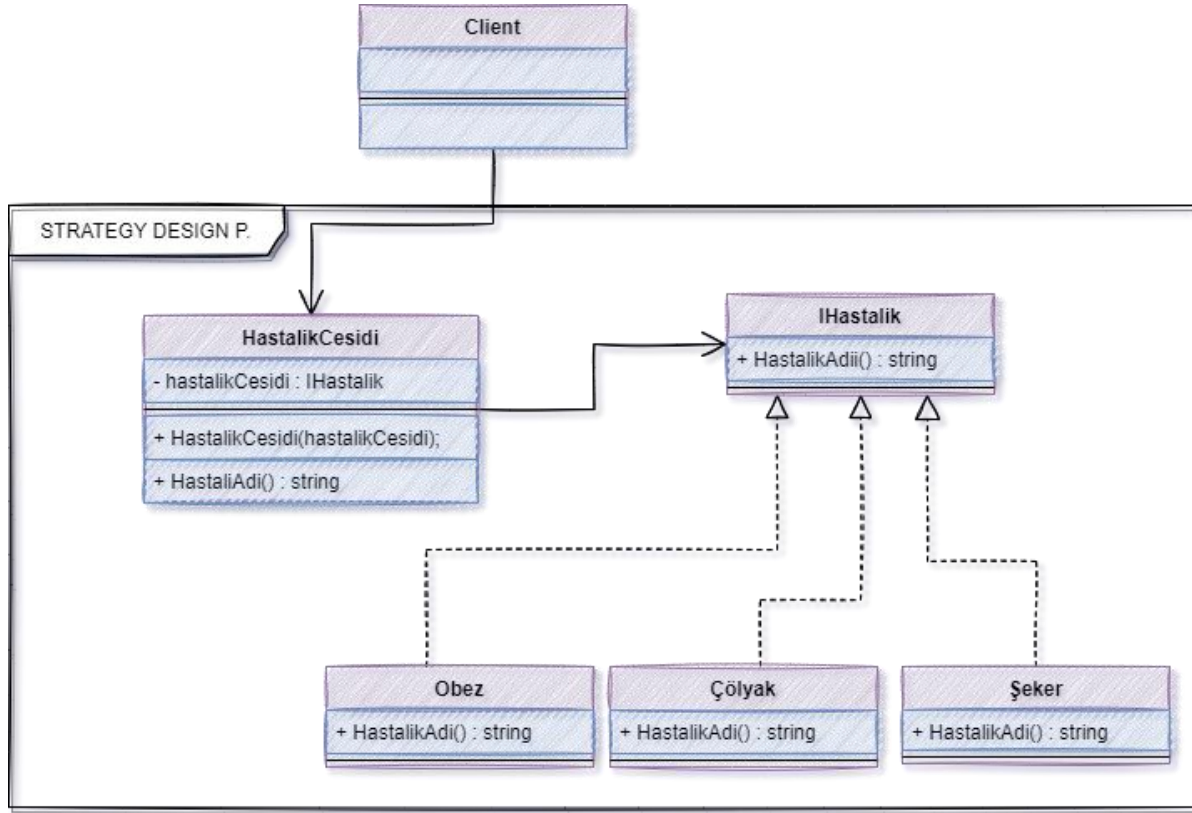
## **Projenin Kapsamı**

Geliştirilecek programda sisteme diyetisyen bilgisi *admin* tarafından, hasta bilgileri ise *diyetisyen* tarafından eklenebiliyor olacak. İlgili diyetisyen ilgili hastanın diyet yöntemini runtime da anlık olarak değiştirebiliyor olacak. Hastaya dair bilgiler iki bölüm halinde raporlanabiliyor olacak. Raporlar JSON veya HTML formatlarında alınabiliyor ve ayrıca raporun içerisindeki bölümlerin sıralanışı isteğe göre değiştirilebiliyor olacaktır. Sistem mimarisinde diyet yöntemleri, hastalık türleri ve rapor yöntemleri arttırılabiliyor olacaktır.

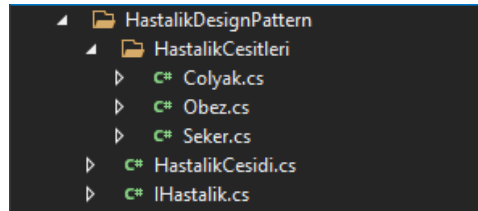
## Diyetisyen Uygulaması UML Diyagramı



## Hastalık Türü İçin Kullanılan Tasarım Deseni – STRATEGY DESIGN P. –



### Gerçekleme



Şekil 1 – Hastalık Tasarım Gerçeklemesi

### IHastalik Sınıfı

```
public interface IHastalik
{
    string HastalikAdi();
}
```

### Obez Sınıfı

```
public class Obez : IHastalik
{
    public string HastalikAdi()
    {
        return "Obez";
    }
}
```

## Colyak Sınıfı

```
public class Colyak : IHastalik
{
    public string HastalikAdi()
    {
        return "Çölyak";
    }
}
```

## Seker Sınıfı

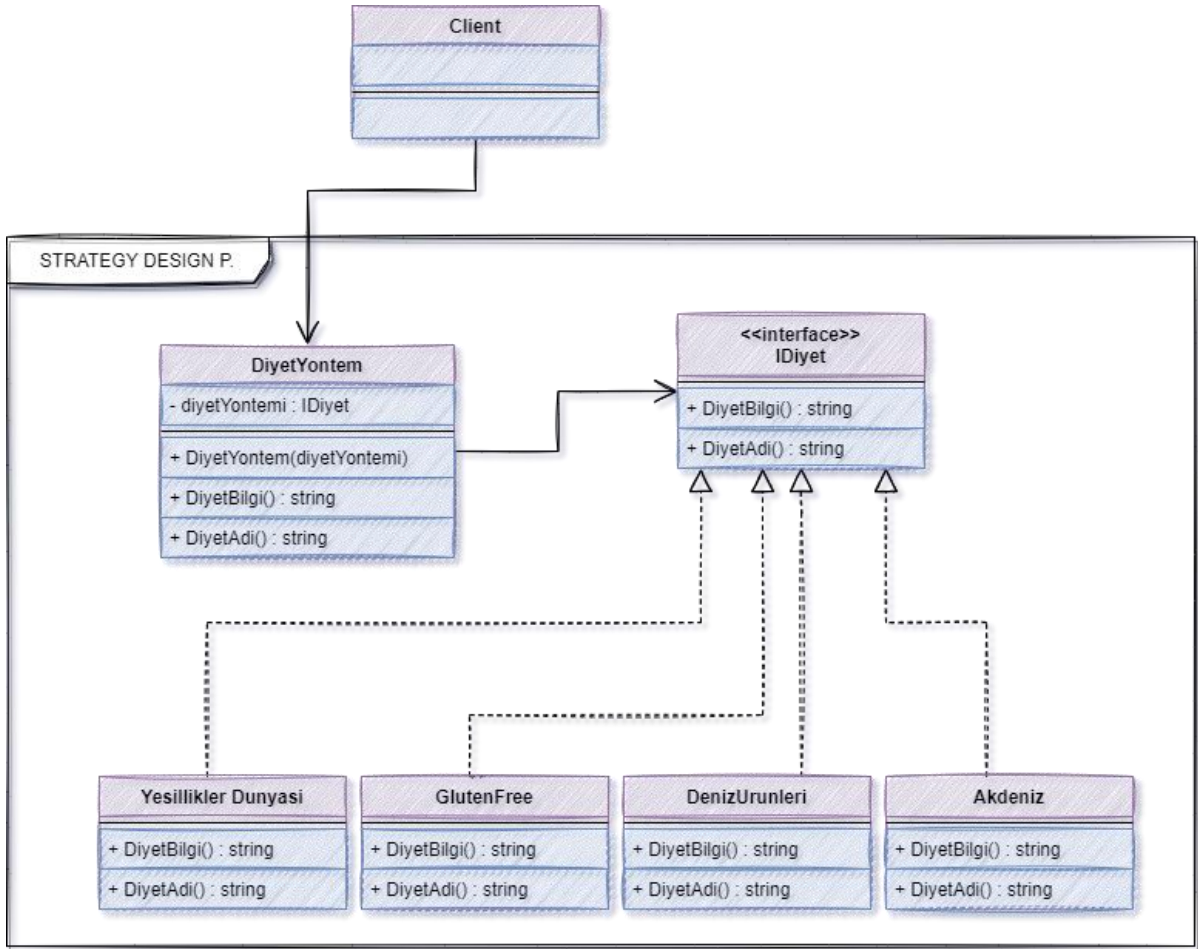
```
public class Seker : IHastalik
{
    public string HastalikAdi()
    {
        return "Şeker";
    }
}
```

## HastalikCesidi Sınıfı

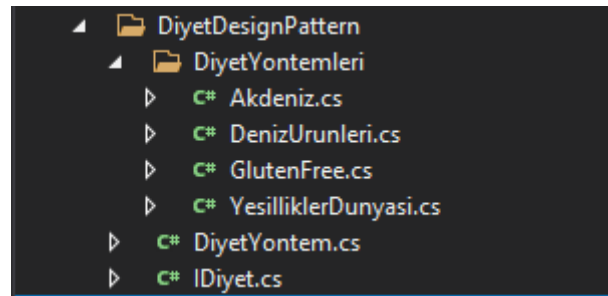
```
public class HastalikCesidi
{
    private IHastalik hastalikCesidi;
    public HastalikCesidi(IHastalik _hastalikCesidi)
    {
        this.hastalikCesidi = _hastalikCesidi;
    }

    public string HastalikAdi()
    {
        return this.hastalikCesidi.HastalikAdi();
    }
}
```

## Diyet Yöntemi İçin Kullanılan Tasarım Deseni – STRATAGY DESIGN P. –



### Gerçekleme



### IDiyet Sınıfı

```
public interface IDiyet
{
    string DiyetBilgi();
    string DiyetAdi();
}
```

## Akdeniz Sınıfı – Akdeniz Diyeti –

```
public class Akdeniz : IDiyet
{
    public string DiyetAdi()
    {
        return "Akdeniz";
    }

    public string DiyetBilgi()
    {
        StringBuilder sb = new StringBuilder();

        sb.AppendLine("Pazartesi");
        sb.AppendLine("Kahvaltı: 1 adet yumurta +1 dilim beyaz peynir + kepekli ekmek + domates, salatalık");
        sb.AppendLine("Öğlen: Ton balıklı bol yeşillikli salata + 1 dilim kepekli ekmek");
        sb.AppendLine("Ara: Yarım yağlı süt + Badem");
        sb.AppendLine("Akşam: Zeytinyağlı sebze yemeği + yarım yağlı yoğurt +salata");
        sb.AppendLine("Ara öğün: Meyve");
        sb.AppendLine(
            // ...
            return sb.ToString();
        )
    }
}
```

## DenizUrunleri Sınıfı – Deniz Ürünleri Diyeti –

```
public class DenizUrunleri : IDiyet
{
    public string DiyetAdi()
    {
        return "Deniz Ürünleri";
    }

    public string DiyetBilgi()
    {
        StringBuilder sb = new StringBuilder();

        sb.AppendLine("SABAH:");
        sb.AppendLine("> Kahvaltı öncesi 2 bardak su.");
        sb.AppendLine("> 1 adet haşlanmış yumurta");
        sb.AppendLine("> 2 yemek kaşığı lor peyniri");
        sb.AppendLine("> 1 adet tam ceviz");
        sb.AppendLine("> 1 ince dilim tam buğday ekmeği(25 gr)");
        sb.AppendLine("> Kırmızı biber, 7 - 8 sap maydanoz");
        sb.AppendLine("> 1 kupa yeşil çay");
        sb.AppendLine(
            // ...
            return sb.ToString();
        )
    }
}
```



## GlutenFree Sınıfı – Gluten Free Diyeti –

```
public class GlutenFree : IDiyet
{
    public string DiyetAdi()
    {
        return "Gluten Free";
    }

    public string DiyetBilgi()
    {
        StringBuilder sb = new StringBuilder();

        sb.AppendLine("Glutensiz Diyet Listesi - Menü");
        sb.AppendLine("");
        sb.AppendLine("Pazartesi");
        sb.AppendLine("Kahvaltı: Glutensiz ekmek, çiğ sebze, beyaz peynir, kuru erik. ");
        sb.AppendLine("Öğle: Izgara balık, rokalı salata, mısır ekmeği.");
        sb.AppendLine("Akşam: Tavuk sote, pirinçli yayla çorbası, mevsim salatası. ");
        sb.AppendLine("");
        // ...
        return sb.ToString();
    }
}
```

## YesilliklerDunyasi Sınıfı –Yeşillikler Dünyası Diyeti –

```
public class YesilliklerDunyasi : IDiyet
{
    public string DiyetAdi()
    {
        return "Yeşillikler Dünyası";
    }

    public string DiyetBilgi()
    {
        StringBuilder sb = new StringBuilder();

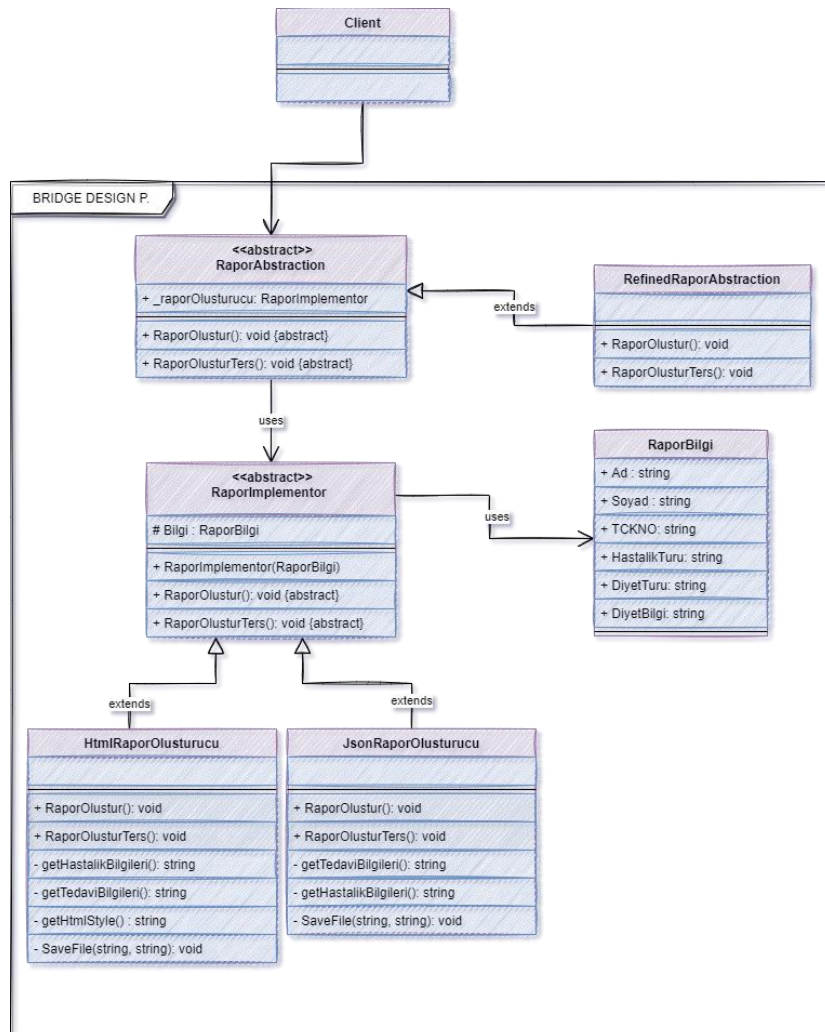
        sb.AppendLine("YEŞİLLİKLER DÜNYASI DİYETİ ");
        sb.AppendLine("");
        sb.AppendLine("Kahvaltı:");
        sb.AppendLine("1 dilim diyet ekmeği,");
        sb.AppendLine("2 - 3 dilim domates,");
        sb.AppendLine("2 - 3 dilim salatalık,");
        sb.AppendLine("Şekersiz çay,");
        sb.AppendLine("Maydanoz,");
        sb.AppendLine("Marul,");
        sb.AppendLine("");
        // ...
        return sb.ToString();
    }
}
```

## DiyetYontemi Sınıfı

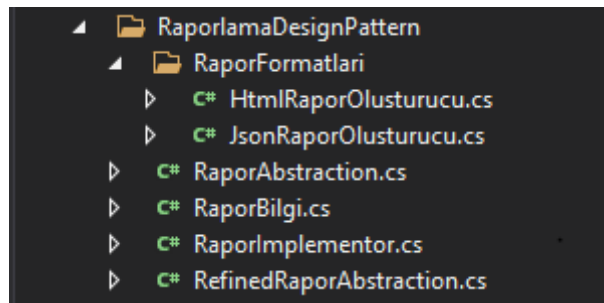
```
public class DiyetYontem
{
    private IDiyet diyetYontemi;
    public DiyetYontem(IDiyet _diyetYontemi)
    {
        this.diyetYontemi = _diyetYontemi;
    }

    public string DiyetBilgi()
    {
        return this.diyetYontemi.DiyetBilgi();
    }
    public string DiyetAdi()
    {
        return this.diyetYontemi.DiyetAdi();
    }
}
```

## Raporlama Sistemi İçin Kullanılan Tasarım Deseni – BRIDGE DESIGN P. –



## Gerçekleme



## RaporImplementor Soyut Sınıfı

```
public abstract class RaporImplementor
{
    protected RaporBilgi raporBilgi;
    public RaporImplementor(RaporBilgi _raporBilgi)
    {
        this.raporBilgi = _raporBilgi;
    }

    public abstract void RaporOlustur();
    public abstract void RaporOlusturTers();
}
```

# HTML Rapor Somut Sınıfı

```

class HtmlRaporOlusturucu : RaporImplemantor
{
    public HtmlRaporOlusturucu(RaporBilgi _raporBilgi) : base(_raporBilgi) { }
    private string getHastaBilgileri()
    {
        StringBuilder sb = new StringBuilder();
        sb.AppendLine(string.Format("<table>"));
        sb.AppendLine(string.Format("<tr>"));
        sb.AppendLine(string.Format("<t\t <td style=\"width:200px;\"><b>Ad:</b></td>"));
        sb.AppendLine(string.Format("<t\t <td> {0} </td>", raporBilgi.ad));
        sb.AppendLine(string.Format("<t\t <td style=\"width:200px;\"><b>Soyad:</b></td>"));
        sb.AppendLine(string.Format("<t\t <td> {0} </td>", raporBilgi.soyad));
        sb.AppendLine(string.Format("<t\t <td style=\"width:200px;\"><b>TCKNO:</b></td>"));
        sb.AppendLine(string.Format("<t\t <td> {0} </td>", raporBilgi.tckno + "<t\t <td>"));
        sb.AppendLine(string.Format("</table>"));
        return sb.ToString();
    }
    private string getTedaviBilgileri()
    {
        StringBuilder sb = new StringBuilder();
        sb.AppendLine(string.Format("<table>"));
        sb.AppendLine(string.Format("<tr>"));
        sb.AppendLine(string.Format("<t\t <td style=\"width:200px;\"><b>Hastalık:</b></td>"));
        sb.AppendLine(string.Format("<t\t <td> {0} </td>", raporBilgi.hastalikTuru));
        sb.AppendLine(string.Format("<t\t <td style=\"width:200px;\"><b>Diyet Yöntemi:</b></td>"));
        sb.AppendLine(string.Format("<t\t <td> {0} </td>", raporBilgi.diyetTuru));
        sb.AppendLine(string.Format("<t\t <td style=\"width:200px;\"><b>Diyet Açıklama:</b></td>"));
        sb.AppendLine(string.Format("<t\t <td> {0} </td>", raporBilgi.diyetBilgi.Replace("<n>", "<br>")));
        sb.AppendLine(string.Format("<t\t <td>"));
        sb.AppendLine(string.Format("</table>"));
        return sb.ToString();
    }
    private string getHtmlStyle()
    {
        StringBuilder sb = new StringBuilder();
        sb.AppendLine("<style>!--bir takım css kodları--></style>");
        return sb.ToString();
    }
    private void SaveFile(string path, string text)
    {
        FileStream fParameter = new FileStream(path, FileMode.Create, FileAccess.Write);
        StreamWriter m_WriterParameter = new StreamWriter(fParameter);
        m_WriterParameter.BaseStream.Seek(0, SeekOrigin.End);
        m_WriterParameter.Write(text);
        m_WriterParameter.Flush();
        m_WriterParameter.Close();
    }
    public override void RaporOlustur()
    {
        string outputName = "rapor_" + raporBilgi.ad + raporBilgi.soyad + "_" + DateTime.Now.ToString("ddMMyyyy") + ".html";
        string dirParameter = AppDomain.CurrentDomain.BaseDirectory + @"HTML Raporlar\" + outputName;
        StringBuilder sb = new StringBuilder();
        sb.AppendLine(getHtmlStyle());
        sb.AppendLine(string.Format("<h3>Hasta Bilgileri:</h3>"));
        sb.AppendLine(getHastaBilgileri());
        sb.AppendLine(string.Format("<hr>"));
        sb.AppendLine(string.Format("<h3>Tedavi Bilgileri:</h3>"));
        sb.AppendLine(getTedaviBilgileri());
        SaveFile(dirParameter, sb.ToString());
        System.Windows.Forms.MessageBox.Show(dirParameter + "<nKonumuna rapor oluşturulmuştur.</n>");
    }
    public override void RaporOlusturTers()
    {
        string outputName = "rapor_Ters_" + raporBilgi.ad + raporBilgi.soyad + "_" + DateTime.Now.ToString("ddMMyyyy") + ".html";
        string dirParameter = AppDomain.CurrentDomain.BaseDirectory + @"HTML Raporlar\" + outputName;
        StringBuilder sb = new StringBuilder();
        sb.AppendLine(getHtmlStyle());
        sb.AppendLine(string.Format("<h3>Tedavi Bilgileri:</h3>"));
        sb.AppendLine(getTedaviBilgileri());
        sb.AppendLine(string.Format("<hr>"));
        sb.AppendLine(string.Format("<h3>Hasta Bilgileri:</h3>"));
        sb.AppendLine(getHastaBilgileri());
        SaveFile(dirParameter, sb.ToString());
        System.Windows.Forms.MessageBox.Show(dirParameter + "<nKonumuna rapor oluşturulmuştur.</n>");
    }
}

```

## JSON Rapor Somut Sınıfı

```
public class JsonRaporOlusturucu : RaporImplementor
{
    public JsonRaporOlusturucu(RaporBilgi _raporBilgi) : base(_raporBilgi)
    {
    }

    private string getHastaBilgileri()
    {
        StringBuilder sb = new StringBuilder();

        sb.AppendLine("{");
        sb.AppendLine(string.Format("\\"hastaAdi\\": \"{0}\\",", raporBilgi.ad));
        sb.AppendLine(string.Format("\\"hastaSoyadi\\": \"{0}\\",", raporBilgi.soyad));
        sb.AppendLine(string.Format("\\"hastaTckno\\": \"{0}\\",", raporBilgi.tckno));
        sb.Append("}");
        return sb.ToString();
    }

    private string getTedaviBilgileri()
    {
        StringBuilder sb = new StringBuilder();

        sb.AppendLine("{");
        sb.AppendLine(string.Format("\\"hastalik\\": \"{0}\\",", raporBilgi.hastalikTuru));
        sb.AppendLine(string.Format("\\"diyetYontemi\\": \"{0}\\",", raporBilgi.diyetTuru));
        sb.AppendLine(string.Format("\\"diyetBilgi\\": \"{0}\\",", raporBilgi.diyetBilgi.Replace(Environment.NewLine, "\\n")));
        sb.Append("}");

        return sb.ToString();
    }

    private void SaveFile(string path, string text)
    {
        FileStream fParameter = new FileStream(path, FileMode.Create, FileAccess.Write);
        StreamWriter m_WriterParameter = new StreamWriter(fParameter);
        m_WriterParameter.BaseStream.Seek(0, SeekOrigin.End);
        m_WriterParameter.Write(text);
        m_WriterParameter.Flush();
        m_WriterParameter.Close();
    }

    public override void RaporOlustur()
    {
        string outputName = "rapor_" + raporBilgi.ad + raporBilgi.soyad + "_" + DateTime.Now.ToString("ddMMyyyy") + ".json";
        string dirParameter = AppDomain.CurrentDomain.BaseDirectory + @"\\JSON Raporlar\\" + outputName;

        StringBuilder sb = new StringBuilder();
        sb.Append("[");
        sb.Append(getHastaBilgileri());
        sb.AppendLine(",");
        sb.Append(getTedaviBilgileri());
        sb.AppendLine("]");
        SaveFile(dirParameter, sb.ToString());
        System.Windows.Forms.MessageBox.Show(dirParameter + "\\nKonumuna rapor oluşturulmuştur.");
    }

    public override void RaporOlusturTers()
    {
        string outputName = "rapor_Ters_" + raporBilgi.ad + raporBilgi.soyad + "_" + DateTime.Now.ToString("ddMMyyyy") + ".json";
        string dirParameter = AppDomain.CurrentDomain.BaseDirectory + @"\\JSON Raporlar\\" + outputName;
        StringBuilder sb = new StringBuilder();

        sb.Append("[");
        sb.Append(getTedaviBilgileri());
        sb.AppendLine(",");
        sb.Append(getHastaBilgileri());
        sb.AppendLine("]");
        SaveFile(dirParameter, sb.ToString());
        System.Windows.Forms.MessageBox.Show(dirParameter + "\\nKonumuna rapor oluşturulmuştur.");
    }
}
```

## RaporAbstraction Sınıfı

```
public abstract class RaporAbstraction
{
    public RaporImplementor _raporOlusturucu;
    public abstract void RaporOlustur();
    public abstract void RaporOlusturTers();
}
```

## RefinedRaporAbstraction Sınıfı

```
public class RefinedRaporAbstraction : RaporAbstraction
{
    public override void RaporOlustur()
    {
        this._raporOlusturucu.RaporOlustur();
    }

    public override void RaporOlusturTers()
    {
        this._raporOlusturucu.RaporOlusturTers();
    }
}
```

## RaporBilgi Sınıfı

```
public class RaporBilgi
{
    public string ad;
    public string soyad;
    public string tckno;
    public string hastalikTuru;
    public string diyetTuru;
    public string diyetBilgi;
}
```

**Açıklama:** *RaporBilgi* sınıfı raporlama somut sınıfları tarafından raporlanacak bilgileri tutmak için kullanılır.

## Veriler Sınıfı

Veriler static sınıfı uygulama verilerini veritabanından çekmek için tasarlanmıştır. Fakat bu örneğimizde verileri run-time’da tuttuğumuz için verileri bellekteki liste yapılarından çekiyoruz.

```
public static class Veriler
{
    public static List<Admin> adminler = new List<Admin>();
    public static List<Diyetisyen> diyetisyenler = new List<Diyetisyen>();
    public static List<Hasta> hastalar = new List<Hasta>();
    public static Diyetisyen getDiyetisyen(string kullanicAdi)
    {
        try
        {
            var _diyetisyen = (from k in Veriler.diyetisyenler where k.kullanicAdi == kullanicAdi select k).ToList()[0];
            return _diyetisyen;
        }
        catch (Exception)
        {
            return null;
        }
    }

    public static bool EkleDiyetisyen(Diyetisyen _diyetisyen)
    {
        bool eklendiMi = false;
        //Bak bakayım aynı kullanicAdini kullanan baska bir diyetisyen var mi.
        var diyetisyen = Veriler.getDiyetisyen(_diyetisyen.kullanicAdi);
        //Eger zaten öyle bir diyetisyen yoksa ekle.
        if (diyetisyen == null)
        {
            Veriler.diyetisyenler.Add(_diyetisyen);
            eklendiMi = true;
        }
        return eklendiMi;
    }

    public static Hasta getHasta(string kullanicAdi)
    {
        try
        {
            var _hasta = (from k in Veriler.hastalar where k.kullanicAdi == kullanicAdi select k).ToList()[0];
            return _hasta;
        }
        catch (Exception)
        {
            return null;
        }
    }

    public static List<Hasta> getHastalar(string diyetisyenKullanicAdi)
    {
        try
        {
            var _hasta = (from k in Veriler.hastalar
                           where k.diyetisyenAdi == diyetisyenKullanicAdi
                           select k).ToList();

            return _hasta;
        }
        catch (Exception)
        {
            return null;
        }
    }

    public static bool EkleHasta(Hasta _hasta)
    {
        bool eklendiMi = false;
        //Bak bakayım aynı kullanicAdini kullanan baska bir hasta var mi.
        var diyetisyen = Veriler.getHasta(_hasta.kullanicAdi);
        //Eger zaten öyle bir hasta yoksa ekle.
        if (diyetisyen == null)
        {
            Veriler.hastalar.Add(_hasta);
            eklendiMi = true;
        }
        return eklendiMi;
    }

    public static Admin getAdmin(string kullanicAdi)
    {
        try
        {
            var _admin = (from k in Veriler.adminler where k.kullanicAdi == kullanicAdi select k).ToList()[0];
            return _admin;
        }
        catch (Exception)
        {
            return null;
        }
    }
}
```

## Kullanıcı Sınıfı

```
public abstract class Kullanici
{
    public string kullaniciAdi { get; protected set; }
    protected string parola = null;
    public string ad = null;
    public string soyad = null;
    public string tckno = null;
    public Kullanici(string _kullaniciAdi, string _parola)
    {
        this.kullaniciAdi = _kullaniciAdi;
        this.parola = _parola;
    }
    public virtual bool GirisYap(string _parola)
    {
        bool sonuc = this.parola == _parola;
        return sonuc;
    }
}
```

## Hasta Sınıfı

```
public class Hasta : Kullanici
{
    private HastalikCesidi hastalik;
    private DiyetYontem diyet;
    public string diyetisyenAdi;

    public Hasta(string _kullaniciAdi, string _parola, IHastalik _hastalik) : base(_kullaniciAdi, _parola)
    {
        this.HastalikDegis(_hastalik);
    }
    public void HastalikDegis(IHastalik _hastalik)
    {
        this.hastalik = new HastalikCesidi(_hastalik);
    }
    public string HastalikAdi()
    {
        return this.hastalik.HastalikAdi();
    }
    public void DiyetDegis(IDiyet _diyet)
    {
        this.diyet = new DiyetYontem(_diyet);
    }
    public string DiyetBilgi()
    {
        return this.diyet.DiyetBilgi();
    }
    public string DiyetAdi()
    {
        return this.diyet.DiyetAdi();
    }
    public void Raporla(RaporImplementor _raporOlusturucu)
    {
        RaporAbstraction raporOlusturucu = new RefinedRaporAbstraction();
        raporOlusturucu._raporOlusturucu = _raporOlusturucu;
        raporOlusturucu.RaporOlustur();
    }
    public void RaporlaTers(RaporImplementor _raporOlusturucu)
    {
        RaporAbstraction raporOlusturucu = new RefinedRaporAbstraction();
        raporOlusturucu._raporOlusturucu = _raporOlusturucu;
        raporOlusturucu.RaporOlusturTers();
    }
}
```



## Diyetisyen Sınıfı

```
public class Diyetisyen : Kullanici
{
    public List<Hasta> hastalar;
    public Diyetisyen(string kullanıcıAdi, string parola) : base(kullanıcıAdi, parola)
    {
        hastalar = Veriler.getHastalar(this.kullanıcıAdi);
    }

    public bool HastaEkle(Hasta hasta)
    {
        bool eklendiMi = false;

        hasta.diyetisyenAdi = this.kullanıcıAdi;
        eklendiMi = Veriler.EkleHasta(hasta);
        //Eger verilere hasta eklendiyse, yereldeki listeye de ekle.
        if (eklendiMi)
            this.hastalar.Add(hasta);

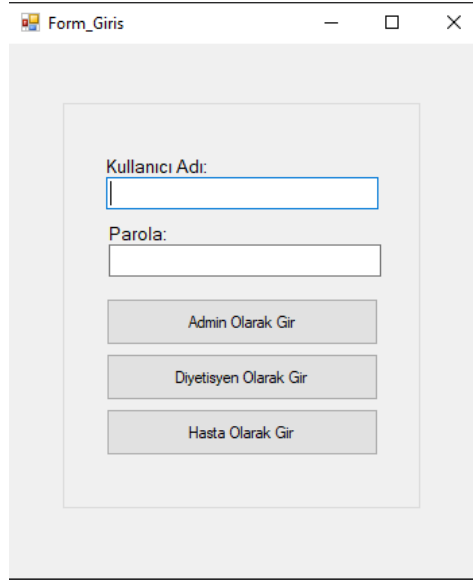
        return eklendiMi;
    }
}
```

## Admin Sınıfı

```
public class Admin : Kullanici
{
    public Admin(string _kullanıcıAdi, string _parola) : base(_kullanıcıAdi, _parola) { }
    public bool DiyetisyenEkle(string _kullanıcıAdi, string _parola, string _tckno, string _ad, string _soyad)
    {
        bool eklendiMi = false;
        Diyetisyen diyetisyen = new Diyetisyen(_kullanıcıAdi, _parola)
        {
            tckno = _tckno,
            ad = _ad,
            soyad = _soyad
        };

        eklendiMi = Veriler.EkleDiyetisyen(diyetisyen);
        return eklendiMi;
    }
}
```

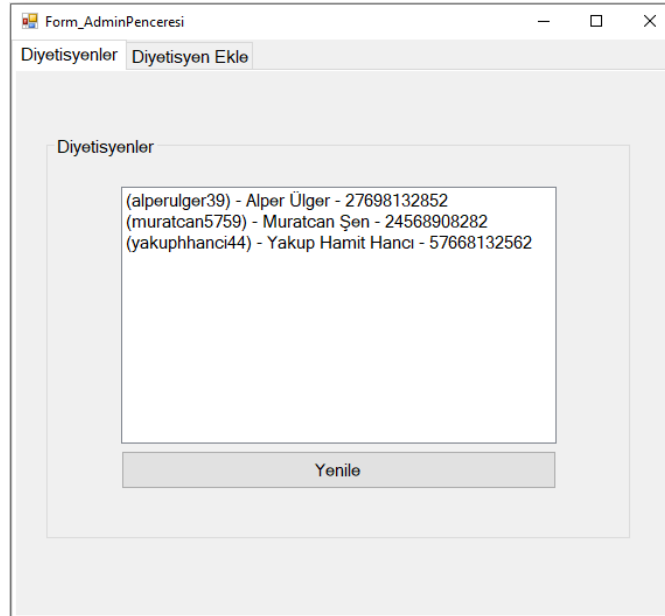
## Ekran Görüntüleri



The screenshot shows a Windows application window titled "Form\_Giris". Inside the window, there is a central panel with the following elements:

- A label "Kullanıcı Adı:" followed by a text input field.
- A label "Parola:" followed by a password input field.
- Three buttons stacked vertically: "Admin Olarak Gir", "Diyetisyen Olarak Gir", and "Hasta Olarak Gir".

Şekil 2 – Giriş Ekranı



The screenshot shows a Windows application window titled "Form\_AdminPenceresi". It has two tabs: "Diyetisyenler" (selected) and "Diyetisyen Ekle". The main content area of the "Diyetisyenler" tab is titled "Diyetisyenler" and contains a list of three dietitians:

- (alperulger39) - Alper Ülger - 27698132852
- (muratcan5759) - Muratcan Şen - 24568908282
- (yakuphanci44) - Yakup Hamit Hancı - 57668132562

Below the list is a button labeled "Yenile".

Şekil 3 – Admin Ekranı - 1

Form\_AdminPenceresi

Diyetisyenler Diyetisyen Ekle

Diyetisyen Ekle

Kullanıcı Adı:

Parola:

Adı:

Soyadı:

TC Kimlik Numarası:

Sisteme Diyetisyen Ekle

Şekil 4 – Admin Ekranı - 2

Diyetisyen Paneli

Yakup Hamit Hancı - (yakuphanci44)

Hastalar Hasta Ekle

Hasta Ekle

Kullanıcı Adı:

Parola:

Adı:

Soyadı:

TC Kimlik Numarası:

Hastalık:

Diyet:

Sisteme Hasta Ekle


Başarılı

Hasta sisteme başarılı bir şekilde eklendi.

Tamam


Şekil 5 – Diyetisyen Ekranı - 1

Diyetisyen Paneli




Yakup Hamit Hancı - (yakuphanci44)


HastalarHasta Ekle




**Hasta Adı:**  
Refika Aymış - (0906202104)  
**Hasta TCKNO:**  
5937425231  
**Hastalık:**  
Şeker  
**Diyet:**  
Yeşillikler Dünyası



**Hasta Adı:**  
Meliha Taymış - (0906202105)  
**Hasta TCKNO:**  
39559085831  
**Hastalık:**  
Obez  
**Diyet:**  
Gluten Free



**Hasta Adı:**  
Sabiha Yanmış - (0906202106)  
**Hasta TCKNO:**  
79834785351  
**Hastalık:**  
Çölyak  
**Diyet:**  
Akdeniz



**Hasta Adı:**  
Cansur Gültekin - (cansurable)  
**Hasta TCKNO:**  
...

Yenile

Şekil 6 – Diyetisyen Ekranı - 2

Hasta Bilgileri

Hasta Kişisel Bilgileri

Adı:

Cansur

Soyadı:

Gültekin

TC Kimlik No:

590441181214

Hastalık:

Şeker

Değiştir

Kullanıcı Adı:

cansurable

Diyet Bilgileri

Diyet:

Gluten Free

Değiştir

Diyet Bilgi:

Glutensiz Diyet Listesi - Menü

Pazartesi  
Kahvaltı: Glutensiz ekmek, çiğ sebze(buharda yapılabilir), beyaz peynir, 2 tane kuru erik.  
Öğle: Izgara balık, rokalı salata, mısır ekmeği.

Rapor İşlemleri

Format

☒ HTML ☐ JSON

Düzen

☒ Hasta Bilgileri ☐ Diyet Bilgileri

☐ Diyet Bilgileri ☐ Hasta Bilgileri

Raporla

Şekil 8 – Diyetisyen Gözünden Hasta Ekranı

Hasta Bilgileri

Hasta Kişisel Bilgileri

Adı:

Cansur

Soyadı:

Gültekin

TC Kimlik No:

590441181214

Hastalık:

Şeker

Değiştir

Kullanıcı Adı:

cansurable

Diyet Bilgileri

Diyet:

Gluten Free

Değiştir

Diyet Bilgi:

Glutensiz Diyet Listesi - Menü

Pazartesi  
Kahvaltı: Glutensiz ekmek, çiğ sebze(buharda yapılabilir), beyaz peynir, 2 tane kuru erik.  
Öğle: Izgara balık, rokalı salata, mısır ekmeği.

Rapor İşlemleri

Format

☒ HTML ☐ JSON

Düzen

☒ Hasta Bilgileri ☐ Diyet Bilgileri

☐ Diyet Bilgileri ☐ Hasta Bilgileri

Raporla

Şekil 7 – Hasta Gözünden Hasta Ekranı