

# KÜTÜPHANE OTOMASYONU PROJESİ

## 1. Projenin Amacı ve İşlevselliği

Kütüphane otomasyon sisteminin amacı, kütüphanedeki tüm kitap, üye ve ödünç alma işlemlerini dijital ortamda yönetmektir. Bu, kütüphanecilere, üyelerine ve yöneticilere zaman kazandıracak ve veri kayıplarını en aza indirecektir. Sistemin temel işlevleri şunlar olacaktır.

- Kitapların eklenmesi, güncellenmesi ve silinmesi
- Üyelerin kaydı, güncellenmesi ve silinmesi
- Kitap ödünç alma ve iade işlemleri
- Kitapların stok takibi
- Kitap ve üye verilerinin raporlanması

Bu sistemin kullanıcılara sunduğu temel faydalar şunlar olacaktır:

- Kütüphanede bulunan kitapların düzenli takibi
- Üyelerin ödünç alma işlemlerinin izlenmesi
- Hızlı veri erişimi ve işlemler
- Kütüphane yönetiminden daha verimli bir iş akışı

## 2. Projede Kullanılacak Teknolojiler:

MySQL, C#

## 3. Projenin Modülleri (Form Sayfaları)

Projenin temel modülleri veya form sayfaları aşağıdaki gibi olabilir:

### A. Ana Menü Formu

- Sistemin başlangıç noktasıdır.
- Kullanıcı, diğer modüllere buradan erişir.
- "Kitaplar", "Üyeler", "Ödünç İşlemleri" gibi modüllere yönlendiren butonlar içerir.

### B. Kitaplar Formu

- Kitap ekleme, düzenleme ve silme işlemleri yapılabilir.
- Kitap adı, yazar, yayınevi, yayın yılı, kategori gibi bilgiler girilebilir.
- Kitaplar, bir liste veya tablo şeklinde görüntülenebilir.
- Arama fonksiyonu ile kitaplara hızlı erişim sağlanabilir.

### C. Üyeler Formu

- Üye kaydı yapılabilir, mevcut üyeler listelenebilir.
- Üye adı, soyadı, adres, telefon numarası, e-posta adresi gibi bilgilerin tutulması sağlanabilir.
- Üye silme, güncelleme işlemleri yapılabilir.

### D. Ödünç İşlemleri Formu

- Kitap ödünç alma ve iade işlemleri gerçekleştirilir.
- Ödünç alınan kitap ve üye bilgileri kaydedilir.
- İade tarihleri kontrol edilerek geç ödeme durumunda uyarılar verilir.
- Üyelerin ödünç alma geçmişi izlenebilir.

## E. Raporlar Formu

- Kitaplar ve üyeler hakkında raporlar oluşturulabilir.
- Ödünç alınan kitapların takibi, iade tarihlerinin geçip geçmediği raporlanabilir.
- Kütüphane yöneticisi için borç takibi, en çok ödünç alınan kitaplar gibi raporlar sunulabilir.

## F. Arama ve Filtreleme

- Kitaplar ve üyeler üzerinde arama yapılabilir.
- Kriterlere göre filtreleme (örneğin, kategori bazında kitap arama veya üyelerin durumuna göre filtreleme).

## 4. Veritabanı Yapısı

- **Kitaplar Tablosu:** Kitap ID, başlık, yazar, yayınevi, basım yılı, kategori gibi bilgiler içerir.
- **Öğrenciler Tablosu:** Öğrenci ID, adı, soyadı, okul numarası, adres, telefon, e-posta gibi bilgiler içerir.
- **Ödünç Kitaplar Tablosu:** Ödünç alınan kitaplar ile ilgili öğrenci ID, kitap ID, ödünç alınan tarih, iade tarihi gibi veriler yer alır.

## 5. Raporlama ve İstatistikler

- Detaylı raporlama özellikleri eklenebilir. Örneğin, kitap kategorileri bazında popülerlik analizi, üye bazında ödünç alma sıklığı gibi istatistikler oluşturulabilir.

## 6. Sonuç ve Değerlendirme

Kütüphane otomasyonu projesi, kütüphanedeki tüm işlemleri dijitalleştirerek verimliliği artırır ve kütüphane yönetimini kolaylaştırır. Kitapların ve öğrenci bilgilerin doğru bir şekilde yönetilmesini ve ödünç işlemlerinin etkin bir şekilde izlenmesini sağlar.

Bu sistem ile hem kütüphanelerin günlük işleyişini iyileştirmek hem de kullanıcıların daha verimli bir şekilde kütüphaneyi kullanmalarını sağlamak için ideal bir çözüm olacaktır. C# kullanarak geliştirilecek olan bu proje, kullanıcı dostu bir arayüz ile kullanıcıya büyük kolaylık sağlayacaktır.

Bu proje, kütüphanelerin yönetim süreçlerini iyileştirecek ve kullanıcıların deneyimini geliştirecektir. Sistemin gelecekteki gelişmeleri, kullanıcı ihtiyaçlarına göre şekillendirilebilir ve daha fazla özellik eklenebilir.

## KÜTÜPHANE OTOMASYONU PROJESİ ZAMAN PLANLAMASI

10.03.2025 – 17.03.2025 →	Veri tabanı tasarımının oluşturulması
17.03.2025 – 24.03.2025 →	Kütüphane Otomasyonu Arayüzünün genel taslağının hazırlanması (Süreç içerisinde eklemeler yapılacak)
24.03.2025 – 31.03.2025 →	Ana Menü Formunun oluşturulması. Kitaplar Formunun oluşturulması (Kitap kaydı, listeleme, silme, güncelleme, arama) Veri tabanı ile c# bağlantısının kurulması
31.03.2025 – 07.04.2025 →	Üyeler Formunun oluşturulması. (Üye kaydı, listeleme, silme ve güncelleme)
07.04.2025 – 14.04.2025 →	Ödünç İşlemleri Formunun oluşturulması
14.04.2025 – 21.04.2025 →	Raporlar Bölümünün oluşturulması
21.04.2025 – 28.04.2025 →	Arama Filtreleme Bölümünün oluşturulması

# KÜTÜPHANE OTOMASYONU PROJESİ

## HAFTALIK SUNUM VE RAPORLAMA

### 1. HAFTA

1. **GİRİŞ:** Bu rapor, kütüphane otomasyonu projesi kapsamında oluşturulacak veri tabanının tasarımı ve uygulanması ile ilgili temel bilgileri içermektedir. Proje, öğrencilerin kitap alıp iade etme süreçlerini takip etmeyi ve kütüphane yönetimini kolaylaştırmayı amaçlamaktadır. Veri tabanı, MySQL kullanılarak tasarlanmıştır.

2. **VERİ TABANI TASARIMI:** Kütüphane otomasyonu veri tabanı, aşağıdaki üç temel tabloyu ve ilgili alan adlarını içermektedir:

✧ **Kitaplar Tablosu**

- Kitap ID (Primary Key)
- Kitap Adı
- Yazar
- Yayınevi
- Sayfa Sayısı
- Kategori (Kitap Türü)

✧ **Öğrenciler Tablosu**

- Öğrenci No (Primary Key)
- Adı
- Soyadı
- Sınıfı
- Cinsiyet
- Telefon

✧ **Ödünç Kitaplar Tablosu**

- Ödünç ID (Primary Key)
- Öğrenci No (Foreign Key)
- Kitap ID (Foreign Key)
- Ödünç Alınan Tarih
- İade Tarihi
- Açıklama

Bu tasarım, kitap ve öğrenci bilgilerinin düzenli saklanması ve ödünç alma-iade süreçlerinin takip edilmesini sağlar.

3. **TABLO TANIMLARI VE SQL KODLARI** Aşağıda her bir tablonun MySQL kullanılarak oluşturulmasına dair SQL kodları aşağıdaki gibidir:

#### Kitaplar Tablosu

```
1 • create table kitaplar(  
2   kitap_id int primary key auto_increment,  
3   tur_id tinyint not null,  
4   kitap_adi varchar(255) not null,  
5   yazar varchar(255) not null,  
6   yayinevi varchar(255) not null,  
7   sayfa_sayisi smallint not null  
8 );
```

#### Öğrenciler Tablosu

```
1 • create table ogrenciler(  
2   ogrenci_no int primary key,  
3   ad varchar(100) not null,  
4   soyad varchar(100) not null,  
5   sinif tinyint not null,  
6   cinsiyet varchar(10) not null,  
7   telefon varchar(15) not null  
8 );
```

#### Ödünç Kitaplar Tablosu

```
1 • create table odunc_kitaplar(  
2   id int primary key auto_increment,  
3   ogr_no int not null,  
4   kitap_id int not null,  
5   verilis_tarihi date not null,  
6   teslim_tarihi date,  
7   aciklama varchar(200)  
8 );
```

#### Kitap Türleri Tablosu

```
1 • create table kitap_turleri(  
2   tur_id tinyint primary key auto_increment,  
3   tur_adi varchar(40) not null  
4 );
```

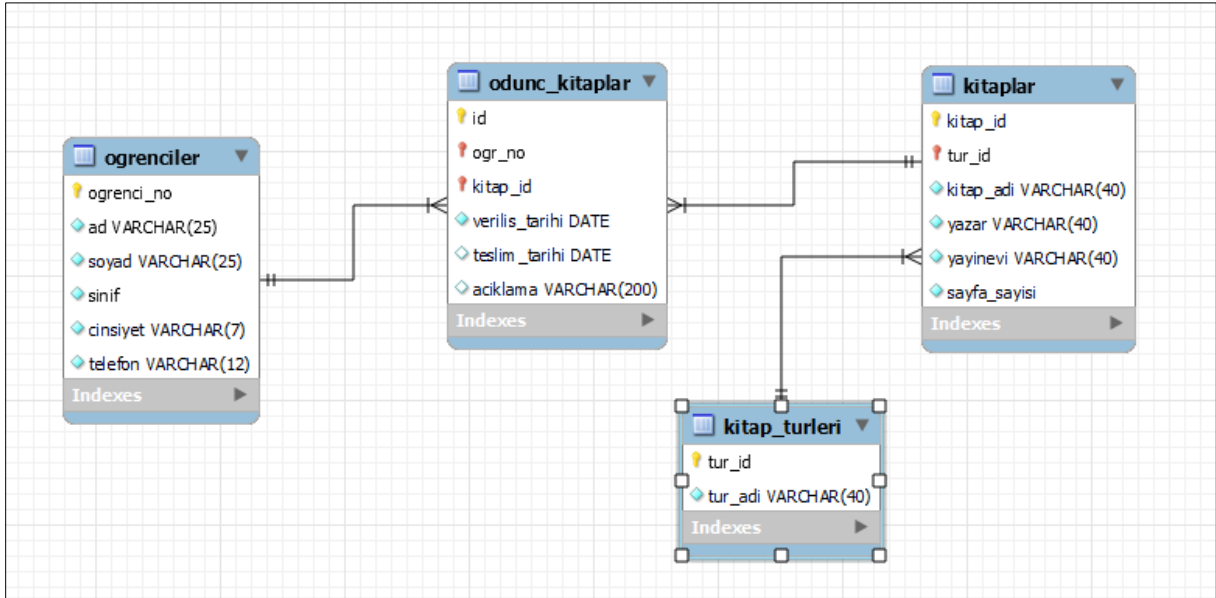
#### 4. İLİŞİKİSEL VERİ TABANI TASARIMI:

İlişkisel veri tabanı tasarımına uygun olarak bir MySQL EER (Enhanced Entity-Relationship) diyagramı aşağıdaki gibidir.

**Kitaplar** ve **Ödünç Kitaplar** tablosu arasında **1-N** ilişkisi vardır. (Bir kitap birden fazla kez ödünç alınabilir).

**Öğrenciler** ve **Ödünç Kitaplar** tablosu arasında **1-N** ilişkisi vardır. (Bir öğrenci birden fazla kitap ödünç alabilir).

Kitaplar tablosunda kullanılacak tür değerlerini içeren kitap\_turleri tablosu oluşturulmuştur. Kitaplar tablosundaki tur\_id alanına yalnızca bu tablodaki türlerden biri girilebilir. Böylelikle veri doğruluğu sağlanmış olur.



#### 5. ÖRNEK VERİ GİRİŞİ: Test amacıyla “*kitaplar, ogrenciler, kitap\_turleri ve odunc\_kitaplar*” tablolarına ilişkisel veri tabanına aşağıdaki örnek veriler eklenmiştir:

```
1 • insert into kitaplar(tur_id,kitap_adi,yazar,yayinevi,sayfa_sayisi)
2 values (1,"Suç ve Ceza","Dostoyevski","Deneme",687),
3 (1,"Çalılıkuşu","Reşat Nuri Gültekin","Örnek",544),
4 (1,"Sefiller","Victor Hugo","Örnek",520),
5 (1,"Küçük Ağa","Tarık Buğra","Deneme",477),
6 (2,"İnsan Neyle Yaşar?","L. N. Tolstoy","Deneme",96),
7 (2,"Ömer Seyfettin Hikayelerinden Seçmeler","Ömer Seyfettin","Deneme",176),
8 (3,"Otuz Beş Yaş","Cahit Sıtkı Tarancı","Örnek",120),
9 (3,"Safahat","Mehmet Akif Ersoy","Örnek",560),
10 (3,"Bütün Şiirleri - Orhan Veli","Orhan Veli Kanık","Örnek",247),
11 (4,"Seyahatname","Evliya Çelebi","Deneme",828)
```

```

1• insert into ogrenciler(ogrenci_no,ad,soyad,sinif,cinsiyet,telefon)
2 values (25,"Esat","Erkin","11","Erkek",5358425555),
3 (28,"Yakup","Bolat","11","Erkek",5362582525),
4 (61,"Esra","Öztürk","10","Kız",5552458888),
5 (45,"Ayşe","Yılmaz","9","Kız",5403528877),
6 (152,"Zeynep","Öztürk","10","Kız",5522369955),
7 (65,"Ali","Kılıç","9","Erkek",5369874521)

```

```

1• insert into kitap_turleri(tur_adi)
2 values("Roman"),
3 ("Hikaye"),
4 ("Şiir"),
5 ("Gezi"),
6 ("Çocuk"),
7 ("Kişisel Gelişim"),
8 ("Sağlık")

```

```

1• insert into odunc_kitaplar(ogr_no,kitap_id,verilis_tarihi,teslim_tarihi,aciklama)
2 values(25,18,"2025-03-10","2025-03-20",""),
3 (28,21,"2025-03-10","2025-03-20","Zamanında getirmeli"),
4 (45,20,"2025-03-12","2025-03-22","Kitabı paylaşmak gerek"),
5 (152,24,"2025-03-12","2025-03-22",""),
6 (152,19,"2025-03-13","2025-03-15",""),
7 (28,25,"2025-03-13","2025-03-20","")

```

Tablolardan biri olan “**odunc\_kitaplar**” tablosuna girilen değerlerin doğruluğu aşağıdaki SQL komutuyla test edildi. Ayrıca diğer tabloların da doğruluğu test edildi.

```
1• SELECT * FROM kutuphane.odunc_kitaplar;
```

Result Grid   Filter Rows:   Edit:   Export/Import:   Wrap Cell Content:						
	id	ogr_no	kitap_id	verilis_tarihi	teslim_tarihi	aciklama
▶	1	25	18	2025-03-10	2025-03-20	
	2	28	21	2025-03-10	2025-03-20	Zamanında getirmeli
	3	45	20	2025-03-12	2025-03-22	Kitabı paylaşmak gerek
	4	152	24	2025-03-12	2025-03-22	
	5	152	19	2025-03-13	2025-03-15	
	6	28	25	2025-03-13	2025-03-20	
	7	25	17	2025-03-15	2025-03-25	Eski kitap verildi

6. **SONUÇ VE GELECEK GELİŞTİRMELER:** Bu veri tabanı tasarımı, kütüphane işleyişini kolaylaştırmak ve kitap takibini düzenlemek için temel bir alt yapı sunmaktadır. Gelecekte, sisteme eklemeler yapılabilir: (Yayınevi, Kitap stok takibi)

# KÜTÜPHANE OTOMASYONU PROJESİ

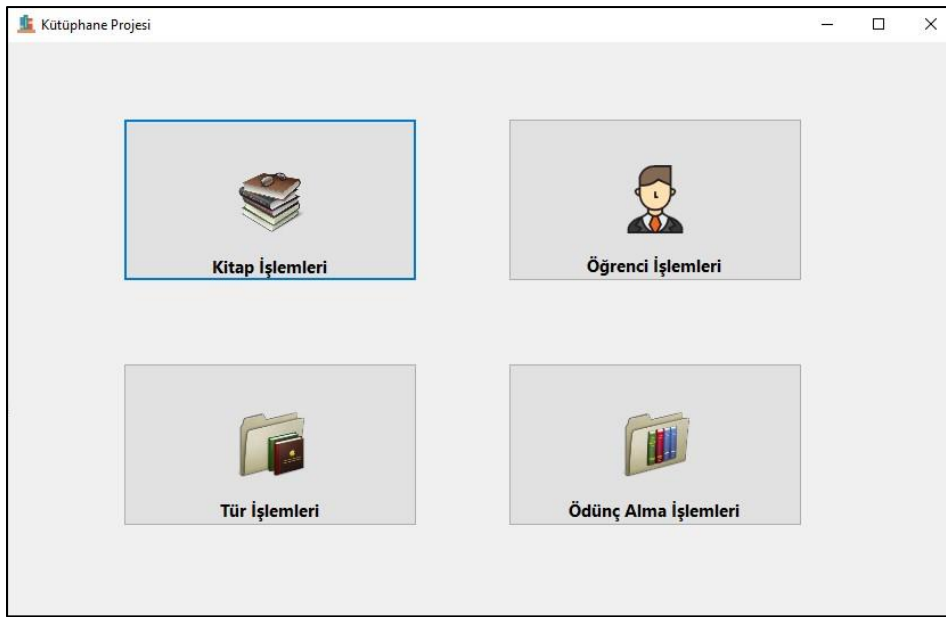
## HAFTALIK SUNUM VE RAPORLAMA

### 2. HAFTA

1. **GİRİŞ:** Bu rapor, Kütüphane Otomasyonu Arayüzünün genel taslağını içermektedir. **Ana Form, Kitap İşlemleri, Öğrenci İşlemleri, Ödünç İşlemleri ve Tür İşlemleri bölümlerinin** bulunduğu sayfalarla ilgili yapılan görsel çalışmaları içermektedir. Arayüz tasarımında kullanılmak üzere Geliştirme ortamı olarak(IDE) **Visual Studio 2022**, programlama dili olarak **C#** kullanılmıştır.

#### 2. KÜTÜPHANE OTOMASYONU ARAYÜZ TASARIMI:

##### A. Proje Ana Sayfasının Tasarımı



Projede yönlendirme işlemlerinin yapılması için ilk olarak bir Ana Sayfa hazırlanır. Bu sayfa içinde diğer sayfaların açılmasını sağlayacak butonlar bulunur.

Form nesnesinde ve içindeki form elemanlarında bazı özelliklerin aşağıdaki gibidir.

##### Form Nesnesi:

"Text" özelliği	: Kütüphane Projesi
"Name" özelliği	: formAnaSayfa
"Size" özelliği	:850;615
"FormBorderStyle" özelliği	: FixedSingle
"StartPosition" özelliği	: CenterScreen

##### Button Nesneleri

"Text" özellikleri	: Kitap İşlemleri, Öğrenci İşlemleri, Tür İşlemleri ve Ödünç Alma İşlemleri
"Name" özellikleri	: btnKitap, btnOgrenci, btnTur, btnOdunc
"Size" özellikleri	: 260;140

## B. Kitap İşlemleri Sayfasının Tasarımı

Kitap İşlemleri sayfası; kütüphanedeki kitaplar ile ilgili ekleme, silme, güncelleme, listeleme ve arama gibi işlemlerin yapılacağı sayfadır. Formun alt bölümündeki “DataGridView” nesnesi, verilerin listelenmesi için kullanılacaktır.

Form nesnesinde ve içindeki form elemanlarında bazı özelliklerin aşağıdaki gibidir.

### Form Nesnesi

“Text” özelliği	: Kitap İşlemleri
“Name” özelliği	: formKitap
“Size” özelliği	: 750;550
“FormBorderStyle” özelliği	: FixedSingle
“StartPosition” özelliği	: CenterScreen

### TextBox Nesneleri

“Name” özellikleri	: txtKitapAdi, txtYazar, txtYayinEvi, txtSayfaSayisi ve txtKitapAra
--------------------	---

### ComboBox Nesnesi

“Name” özelliği	: comboKitapTur
-----------------	-----------------

### Button Nesneleri

“Name” özellikleri	: btnKaydet, btnSil, btnGuncelle
“Text” özellikleri	: Kaydet, Sil, Güncelle
“Size” özellikleri	: 120,60

### DataGridView Nesnesi

“Name” özelliği	: gridKitap
“EditMode” özelliği	: EditProgrammatically
“SelectionMode” özelliği	: FullRowSelect
“AutoSizeColumnsMode”	: Fill



### C. Öğrenci İşlemleri Sayfasının Tasarımı

Öğrenci İşlemleri sayfası; öğrenciler ile ilgili ekleme, silme, güncelleme, listeleme ve arama gibi işlemlerin yapılacağı sayfadır.

Form nesnesinde ve içindeki form elemanlarında bazı özelliklerin aşağıdaki gibidir.

#### Form Nesnesi

"Text" özelliği	: Öğrenci İşlemleri
"Name" özelliği	: formOgrenci
"Size" özelliği	: 750;600
"FormBorderStyle" özelliği	: FixedSingle
"StartPosition" özelliği	: CenterScreen

#### TextBox Nesneleri

"Name" özellikleri	: txtNo, txtAd, txtSoyad, txtTelefon ve txtOgrenciAra
--------------------	---

#### ComboBox Nesnesi

"Name" özelliği	: comboSinif, comboCinsiyet
-----------------	-----------------------------

#### Button Nesneleri

"Name" özellikleri	: btnKaydet, btnSil, btnGuncelle
"Text" özellikleri	: Kaydet, Sil, Güncelle
"Size" özellikleri	: 120,60

#### DataGridView Nesnesi

"Name" özelliği	: gridOgrenci
"EditMode" özelliği	: EditProgrammatically
"SelectionMode" özelliği	: FullRowSelect
"AutoSizeColumnsMode"	: Fill

#### D. Kitap Tür İşlemleri Sayfasının Tasarımı

Kitap Tür İşlemleri sayfası; kitap türleri ile ilgili ekleme, silme, güncelleme, listeleme gibi işlemlerin yapılacağı sayfadır.

Form nesnesinde ve içindeki form elemanlarında bazı özelliklerin aşağıdaki gibidir.

##### Form Nesnesi

"Text" özelliği	: Kitap Tür İşlemleri
"Name" özelliği	: formKitapTur
"Size" özelliği	: 450;485
"FormBorderStyle" özelliği	: FixedSingle
"StartPosition" özelliği	: CenterScreen

##### TextBox Nesneleri

"Name" özellikleri	: txtTurAdi
--------------------	-------------

##### Button Nesneleri

"Name" özellikleri	: btnKaydet, btnSil, btnGuncelle
"Text" özellikleri	: Kaydet, Sil, Güncelle
"Size" özellikleri	: 120,60

##### DataGridView Nesnesi

"Name" özelliği	: gridKitapTur
"EditMode" özelliği	: EditProgrammatically
"SelectionMode" özelliği	: FullRowSelect
"AutoSizeColumnsMode"	: Fill

## E. Ödünç Kitap İşlemleri Sayfasının Tasarımı

Ödünç Kitap İşlemleri sayfası, kütüphanedeki kitapların öğrencilere ödünç olarak verilmesi ve geri alınması işlemlerinin yapılacağı sayfadır. Ödünç kitap verme ve geri alma işlemlerinde tarih olarak, sistem tarafından o günün tarihi otomatik olarak verilecektir.

Form nesnesinde ve içindeki form elemanlarında bazı özelliklerin aşağıdaki gibidir.

### Form Nesnesi

"Text" özelliği	: Ödünç Kitap İşlemleri
"Name" özelliği	: formOduncKitap
"Size" özelliği	: 800;550
"FormBorderStyle" özelliği	: FixedSingle
"StartPosition" özelliği	: CenterScreen

### TextBox Nesneleri

"Name" özellikleri	: txtNo, txtAciklama ve txtAramaOgrenci
--------------------	---

### ComboBox Nesnesi

"Name" özelliği	: comboKitap
-----------------	--------------

### Button Nesneleri

"Name" özellikleri	: btnKitapVer, btnSil, btnKitapAl
"Text" özellikleri	: Kitap Ver, Sil, Kitap Al
"Size" özellikleri	: 120,60

### DataGridView Nesnesi

"Name" özelliği	: gridOduncKitaplar
"EditMode" özelliği	: EditProgrammatically
"SelectionMode" özelliği	: FullRowSelect
"AutoSizeColumnsMode"	: Fill

## “Image” Özellikleri

Form ve Butonlara eklenecek iconlar ücretsiz resimlerin olduğu <https://icon-icons.com/> web sayfasından indirilmiştir. İndirilen resimler araçlar menüsünden eklenen **ImageList** bileşeniyle birlikte Form nesnelere ve Butonlara eklenmiştir.

### 3. FORMLAR ARASI GEÇİŞ KODLARI

Oluşturulan Ana Form bölümünde butonlara tıklandığında ilgili sayfalara yönlendirme işleminin yapılması için butonların “click” olayına bazı kodların yazılması gerekir. Kodlar aşağıdaki gibidir.

```
private void btnOgrenci_Click(object sender, EventArgs e)
{
    formOgrenci ogrenci = new formOgrenci();
    ogrenci.ShowDialog();
}

private void btnTur_Click(object sender, EventArgs e)
{
    formKitapTur kitapTur = new formKitapTur();
    kitapTur.ShowDialog();
}

private void btnKitap_Click(object sender, EventArgs e)
{
    formKitap kitap = new formKitap();
    kitap.ShowDialog();
}

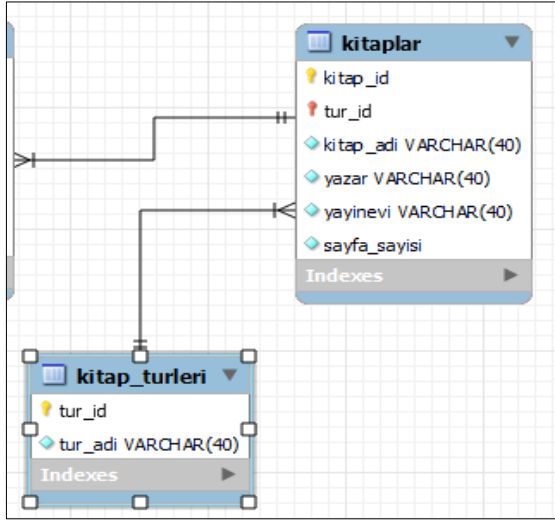
private void btnOdunc_Click(object sender, EventArgs e)
{
    formOduncKitap oduncKitap = new formOduncKitap();
    oduncKitap.ShowDialog();
}
```

# KÜTÜPHANE OTOMASYONU PROJESİ

## HAFTALIK SUNUM VE RAPORLAMA

### 3. HAFTA

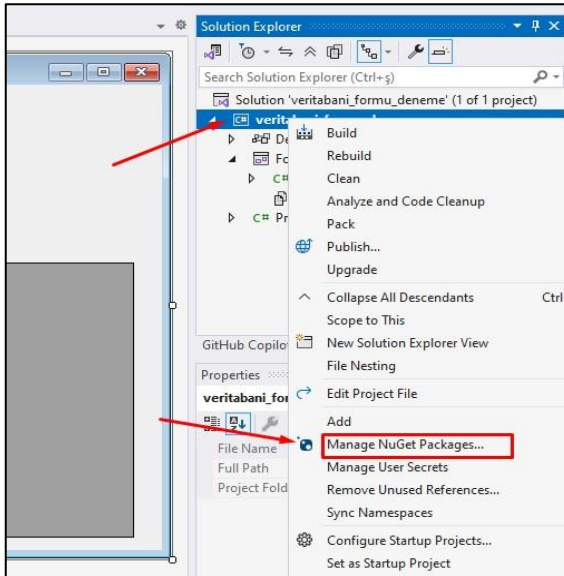
- GİRİŞ:** Bu rapor, **Ana Form** üzerinde bulunan **Kitap Tür İşlemleri formunun** veri tabanı ile bağlantı kurularak, Listeleme, Kitap Kaydı, Silme, Güncelleme, Arama işlemlerinin yapıldığı çalışmaları içermektedir.
- KİTAP TÜR İŞLEMLERİ FORM SAYFASI KODLAMALARI VE VERİ TABANI TASARIMI**



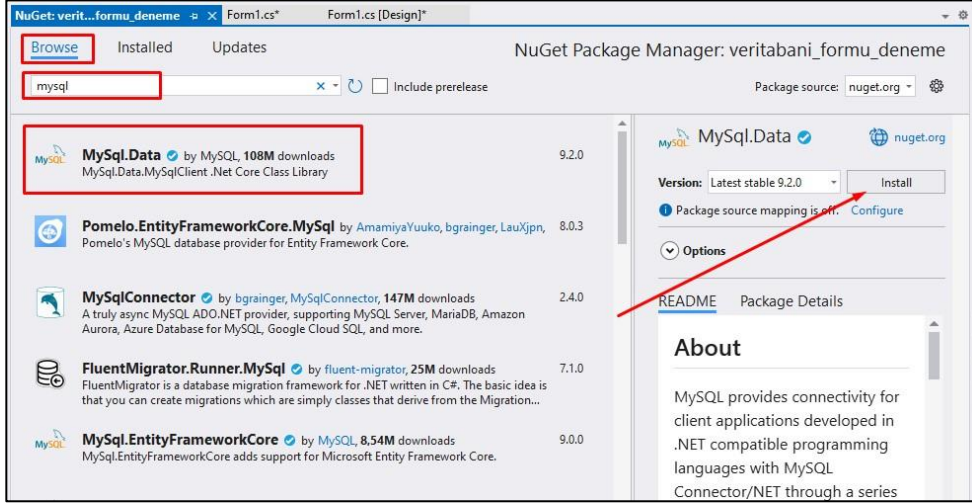
İlişkisel veri tabanında **kitaplar** tablosundaki foreign key alanı **kitap\_turleri** tablosundaki **primary key** alanına bağladığımız için öncelikle **Kitap Tür İşlemleri Form Sayfasını** tasarlamamız gerekmektedir.

#### a) MySql.Data paketinin yüklenmesi

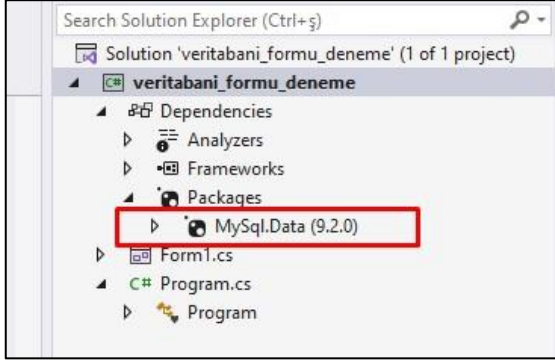
C# ile MySQL veri tabanı bağlantısı kurmak için **MySql.Data** Referansını projemize eklememiz gerekmektedir. **NuGet** Paket Yöneticisini kullanarak **MySql.Data** yazılımını projemize ekliyoruz.



Paket yöneticisi için Solution Explorer penceresinde proje üzerinde sağ tıklayarak **Manage NuGet Packages** tıklanır.

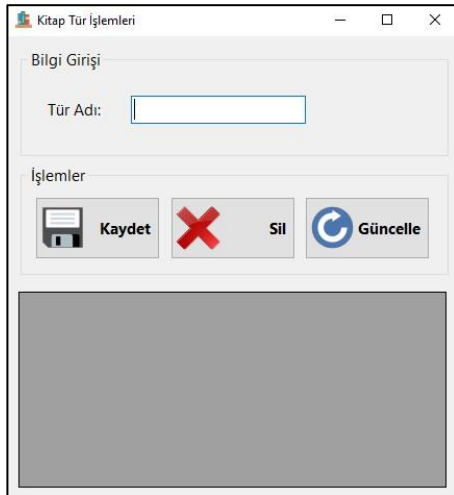


Açılan pencerede **Browse** sekmesinde *MySQL* araması yaparak *MySQL.Data* paketinin kurulumunu yapılır.



Bu işlem sonucunda *MySQL.Data* referansı projemize eklenmiş olacaktır.

#### b) Kitap Tür Formu ve Veri Tabanı Bağlantısı (Listeleme İşlemi)



\* *Önceki hafta oluşturulan raporda belirtildiği gibi yandaki gibi bir form oluşturulur.*

Kitap türlerini listeleme kodları **TurleriListele()** metodu içine yazılır ve bu metot, formun yüklenmesi olayında çağrılır. Bu işlemten sonra Veri tabanından çekilecek verilerin (kitap türlerinin) **DataGridView** nesnesinde listelendiği görülür

**TurleriListele()** fonksiyonunun içindeki kod bölümleriyle ilgili açıklamalar açıklama satırı olarak eklenmiştir.

```

private void Form1_Load(object sender, EventArgs e)
{
    TurleriListele();
}

2 references
public void TurleriListele()
{
    string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
    //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.

    MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

    string sorguCumlesi = "select * from kitap_turleri"; //verileri listeleyecek olan SQL sorgusu yazılır
    MySqlDataAdapter dataadapter = new MySqlDataAdapter(sorguCumlesi, baglanti); //dataadapter nesnesi oluşturulur
    DataTable dataTable = new DataTable(); //Verileri tablo hâlinde saklamada kullanılan nesnedir.

    dataadapter.Fill(dataTable); //sorgu sonucunda dönen kayıtlar dataTable nesnesine aktarılır
    gridKitapTur.DataSource = dataTable; //DataTable nesnesindeki kayıtlar datagridView de listelenir.

    gridKitapTur.Columns["tur_id"].HeaderText = "ID"; //DataGridView nesnesinde sütun başlıkları listelenir
    gridKitapTur.Columns["tur_id"].Width = 100; // Sütun genişliği belirlenir

    gridKitapTur.Columns["tur_adi"].HeaderText = "Tür Adı";
    gridKitapTur.Columns["tur_adi"].Width = 100;
}

```

```

using MySql.Data.MySqlClient;
using System.Data;
using System.Data.Common;

```

Yukarıdaki bazı kodların çalışması için yandaki kütüphanelerin eklenmesi otomatik olarak yapılır. Eklenme yapılmadıysa elle ekleme yapılır.

ID	Tür Adı
1	Roman
2	Hikaye
3	Şiir
4	Gezi
5	Çocuk
6	Kişisel Gelişim
7	Çocuk

Veri tabanıyla bağlantı kurulmuş olup yandaki gibi bir görsel elde edilir.

### c) Kitap Tür Formu Kayıt İşlemi

Kitap Tür işlemleri form ekranına tür adı bilgisi girişi yapıldıktan sonra “**Kaydet**” butonuna tıklandığında aşağıdaki kodlar çalıştırılır ve kitap türü kayıt işlemi gerçekleştirilir. Ekleme kodları “Kaydet” butonunun tıklanma olayına yazılır. Try-Catch bloğu içine yazılan kodlarda hata olması durumunda uyarı mesajı devreye girecektir.

```
private void btnKaydet_Click(object sender, EventArgs e)
{
    try
    {
        string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
        //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
        MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

        if (baglanti.State != ConnectionState.Open) //bağlantının durumu kontrol edilir
        {
            baglanti.Open(); //eğer bağlantı açık değilse açılır
        }

        string komutsatiri = "insert into kitap_turleri (tur_adi) values(@tur_adi)";
        MySqlCommand komut = new MySqlCommand(komutsatiri, baglanti); //komut çalıştırmak için MySqlCommand nesnesi oluşturulur

        komut.Parameters.AddWithValue("@tur_adi", txtTurAdi.Text); //Sorguda verilen parametrelerin değerleri belirlenir
        komut.ExecuteNonQuery(); //Ekleme sorgusu çalıştırılır ve hata oluşmazsa öğrenci eklenir
        baglanti.Close(); //Bağlantı kapatılır
        txtTurAdi.Clear();

        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        TurleriListele(); //Eklenen Kitap türlerinin DataGridView'de görülebilmesi için veriler tekrar listelenir
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

### d) Kitap Tür Formu Silme İşlemi

Kitap türünü **silme ve güncelleme** işlemleri yapılmadan önce kitap türünün **DataGridView** üzerinde seçilmesi ve o kayıt ile ilgili bilginin **TextBox** nesnesinde gösterilmesi gerekir. Bunun için nesnenin “CellClick” olayına aşağıdaki kodlar yazılır.

```
1 reference
private void gridKitapTur_CellClick(object sender, DataGridViewCellEventArgs e)
{
    try
    {
        //DataGridView'de seçili olan Kitap türüne ait bilgiler textBox içine yazdırılır
        txtTurAdi.Text = gridKitapTur.CurrentRow.Cells["tur_adi"].Value.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```



**DataGridView** üzerinde silinmesi istenen kitap türü seçildikten sonra o türün adı **TextBox** nesnesinde gösterilir. “**Sil**” butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve kayıt, veri tabanından silinir. Kodlar “**Sil**” butonunun tıklanma olayına yazılır.

**Ek bilgi:** “**Kitaplar**” ve “**kitap\_turleri**” tabloları arasında ilişki oluşturulduğu için kayıtlı bir kitaba ait tür sistemden silinemez.

```
1 reference
private void btnSil_Click(object sender, EventArgs e)
{
    try
    {
        string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
        //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
        MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

        if (baglanti.State != ConnectionState.Open) //Bağlantı durumu kontrol edilir
        {
            baglanti.Open(); //Eğer açık değilse bağlantı açılır
        }

        string komutsatiri = "delete from kitap_turleri where tur_id= @tur_id"; //@tur_id yerine parametre gelecektir.
        MySqlCommand komut = new MySqlCommand(komutsatiri, baglanti); //komut çalıştırmak için MySqlCommand nesnesi oluşturulur

        komut.Parameters.AddWithValue("@tur_id", gridKitapTur.CurrentRow.Cells["tur_id"].Value.ToString());
        //Üst satırda sorguya parametre olarak DataGridView'de seçili olan tur_id bilgisi gönderilir.

        komut.ExecuteNonQuery(); //Sorgu çalıştırılır ve hata oluşmazsa öğrenci silinir.
        baglanti.Close(); //Bağlantı kapatılır
        txtTurAdi.Clear();

        MessageBox.Show("İşlem başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        TurleriListele(); //Silinen kaydın görünmemesi için tekrar listeleme yapılır
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluşturdu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

#### e) Kitap Tür Formu Güncelleme İşlemi

DataGridView üzerinde güncellenmesi istenen kitap türü seçildikten sonra o türün adı TextBox nesne sinde gösterilir. Yeni değer girildikten sonra “**Güncelle**” butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve kitap türünü güncelleme işlemi gerçekleştirilir. Kodlar “**Güncelle**” butonunun tıklanma olayına yazılır.

1 reference

```
private void btnGuncelle_Click(object sender, EventArgs e)
{
    try
    {
        string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
        //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
        MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

        if (baglanti.State != ConnectionState.Open) //Bağlantı durumu kontrol edilir
        {
            baglanti.Open(); //Eğer açık değilse bağlantı açılır
        }

        string komutsatiri = "update kitap_turleri set tur_adi=@tur_adi where tur_id=@tur_id"; //@tur_id yerine parametre gelecektir.
        MySqlCommand komut = new MySqlCommand(komutsatiri, baglanti); //komut çalıştırmak için MySqlCommand nesnesi oluşturulur

        komut.Parameters.AddWithValue("@tur_id", int.Parse(gridKitapTur.CurrentRow.Cells["tur_id"].Value.ToString()));
        //Üst satırda sorguya parametre olarak DataGridView'de seçili olan tur_id bilgisi gönderilir.
        komut.Parameters.AddWithValue("@tur_adi", txtTurAdi.Text);

        komut.ExecuteNonQuery();
        baglanti.Close();
        txtTurAdi.Clear();
        MessageBox.Show("İşlem başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);

        TurleriListele();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

# KÜTÜPHANE OTOMASYONU PROJESİ

## HAFTALIK SUNUM VE RAPORLAMA

### 4. HAFTA

1. **GİRİŞ:** Bu rapor, **Ana Form** üzerinde bulunan **Kitap İşlemleri formunun** veri tabanı ile bağlantı kurularak, Listeleme, Kitap Kaydı, Silme, Güncelleme, Arama işlemlerinin yapıldığı çalışmaları içermektedir.

### 2. KİTAP İŞLEMLERİ FORM SAYFASI KODLAMALARI VE VERİ TABANI TASARIMI

#### a) Kitap İşlemleri Formu ve Veri Tabanı Bağlantısı (Listeleme İşlemi)

\* Yukarıdaki gibi bir form oluşturulur.

#### Kitap Türlerini (Combobox) Listeleme İşlemi

ComboBox nesnesinde kitap türlerinin listelenmesi için gerekli olan kodlar **KitapTurYukle()** metodunun içine yazılır. Daha sonra bu metod, formun yüklenmesi olayında çağrılır.

```

1 reference
private void formKitap_Load(object sender, EventArgs e)
{
    KitapTurYukle();
    KitapListele();
}

1 reference
public void KitapTurYukle()
{
    string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
    //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
    MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

    try
    {
        string sorguCumlesi = "select * from kitap_turleri"; //verileri listeleyecek olan SQL sorgusu yazılır
        MySqlDataAdapter dataadapter = new MySqlDataAdapter(sorguCumlesi, baglanti); //dataadapter nesnesi oluşturulur
        DataTable dataTable = new DataTable(); //Verileri tablo hâlinde saklamada kullanılan nesnedir.

        dataadapter.Fill(dataTable); //sorgu sonucunda dönen kayıtlar dataTable nesnesine aktarılır
        comboKitapTur.DataSource = dataTable; //ComboBox nesnesinin veri kaynağı ayarlanır.
        comboKitapTur.ValueMember = "tur_id"; //Arka planda tutulup veri tabanına kaydedilecek alan belirlenir.
        comboKitapTur.DisplayMember = "tur_adi"; //Kullanıcıya gösterilecek alan belirlenir
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

## Kitapların Listeleme İşlemi

DataGridView nesnesinde kitapların listelenmesi için gerekli olan kodlar **KitapListele()** metodunun içine yazılır. Daha sonra bu metot, formun yüklenmesi olayında çağrılır.

```

public void KitapListele()
{
    string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
    //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
    MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

    try
    {
        string komutsatiri = "select kitap_id,tur_adi,kitap_adi,yazar,yayinevi,sayfa_sayisi from kitaplar,kitap_turleri where kitaplar.tur_id=kitap_turleri.tur_id";
        MySqlDataAdapter dataAdapter = new MySqlDataAdapter(komutsatiri, baglanti);
        DataTable dataTable = new DataTable();
        dataAdapter.Fill(dataTable);
        gridKitap.DataSource = dataTable;

        gridKitap.Columns["kitap_id"].HeaderText = "ID";
        gridKitap.Columns["kitap_id"].Width = 20;
        gridKitap.Columns["tur_adi"].HeaderText = "Tür";
        gridKitap.Columns["tur_adi"].Width = 30;
        gridKitap.Columns["kitap_adi"].HeaderText = "Adı";
        gridKitap.Columns["kitap_adi"].Width = 90;
        gridKitap.Columns["yazar"].HeaderText = "Yazar";
        gridKitap.Columns["yazar"].Width = 80;
        gridKitap.Columns["yayinevi"].HeaderText = "Yayınevi";
        gridKitap.Columns["yayinevi"].Width = 80;
        gridKitap.Columns["sayfa_sayisi"].HeaderText = "Sayfa Sayısı";
        gridKitap.Columns["sayfa_sayisi"].Width = 50;
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

ID	Tür	Yazar	Yayınevi	Sayfa Sayısı	
1	Roman	Sabahattin Ali	Deneme	221	
2	Roman	Dostoyevski	Deneme	687	
3	Roman	Beyaz Gemi	Cengiz Aytmatov	Deneme	168
4	Roman	Sinekli Bakkal	Halide Edip Adivar	Örnek	476
5	Roman	Çalıkuşu	Reşat Nuri Gültekin	Örnek	544
6	Roman	Sefiller	Victor Hugo	Örnek	520
7	Hikaye	Ömer Seyfettin Hikayelerinden...	Ömer Seyfettin	Deneme	176
8	Roman	Küçük Ağa	Tarık Buğra	Deneme	477
9	Roman	Yaban	Yakup Kadri Karaosmanoğlu	Deneme	215

Bu işlemlerden sonra kitap bilgilerinin **DataGridView** nesnesinde, kitap türlerinin de **ComboBox** nesnesinde listelendiği görülür.

## b) Kitap İşlemleri Formu Kayıt İşlemi

Kitap İşlemleri form ekranına veri girişi yapıldıktan sonra “**Kaydet**” butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve kitap kayıt işlemi gerçekleştirilir. Kodlar “**Kaydet**” butonunun tıklanma olayına yazılır.

```
1 reference
private void btnKaydet_Click(object sender, EventArgs e)
{
    try
    {
        string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
        //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
        MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

        if (baglanti.State != ConnectionState.Open)//bağlantının durumu kontrol edilir
        {
            baglanti.Open(); //eğer bağlantı açık değilse açılır
        }

        string komutsatiri = "insert into kitaplar (tur_id,kitap_adi,yazar,yayinevi,sayfa_sayisi)" +
            "values (@tur_id,@kitap_adi,@yazar,@yayinevi,@sayfa_sayisi)";
        MySqlCommand komut = new MySqlCommand(komutsatiri, baglanti); //komut çalıştırmak için MySqlCommand nesnesi oluşturulur
        //alttaki satırda Combobox nesnesinin arka planda tuttuğu id değeri tur_id olarak kayıt edilir.

        komut.Parameters.AddWithValue("@tur_id", int.Parse(ComboBox1.SelectedItem.ToString()));
        komut.Parameters.AddWithValue("@kitap_adi", txtKitapAdi.Text);
        komut.Parameters.AddWithValue("@yazar", txtYazar.Text);
        komut.Parameters.AddWithValue("@yayinevi", txtYayinEvi.Text);
        komut.Parameters.AddWithValue("@sayfa_sayisi", int.Parse(txtSayfaSayisi.Text));
        komut.ExecuteNonQuery();
        baglanti.Close();
        Temizle();
        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        KitapListele();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

### 3. KİTAP İŞLEMLERİ FORM SAYFASI KODLAMALARI VE VERİ TABANI TASARIMI

#### c) Kitap İşlemleri Formu Silme İşlemi

Kitap silme ve güncelleme işlemleri yapılmadan önce işlem yapılacak kitabın **DataGridView** üzerinde seçilmesi ve o kayıt ile ilgili bilgilerin form elemanlarında gösterilmesi gerekir. Bunun için nesnenin "CellClick" olayına aşağıdaki kodlar eklenmiştir.

```
1 reference
private void gridKitap_CellClick(object sender, DataGridViewCellEventArgs e)
{
    try
    {
        //DataGridView'de seçili olan Kitaba ait bilgiler boş kutular içine yazdırılır
        txtKitapAdi.Text = gridKitap.CurrentRow.Cells["kitap_adi"].Value.ToString();
        txtSayfaSayisi.Text = gridKitap.CurrentRow.Cells["sayfa_sayisi"].Value.ToString();
        txtYayinevi.Text = gridKitap.CurrentRow.Cells["yayinevi"].Value.ToString();
        txtYazar.Text = gridKitap.CurrentRow.Cells["yazar"].Value.ToString();
        comboKitapTur.Text = gridKitap.CurrentRow.Cells["tur_adi"].Value.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluşturdu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

**DataGridView** üzerinde silinmesi istenen kitap seçildikten sonra o kitaba ait bilgiler form elemanların da gösterilir. "**Sil**" butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve kitap bilgileri veri tabanından silinir. Kodlar "**Sil**" butonunun tıklanma olayına yazılır. Ödünç verilen bir kitap tablolar arası bağlantıdan dolayı silinemez.

```
1 reference
private void btnSil_Click(object sender, EventArgs e)
{
    try
    {
        string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
        //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
        MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

        if (baglanti.State != ConnectionState.Open) //Bağlantı durumu kontrol edilir
        {
            baglanti.Open(); //Eğer açık değilse bağlantı açılır
        }

        string komutsatiri = "delete from kitaplar where kitap_id= @kitap_id"; //@kitap_id yerine parametre gelecektir.
        MySqlCommand komut = new MySqlCommand(komutsatiri, baglanti); //komut çalıştırmak için MySqlCommand nesnesi oluşturulur

        komut.Parameters.AddWithValue("@kitap_id", gridKitap.CurrentRow.Cells["kitap_id"].Value.ToString());
        //Üst satırda sorguya parametre olarak DataGridView'de seçili olan kitap_id bilgisi gönderilir.

        komut.ExecuteNonQuery(); //Sorgu çalıştırılır ve hata oluşmazsa öğrenci silinir.
        baglanti.Close(); //Bağlantı kapatılır
        Temizle();

        MessageBox.Show("İşlem başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        KitapListele(); //Silinen kayıttan görünmemesi için tekrar listeleme yapılır
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluşturdu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

#### d) Kitap İşlemleri Formu Güncelleme İşlemi

```
1 reference
private void btnGuncelle_Click(Object sender, EventArgs e)
{
    try
    {
        string baglanticumlesi = "Server=localhost;Database=kutuphane;Uid=root;Pwd=12345678";
        //MySQL veri tabanına bağlanmak için gerekli olan bağlantı cümlesidir.
        MySqlConnection baglanti = new MySqlConnection(baglanticumlesi); //Veri Tabanı bağlantısı oluşturuldu

        if (baglanti.State != ConnectionState.Open) //Bağlantı durumu kontrol edilir
        {
            baglanti.Open(); //Eğer açık değilse bağlantı açılır
        }
        //string komutsatiri = "update kitap_turleri set tur_adi=@tur_adi where tur_id=@tur_id";
        string komutsatiri = "update kitaplar set tur_id=@tur_id,kitap_adi=@kitap_adi," +
            "yazar=@yazar,yayinevi=@yayinevi,sayfa_sayisi=@sayfa_sayisi where kitap_id=@kitap_id";
        MySqlCommand komut = new MySqlCommand(komutsatiri, baglanti); //komut çalıştırmak için MySqlCommand nesnesi oluşturulur

        komut.Parameters.AddWithValue("@kitap_id", int.Parse(gridKitap.CurrentRow.Cells["kitap_id"].Value.ToString()));
        komut.Parameters.AddWithValue("@tur_id", int.Parse(comboKitapTur.SelectedValue.ToString()));
        komut.Parameters.AddWithValue("@kitap_adi", txtKitapAdi.Text);
        komut.Parameters.AddWithValue("@yazar", txtYazar.Text);
        komut.Parameters.AddWithValue("@yayinevi", txtYayinEvi.Text);
        komut.Parameters.AddWithValue("@sayfa_sayisi", int.Parse(txtSayfaSayisi.Text));

        komut.ExecuteNonQuery();
        baglanti.Close();
        Temizle();
        MessageBox.Show("İşlem başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);

        KitapListele();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

## KÜTÜPHANE OTOMASYONU PROJESİ

### HAFTALIK SUNUM VE RAPORLAMA

#### 5. HAFTA

1. **GİRİŞ:** Bu rapor, **Ana Form** üzerinde bulunan **Öğrenci İşlemleri formunun** veri tabanı ile bağlantı kurularak, Listeleme, Kitap Kaydı, Silme, Güncelleme, Arama işlemlerinin yapıldığı çalışmaları içermektedir.

#### 2. ÖĞRENCİ İŞLEMLERİ FORM SAYFASI KODLAMALARI VE VERİ TABANI TASARIMI

##### a) Öğrenci Formu ve Veri Tabanı Bağlantısı (Listeleme İşlemi)

The screenshot shows a Windows application window titled 'formOgrenci'. The window is divided into several sections. At the top, there is a 'Bilgi Girişi' (Information Entry) section with input fields for 'Okul no:', 'Ad:', 'Soyad:', 'Sınıf:', 'Cinsiyet:', and 'Telefon:'. Below this, there is an 'Arama' (Search) section with an 'Öğrenci Adı:' input field. To the right of the search field, there is an 'İşlemler' (Operations) section with three buttons: 'Kaydet' (Save) with a floppy disk icon, 'Sil' (Delete) with a red 'X' icon, and 'Güncelle' (Update) with a circular arrow icon. The bottom half of the window is a large, empty gray rectangular area, likely intended for a DataGridView to display the search results.

\* Yukarıdaki gibi bir form oluşturulur.

Öğrenci listeleme kodları bir metot içine yazılır ve bu metot, formun yüklenmesi olayında çağrılır.

Bu işlemten sonra öğrenci bilgilerinin DataGridView nesnesinde listelendiği görülür.



```

namespace kutuphane
{
    5 references
    public partial class formOgrenci : Form
    {
        1 reference
        public formOgrenci()
        {
            InitializeComponent();
        }

        VeriTaniIslemleri vtIslemleri = new VeriTaniIslemleri();
        MySqlConnection baglanti;
        string komutsatiri;
        MySqlCommand komut;

        1 reference
        private void formOgrenci_Load(object sender, EventArgs e)
        {
            Listele();
        }
    }
}

```

```

4 references
public void Listele()
{
    try
    {
        baglanti = vtIslemleri.baglan();
        komutsatiri = "Select * From ogrenciler";

        MySqlDataAdapter dataAdapter = new MySqlDataAdapter(komutsatiri, baglanti);
        DataTable dataTable = new DataTable();
        dataAdapter.Fill(dataTable);
        gridOgrenci.DataSource = dataTable;

        gridOgrenci.Columns["ogrenci_no"].HeaderText = "Öğrenci Numarası";
        gridOgrenci.Columns["ad"].HeaderText = "Ad";
        gridOgrenci.Columns["soyad"].HeaderText = "Soyad";
        gridOgrenci.Columns["sinif"].HeaderText = "Sınıf";
        gridOgrenci.Columns["cinsiyet"].HeaderText = "Cinsiyet";
        gridOgrenci.Columns["telefon"].HeaderText = "Telefon";
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

## b) Öğrenci İşlemleri Formu Kayıt İşlemi

Öğrenci İşlemleri form ekranına veri girişi yapıldıktan sonra **“Kaydet”** butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve öğrenci kayıt işlemi gerçekleştirilir. Kodlar **“Kaydet”** butonunun tıklanma olayına yazılır. Öğrenci eklenirken aynı numaraya sahip bir öğrenci daha kaydedilmek istenirse program hata verecektir. Bunun nedeni, **öğrenci\_no** alanının **PK** olarak belirlenmesidir. Kayıt işleminin başarılı olması için bilgilerin doğru bir şekilde girilmesi gerekir.

```

private void btnKaydet_Click(object sender, EventArgs e)
{
    try
    {
        if (baglanti.State != ConnectionState.Open)
        {
            baglanti.Open();
        }

        komutsatiri = "insert into ogrenciler (ogrenci_no,ad,soyad,sinif,cinsiyet,telefon) values(@no,@ad,@soyad,@sinif,@cinsiyet,@telefon)";
        komut = new MySqlCommand(komutsatiri, baglanti);

        komut.Parameters.AddWithValue("@no", int.Parse(txtNo.Text));
        komut.Parameters.AddWithValue("@ad", txtAd.Text);
        komut.Parameters.AddWithValue("@soyad", txtSoyad.Text);
        komut.Parameters.AddWithValue("@sinif", int.Parse(comboSinif.SelectedItem.ToString()));
        komut.Parameters.AddWithValue("@cinsiyet", comboCinsiyet.SelectedItem.ToString());
        komut.Parameters.AddWithValue("@telefon", txtTelefon.Text);

        komut.ExecuteNonQuery();
        baglanti.Close();
        Temizle();
        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        ListeLe();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

3 references

```

public void Temizle()
{
    txtAd.Clear();
    txtSoyad.Clear();
    txtNo.Clear();
    txtTelefon.Clear();
}

```

### c) Öğrenci İşlemleri Formu Silme İşlemi

Öğrenci silme ve güncelleme işlemleri yapılmadan önce öğrencinin DataGridView üzerinde seçilmesi ve o kayıt ile ilgili bilgilerin form elemanlarında gösterilmesi gerekir. Bunun için nesnenin “CellClick (Hücrenin tıklanması)” olayına aşağıdaki kodların yazılması gerekir.

```

private void gridOgrenci_CellClick(object sender, DataGridViewCellEventArgs e)
{
    try
    {
        txtNo.Text = gridOgrenci.CurrentRow.Cells["ogrenci_no"].Value.ToString();
        txtAd.Text = gridOgrenci.CurrentRow.Cells["ad"].Value.ToString();
        txtSoyad.Text = gridOgrenci.CurrentRow.Cells["soyad"].Value.ToString();
        txtTelefon.Text = gridOgrenci.CurrentRow.Cells["telefon"].Value.ToString();
        comboSinif.SelectedItem = gridOgrenci.CurrentRow.Cells["sinif"].Value.ToString();
        comboCinsiyet.SelectedItem = gridOgrenci.CurrentRow.Cells["cinsiyet"].Value.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

DataGridView üzerinde silinmesi istenen öğrenci seçildikten sonra o öğrenciye ait bilgiler form elemanlarında gösterilir. “Sil” butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve öğrenci bilgileri veri tabanından silinir. Kodlar “Sil” butonunun tıklanma olayına yazılır. **“Oğrenciler”** ve **“odunc\_kitaplar”** tabloları arasında ilişki oluşturulduğu için ödünç kitap alan öğrenci sistemden silinemez.

```
private void btnSil_Click(object sender, EventArgs e)
{
    try
    {
        if (baglanti.State != ConnectionState.Open)
        {
            baglanti.Open();
        }

        komutsatiri = "delete from ogrenciler where ogrenci_no=@no";
        komut = new MySqlCommand(komutsatiri, baglanti);
        komut.Parameters.AddWithValue("@no", gridOgrenci.CurrentRow.Cells["ogrenci_no"].Value.ToString());

        komut.ExecuteNonQuery();

        baglanti.Close();
        Temizle();
        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        Listele();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

#### d) Öğrenci İşlemleri Formu Güncelleme İşlemi

DataGridView üzerinde güncellenmesi istenen öğrenci seçildikten sonra o öğrenciye ait bilgiler form elemanlarında gösterilir. Yeni değerler girildikten sonra **“Güncelle”** butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve öğrenci bilgileri güncellenir. Kodlar **“Güncelle”** butonunun tıklanma olayına yazılır.

```
private void btnGuncelle_Click(object sender, EventArgs e)
{
    try
    {
        if (baglanti.State != ConnectionState.Open)
        {
            baglanti.Open();
        }

        komutsatiri = "update ogrenciler set ad=@ad,soyad=@soyad,sinif=@sinif,cinsiyet=@cinsiyet,telefon=@telefon where ogrenci_no=@no";
        komut = new MySqlCommand(komutsatiri, baglanti);

        komut.Parameters.AddWithValue("@no", int.Parse(gridOgrenci.CurrentRow.Cells["ogrenci_no"].Value.ToString()));
        komut.Parameters.AddWithValue("@ad", txtAd.Text);
        komut.Parameters.AddWithValue("@soyad", txtSoyad.Text);
        komut.Parameters.AddWithValue("@sinif", int.Parse(comboSinif.SelectedItem.ToString()));
        komut.Parameters.AddWithValue("@cinsiyet", comboCinsiyet.SelectedItem.ToString());
        komut.Parameters.AddWithValue("@telefon", txtTelefon.Text);

        komut.ExecuteNonQuery();
        baglanti.Close();
        Temizle();
        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
        Listele();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

#### e) Öğrenci İşlemleri Formu Arama İşlemi

Form üzerindeki arama bölümünde öğrenci adına göre arama yapılabilmesini sağlayan kodlar aşağıdaki gibidir. Kodlar “**TextChanged**” olayına yazıldığı için arama kutusundaki her değişiklikte arama işlemi tekrarlanır ve sonuçlar DataGridView nesnesinde gösterilir.

```
private void txtOgrenciAra_TextChanged(object sender, EventArgs e)
{
    OgrenciArama(txtOgrenciAra.Text);
}

1 reference
public void OgrenciArama(string aranacakKelime)
{
    try
    {
        if (baglanti.State != ConnectionState.Open)
        {
            baglanti.Open();
        }
        komut = new MySqlCommand();
        komut.Connection = baglanti;
        komut.CommandText = "select * from ogrenciler where ad like '%" + aranacakKelime + "%'";

        MySqlDataAdapter dataAdapter = new MySqlDataAdapter(komut);
        DataTable dataTable = new DataTable();
        dataAdapter.Fill(dataTable);
        baglanti.Close();
        gridOgrenci.DataSource = dataTable;
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

## KÜTÜPHANE OTOMASYONU PROJESİ

### HAFTALIK SUNUM VE RAPORLAMA

#### 6. HAFTA

1. **GİRİŞ:** Bu rapor, **Ana Form** üzerinde bulunan **Ödünç Kitap İşlemleri formunun** veri tabanı ile bağlantı kurularak, Listeleme, Ödünç Kitap Kaydı, Silme, Kitap Geri Alma, Arama işlemlerinin yapıldığı çalışmaları içermektedir.

#### 2. ÖDÜNÇ KİTAP İŞLEMLERİ FORM SAYFASI KODLAMALARI VE VERİ TABANI TASARIMI

##### a) Ödünç Kitap İşlemleri Formu ve Veri Tabanı Bağlantısı (Listeleme İşlemi)

\* Yukarıdaki gibi bir form oluşturulur.

Ödünç Kitap İşlemleri listeleme kodları **Listele()** metodu içine yazılır ve bu metot, formun yüklenmesi olayında çağrılır. Ayrıca **KitapYukle()** metoduyla veri tabanındaki kitap bilgileri combobox a yüklenir. Bu işlemden sonra Ödünç Kitap bilgilerinin DataGridView nesnesinde listelendiği görülür.

```

1 reference
private void formOduncKitap_Load(object sender, EventArgs e)
{
    KitapYukle();
    Listele();
}

4 references
public void KitapYukle()
{
    try
    {
        baglanti = vtIslemleri.baglan();
        komutsatiri = "Select * From kitaplar where kitap_id not in (select kitap_id from odunc_kitaplar where teslim_tarihi IS NULL)";

        MySqlDataAdapter dataAdapter = new MySqlDataAdapter(komutsatiri, baglanti);
        DataTable dataTable = new DataTable();

        dataAdapter.Fill(dataTable);

        comboKitap.DataSource = dataTable;
        comboKitap.ValueMember = "kitap_id";
        comboKitap.DisplayMember = "kitap_adi";
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluřtu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

```

4 references
public void Listele()
{
    try
    {
        baglanti = vtIslemleri.baglan();
        komutsatiri = "Select id, ogrenci_no, ad, soyad, kitap_adi, verilis_tarihi, teslim_tarihi, aciklama " +
        "From kitaplar, ogrenciler, odunc_kitaplar " +
        "where ogr_no = ogrenci_no and kitaplar.kitap_id = odunc_kitaplar.kitap_id";

        MySqlDataAdapter dataAdapter = new MySqlDataAdapter(komutsatiri, baglanti);
        DataTable dataTable = new DataTable();
        dataAdapter.Fill(dataTable);
        gridOduncKitaplar.DataSource = dataTable;

        gridOduncKitaplar.Columns["id"].HeaderText = "ID";
        gridOduncKitaplar.Columns["id"].Width = 30;
        gridOduncKitaplar.Columns["ogrenci_no"].HeaderText = "Öğrenci No";
        gridOduncKitaplar.Columns["ogrenci_no"].Width = 40;
        gridOduncKitaplar.Columns["ad"].HeaderText = "Ad";
        gridOduncKitaplar.Columns["ad"].Width = 70;
        gridOduncKitaplar.Columns["soyad"].HeaderText = "Soyad";
        gridOduncKitaplar.Columns["soyad"].Width = 70;
        gridOduncKitaplar.Columns["kitap_adi"].HeaderText = "Kitap Adı";
        gridOduncKitaplar.Columns["kitap_adi"].Width = 100;
        gridOduncKitaplar.Columns["verilis_tarihi"].HeaderText = "Veriliř Tarihi";
        gridOduncKitaplar.Columns["verilis_tarihi"].Width = 70;
        gridOduncKitaplar.Columns["teslim_tarihi"].HeaderText = "Teslim Tarihi";
        gridOduncKitaplar.Columns["teslim_tarihi"].Width = 70;
        gridOduncKitaplar.Columns["aciklama"].HeaderText = "Açıklama";
        gridOduncKitaplar.Columns["aciklama"].Width = 150;
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluřtu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

## b) Ödünç Kitap Verme İşlemleri

Ödünç Kitap İşlemleri form ekranında öğrenci ve emanet verilecek kitap seçildikten sonra “**Kitap Ver**” butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır. Kodlar “**Kitap Ver**” butonunun tıklanma olayına yazılır.

```
1 reference
private void btnKitapVer_Click(object sender, EventArgs e)
{
    try
    {
        if (baglanti.State != ConnectionState.Open) baglanti.Open();

        komutsatiri = "insert into odunc_kitaplar (ogr_no,kitap_id,verilis_tarihi,aciklama)" +
            "values (@ogr_no,@kitap_id,@verilis_tarihi,@aciklama)";

        komut = new MySqlCommand(komutsatiri, baglanti);

        komut.Parameters.AddWithValue("@ogr_no", int.Parse(txtNo.Text));
        komut.Parameters.AddWithValue("@kitap_id", int.Parse(comboKitap.SelectedValue.ToString()));
        komut.Parameters.AddWithValue("@verilis_tarihi", DateTime.Now.ToString("yyyy/MM/dd"));
        komut.Parameters.AddWithValue("@aciklama", txtAciklama.Text);

        komut.ExecuteNonQuery();
        baglanti.Close();
        Temizle();
        KitapYukle();
        Listele();
        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

3 references
public void Temizle()
{
    txtNo.Clear();
    txtAciklama.Clear();
}
```

## c) Ödünç Kitap İşlemleri Formu Silme İşlemi

Ödünç Kitap İşlemleri silme ve güncelleme işlemleri yapılmadan önce verilmiş olan kitap kaydının DataGridView üzerinde seçilmesi ve o kayıt ile ilgili bilgilerin form elemanlarında gösterilmesi gerekir. Bunun için nesnenin “**CellClick** (Hücrenin tıklanması)” olayına aşağıdaki kodların yazılması gerekir.

```
private void gridOduncKitaplar_CellClick(object sender, DataGridViewCellEventArgs e)
{
    try
    {
        txtAçıklama.Text = gridOduncKitaplar.CurrentRow.Cells["aciklama"].Value.ToString();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

DataGridView üzerinde silinmesi istenen kayıt seçildikten sonra o kayıta ait bilgiler form elemanlarında gösterilir. “Sil” butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır ve öğrenci bilgileri veri tabanından silinir. Kodlar “Sil” butonunun tıklanma olayına yazılır.

```
private void btnSil_Click(object sender, EventArgs e)
{
    try
    {
        if (baglanti.State != ConnectionState.Open) baglanti.Open();
        komutsatiri = "delete from odunc_kitaplar where id=@id";
        komut = new MySqlCommand(komutsatiri, baglanti);

        komut.Parameters.AddWithValue("@id", gridOduncKitaplar.CurrentRow.Cells["id"].Value.ToString());

        komut.ExecuteNonQuery();
        baglanti.Close();
        Temizle();
        KitapYukle();
        Listele();
        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```

#### d) Ödünç Kitap İşlemleri Formu Geri Alma İşlemi

DataGridView üzerinde teslim alınacak ödünç kitap kaydı seçildikten sonra açıklama bölümü doldurularak “**Kitap Al**” butonu tıklandığında aşağıdaki kodlar çalıştırılır ve kitabı geri alma işlemi gerçekleştirilir.



```

private void btnKitapAl_Click(object sender, EventArgs e)
{
    try
    {
        if (gridOduncKitaplar.CurrentRow.Cells["teslim_tarihi"].Value.ToString() != string.Empty)
        {
            MessageBox.Show("Bu kitap teslim alınmış.", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
            return;
        }

        if (baglanti.State != ConnectionState.Open) baglanti.Open();

        komutsatiri = "update odunc_kitaplar set teslim_tarihi=@teslim_tarihi, aciklama=@aciklama where id=@id";
        komut = new MySqlCommand(komutsatiri, baglanti);

        komut.Parameters.AddWithValue("@id", int.Parse(gridOduncKitaplar.CurrentRow.Cells["id"].Value.ToString()));
        komut.Parameters.AddWithValue("@teslim_tarihi", DateTime.Now.ToString("yyyy/MM/dd"));
        komut.Parameters.AddWithValue("@aciklama", txtAciklama.Text);

        komut.ExecuteNonQuery();
        baglanti.Close();
        Temizle();
        KitapYukle();
        Listele();
        MessageBox.Show("İşlem Başarılı", "Mesaj", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

#### e) Ödünç Kitap İşlemleri Formu Arama İşlemi

Form üzerindeki arama bölümünde öğrenci adına göre arama yapılabilmesini sağlayan kodlar aşağıdaki gibidir. Kodlar **“TextChanged”** olayına yazıldığı için arama kutusundaki her değişiklikte arama işlemi tekrarlanır ve sonuçlar DataGridView nesnesinde gösterilir.

```

private void txtAramaOgrenci_TextChanged(object sender, EventArgs e)
{
    OduncKitapOgrenciArama(txtAramaOgrenci.Text);
}

1 reference
public void OduncKitapOgrenciArama(string aranacakkelime)
{
    try
    {
        if (baglanti.State != ConnectionState.Open) baglanti.Open();

        komutsatiri = "SELECT id, ogrenci_no, ad, soyad, kitap_adi, verilis_tarihi, teslim_tarihi, aciklama " +
            "FROM kitaplar, ogrenciler, odunc_kitaplar " +
            "WHERE ogr_no=ogrenci_no AND kitaplar.kitap_id=odunc_kitaplar.kitap_id AND ad LIKE '%" + aranacakkelime + "%'";

        MySqlDataAdapter dataAdapter = new MySqlDataAdapter(komutsatiri, baglanti);
        DataTable dataTable = new DataTable();
        dataAdapter.Fill(dataTable);

        gridOduncKitaplar.DataSource = dataTable;
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata oluştu", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

## KÜTÜPHANE OTOMASYONU PROJESİ

### HAFTALIK SUNUM VE RAPORLAMA

#### 7. HAFTA

1. **GİRİŞ:** Bu rapor, ayrı bir Giriş Formu ile Kullanıcı adı ve Şifre girişiyle veri tabanı ile bağlantı kurularak, diğer form sayfalarına erişim izni verilen işlemlerinin yapıldığı çalışmaları içermektedir.
2. **KÜTÜPHANE OTOMASYONU GİRİŞ PANELİ VERİ TABANI TASARIMI**

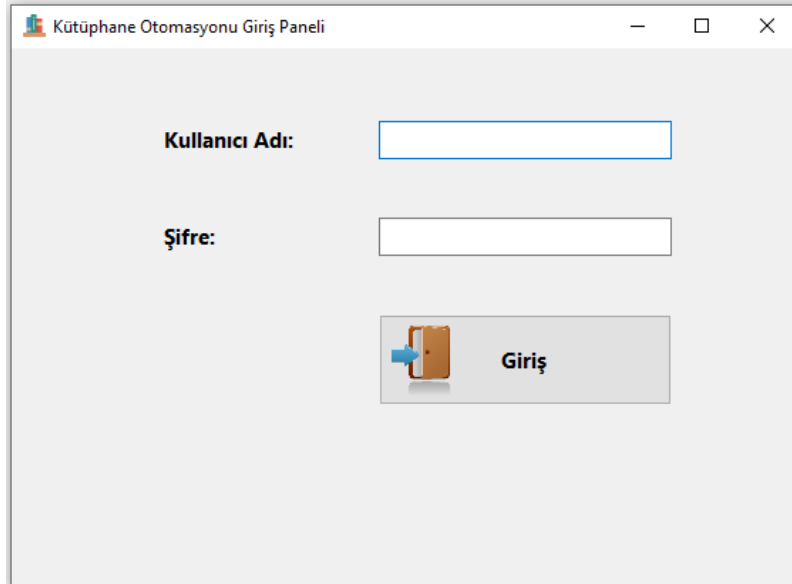
##### Kullanıcılar Tablosu Oluşturulması

```
create table kullanicilar(  
id int primary key auto_increment,  
kullanici_adi varchar(50) not null,  
sifre varchar(50) not null  
);
```

##### Kullanıcılar Tablosu Veri girişi

```
insert into kullanicilar1(kullanici_adi,sifre) values("admin","123")
```

3. **KÜTÜPHANE OTOMASYONU GİRİŞ PANELİ FORM SAYFASI KODLAMALARI**



Yukardaki gibi bir görsel panel hazırlanır. Kullanıcı adı ve şifre alanları doldurulduktan sonra **“Giriş”** butonuna tıklandığı zaman aşağıdaki kodlar çalıştırılır.

Kullanıcı adı ve şifre alanlarının boş bırakılması **“IsNullOrEmpty”** metoduyla kontrol edilmiştir. Veri tabanı ile bağlantı kurularak kayıtlar kontrol edilerek eşleşen kayıt varsa AnaForm sayfamıza erişim sağlanacaktır.

```
VeriTabaniIslemleri vtIslemleri = new VeriTabaniIslemleri();
 MySqlConnection baglanti;
 string komutsatiri;

1 reference
private void button1_Click(object sender, EventArgs e)
{
    string kullanıcıAdi = txtKullaniciAdi.Text.Trim();
    string sifre = txtSifre.Text;

    if (string.IsNullOrEmpty(kullanıcıAdi) || string.IsNullOrEmpty(sifre))
    {
        lblDurum.Text = "Lütfen tüm alanları doldurun.";
        return;
    }

    try
    {
        baglanti = vtIslemleri.baglan(); //Veri Tabanı bağlantı nesnesi oluşturulur
        baglanti.Open();
        komutsatiri = "SELECT * FROM kullanicilar WHERE kullanıcı_adi=@kadi AND sifre=@sifre";

        MySqlCommand cmd = new MySqlCommand(komutsatiri, baglanti);
        cmd.Parameters.AddWithValue("@kadi", kullanıcıAdi);
        cmd.Parameters.AddWithValue("@sifre", sifre);

        MySqlDataReader dr = cmd.ExecuteReader();
        if(dr.Read())
        {
            dr.Close();
            baglanti.Close();
            formAnaSayfa ana = new formAnaSayfa();
            ana.Show();
            this.Hide();
        }
        else
        {
            lblDurum.Text = "Kullanıcı adı veya şifre hatalı.";
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message, "Hata Oluştı", MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}
```