



T.C

BOLU ABANT İZZET BAYSAL ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

VERİTABANI YÖNETİM SİSTEMLERİ

DÖNEM PROJESİ ÖDEVİ

Eren ŞAŞKIN

Mürsel ELİBOL

Önsöz

Veritabanı Yönetim Sistemleri dersinden edindiğimiz bilgileri bir uygulama üzerinde pratiğe dökülebilmek ve pekiştirebilmek için verilen dönem projesinde kullanmayı hedefledik. Proje konusu olarak, Kütüphane Otomasyon Sistemini seçtik. Bu raporda, otomasyon sistemi için gerekli senaryoyu daha sonra uygulama tasarımı için veritabanı ve arayüz işlemlerine açıklık getirerek sistemin çalışır durumu ile ilgili bilgiler verilecektir.

İçindekiler

1. Giriş.....	4
2. Genel Bakış.....	5
3. Uygulama Tasarımı.....	6
3.1 Veritabanı Tasarımı.....	7
3.1.1 Veritabanı diyagramı.....	11
3.1.2 Saklı Yordamlar (Stored Procedures).....	12
3.1.3 Tetikleyiciler (Triggers).....	18
3.2 Arayüz Tasarımı.....	21
3.2.1 Formların Oluşturulması.....	21
3.2.2 Sınıf Kütüphanesinin oluşturulması.....	30
4. Tasarımların Entegre Edilmesi ve Çalıştırılması.....	33

1. Giriş

Günümüzde bilgisayar sistemlerinin her alanda kullanılmasıyla her türlü bilgiye hızlı bir şekilde ulaşım artmıştır. Daha doğru ve kaliteli bilgiye ulaşabilmek için kütüphanelerde de bilgisayar kullanımı yaygınlaşmaya başlamıştır. Böylelikle kütüphanelerde otomasyon sistemi oluşmaya başlamıştır. Bilgi paylaşımını daha hızlı ve doğru kaynaklar kullanılarak ulaşılabilmesi için kütüphane sistemleri oluşturulmuştur.

Kütüphane otomasyon sistemleri adı verilen bu kütüphane sistemleri sayesinde döküman sağlama ya da kaynakları ödünç verme veya bilgi danışma hizmeti vermektedir. Bilişim sistemleri kullanılarak erişilen her türlü bilgiyi kaynak yönetimini ve paylaşımını sağlar. Kütüphane sistemleri içerisinde en çok kullanılan barkod kullanımı sayesinde kullanılan kaynakların takibi sağlanmaktadır. Barkod sistemi sayesinde kaynaklar üzerinde yapılan hatalar önlenir bilgi ihtiyacı olan kişinin kaynak kullanım süresi belirlenir isteğe göre bu süre uzatılabilir. Bilgisayar ile yapılan taramalar sayesinde kaynaklara ister tek tek ister toplu olarak ulaşılır.

Online katalog adı verilen bu sistem sayesinde hem kütüphane içinden hem de kütüphane dışından kaynak taraması yapılabilir ve böylece istenilen kaynaklara rahatça ulaşılabilir. Kütüphane envanterinin tutulması kaynak satın alınması ve kurs rezervasyonlarını içine alan bir diğer sistem ise bütünleşik kütüphane otomasyon sistemidir. Bu sistem sayesinde istenilen tüm kaynaklara katalog şeklinde ulaşılabilir. Kütüphane otomasyon sistemlerinin kullanıldığı tüm kütüphanelerde istenilen bilgiye en doğru kaynaktan en hızlı bir şekilde ulaşım sağlanmaktadır. Ayrıca hangi kaynağın kimin kullandığını gösteren barkod sistemi sayesinde de kaynak takibi kolayca yapılabilir.

2. Genel Bakış

Kütüphane otomasyon sistemi sayesinde öncelikli olarak kütüphanede bulunan kitaplar kayıt altına alınmalı. Kitaplar bir kez kaydedildikten sonra silinebiliyor ve ya değiştirilebiliyor olmalı. Daha sonra üye kayıt edilebilir. Üyelerin bilgileri kaydedildikten sonra değiştirilebiliyor ve ya üyeler silinebiliyor olmalıdır.

Emanet işlemleri yapılmalıdır. Bir kitap bir üyeye emanet edilmeli ve işlem kayıt altına alınmalıdır. Emanetteki kitapların kimlere verildiği görüntülenip emanet ile ilgili bilgiler görüntülenebiliyor olmalıdır. Üyenin kitabı iade tarihinden geç teslim etmesi durumunda üyeye ceza işlemi uygulanarak bir süre kütüphane hizmetinden yararlanması engellenmelidir.

Üye sisteme giriş yaptığında Aradığı kitabın kütüphanede olup olmadığını görebilmeli, üzerinde bulunan kitapları, daha önce almış olduğu kitapları kendi profil ayarlarını düzenleme, emanet olarak aldığı kitabın uzatma işlemi yapabilmelidir.

3. Uygulama Tasarımı

3.1 Veritabanı Tasarımı

Admin tablosunun oluşturulması: Veritabanı işlemlerinin tamamının yönetebilecek kişinin bilgilerinin tutulduğu tablodur. Fotoğraftaki verilerden daha fazla admin kişisini tanımak için bilgiler alınabilirdi. Oluşturulan yeni admin sisteme emaili ve şifresi ile giriş yapabilecektir. Bu yüzden eMail alanı *unique* tanımlanmıştır. Şifre alanının ise *default* değer olarak, admin giriş yaptıktan sonra kendi şifresini kendisi belirlemesini sağladık.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
adminID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
ad	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
soyad	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
eMail	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
sifre	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'123'
telefon	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
adres	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

```
CREATE TABLE `admin` (  
  `adminID` int NOT NULL AUTO_INCREMENT,  
  `ad` varchar(45) NOT NULL,  
  `soyad` varchar(45) NOT NULL,  
  `eMail` varchar(100) NOT NULL,  
  `sifre` varchar(45) NOT NULL DEFAULT '123',  
  `telefon` varchar(45) DEFAULT NULL,  
  `adres` varchar(45) DEFAULT NULL,  
  PRIMARY KEY (`adminID`),  
  UNIQUE KEY `adminID_UNIQUE` (`adminID`),  
  UNIQUE KEY `eMail_UNIQUE` (`eMail`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Üye tablosunun oluşturulması: Kitapları takip edebilmek için hangi kitabın hangi üyeye verildiği bilgisi tutulmalı. Bu nedenle üye bilgilerine yer verilen üye tablosu oluşturulmalı. Sisteme admin gibi email ve şifresiyle giriş yapabilmektedir. Email bilgisi *unique* olarak ayarlanmıştır. Ve şifre *default* değer olan '123' olarak belirlenmektedir. Üye sisteme giriş yaptığında şifre, telefon ve adres bilgisini kendisi düzenleyebilmelidir. Diğer alanları ise görmemelidir.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
uyeID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
ad	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
soyad	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
eMail	VARCHAR(100)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
sifre	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'123'
telefon	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
adres	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
uyelikDurumu		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'1'
adminID		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
cezaTarihi	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

Bu tabloda açıklanması gereken diğer noktalar ise;

ÜyelikDurumu: Bu alan kişinin aldığı kitabı iade etmesi gerektiği tarihi kaçırmaması durumunda *true-false* değer olarak kitap alamaz duruma düşmesi için kullanılmaktadır.

adminID: Bu alan, bir üye üzerinde ki oluşturma ve silme işlemini hangi adminin yaptığı bilgisini tutmak için oluşturulmuştur.

cezaTarihi: Bu alan bir üye kitabını iade tarihinden sonra teslim etmesi durumunda kütüphaneden tekrar kitap alabileceği tarihi tutacaktır. Bir üye kitabı geç teslim ettiğinde, teslim ettiği tarihten 15 gün süreyle cezalı duruma düşecektir.

```
CREATE TABLE `uye` (  
  `uyeID` int NOT NULL AUTO_INCREMENT,  
  `ad` varchar(45) NOT NULL,  
  `soyad` varchar(45) NOT NULL,  
  `eMail` varchar(100) NOT NULL,  
  `sifre` varchar(45) NOT NULL DEFAULT '123',  
  `telefon` varchar(45) DEFAULT NULL,  
  `adres` varchar(45) DEFAULT NULL,  
  `uyelikDurumu` int DEFAULT '1',  
  `adminID` int DEFAULT NULL,  
  `cezaTarihi` date DEFAULT NULL,  
  PRIMARY KEY (`uyeID`),  
  UNIQUE KEY `uyeID_UNIQUE` (`uyeID`),  
  UNIQUE KEY `eMail_UNIQUE` (`eMail`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Kitap tablosunun oluşturulması: Kütüphane sisteminde emanet verilen kitapların verilerinin tutulduğu tablodur. Kitaplar ISBN numarasına göre *unique* olarak tutulur. Ve fotoğraftaki gibi bilgileri saklanmaktadır.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
🔑 kitapID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
🔹 ISBN	VARCHAR(13)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔹 ad	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔹 yazar	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔹 baskiYili	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
🔹 yayinEvi	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
🔹 sayfaSayisi	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
🔹 fotograF	BLOB	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
🔹 aciklama	LONGTEXT	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
🔹 kitapDurumu		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'1'
🔹 islemSayisi		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
🔹 adminID		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Bu tabloda açıklanması gereken diğer noktalar ise;



kitapDurumu: Bu alan ait olduğu kitabın işlem görüp görmediği bilgisini true-false olarak tutar. Bu sayede işlem görmüş bir kitabın başka bir kullanıcıya tahsis edilmesi durumunu engeller. İlk oluşturulduğunda default olarak 1 değeri alır.

adminID: Bu alan, bir kitap üzerinde ki oluşturma ve silme işlemini hangi adminin yaptığı bilgisini tutmak için oluşturulmuştur.

islemSayisi: Bu alan bir kitabın kaç kez işlem gördüğü bilgisini tutacaktır. En çok okunan kitabın görüntülenmesi için oluşturulmuştur. Default değer olarak 0 verilmiştir. İşlem gördükçe artacaktır.

```
CREATE TABLE `kitap` (  
  `kitapID` int NOT NULL AUTO_INCREMENT,  
  `ISBN` varchar(13) NOT NULL,  
  `ad` varchar(45) NOT NULL,  
  `yazar` varchar(45) NOT NULL,  
  `baskiYili` varchar(45) DEFAULT NULL,  
  `yayinEvi` varchar(45) DEFAULT NULL,  
  `sayfaSayisi` varchar(45) DEFAULT NULL,  
  `fotograf` blob,  
  `aciklama` longtext,  
  `kitapDurumu` tinyint NOT NULL DEFAULT '1',  
  `islemSayisi` int DEFAULT '0',  
  `adminID` int DEFAULT NULL,  
  PRIMARY KEY (`kitapID`),  
  UNIQUE KEY `kitapID_UNIQUE` (`kitapID`),  
  UNIQUE KEY `ISBN_UNIQUE` (`ISBN`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Kategori tablosunun oluşturulması: Kitapların sadece bir kategoride tutulması, gerektiğinde bu kategoriden aratılarak üyeye kolaylık sağlanması işlemi için oluşturulmuştur. Kitap tablosuyla ayrı bir tabloda ilişkilendirilerek işlem yapılması sağlanır.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 kategoriID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
 kategoriAdi	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

```
CREATE TABLE `kategori` (  
  `kategoriID` int NOT NULL AUTO_INCREMENT,  
  `kategoriAdi` varchar(45) NOT NULL,  
  PRIMARY KEY (`kategoriID`),  
  UNIQUE KEY `ad_UNIQUE` (`kategoriAdi`)  
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```


Kitap_kategori tablosunun oluşturulması: Her bir kitap bir kategoriyle eşleştirilmektedir. Kitap tablosundan kitapID ve kategori tablosundan kategoriID PK ler olarak bu tabloda saklanır.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
🔑 kitapID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔑 kategoriID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

```
CREATE TABLE `kitap_kategori` (
  `kitapID` int NOT NULL,
  `kategoriID` int NOT NULL,
  PRIMARY KEY (`kitapID`,`kategoriID`),
  KEY `FK_kategori` (`kategoriID`),
  CONSTRAINT `FK_kategori` FOREIGN KEY (`kategoriID`) REFERENCES `kategori` (`kategoriID`),
  CONSTRAINT `FK_kitap` FOREIGN KEY (`kitapID`) REFERENCES `kitap` (`kitapID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

İşlem tablosunun oluşturulması: Admin bir üyeye kitap emanet verdiğinde bu verilerin tutularak kitap takibinin yapılması gerekir. Bu tabloda ise bu veriler saklanır. Hangi üyenin hangi kitabı aldığını ve bu işlemi hangi adminin gerçekleştirdiği verisi tutulur. Ek olarak kitabın emanet verildiği işlem tarihi ve iade etmesi istenen tarih tutulmalıdır. Bu tabloda kitapID *unique* seçilerek bir kitabın aynı anda iki farklı üyeye tanımlanmasının engellenmiştir.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
🔑 işlemID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
🔑 üyeID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔑 kitapID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
📅 işlemTarihi	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
📅 iadeTarihi	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
📌 işlemDurumu		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0'
📌 adminID		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
📌 emanetDurumu	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'RAFTA'
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Bu tabloda açıklanması gereken diğer noktalar ise;

emanetDurumu: Bu alan ait olduğu kitabın iade edileceği tarihi tutar. Eğer kitap işlem görmemiş ise default olarak 'rafta' tutar ve bu sayede kitapları listelediğimiz zaman yazacağımız sorgular ile üyelere aradıkları kitabın mevcut olup olmadığı bilgisi verilebilir

işlemDurumu: Bir üye kitabın emanet süresini yenileyerek uzatmak istediğinde kaç kez uzattığı bilgisini tutar.

```

CREATE TABLE `islem` (
  `islemID` int NOT NULL AUTO_INCREMENT,
  `uyeID` int NOT NULL,
  `kitapID` int NOT NULL,
  `islemTarihi` date DEFAULT NULL,
  `iadeTarihi` date DEFAULT NULL,
  `islemDurumu` int NOT NULL DEFAULT '0',
  `adminID` int DEFAULT NULL,
  `emanetDurumu` varchar(45) DEFAULT 'RAFTA',
  PRIMARY KEY (`islemID`,`uyeID`,`kitapID`),
  UNIQUE KEY `kitapID_UNIQUE` (`kitapID`),
  KEY `fk_uyeID_2` (`uyeID`),
  KEY `fk_admin_2` (`adminID`),
  CONSTRAINT `fk_admin_2` FOREIGN KEY (`adminID`) REFERENCES `admin` (`adminID`),
  CONSTRAINT `fk_kitapID` FOREIGN KEY (`kitapID`) REFERENCES `kitap` (`kitapID`),
  CONSTRAINT `fk_uyeID_2` FOREIGN KEY (`uyeID`) REFERENCES `uye` (`uyeID`)
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

Log tablosunun oluşturulması: Veritabanında yapılan kitap ekleme, silme ve güncelleme, üye için yapılan ekleme ve silme, işlem için yapılan kitap verme ve teslim alma işlemlerini hangi adminin yaptığı verilerini çekebilmek için ve hangi üye hangi kitabı daha önce iade etmiş bilgilerini tutmak için oluşturulmuştur. Bu işlemler için triggerdan yararlandık.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
logID		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
islemID		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
uyeID		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
kitapID		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
islemTarihi	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
iadeTarihi	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
adminID		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
aciklama	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Bu tabloda açıklanması gereken diğer noktalar ise;

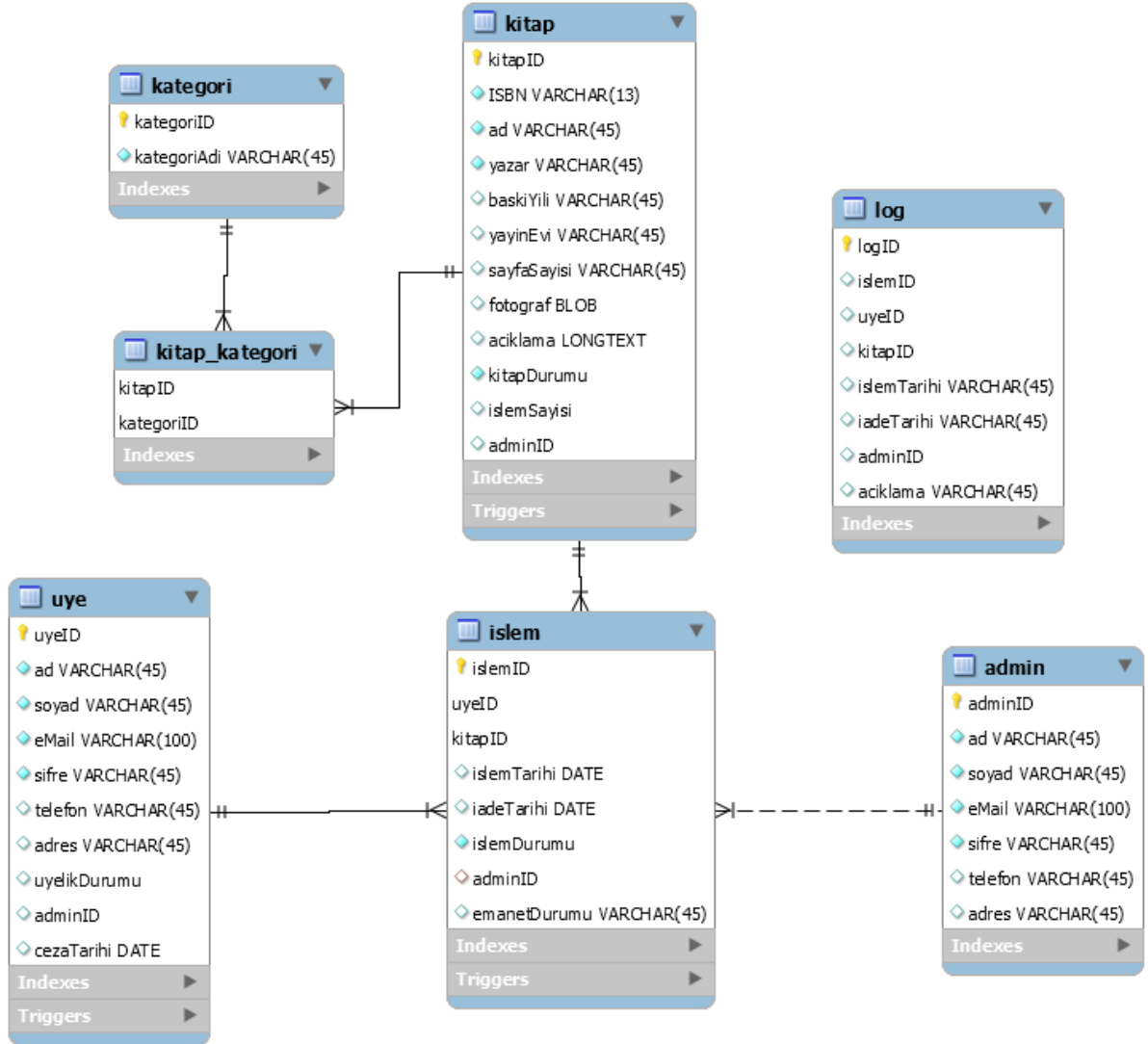
aciklama: Bu alan yapılan işlemin belirtildiği alandır. Kitap silindi, kitap eklendi, üye oluşturuldu, kitap teslim alındı, kitap emanet verildi bilgileri yer alır.

```

CREATE TABLE `log` (
  `logID` int NOT NULL AUTO_INCREMENT,
  `islemID` int DEFAULT NULL,
  `uyeID` int DEFAULT NULL,
  `kitapID` int DEFAULT NULL,
  `islemTarihi` varchar(45) DEFAULT NULL,
  `iadeTarihi` varchar(45) DEFAULT NULL,
  `adminID` int DEFAULT NULL,
  `aciklama` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`logID`)
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;

```

3.1.1 Veritabanı diyagramı:



3.1.2. Saklı yordam (Stored Procedure): Belirlediğimiz şartlara göre, program içinden çağrılarak kullanılan, otomasyon sistemine entegre etmek üzere prosedürler oluşturduk. Bunlar;

1. Admin Ekle (p_adminEkle) : Program içinde verileri alarak tablodaki alanlara ekleme.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_adminEkle`(
  IN p_ad varchar(45),
  IN p_soyad varchar(45),
  IN p_eMail varchar(100),
  IN p_telefon varchar(45),
  IN p_adres varchar(45)
)
BEGIN
insert into admin(ad,soyad,eMail,telefon,adres) values (p_ad,p_soyad,p_eMail,p_telefon,
  p_adres);
END$$
DELIMITER ;
```

2. Admin Güncelle (p_adminGüncelle) : E mail verisine göre o kişinin verilerini güncelleme.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_adminGuncelle`(
  IN p_ad varchar(45),
  IN p_soyad varchar(45),
  IN p_eMail varchar(100),
  IN p_telefon varchar(45),
  IN p_adres varchar(45)
)
BEGIN
UPDATE admin SET ad = p_ad, soyad = p_soyad, telefon = p_telefon, adres = p_adres
  where eMail = p_eMail;
END$$
DELIMITER ;
```

3. Admin Silme (p_adminSil) : E mail verisine göre o kişinin verilerini silme.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_adminSil`(IN p_eMail varchar(100))
BEGIN
  DELETE FROM admin WHERE eMail = p_eMail;
END$$
DELIMITER ;
```

4. Üye ekleme (p_uyeEkle) : Program içinde verileri alarak tablodaki alanlara ekleme.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_uyeEkle`(
  IN p_ad varchar(45),
  IN p_soyad varchar(45),
  IN p_eMail varchar(100),
  IN p_telefon varchar(45),
  IN p_adres varchar(45),
  IN p_adminID int,
  IN p_cezaTarihi date
)
BEGIN
  insert into uye(ad,soyad,eMail,telefon,adres,adminID,cezaTarihi) values (p_ad,p_soyad,p_eMail,p_telefon,p_adres,p_adminID,p_cezaTarihi);
END$$
DELIMITER ;
```

5. Üye güncelleme (p_uyeGüncelle) : E mail verisine göre o kişinin verilerini güncelleme.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_uyeGuncelle`(
  IN p_ad varchar(45),
  IN p_soyad varchar(45),
  IN p_eMail varchar(100),
  IN p_telefon varchar(45),
  IN p_adres varchar(45)
)
BEGIN
  UPDATE uye SET ad = p_ad, soyad = p_soyad, telefon = p_telefon, adres = p_adres
  where eMail = p_eMail;
END$$
DELIMITER ;
```

6. Üye silme (p_uyeSilme) : E mail verisine göre o kişinin verilerini silme.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_uyeSil`(
  IN p_eMail varchar(100),
  IN p_adminID int)
BEGIN
  UPDATE uye SET adminID = p_adminID where eMail = p_eMail;
  DELETE FROM uye WHERE eMail = p_eMail;
END$$
DELIMITER ;
```

7. Sisteme girişi kontrol etme (p_girisKontrol) : Sisteme giriş olduğunda çalışacak olan prosedürdür. Kullanıcıdan mail ve şifresiyle giriş yapması istenir. Bu veriler alınarak sistemde kullanılır.

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_girisKontrol`(  
  IN f_eMail varchar(100),  
  IN f_sifre varchar(45),  
  IN f_tablo varchar(45)  
)  
BEGIN  
  SELECT ad,soyad FROM f_tablo WHERE eMail = f_eMail AND sifre = f_sifre;  
END$$  
DELIMITER ;
```

8. Kitap ekleme (p_kitapEkle) : Admin tarafından kitap oluşturulmak istendiğinde çalışacak prosedürdür. Bir kitap eklendiğinde o kitabın verilerini veritabanına kayıt eder. Daha sonra gelen kategori adına ait kategoriID sini bulup, bu ID ye göre kitap_kategori tablosuna kitapID yi oluşturur.

```
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_kitapEkle`(  
  IN p_ISBN varchar(13),  
  IN p_ad varchar(45),  
  IN p_yazar varchar(45),  
  IN p_baskiYili varchar(45),  
  IN p_yayinEvi varchar(45),  
  IN p_sayfaSayisi varchar(45),  
  IN p_fotograf blob,  
  IN p_aciklama longtext,  
  IN p_kategoriAd varchar(45),  
  IN p_adminID int  
)  
BEGIN  
  declare p_kitapID INT;  
  declare p_kategoriID INT;  
  
  insert into kitap(ISBN,ad,yazar,baskiYili,yayinEvi,sayfaSayisi,fotograf,aciklama,adminID) values (p_ISBN,p_ad,p_yazar,p_baskiYili,p_yayinEvi,p_sayfaSayisi,p_fotograf,p_aciklama,p_adminID);  
  
  set p_kitapID = (select kitapID from kitap where ISBN = p_ISBN);  
  set p_kategoriID = (select kategoriID from kategori where kategoriAdi = p_kategoriAd  
);  
  
  insert into kitap_kategori(kitapID,kategoriID) values (p_kitapID,p_kategoriID);  
END$$  
DELIMITER ;
```

9. Kitap güncelleme (p_kitapGüncelleme) : Admin tarafından kitap güncellenmek istendiğinde çalışacak prosedürdür. ISBN numarasına göre güncellenir. Bir kitap güncellendiğin o kitabın verilerini veritabanında düzenler. Daha sonra gelen kategori adına ait kategoriID sini bulup, bu ID ye göre kitap_kategori tablosuna kitapID yi düzenler.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_kitapGuncelle`(
  IN p_ISBN varchar(13),
  IN p_ad varchar(45),
  IN p_yazar varchar(45),
  IN p_baskiYili varchar(45),
  IN p_yayinEvi varchar(45),
  IN p_sayfaSayisi varchar(45),
  IN p_fotograf blob,
  IN p_aciklama longtext,
  IN p_kategoriAd varchar(45),
  IN p_adminID int
)
BEGIN
  declare p_kitapID INT;
  declare p_kategoriID INT;

  UPDATE kitap SET
    ad = p_ad,
    yazar = p_yazar,
    baskiYili = p_baskiYili,
    yayinEvi = p_yayinEvi,
    sayfaSayisi = p_sayfaSayisi,
    fotograf = p_fotograf,
    aciklama = p_aciklama,
    adminID = p_adminID
  where ISBN = p_ISBN;

  set p_kitapID = (select kitapID from kitap where ISBN = p_ISBN);
  set p_kategoriID = (select kategoriID from kategori where kategoriAdi = p_kategoriAd
  );
  UPDATE kitap_kategori SET kategoriID = p_kategoriID where kitapID = p_kitapID;
END$$
DELIMITER ;
```

10. Kitap silme (p_kitapSilme) : Kitap silinmek istendiğinde çalışacak prosedürdür. ISBN numarasına göre siler. AdminID alınmasının sebebi ise hangi adminin silme işlemini yaptığı bilgisini log tablosunda tutabilmek. Eğer bir kitap işlem görüyorsa o kitap silemeyeceği için uygulamada gerekli hata mesajları verilecektir. Bir kitabı silerken ait olduğu kategoriden de çıkarmak gerekir. Bu yüzden bu prosedürde kategorideki kayıdını silmek için iç içe sorguyla ait olduğu kategori alınarak kitap_kategori tablosunda karşılık gelen satır silinmesi ayarlanmıştır.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_kitapSil`(IN p_ISBN varchar(13), IN p_adminID int)
BEGIN
declare islemVarMi int;

set islemVarMi = (SELECT count(kitapID) FROM islem where kitapID =(SELECT kitapID FROM kitap WHERE ISBN = p_ISBN)) ;

if islemVarMi = 0 THEN
    UPDATE kitap SET adminID = p_adminID where ISBN = p_ISBN;
    DELETE FROM kitap_kategori WHERE kitapID = (Select kitapID from kitap where ISBN = p_ISBN);# Kategorideki kaydı siler.
    DELETE FROM kitap WHERE ISBN = p_ISBN; # islemde fk olduğu için varsa zaten silemez.
elseif islemVarMi = 1 THEN
    DELETE FROM kitap WHERE ISBN = p_ISBN;
end if;

END$$
DELIMITER ;
```

11. İşlem ekleme (p_islemEkle) : Bir emanet işlemi oluşturduğumuzda çalışacak prosedürdür. Hangi adminin, hangi üyeye, hangi kitabı verdiği, bu işlemi ne zaman yaptığı ve ne zaman iade etmesi gerektiği verileri alınır. İşlem oluşturulmadan önce kitabın *kitapDurumu* verisi 0 yapılarak o kitabın pasif duruma geçmesi sağlanır.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_islemEkle`(
    IN p_uyeID int,
    IN p_kitapID int,
    IN p_adminID int,
    IN p_islemTarihi date,
    IN p_iadeTarihi date,
    IN p_emanetDurumu varchar(45)
)
BEGIN
#kitabı pasif yapve işlem sayisini arttır.
```



```

UPDATE kitap SET kitapDurumu = 0, islemSayisi=islemSayisi + 1 where kitapID=p_kitapID;
insert into islem(uyeID,kitapID,adminID,islemTarihi,iadeTarihi,emanetDurumu) values
(p_uyeID,p_kitapID,p_adminID,p_islemTarihi,p_iadeTarihi,p_emanetDurumu);
END$$
DELIMITER ;

```

12. İşlem silme (p_islemSilme) : Bir emanet işlemi sonlandığında yani kitap üye tarafından iade edildiğinde sırasıyla;

- Kitabın kitapDurumu 1 yapılarak aktif hale çevrilir.
- islemID verisine göre islem tablosundan veri silinir.
- Bazı kontroller yapılmaktadır bunlar; üye kitabı iade tarihinden geç bir tarihte getirdiyse, uyelikDurumu verisine bakılarak if bloğu çalışır. Bu if bloğu üye geç getirdiyse o üyeye getirdiği tarihin üzerine +15 gün ekleyerek bu süre zarfında kitap alamaması için gerekli değişikliği yapar. Ceza tarihini 15 gün sonraki tarihle günceller.

```

DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_islemSil`(IN p_islemID int)
BEGIN

    declare p_onbesGun date;
    declare p_uyeninDurumu INT;
    declare p_uyeID INT;

    set p_uyeID = (select uyeID from islem where islemID = p_islemID);
    set p_onbesGun = current_date() + INTERVAL 15 DAY;
    set p_uyeninDurumu = (SELECT uyelikDurumu from uye where uyeID = p_uyeID);

    #kitap durumunu aktif yapmak için
    UPDATE kitap SET kitapDurumu = 1 where kitapID = (select kitapID from islem where
    islemID=p_islemID);

    DELETE FROM islem WHERE islemID = p_islemID;

    if p_uyeninDurumu = 0 then
        UPDATE uye SET cezaTarihi = p_onbesGun where uyeID = p_uyeID;
    end if;

END$$
DELIMITER ;

```

13. Kitap uzatma (p_kitapUzatma) : Üye kitabın süresini uzatmak isteyebilir. Bu durumda kitap uzatma prosedürü çalışılır. Kaç kez uzatıldı bilgisi için işlemDurumu alanı 1 arttırılır. İade tarihi ise uygulama dan gelecek olan +15 gün yeni ek süre verir. Bu işlemi üye ancak iade tarihine 1 gün kala yapabilecektir.

```
DELIMITER $$
CREATE DEFINER=`root`@`localhost` PROCEDURE `p_kitapUzat`(
    IN p_islemID int,
    IN p_iadeTarihi varchar(45)
)
BEGIN

    UPDATE işlem SET iadeTarihi = p_iadeTarihi,
    emanetDurumu=p_iadeTarihi, işlemDurumu = işlemDurumu + 1 WHERE işlemID = p_islemID;

END$$
DELIMITER ;
```

3.1.3 Tetikleyiciler (Triggers): Belirlediğimiz şartlara göre, bir tabloda belirli olaylar meydana geldiğinde veya gelmeden önce otomatik olarak çalışan özel bir store procedure türü olan tetikleyicileri ile log verilerini ayarlamak istedik. Tetikleyicileri kullandığımız tablolar ve işlemler ise şöyle;

1. Kitap Verme İşlemi Oluşturduktan Sonra

```
CREATE DEFINER=`root`@`localhost` TRIGGER `islem_AFTER_INSERT` AFTER INSERT ON `islem` FOR EACH ROW BEGIN
INSERT INTO log(
    işlemID,
    üyeID,
    kitapID,
    işlemTarihi,
    iadeTarihi,
    adminID,
    aciklama
) VALUES(NEW.islemID, NEW.üyeID, NEW.kitapID, NEW.islemTarihi, NEW.iadeTarihi, NEW.adminID, 'Kitap verildi..');
END;
```

2. Kitap Teslim Alma İşlemini Oluşturduktan Sonra

```
CREATE DEFINER=`root`@`localhost` TRIGGER `islem_AFTER_DELETE` AFTER DELETE ON `islem` FOR EACH ROW BEGIN
INSERT INTO log(
    islemID,
    uyeID,
    kitapID,
    islemTarihi,
    iadeTarihi,
    adminID,
    aciklama
) VALUES(OLD.islemID, OLD.uyeID, OLD.kitapID, OLD.islemTarihi,now(), OLD.adminID,"Kitap alındı..");
END;
```

3. Yeni Bir Kitap Ekledikten Sonra

```
CREATE DEFINER=`root`@`localhost` TRIGGER `kitap_AFTER_INSERT` AFTER INSERT ON `kitap` FOR EACH ROW BEGIN
INSERT INTO log(
    kitapID,
    islemTarihi,
    adminID,
    aciklama
) VALUES(NEW.kitapID, now(), NEW.adminID,"Kitap oluşturuldu...");
END;
```

4. Kitap Güncelledikten Sonra

```
CREATE DEFINER=`root`@`localhost` TRIGGER `kitap_AFTER_UPDATE` AFTER UPDATE ON `kitap` FOR EACH ROW BEGIN
INSERT INTO log(
    kitapID,
    islemTarihi,
    adminID,
    aciklama
) VALUES(OLD.kitapID, now(), new.adminID,"Kitap güncellendi...");
END;
```

5. Kitap Sildikten Sonra

```
CREATE DEFINER=`root`@`localhost` TRIGGER `kitap_AFTER_DELETE` AFTER DELETE ON `kitap` FOR EACH ROW BEGIN
INSERT INTO log(
    kitapID,
    islemTarihi,
    adminID,
    aciklama
) VALUES(OLD.kitapID, now(), OLD.adminID, "Kitap silindi...");
END;
```

6. Uye Olusturduktan Sonra

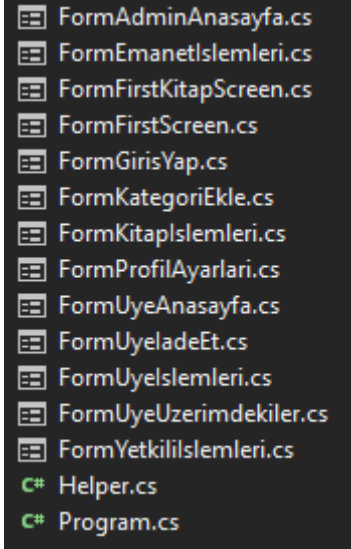
```
CREATE DEFINER=`root`@`localhost` TRIGGER `uye_AFTER_INSERT` AFTER INSERT ON `uye` FOR EACH ROW BEGIN
INSERT INTO log(
    uyeID,
    islemTarihi,
    adminID,
    aciklama
) VALUES(NEW.uyeID, now(), NEW.adminID, "Üye oluşturuldu...");
END;
```

7. Uye Sildikten Sonra

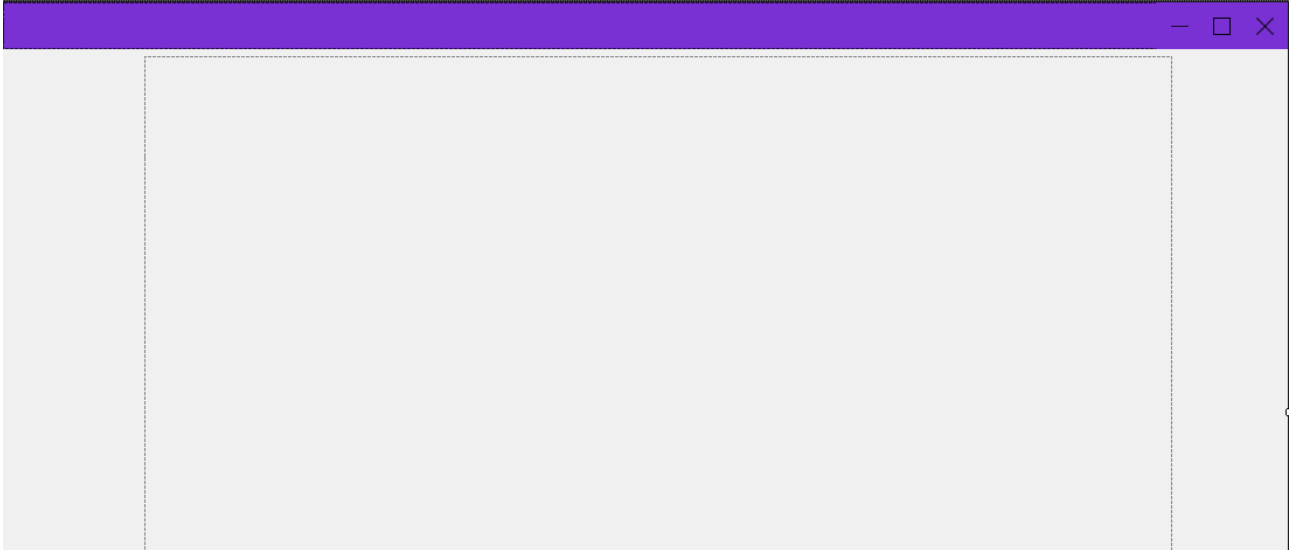
```
CREATE DEFINER=`root`@`localhost` TRIGGER `uye_AFTER_DELETE` AFTER DELETE ON `uye` FOR EACH ROW BEGIN
INSERT INTO log(
    uyeID,
    islemTarihi,
    adminID,
    aciklama
) VALUES(OLD.uyeID, now(), OLD.adminID, "Üye silindi...");
END;
```

3.2 Arayüz Tasarımı

3.2.1 Formların oluşturulması



FormFirstScreen.cs : Bu form içerisinde Panel Content bulundurur ve FormFirstKitapScreen formunu bu panel içerisinde açmaktadır. Uygulama bu form üzerinden başlatılır.




FormFirstKitapScreen.cs : Bu form FormFirstScreen formunun içerisinde çalışır. Ve tasarım amaçlı aşağıdaki gibi bir görüntüye sahiptir. Sistem ilk açıldığında görüntülen formdur. Çalıştığında kitap adı, yazar adı, ISBN ye göre arama yapabileceği textbox, kategoriye göre arama yapabileceği combobox bulunur. Kullanıcı arama yaptığıda tasarım paneli kapanır ve arama yaptığı alana göre veritabanında bulunan tüm kitaplar datagridviewde listelenir. Listenen kitaplar uygunsa 'Rafta' yazısı, eğer uygun değilse (bir başkası tarafından alınmışsa) iade tarihleri görüntülenmektedir. Burada kullanıcı giriş yapmadan da kitap durumunu sorgulaması sağlanmıştır. Sisteme giriş yapmak için giriş yap butonuna tıklanır.

Kitap Adı, Yazar Adı, ISBN

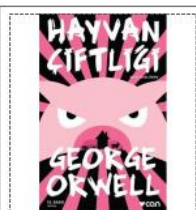
Ara

Giriş Yap

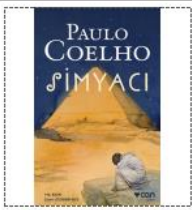
En Çok Okunanlar



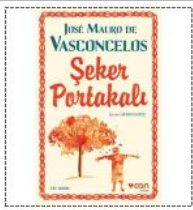
Bir Ömür Nasıl Yaşanır?
İLBER ORTAYLI



Hayvan Çiftliği
GEORGE ORWELL

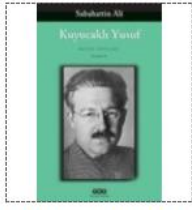


Simyacı
PAULO COELHO

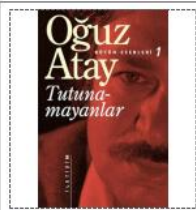


Şeker Portakalı
J. M. VASCONCELOS

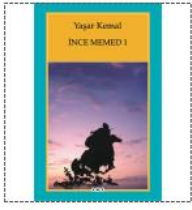
Türk Klasikleri



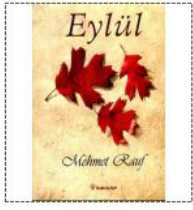
Kuyucaklı Yusuf
SABAHATTİN ALİ



Tutunamayanlar
OĞUZ ATAY



İnce Memed - I
YAŞAR KEMAL



Eylül
MEHMET RAUF

En Yeniler

FormGirisYap.cs: Sisteme girmek için kayıtlı olan kullanıcılardan email, şifre ve giriş türünün belirlenmesi istenmektedir. Kaydı olmayan kişilerin admin tarafından kayıt açtırmaları gerekmektedir. Admin girişi gerçekleştiğinde FormAdminAnasayfa formu, üye girişi olursa FormUyeAnasayfa formu ekrana gelecektir. Eğer hatalı giriş gerçekleşirse gerekli hata label ile gösterilecektir.

FormAdminAnasayfa.cs : Sisteme admin bilgileriyle giriş yapıldığında karşımıza çıkan ilk ekrandır. Mobile-Menü butonuna tıklayarak panel küçültülüp büyültülebilir. Menüdeki kitap işlemleri, üye işlemleri, emanet işlemleri, yetkili işlemleri butonlarına tıklandığında ilgili formlar ortadaki panel içinde açılır. Oturum açan kullanıcının ad soyad ve mail bilgisi ve tarih saat bilgileri gösterilmektedir. Oturum kapatmak için çıkış butonu bulunmaktadır.

FormKitapIslemleri.cs: Bir kitap eklemek istediğimizde, tüm alanların eksiksiz doldurulması gerekmektedir. Eksik veya yanlış bir veri girildiğinde uyarı mesajı görüntülenmektedir. Eğer ilgili kategori bulunamamışsa kategori ekle formuna gidilir ve ilgili kategori eklenebilir.

Kaydet butonuna tıklıldığında kitap veritabanına kaydedilir ve anlık olarak gridview de görüntülenir.

Bir kitabı aramak istediğimizde, istenilen kitap ile ilgili bilgiler kitapAratextboxa girilir, veritabanında arama yapılır ve istenilen sonuçlar gridview de listelenir.

Bir kitabın bilgilerini düzenlemek istediğimizde, gridview da ilgili alana ait satır seçilir. Düzenle butonuna bastığımızda seçilen satırdaki veriler textboxlara otomatik olarak yerleşir. Değişiklik yapılmak istenilen alanları düzeltip kaydet butona basılır ve verileri veriler güncellenerek veritabanına kaydedilir

Bir kitabı silmek istediğimizde, gridviewda ilgili alana ait satır seçilir. Düzenle butonuna bastığımızda seçilen satırdaki veriler textboxlara otomatik olarak yerleşir. Ardından sil butonuna bastığımızda kitap veritabanından silinir. Eğer silmek istediğimiz kitap bir kullanıcıya tahsis edilmişse o kitabın neden silinemediğine dair bir uyarı mesajı gösterilir. Kitap üyeden teslim alındığında silinebilir duruma gelir.

ISBN :	<input type="text"/>	Kitap Adı, Yazar Adı, ISBN, Kategori Adı	Ara
Kitap Ad :	<input type="text"/>		
Yazar :	<input type="text"/>		
Kategori :	<input type="text"/>		
	Kategori Ekle		
Baskı Yılı :	<input type="text"/>		
Yayın Evi :	<input type="text"/>		
Sayfa Sayısı :	<input type="text"/>		
Fotoğraf :	<input type="button" value="Resim Seç"/>		
	<div></div>		
Açıklama :	<input type="text"/>		
<input type="button" value="Kaydet"/>			
<input type="button" value="Düzenle"/>	<input type="button" value="Sil"/>		

FormKategoriEkle.cs: Kitap işlemlerinde bir kitabı eklerken ilgili kategori bulunamayıp, kategori ekle butonuna tıklıldığında açılan formdur.

Bir kategori ekleme istediğimizde, textboxa istenilen kategori adı girilir ve Kaydı butona tıklıldığında kategori kaydedilir. Eğer kaydedilmek istenilen kategori adı mevcutsa hata mesajı gösterilir.(label). Bir kategoriye güncellemek istediğimizde, comboboxdan ilgili kategori seçilir. Düzenle butonuna basıldığında kategori adı textboxa yerleşir. Düzeltilip kaydet

butonuna basılır ve veritabanında güncellenir. Eğer o kategoriye ait kitaplar mevcutsa, güncellenen kategori adı tüm kitaplarda da güncellenir. Bir kategoriye silmek istediğimizde, comboboxdan ilgili kategori seçilir. Düzenle butonuna basıldığında kategori adı textboxa yerleşir ve sil butonuna basıldığında siler. Eğer o kategoriye ait kitap mevcutsa silinemeyeceğine dair bir uyarı mesajı gösterilir.(label)

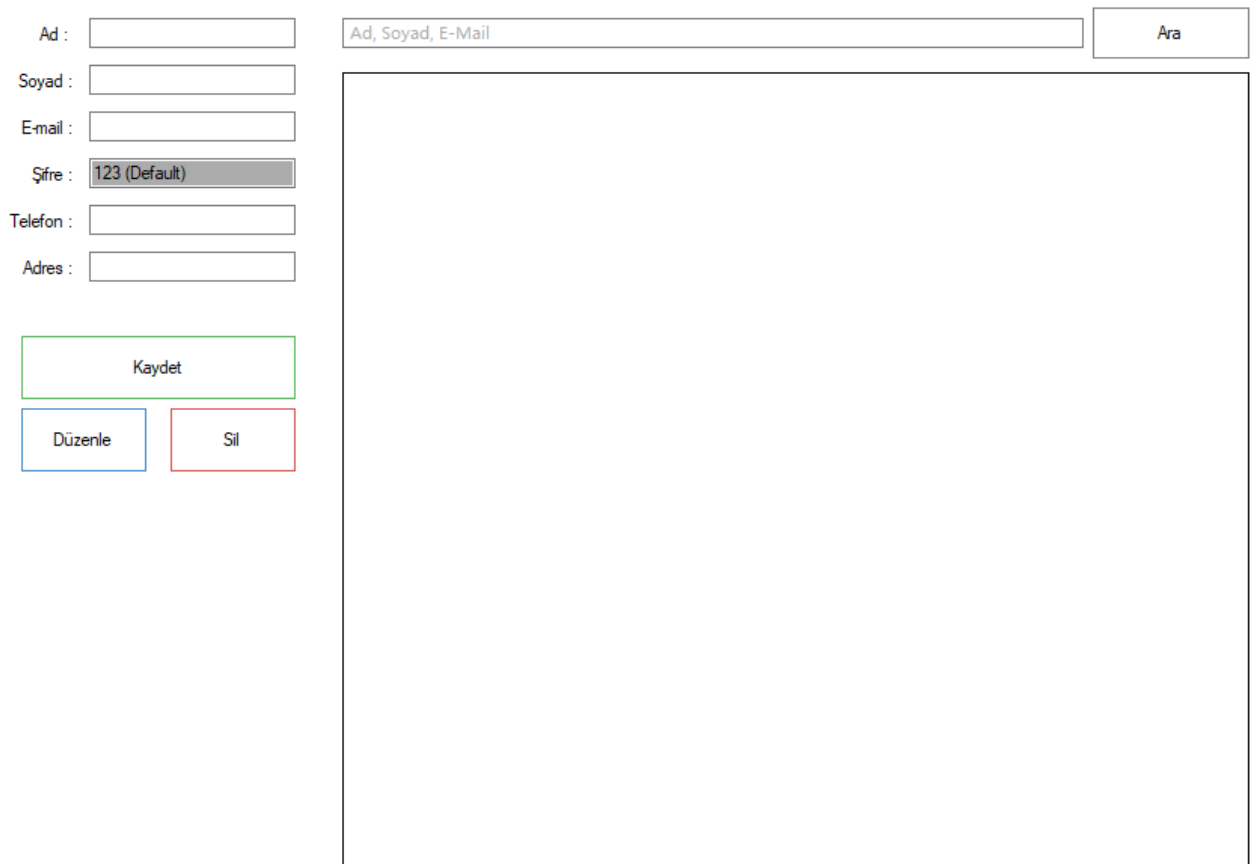


FormUyeIslemleri.cs: Bir üye eklemek istediğimizde, tüm alanların eksiksiz doldurulması gerekmektedir. Eksik veya yanlış bir veri girildiğinde uyarı mesajı görüntülenmektedir. Kaydet butonuna tıklandığından üye veritabanına kaydedilir ve anlık olarak gridviewde görüntülenir. Kaydedilen her üyenin ilk şifresi default olarak '123' olur.

Bir üyeyi aramak istediğimizde, istenilen üye ile ilgili bilgiler uyeAratextboxa girilir, veritabanında arama yapılır ve istenilen sonuçlar gridview de listelenir.

Bir üye bilgilerini düzenlemek istediğimizde, gridview da ilgili alana ait satır seçilir. Düzenle butonuna bastığımızda seçilen satırdaki veriler textboxlara otomatik olarak yerleşir. Değişiklik yapılmak istenilen alanları düzeltip kaydet butona basılır ve veriler güncellenerek veritabanına kaydedilir.

Bir üyeyi silmek istediğimizde, gridview da ilgili alana ait satır seçilir. Düzenle butonuna bastığımızda seçilen satırdaki veriler textboxlara otomatik olarak yerleşir. Ardından sil butonuna bastığımızda üye veritabanından silinir. Eğer silmek istediğimiz üyeye bir kitap tahsis edilmişse o üyenin neden silinemediğine dair bir uyarı mesajı gösterilir. Kitap üyeden teslim alındığında üye silinebilir duruma gelir.



FormEmanetIslemleri.cs: Kütüphane yetkilisi kitabı bir üyeye tahsis etmek istediğinde 1. datagridviewden ilgili kullanıcıya ait satırı seçer, seçilen o kullanıcının üyeID'si textboxa yerleşir. (uyeID 'si textbox'ının yanında bulunan kutucuk eğer yeşil renkte olursa kitap alabilir. Eğer kırmızı renkte olursa cezası bitene kadar kitap alamaz durumunu gösterir.) 2.datagridviewden istenilen kitaba ait satır seçilir, seçilen o kitabın kitapID'si textboxa yerleşir. İşlem tarihi o günün tarihini ve teslim tarihi ise 15 gün sonrası olarak otomatik bir şekilde gösterilir. Kitabı Ver butonuna tıklandığında 3.datagridviewda emanet işlemi listelenir.

Bir kullanıcı kitabı teslim etmek istediğinde kullanıcı bilgileri ya da kitap bilgileri islemAraTextboxa girilerek 3.datagridviewda listelenir. Eğer kullanıcı kitabı teslim tarihinden önce getirmişse teslim al butonuna tıklanarak teslim alınır. Eğer teslim tarihini geciktirmişse 15 gün kitap alamama cezası verilerek kitap teslim alınır.

Yetkili ID :	<input type="text"/>	Ad, Soyad, E-Mail	<input type="text"/>	Ara
Uye ID :	<input type="text"/>			
Kitap ID :	<input type="text"/>			
İşlem Tarihi :	19.05.2020 <input type="text"/>			
Teslim Tarihi :	23.05.2020 <input type="text"/>			
<input type="button" value="Temizle"/>	<input type="button" value="Kitap Ver"/>	Kitap Adı, Yazar Adı, ISBN, Kategori Adı	<input type="text"/>	Ara
uyan				
<input type="button" value="Teslim Al"/>		İşlem ID, Uye Adı, Kitap Adı, ISBN	<input type="text"/>	Ara

FormYetkiliIslemleri.cs: Bir yetkili eklemek istediğimizde, tüm alanların eksiksiz doldurulması gerekmektedir. Eksik veya yanlış bir veri girildiğinde uyarı mesajı görüntülenmektedir. Kaydet butonuna tıklandığından yetkili veritabanına kaydedilir ve anlık olarak gridviewde görüntülenir. Kaydedilen her yetkilinin ilk şifresi default olarak '123' olur.

Bir yetkiliyi aramak istediğimizde, istenilen yetkili ile ilgili bilgiler textboxa girilir, veritabanında arama yapılır ve istenilen sonuçlar gridviewde listelenir.

Bir yetkili bilgilerini düzenlemek istediğimizde, gridviewda ilgili alana ait satır seçilir. Düzenle butonuna bastığımızda seçilen satırdaki veriler textboxlara otomatik olarak yerleşir. Değişiklik yapılmak istenilen alanları düzeltip kaydet butona basılır ve veriler güncellenerek veritabanına kaydedilir.

Bir yetkiliyi silmek istediğimizde, gridviewda ilgili alana ait satır seçilir. Düzenle butonuna bastığımızda seçilen satırdaki veriler textboxlara otomatik olarak yerleşir. Ardından sil butonuna bastığımızda yetkili veritabanından silinir.

Log datagridviewında ise tüm yapılan işlemlerin bilgileri tutulur. (Kitap hangi admin tarafından eklendi? Üyeyi hangi admin sildi? Kitabı hangi admin verdi? Kitap hangi admin tarafından teslim alındı?) Her admin diğer her adminin yaptığı işlemleri görebilir.

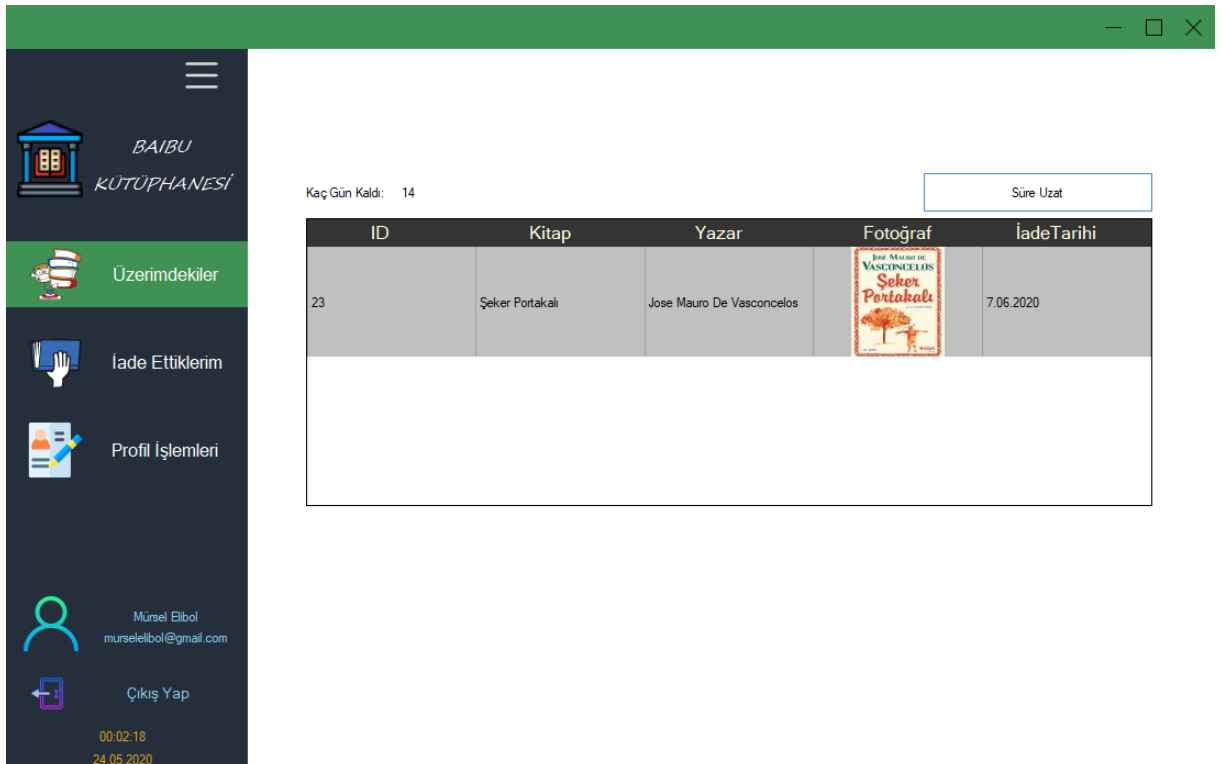
Yetkili ID :	<input type="text"/>	<input type="text" value="Ad, Soyad, E-Mail"/>	<input type="button" value="Ara"/>
Ad :	<input type="text"/>	<div></div>	
Soyad :	<input type="text"/>		
E-mail :	<input type="text"/>		
Şifre :	<input type="text" value="123456 (Default)"/>		
Telefon :	<input type="text"/>		
Adres :	<input type="text"/>		
	<input type="button" value="Kaydet"/>	<input type="button" value="Düzenle"/>	<input type="button" value="Sil"/>

LOG

FormUyeAnasayfa.cs: Sisteme üye bilgileriyle giriş yapıldığında karşımıza çıkan ilk ekrandır. Mobile-Menü butonuna tıklayarak panel küçültülüp büyültülebilir. Menüdeki üzerimdekimler, iade ettiklerim ve profil işlemleri butonlarına tıklandığında ilgili formlar ortadaki panel içinde açılır. Oturum açan kullanıcının ad soyad ve mail bilgisi ve tarih saat bilgileri gösterilmektedir. Oturum kapatmak için çıkış butonu bulunmaktadır.



FormUyeUzerimdekimler.cs: Üzerimdekimler formunda kullanıcının üzerinde bulunan kitaplar, kitapları iade etmesi için kalan gün sayısı gösterilir. Kullanıcı kitabın süresini uzatmak istediğinde Süre Uzat butonuna basabilir. Ancak kitaplar sadece iade tarihinden 1 gün önce uzatılabilir. Kullanıcı butona tıkladığında bu bilgi ona uyarı olarak gösterilir.



FormUyeIadeEttiklerim.cs: Kullanıcının bu zamana kadar alıp iade ettiği tüm kitaplar gösterilir.(Datagridview)

BAIBU
KÜTÜPHANESİ

Üzerimdekiler

Iade Ettiklerim

Profil İşlemleri

Mürsel Elbol
murselelbol@gmail.com

Çıkış Yap

00.07.28
24.05.2020

ad	yazar	fotograf	iadeTarihi
Bir Ömür Nasıl Yaşanır	İlber Ortaylı		2020-05-23 23:32:49
Tutunamayanlar	Oğuz Atay		2020-05-23 23:33:05

FormProfilAyarlari.cs: Üye veya admin ad, soyad, e-mail bilgilerini güncelleyemez. Bu alanlar sadece okunabilir bir şekilde textboxlara yerleşir. Şifre, telefon, adres bilgilerini güncellemek için gerekli alanlar doldurulur ve ardından Kaydet butonuna basıldığında veritabanında kayıt güncellenir.

×

Ad :

Soyad :

E-mail :

Şifre :

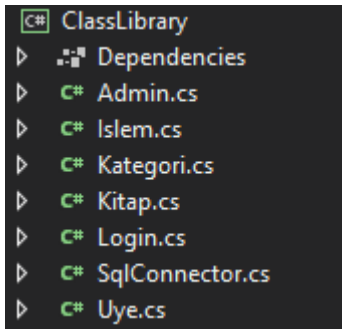
Telefon :

Adres :

Kaydet

Başarıyla Değiştirildi...

3.2.2 Sınıf Kütüphanesinin Oluşturulması



Admin.cs: Admin nesneleri oluşturmak için;

```
public class Admin
{
    public string adminID { get; set; }
    public string ad { get; set; }
    public string soyad { get; set; }
    public string eMail { get; set; }
    public string sifre { get; set; }
    public string telefon { get; set; }
    public string adres { get; set; }
}
```

Uye.cs: Üye nesneleri oluşturmak için;

```
public class Uye
{
    public int uyeID { get; set; }
    public string ad { get; set; }
    public string soyad { get; set; }
    public string eMail { get; set; }
    public string sifre { get; set; }
    public string telefon { get; set; }
    public string adres { get; set; }
    public int adminID { get; set; }
    public bool uyelikDurumu { get; set; }
    public string cezaTarihi { get; set; }
}
```

Kategori.cs Kategori nesneleri oluşturmak için;

```
public class Kategori
{
    public int kategoriID { get; set; }
```

```
        public string ad { get; set; }  
    }
```

Kitap.cs: Kitap nesneleri oluşturmak için;

```
public class Kitap  
{  
    public int kitapID { get; set; }  
    public string ISBN { get; set; }  
    public string ad { get; set; }  
    public string yazar { get; set; }  
    public string baskiYili { get; set; }  
    public string yayinEvi { get; set; }  
    public string sayfaSayisi { get; set; }  
    public byte[] fotograf { get; set; }  
    public string aciklama { get; set; }  
    public string kategori { get; set; }  
    public bool kitapDurumu { get; set; }  
    public int islemSayisi { get; set; } //(En çok okunanları tutabilmek için)  
    public int adminID { get; set; }  
}
```

Buraya veri girişi olmayacak işlem gördükçe artacak.

Login.cs: Giriş yapan kişiyi oluşturup gerekli yerlerde çağırabilmek için;

```
public class Login  
{  
    public string girisYapID { get; set; }  
    public string eMail { get; set; }  
    public string sifre { get; set; }  
  
    public string adSoyad { get; set; }  
}
```

Islem.cs: İşlem nesnesi oluşturmak için;

```
public class Islem  
{  
    public int islemID { get; set; }  
    public int uyeID { get; set; }  
    public int kitapID { get; set; }  
    public string islemTarihi { get; set; }  
    public string iadeTarihi { get; set; }  
    public bool islemDurumu { get; set; }  
  
    public int adminID { get; set; }  
    public string emanetDurumu { get; set; }  
}
```

SqlConnection.cs: Bu sınıf mysql de oluşturulmuş procedurelerin çalıştırılmasını sağlar. Veritabanı bağlantı adresinin tutulduğu tek yerdir.

```
public class SqlConnection
{
    public static string baglanti_adresi = @"Server=localhost; Database =
kutuphane; Uid = root; Pwd =1234 ;";
    bool status; //Prosedürler bool değer gönderirler.

    //KİTAP İŞLEMLERİ.....
    public bool kitapOlustur(Kitap kitap)
    public bool kitapGuncelle(Kitap kitap)
    public bool kitapSil(Kitap kitap)
    public bool kitapUzat(Islem işlem)
    //.....

    //ADMİN İŞLEMLERİ.....
    public bool adminOlustur(Admin admin)
    public bool adminGuncelle(Admin admin)
    public bool adminSil(Admin admin)
    //.....

    //UYE İŞLEMLERİ.....
    public bool uyeOlustur(Uye uye)
    public bool uyeGuncelle(Uye uye)
    public bool uyeSil(Uye uye)
    //.....

    //EMANET İŞLEMLERİ.....
    public bool işlemOlustur(Islem işlem)
    public bool işlemSil(Islem işlem)
    //.....

    //GİRİŞ YAP.....
    public bool girisYap(Login login, string secim)
    //.....
}
```


4. Tasarımların Entegre Edilmesi ve Çalıştırılması

Helper.cs: Uygulamamıza arayüz tarafında yardım sağlayan datagridview ve comboboxlara verileri çekmemiz için contractor çalıştırdığımız ve veritabanına select sorguları gönderen, ek olarak giriş yapan kişinin tipini ve ID sini tutan sınıftır.

```
public class Helper
{
    public static string baglanti_adresi = SqlConnection.baglanti_adresi;
    public static string girisYapID; //sisteme kim giriş yaparsa ID sini burada tutar.
    Gerekli yerlere gönderir.
    public static string secim;

    //datagridde veri çekme
    public Helper(string sorgu, string v, DataGridView dataGridView1)
    {
        try
        {
            using (MySqlConnection connection = new MySqlConnection(baglanti_adresi))
            {
                if (v == "uye")
                {
                    connection.Open();

                    // Bir üye kitabı geç getirdiğinde ceza tarihi verilir. Bu ceza süresi bitmişse üyeyi aktif yapar.
                    MySqlCommand command = new MySqlCommand($"UPDATE uye SET uyelikDurumu = 1 where cezaTarihi < current_date() AND uyelikDurumu = 0", connection);
                    // Bir üye kitabı geç getirdiğinde ceza tarihi verilir. Bu ceza süresi bitmişse üyeyi aktif yapar.
                    MySqlCommand comman2 = new MySqlCommand($"UPDATE uye SET uyelikDurumu = 0 where uyelikDurumu = 1 AND uyeID IN (SELECT uyeID FROM islem where iadeTarihi < current_date())", connection);

                    command.ExecuteNonQuery();
                    comman2.ExecuteNonQuery();
                    connection.Close();

                }

                MySqlDataAdapter da = new MySqlDataAdapter($"Select {sorgu} From {v}", connection);

                DataSet ds = new DataSet();
                connection.Open();
                da.Fill(ds, v);
                dataGridView1.DataSource = ds.Tables[v];
                connection.Close();

            }
        }
        catch (Exception)
        {
            MessageBox.Show("Veritabanı hatası...");
        }
    }

    //combobox
    public Helper(string sorgu, string v, ComboBox combo)
    {
        try
        {
            using (MySqlConnection connection = new MySqlConnection(baglanti_adresi))
```

```

        {
            MySqlCommand command = new MySqlCommand($"Select {sorgu} From {v}",
connection);
            connection.Open();

            MySqlDataReader reader = command.ExecuteReader();
            while (reader.Read())
            {
                combo.Items.Add(reader.GetString(sorgu));
            };
            connection.Close();
        }
    }
    catch (Exception)
    {
        MessageBox.Show("Veritabanı hatası...");
    }
}

// Uye ve admin
public Helper(string sorgu, string v, string arama, DataGridView dataGridView1)
{
    try
    {
        using (MySqlConnection connection = new MySqlConnection(baglanti_adresi))
        {
            //UPDATE uye SET uyelikDurumu = 1 where cezaTarihi<current_date()
            MySqlDataAdapter da = new MySqlDataAdapter($"Select {sorgu} From {v} where ad
LIKE '{arama}%' OR soyad LIKE '{arama}%' OR eMail LIKE '{arama}%', connection);
            DataSet ds = new DataSet();
            connection.Open();
            da.Fill(ds, v);
            dataGridView1.DataSource = ds.Tables[v];
            connection.Close();
        }
    }
    catch (Exception)
    {
        MessageBox.Show("Veritabanı hatası...");
    }
}

// Kitap
public Helper(string kitap, string sorgu, string v, string arama, DataGridView
dataGridView1)
{
    try
    {
        using (MySqlConnection connection = new MySqlConnection(baglanti_adresi))
        {
            MySqlDataAdapter da = new MySqlDataAdapter($"Select {sorgu} From {v} where
t1.ad LIKE '{arama}%' OR t1.yazar LIKE '{arama}%' OR t3.kategoriAdi LIKE '{arama}%' OR t1.ISBN
LIKE '{arama}%', connection);
            DataSet ds = new DataSet();
            connection.Open();
            da.Fill(ds, v);
            dataGridView1.DataSource = ds.Tables[v];
            connection.Close();
        }
    }
    catch (Exception)

```

```

        {
            MessageBox.Show("Veritabanı hatası...");
        }
    }

    // İşlem
    public Helper(int kitap, string sorgu, string v, string arama, DataGridView
dataGridView1)
    {
        try
        {
            using (MySqlConnection connection = new MySqlConnection(baglanti_adresi))
            {
                MySqlDataAdapter da = new MySqlDataAdapter($"Select {sorgu} From {v} where
t1.islemID LIKE '{arama}%' OR t3.ad LIKE '{arama}%' OR t2.ad LIKE '{arama}%' OR t3.soyad LIKE
'{arama}%' OR t2.ISBN LIKE '{arama}%', connection);
                DataSet ds = new DataSet();
                connection.Open();
                da.Fill(ds, v);
                dataGridView1.DataSource = ds.Tables[v];
                connection.Close();
            }
        }
        catch (Exception)
        {
            MessageBox.Show("Veritabanı hatası...");
        }
    }
}

```

FormGirisYap.cs: Bu formda giriş yap butonuna tıkladığımızda SqlConnection.cs formunda çalıştırılan prosedürün kodları şu şekilde;

```

//GİRİŞ YAP
public bool girisYap(Login login, string secim)
{
    using (MySqlConnection connection = new MySqlConnection(baglanti_adresi))
    {
        MySqlCommand command = new MySqlCommand();
        MySqlDataReader rd;
        command.Connection = connection;
        connection.Open();
        command.CommandText = $"SELECT {secim}ID, ad, soyad FROM " + secim + " where
eMail = '" + login.eMail + "' AND sifre='" + login.sifre + "'";

        rd = command.ExecuteReader();

        if (rd.Read())
        {
            login.girisYapID = rd.GetValue(0).ToString();
            login.adSoyad = rd.GetValue(1).ToString() + " " + rd.GetValue(2).ToString();
            connection.Close();
            return true;
        }
    }
}

```

```

        else
        {
            connection.Close();
            return false;
        }

    }

}

```

FormKitapIslemleri.cs: Bu formda kitap işlemleri yapıldığında SqlConnection.cs de çalışan kodlar;

Kitap Oluştur Fonksiyonu:

```

public bool kitapOlustur(Kitap kitap)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[10];
        pms[0] = new MySqlParameter("p_ISBN", MySqlDbType.VarChar);
        pms[0].Value = kitap.ISBN;

        pms[1] = new MySqlParameter("p_ad", MySqlDbType.VarChar);
        pms[1].Value = kitap.ad;

        pms[2] = new MySqlParameter("p_yazar", MySqlDbType.VarChar);
        pms[2].Value = kitap.yazar;

        pms[3] = new MySqlParameter("p_baskiYili", MySqlDbType.VarChar);
        pms[3].Value = kitap.baskiYili;

        pms[4] = new MySqlParameter("p_yayinEvi", MySqlDbType.VarChar);
        pms[4].Value = kitap.yayinEvi;

        pms[5] = new MySqlParameter("p_sayfaSayisi", MySqlDbType.VarChar);
        pms[5].Value = kitap.sayfaSayisi;
        pms[6] = new MySqlParameter("p_fotograf", MySqlDbType.Blob);
        pms[6].Value = kitap.fotograf;
        pms[7] = new MySqlParameter("p_aciklama", MySqlDbType.VarChar);
        pms[7].Value = kitap.aciklama;
        pms[8] = new MySqlParameter("p_kategoriAd", MySqlDbType.VarChar);
        pms[8].Value = kitap.kategori;
        pms[9] = new MySqlParameter("p_adminID", MySqlDbType.Int32);
        pms[9].Value = kitap.adminID;
        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_kitapEkle";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

Kitap Güncelle Fonksiyonu:

```
public bool kitapGuncelle(Kitap kitap)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[10];
        pms[0] = new MySqlParameter("p_ISBN", MySqlDbType.VarChar);
        pms[0].Value = kitap.ISBN;

        pms[1] = new MySqlParameter("p_ad", MySqlDbType.VarChar);
        pms[1].Value = kitap.ad;

        pms[2] = new MySqlParameter("p_yazar", MySqlDbType.VarChar);
        pms[2].Value = kitap.yazar;

        pms[3] = new MySqlParameter("p_baskiYili", MySqlDbType.VarChar);
        pms[3].Value = kitap.baskiYili;

        pms[4] = new MySqlParameter("p_yayinEvi", MySqlDbType.VarChar);
        pms[4].Value = kitap.yayinEvi;

        pms[5] = new MySqlParameter("p_sayfaSayisi", MySqlDbType.VarChar);
        pms[5].Value = kitap.sayfaSayisi;
        pms[6] = new MySqlParameter("p_fotograf", MySqlDbType.Blob);
        pms[6].Value = kitap.fotograf;
        pms[7] = new MySqlParameter("p_aciklama", MySqlDbType.LongText);
        pms[7].Value = kitap.aciklama;
        pms[8] = new MySqlParameter("p_kategoriAd", MySqlDbType.VarChar);
        pms[8].Value = kitap.kategori;
        pms[9] = new MySqlParameter("p_adminID", MySqlDbType.Int32);
        pms[9].Value = kitap.adminID;

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_kitapGuncelle";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}
```

Kitap Sil Fonksiyonu:

```
public bool kitapSil(Kitap kitap)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[2];
        pms[0] = new MySqlParameter("p_ISBN", MySqlDbType.VarChar);
        pms[0].Value = kitap.ISBN;
        pms[1] = new MySqlParameter("p_adminID", MySqlDbType.Int32);
        pms[1].Value = kitap.adminID;

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_kitapSil";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}
```

FormAdminIslemleri.cs: Bu formda çalışan işlemlerde SqlConnectorda çalışan prosedür kodları:

Admin Ekle Fonkiyonu:

```
public bool adminOlustur(Admin admin)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[5];
        pms[0] = new MySqlParameter("p_ad", MySqlDbType.VarChar);
        pms[0].Value = admin.ad;

        pms[1] = new MySqlParameter("p_soyad", MySqlDbType.VarChar);
        pms[1].Value = admin.soyad;

        pms[2] = new MySqlParameter("p_eMail", MySqlDbType.VarChar);
        pms[2].Value = admin.eMail;

        pms[3] = new MySqlParameter("p_telefon", MySqlDbType.VarChar);
        pms[3].Value = admin.telefon;

        pms[4] = new MySqlParameter("p_adres", MySqlDbType.VarChar);
        pms[4].Value = admin.adres;

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_adminEkle";
    }
}
```

```

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

Admin Güncelleme Fonksiyonu:

```

public bool adminGuncelle(Admin admin)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[5];
        pms[0] = new MySqlParameter("p_ad", MySqlDbType.VarChar);
        pms[0].Value = admin.ad;

        pms[1] = new MySqlParameter("p_soyad", MySqlDbType.VarChar);
        pms[1].Value = admin.soyad;

        pms[2] = new MySqlParameter("p_eMail", MySqlDbType.VarChar);
        pms[2].Value = admin.eMail;

        pms[3] = new MySqlParameter("p_telefon", MySqlDbType.VarChar);
        pms[3].Value = admin.telefon;
        pms[4] = new MySqlParameter("p_adres", MySqlDbType.VarChar);
        pms[4].Value = admin.adres;
        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_adminGuncelle";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

Admin Sil Fonksiyonu:

```

public bool adminSil(Admin admin)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[1];

        pms[0] = new MySqlParameter("p_eMail", MySqlDbType.VarChar);
        pms[0].Value = admin.eMail;
    }
}

```

```

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_adminSil";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

FormEmanetIslemleri.cs: Bu formda kitap işlemleri yapıldığında SqlConnector.cs de çalışan kod.

Kitap Uzat Fonksiyonu:

```

public bool kitapUzat(Islem islem)
{
    using (MySQLConnection connection = new MySQLConnection(baglanti_adresi))
    {
        MySQLParameter[] pms = new MySQLParameter[2];
        pms[0] = new MySQLParameter("p_islemID", MySQLDbType.Int32);
        pms[0].Value = islem.islemID;

        pms[1] = new MySQLParameter("p_iadeTarihi", MySQLDbType.VarChar);
        pms[1].Value = islem.iadeTarihi;

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_kitapUzat";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

Emanet Oluşturma Fonksiyonu:

```

public bool islemOlustur(Islem islem)
{
    using (MySQLConnection connection = new MySQLConnection(baglanti_adresi))
    {
        MySQLParameter[] pms = new MySQLParameter[6];
        pms[0] = new MySQLParameter("p_uyeID", MySQLDbType.Int32);
        pms[0].Value = islem.uyeID;

        pms[1] = new MySQLParameter("p_kitapID", MySQLDbType.Int32);

```



```

pms[1].Value = islem.kitapID;

pms[2] = new MySqlParameter("p_adminID", MySqlDbType.Int32);
pms[2].Value = islem.adminID;

pms[3] = new MySqlParameter("p_islemTarihi", MySqlDbType.Date);
pms[3].Value = islem.islemTarihi;

pms[4] = new MySqlParameter("p_iadeTarihi", MySqlDbType.Date);
pms[4].Value = islem.iadeTarihi;
pms[5] = new MySqlParameter("p_emanetDurumu", MySqlDbType.VarChar);
pms[5].Value = islem.emanetDurumu;

MySqlCommand command = new MySqlCommand();

command.Connection = connection;
command.CommandType = CommandType.StoredProcedure;
command.CommandText = "p_islemEkle";

command.Parameters.AddRange(pms);

connection.Open();
if (command.ExecuteNonQuery() == 1)
    status = true;
else
    status = false;
connection.Close();
}
return status;
}

```

Emaneti Geri alma Fonksiyonu:

```

public bool islemSil(Islem islem)
{
    using (MySqlConnection connection = new MySqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[1];
        pms[0] = new MySqlParameter("p_islemID", MySqlDbType.Int32);
        pms[0].Value = islem.islemID;

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_islemSil";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

FormUyeIslemleri.cs: Bu formda kitap işlemleri yapıldığında SqlConnection.cs de çalışan kod.

Uye Oluştur Fonksiyonu:

```
public bool uyeOlustur(Uye uye)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[7];
        pms[0] = new MySqlParameter("p_ad", MySqlDbType.VarChar);
        pms[0].Value = uye.ad;

        pms[1] = new MySqlParameter("p_soyad", MySqlDbType.VarChar);
        pms[1].Value = uye.soyad;

        pms[2] = new MySqlParameter("p_eMail", MySqlDbType.VarChar);
        pms[2].Value = uye.eMail;

        pms[3] = new MySqlParameter("p_telefon", MySqlDbType.VarChar);
        pms[3].Value = uye.telefon;

        pms[4] = new MySqlParameter("p_adres", MySqlDbType.VarChar);
        pms[4].Value = uye.adres;
        pms[5] = new MySqlParameter("p_adminID", MySqlDbType.Int32);
        pms[5].Value = uye.adminID;
        pms[6] = new MySqlParameter("p_cezaTarihi", MySqlDbType.Date);
        pms[6].Value = uye.cezaTarihi;

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_uyeEkle";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}
```

Üye Güncelleme Fonksiyonu:

```
public bool uyeGuncelle(Uye uye)
{
    using (SqlConnection connection = new SqlConnection(baglanti_adresi))
    {
        MySqlParameter[] pms = new MySqlParameter[5];
        pms[0] = new MySqlParameter("p_ad", MySqlDbType.VarChar);
        pms[0].Value = uye.ad;

        pms[1] = new MySqlParameter("p_soyad", MySqlDbType.VarChar);
        pms[1].Value = uye.soyad;

        pms[2] = new MySqlParameter("p_eMail", MySqlDbType.VarChar);
        pms[2].Value = uye.eMail;

        pms[3] = new MySqlParameter("p_telefon", MySqlDbType.VarChar);
        pms[3].Value = uye.telefon;

        pms[4] = new MySqlParameter("p_adres", MySqlDbType.VarChar);
        pms[4].Value = uye.adres;
```

```

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_uyeGuncelle";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

Üye silme Fonksiyonu:

```

public bool uyeSil(Uye uye)
{
    using (MySQLConnection connection = new MySQLConnection(baglanti_adresi))
    {
        MySQLParameter[] pms = new MySQLParameter[2];

        pms[0] = new MySQLParameter("p_eMail", MySQLDbType.VarChar);
        pms[0].Value = uye.eMail;
        pms[1] = new MySQLParameter("p_adminID", MySQLDbType.Int32);
        pms[1].Value = uye.adminID;

        MySqlCommand command = new MySqlCommand();

        command.Connection = connection;
        command.CommandType = CommandType.StoredProcedure;
        command.CommandText = "p_uyeSil";

        command.Parameters.AddRange(pms);

        connection.Open();
        if (command.ExecuteNonQuery() == 1)
            status = true;
        else
            status = false;
        connection.Close();
    }
    return status;
}

```

Uygulama tarafında çalıştırılan SQL komutları: Bu komutlar helper.cs sınıfına gönderilerek çalışan ve bazı formlarda çalışan komutlardır.

1. FormEmanetIslemleri.cs Çalışan Sorgular: Uye, kitap ve işlem verilerini getiren sorgular içerir.

ID	Ad	Soyad	Email	ISBN	Kitap	Fotoğraf	İadeTarihi
30	Ahmet	Yürekli	ahmetyurekli@gm...	00-MR-E	Eylül		10.06.2020
46	Fatih	Demir	fatihdemir@gmail...	00-KK-VM	Veritabanı Mantığı		9.06.2020

#İŞLEM LİSTELEME

```
SELECT t1.islemID as ID, t3.ad as Ad, t3.soyad as Soyad, t3.eMail as Email,
t2.ISBN as ISBN,t2.ad as Kitap, t2.fotograf as Fotoğraf, t1.iadeTarihi
as İadeTarihi
FROM islem t1
INNER JOIN kitap t2 ON t1.kitapID = t2.kitapID
INNER JOIN uye t3 ON t1.uyeID = t3.uyeID;
```

ID	Ad	Soyad	Email	Tel	Durumu
33	Ahmet	Yürekli	ahmetyurekli@g...	54599999	1
34	Alperen	Dumaz	alperendumaz@...	54577777	0

#ÜYE LİSTELEME

```
SELECT uyeID as ID,ad as Ad, soyad as Soyad, eMail as Email, telefon as Tel,
uyelikDurumu as Durumu FROM uye;
```

ID	ISBN	Ad	Yazar	Kategori	Yıl	Yayın	Foto
9	00-MR-E	Eylül	Mehmet Rauf	Edebiyat	1950	Can	
11	00-SA-KY	Kuyucaklı ...	Sabahattin ...	Edebiyat	1889	Can	
12	00-OA-T	Tutunamav	Öğüz Atay	Edebiyat	1972	Can	

#KİTAP LİSTELEME

```
SELECT t1.kitapID as ID,t1.ISBN,t1.ad as Ad,t1.yazar as Yazar, t3.kategoriAdi
as Kategori, t1.baskiYili as Yıl, t1.yayinEvi as Yayın, t1.fotograf as Foto
FROM kitap t1
INNER JOIN kitap_kategori t2 ON t1.kitapID = t2.kitapID
INNER JOIN kategori t3 ON t2.kategoriID = t3.kategoriID;
```

2. FormFirstKitapScreen.cs Çalışan Sorgu: Kitabın tüm verilerini getirir. Varsa iade tarihini işlem yoksa 'Rafta' olarak değiştiren sql komutu:

ISBN	Ad	Yazar	Kategori	Yıl	Yayın	Sayfa	Foto	Detay	Durum
00-MR-E	Eylül	Mehmet Rauf	Edebiyat	1950	Can	180		Mehmet Rau...	2020-06-10
00-KK-VM	Veritabanı M...	Kerem Köse...	Bilişim	2018	Toprak	150		Veritabanı...	2020-06-09
00-SA-KY	Kuyucaklı Yu...	Sabahattin Ali	Edebiyat	1889	Can	180		Sabahattin A...	Rafta
00-OA-T	Tutunamaya...	Oğuz Atay	Edebiyat	1972	Can	500		Tutunamıyor	Rafta

#ÜYE TARAFINA GÖRÜNEN KİTAP LİSTELEME (DURUM İÇEREN)

```
SELECT t1.ISBN,t1.ad as Ad,t1.yazar as Yazar, t3.kategoriAdi as Kategori,
t1.baskiYili as Yıl, t1.yayinEvi as Yayın,t1.sayfaSayisi as Sayfa,
t1.fotograf as Foto, t1.aciklama as Detay,
IFNULL(t4.emanetDurumu,'Rafta') as Durum
FROM kitap t1
INNER JOIN kitap_kategori t2 ON t1.kitapID = t2.kitapID
INNER JOIN kategori t3 ON t2.kategoriID = t3.kategoriID
LEFT JOIN islem t4 ON t1.kitapID = t4.kitapID;
```

3. FormKategoriEkle.cs Çalışan Sorgu: Kategori verilerini güncelleme ve silme sorguları çalıştırır.Textteki veriyi geçici olarak temp değişkeninde saklar ve aynı textte yapılan değişiklik ile eski veriyi yenisiyle günceller. Aynı işlemi silme de de yapar.

#COMBOBOX A LİSTELEME

```
SELECT kategoriAdi FROM kategori;
```

#KATEGORİ EKLEME

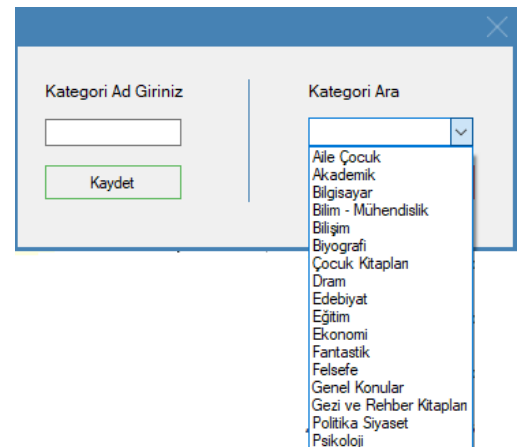
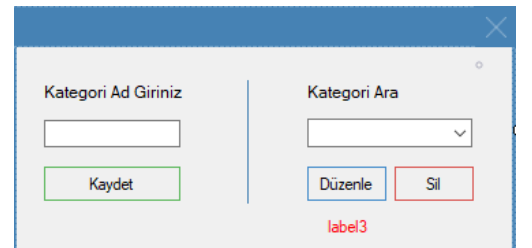
```
INSERT INTO kategori(kategoriAdi)
VALUES ('{KateEkleTxt.Text}');
```

#KATEGORİ GÜNCELLEME

```
Update kategori set kategoriAdi =
'{KateEkleTxt.Text}'
where kategoriAdi = '{temp}';
```

#KATEGORİ SİLME

```
DELETE FROM kategori WHERE kategoriAdi='{temp}';
```



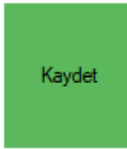
4. FormKitapIslemleri.cs Çalışan Sorgu: Kitapların listelenmesi sağlar.

Kitap Adı, Yazar Adı, ISBN, Kategori Adı							Ara	
ISBN	Ad	Yazar	Kategori	Yıl	Yayın	Sayfa	Foto	Detay
00-MR-E	Eylül	Mehmet ...	Edebiyat	1950	Can	180		Mehmet ...
00-SA-KY	Kuyucaklı...	Sabahatti...	Edebiyat	1889	Can	180		Sabahatti...

#KİTAP LİSTELEME

```
SELECT t1.ISBN,t1.ad as Ad,t1.yazar as Yazar, t3.kategoriAdi as Kategori,
t1.baskiYili as Yıl, t1.yayinEvi as Yayın, t1.sayfaSayisi as Sayfa,
t1.fotograf as Foto, t1.aciklama as Detay
FROM kitap t1
INNER JOIN kitap_kategori t2 ON t1.kitapID = t2.kitapID
INNER JOIN kategori t3 ON t2.kategoriID = t3.kategoriID;
```

5. FormProfilAyarlari.cs Çalışan Sorgu: Bu formda giriş yapan kişinin bilgilerini çeken sorgular yapılır. Helper sınıfından giriş yapanın ID si ve Türü alınır. Kişinin verileri textboxlara aktarılır.

Şifre :	<input type="text" value="samsun"/>	
Telefon :	<input type="text" value="545999999"/>	
Adres :	<input type="text" value="Samsun"/>	

Başarıyla Değiştirildi...

#LİSTELEME

```
SELECT ad, soyad, eMail, sifre, telefon, adres FROM {tablo}
WHERE {tablo}ID={Helper.girisYapID};
```

#GÜNCELLEME

```
UPDATE {tablo}
SET sifre='{sifreTxt.Text}', telefon='{telefonTxt.Text}', adres='{adresTxt.Text}';
```

6. FormUyeIslemleri.cs Çalışan Sorgu: Üye bilgilerini listeler.

Ad, Soyad, E-Mail				Ara
Ad	Soyad	Email	Tel	Adres
Ahmet	Yürekli	ahmetyurekli@gmail....	54599999	Ankara
Alperen	Dumaz	alperendumaz@gmai...	54577777	Karabük

#ÜYE LİSTELEME

```
SELECT ad as Ad, soyad as Soyad, eMail as Email, telefon as Tel,
adres as Adres FROM uye;
```

7. FormAdminIslemleri.cs Çalışan Sorgu: Admin bilgilerini listeler.

Ad, Soyad, E-Mail				Ara
Ad	Soyad	Email	Tel	Adres
Mürsel	Elibol	murselelibol@gmail...	111111	Samsun
Eren	Şaşkın	erensaskin@gmail....	5455555	İstanbul

#ADMİN LİSTELEME

```
SELECT ad as Ad, soyad as Soyad, eMail as Email, telefon as Tel,
adres as Adres FROM admin;
```

8. FormUyeIadeEt.cs Çalışan Sorgu: Üyenin daha önce iade ettiği kitapları log tablosundan alıp listeler.

ad	yazar	fotograf	iadeTarihi
Simyacı	Paulo Coelho		2020-05-25 13:41:42
Köyden Bir Çocuk	Seyyid Kutub		2020-05-25 13:41:48

#ÜYE DAHA ÖNCE ALDIĞI KİTAPLARI LİSTELERKEN...

```
SELECT t1.ad, t1.yazar, t1.fotograf, t2.iadeTarihi
FROM kitap t1
INNER JOIN log t2 ON t1.kitapID = t2.kitapID
WHERE t2.uyeID = {Helper.girisYapID} AND t2.aciklama = 'Kitap alındı..';
```

9. FormUyeUzerimdekiler.cs Çalışan Sorgu: Üyenin emanet aldığı kitapları listeler.

ID	Kitap	Yazar	Fotoğraf	İadeTarihi
46	Veritabanı Mantığı	Kerem Köseoğlu		9.06.2020

#ÜYENİN ÜZERİNDEKİLERİ GÖSTEREN SORGU

```
SELECT t1.islemID as ID, t2.ad as Kitap, t2.yazar as Yazar,
t2.fotograf as Fotoğraf, t1.iadeTarihi as İadeTarihi
FROM islem t1
INNER JOIN kitap t2 ON t1.kitapID = t2.kitapID
INNER JOIN uye t3 ON t1.uyeID = t3.uyeID where t1.uyeID = {Helper.girisYapID};
```


10. Helper.cs Çalışan Sorgu: Üye bir emaneti iade tarihinden geç getirirse +15 gün cezalı duruma düşer, üyelik durumu 0 (pasif) yapar ve 15 gün boyunca kitap alamaz olur. Eğer Ceza tarihi sona erdiyse üyenin durumunu 1 (aktif) yaparak kitap alabilir duruma getirir.

#ÜYENİN DURUMUNU CEZATARİHİNE GÖRE GÜNCELLER

```
UPDATE uye SET uyelikDurumu = 1
where cezaTarihi < current_date() AND uyelikDurumu = 0;

UPDATE uye SET uyelikDurumu = 0
where uyelikDurumu = 1 AND uyeID IN
(SELECT uyeID FROM islem where iadeTarihi < current_date());
```

11. Arama Sorguları: Uygulamada kullanılan arama sorguları.

Ad, Soyad, E-Mail	Ara
-------------------	-----

#ÜYE ARAMA

```
SELECT ad as Ad, soyad as Soyad, eMail as Email, telefon as Tel, adres as Adres
From uye
where ad LIKE '%{arama}%' OR soyad LIKE '%{arama}%' OR eMail LIKE '%{arama}%';
```

Ad, Soyad, E-Mail	Ara
-------------------	-----

#ADMİN ARAMA

```
SELECT ad as Ad, soyad as Soyad, eMail as Email, telefon as Tel, adres as Adres
From admin
where ad LIKE '%{arama}%' OR soyad LIKE '%{arama}%' OR eMail LIKE '%{arama}%';
```

Kitap Adı, Yazar Adı, ISBN, Kategori Adı	Ara
--	-----

#KİTAP ARAMA

```
Select t1.ISBN,t1.ad as Ad,t1.yazar as Yazar, t3.kategoriAdi as Kategori,
t1.baskiYili as Yıl, t1.yayinEvi as Yayın, t1.sayfaSayisi as Sayfa,
t1.fotograf as Foto, t1.aciklama as Detay
From kitap t1
INNER JOIN kitap_kategori t2 ON t1.kitapID = t2.kitapID
INNER JOIN kategori t3 ON t2.kategoriID = t3.kategoriID
where t1.ad LIKE '%{arama}%' OR t1.yazar LIKE '%{arama}%'
OR t3.kategoriAdi LIKE '%{arama}%' OR t1.ISBN LIKE '%{arama}%';
```

#İŞLEM ARAMA

```
SELECT t1.islemID as ID, t3.ad as Ad, t3.soyad as Soyad,  
t3.eMail as Email, t2.ISBN as ISBN,t2.ad as Kitap,  
t2.fotograf as Fotoğraf, t1.iadeTarihi as İadeTarihi  
From islem t1  
INNER JOIN kitap t2 ON t1.kitapID = t2.kitapID  
INNER JOIN uye t3 ON t1.uyeID = t3.uyeID  
where t1.islemID LIKE '%{arama}%'  
OR t3.ad LIKE '%{arama}%'  
OR t2.ad LIKE '%{arama}%'  
OR t3.soyad LIKE '%{arama}%'  
OR t2.ISBN LIKE '%{arama}%';
```