

**KÜTAHYA SAĞLIK BİLİMLERİ ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ**



Final Raporu

Halil Şimşek

2218111004

04.06.2024

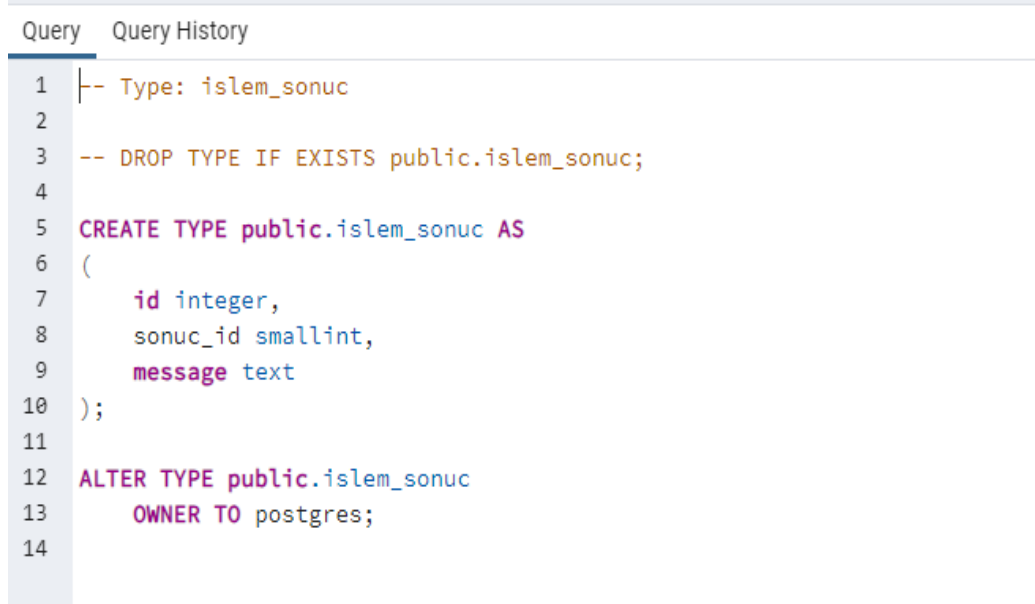
1 PostgreSQL Fonksiyonları ve Veri Girişi

Proje süreci kapsamında bu hafta, PostgreSQL’de fonksiyonların öğrenilmesi ve uygulanması üzerine çalışmalar gerçekleştirilmiştir. Veritabanı katmanında, kitap ekleme ve listeleme işlemleri başarılı bir şekilde tamamlanmıştır. Fonksiyonların öğrenilmesi aşamasında, Murat Yücedağ’ın YouTube kanalından faydalanılmıştır. (1)

```
Query Query History
5 CREATE OR REPLACE FUNCTION kitap.kitap_ekle(
6     p_ad character varying,
7     p_fiyat numeric,
8     p_yazar_ad character varying,
9     p_kategori_ad character varying)
10 RETURNS islem_sonuc
11 LANGUAGE 'plpgsql'
12 COST 100
13 VOLATILE PARALLEL UNSAFE
14 AS $BODY$
15 DECLARE
16     _isonuc public.islem_sonuc%rowtype;
17     _id integer;
18     _yazar_id integer;
19     _kategori_id integer;
20 BEGIN
21 |
22 SELECT id INTO _yazar_id FROM tbl_yazar WHERE yazar_ad = p_yazar_ad LIMIT 1;
23 IF _yazar_id IS NULL THEN
24     INSERT INTO tbl_yazar (yazar_ad) VALUES (p_yazar_ad) RETURNING id INTO _yazar_id;
25 END IF;
26
27 SELECT id INTO _kategori_id FROM tbl_kategori WHERE kategori_ad = p_kategori_ad LIMIT 1;
28 IF _kategori_id IS NULL THEN
29     INSERT INTO tbl_kategori (kategori_ad) VALUES (p_kategori_ad) RETURNING id INTO _kategori_id;
30 END IF;
31
32 IF EXISTS (SELECT * FROM tbl_kitap WHERE ad = p_ad LIMIT 1) THEN
33     SELECT 0, 3, 'Aynı kitap tekrar eklenemez' INTO _isonuc;
34     RETURN _isonuc;
35 END IF;
36
37 INSERT INTO tbl_kitap (ad, fiyat, yazar_id, kategori_id)
38 VALUES (p_ad, p_fiyat, _yazar_id, _kategori_id);
39
40 _id := lastval();
41
42 SELECT _id, 1, 'Kitap başarıyla eklendi' INTO _isonuc;
43
44 RETURN _isonuc;
```

Şekil 1: PostgreSql’de Fonksiyon Kullanımı

1.1 Type Kullanımının Avantajları



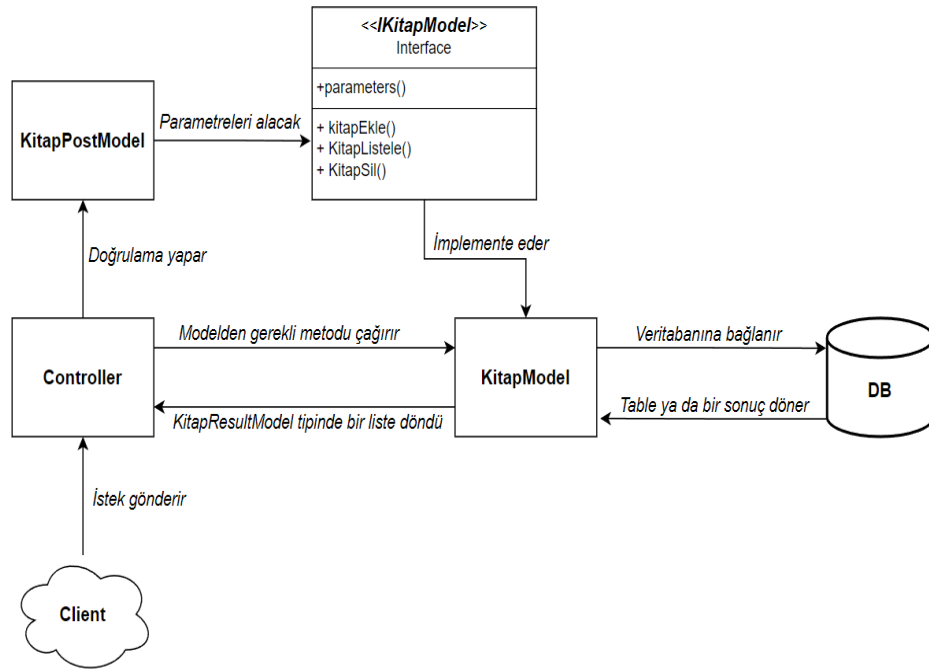
```
Query  Query History
1  |-- Type: islem_sonuc
2
3  -- DROP TYPE IF EXISTS public.islem_sonuc;
4
5  CREATE TYPE public.islem_sonuc AS
6  (
7      id integer,
8      sonuc_id smallint,
9      message text
10 );
11
12 ALTER TYPE public.islem_sonuc
13     OWNER TO postgres;
14
```

Şekil 2: Type Kullanımının Gösterimi

- **Daha İyi Tip Güvenliği:** Geri dönüş değeri olarak tip kullanmak, belirli bir türün dönmesini beklediğinizde daha sağlam bir tip güvenliği sağlar. Tip belirtmek, beklenmeyen veri türü dönüşlerine karşı kodunuzu korur ve hataları önler.
- **Kod Okunabilirliği ve Anlaşılabilirlik:** Fonksiyonun geri dönüş değerini belirtmek, kodun daha okunabilir ve anlaşılabilir olmasını sağlar. Fonksiyonun ne tür bir değer döndüreceği hakkında açık bir bilgi verir, bu da kodu anlamayı ve bakımını daha kolay hale getirir.
- **Dökümantasyon ve API Uyumluluğu:** Geri dönüş değerini belirtmek, fonksiyonun dış dünyayla etkileşimini açıklığa kavuşturur. API'ler arasında tutarlılık sağlar ve dış kullanıcılar veya diğer geliştiriciler için fonksiyonun nasıl kullanılacağı hakkında açık bir belge sağlar.
- **Veri Doğrulama ve Kısıtlamalar:** PostgreSQL'de tip kullanarak, geri dönüş değeri üzerinde veri doğrulama ve kısıtlamalar uygulanabilir. Bu, istenmeyen değerlerin dönmesini önler ve veri bütünlüğünü korur.

Aynı zamanda ekleme,silme,güncelleme gibi ortak işler için bir type oluşturuldu. Bu type sayesinde yapılan işlemlere göre kullanıcıya bir mesaj verilebilecek. Ekleme fonksiyonunu kullanarak veri tabanındaki kitap sayısı da arttırılmıştır. Type Araştırmasında Chatgpt'den yararlanılmıştır. (2)

2 ASP.NET MVC Model Katmanı



Şekil 3: Model Katmanının İşleyişi

2.1 IKitapModel Arayüzü

Bu arayüz, kitaplarla ilgili işlemlerin sözleşmesini tanımlar. Yani, kitap listeleme ve kitap ekleme gibi işlevlerin prototiplerini içerir. KitapListe metodu, kitap listeleme işlemi için gerekli parametreleri (KitapPostModel) alır ve sonuç olarak bir KitapResultModel listesi döndürür. Bu metod, veritabanından kitap listesini almak için kullanılır.

```
1  using Bookcase.Models.Kitap.PostModel;
2  using Bookcase.Models.Kitap.Result;
3  using System.Data;
4
5  namespace Bookcase.Models.Kitap
6  {
7      public interface IKitapModel
8      {
9          List<KitapListeResultModel> KitapListe(KitapListePostModel postModel);
10         islemSonuc KitapEkle(KitapEklePostModel postModel);
11         KitapOkuResultModel KitapOku(KitapOkuPostModel PostModel);
12         List<KitapKategoriListeResultModel> KitapKategoriListe();
13     }
14 }
```

Şekil 4: IKitapModel Interface

2.2 KitapModel Sınıfı

IKitapModel arayüzünü uygular (implement eder) ve bu arayüzde tanımlanan metodların somut davranışlarını içerir. KitapListe metodunun somut uygulaması, PostgreSQL veritabanına bağlanır, bir SQL sorgusu çalıştırır ve sonuçları bir KitapResultModel listesine dönüştürür. Veritabanı bağlantısı ve sorgu işlemleri için NpgsqlConnection ve NpgsqlCommand nesnelerini kullanır.

```

public KitapModel()
{
    con = new NpgsqlConnection("Server=localhost;Port=5432;Database=BookcaseDB;User Id = bookcase_user; Password=Aa123456*");
    con.Open();

    cmd = new NpgsqlCommand
    {
        Connection = con,
        CommandTimeout = 600
    };
}

public List<KitapListeResultModel> KitapListe(KitapListePostModel PostModel)
{
    List<KitapListeResultModel> kitapList = [];

    cmd.CommandText = "select * from kitap.kitap_liste(p_ad=>@p_ad)";

    cmd.Parameters.Add(new NpgsqlParameter()
    {
        NpgsqlDbType = NpgsqlTypes.NpgsqlDbType.Varchar,
        Direction = ParameterDirection.Input,
        ParameterName = "@p_ad",
        Value = PostModel.kitap_ad == null ? DBNull.Value : PostModel.kitap_ad
    });

    DataTable dt = new();
    dt.Load(cmd.ExecuteReader());
    con.Close();

    foreach (DataRow row in dt.Rows)
    {
        KitapListeResultModel kitap = new()
        {
            id = Convert.ToInt32(row["id"]),
            ad = row["ad"].ToString(),
            fiyat = Convert.ToDecimal(row["fiyat"]),
            yazar_ad = row["yazar_ad"].ToString(),
            kategori_ad = row["kategori_ad"].ToString()
        };

        kitapList.Add(kitap);
    }

    return kitapList;
}

```

Şekil 5: KitapModel Sınıfının Kodları

2.3 KitapPostModel Sınıfı

Kullanıcıdan alınan verileri taşımak için kullanılır. Bu model, kitap listeleme işlemi sırasında kullanıcı tarafından sağlanan verileri (örneğin, bir kitap adı) içerir. `IValidatableObject` arayüzünü uygular ve `Validate` metodu ile modelin kendini doğrulamasını sağlar. Bu arabirim, bir sınıfın `Validate` metodunu uygulamasını gerektirir.

Bu metod, sınıfın kendi doğrulama mantığını içerir ve sınıfın durumunu değerlendirir. `Validate` metodu, geçerli bir nesne durumunu döndürdüğünde, sınıfın geçerli olduğu anlamına gelir. Geçerli bir nesne durumu döndürülmediğinde ise, sınıfın geçerliliği başarısız olur ve bu durumda hata mesajları veya hata listeleri ile birlikte geçerli olmayan nesne durumu döndürülebilir.

ASP.NET MVC veya ASP.NET Core MVC projelerinde, özellikle model doğrulaması yapmak için sıklıkla `IValidatableObject` arabirimi kullanılır. Bir modelin belirli doğrulama kurallarını uygulamak ve bu kurallara uygunluğunu kontrol etmek için `IValidatableObject` arabirimi uygulanabilir. Bu, gelen verilerin doğruluğunu sağlamak için önemli bir mekanizmadır ve hatalı girişlerin önlenmesine yardımcı olur.

```
3 namespace Bookcase.Models.Kitap.PostModel
4 {
5     public class KitapListePostModel : IValidatableObject
6     {
7         //[Required]
8         [StringLength(255, ErrorMessage = "Kitap adı 255 karakterden fazla olamaz")]
9         public string kitap_ad { get; set; }
10
11
12         public int start { get; set; }
13         public int length { get; set; }
14
15         public IEnumerable<ValidationResult> Validate(ValidationContext validationContext)
16         {
17             List<ValidationResult> results = [];
18             return results;
19         }
20     }
```

Şekil 6: KitapPostModel Sınıfının Kodları

2.4 KitapResultModel Sınıfı

Veritabanından dönen sonuçları temsil eder. Her bir KitapResultModel nesnesi, bir kitap kaydının id, ad, fiyat, yazarAd, kategoriAd gibi özelliklerini içerir.

```
1  namespace Bookcase.Models.Kitap.Result
2  {
3      public class KitapKategoriListeResultModel
4      {
5          public int id { get; set; }
6          public string kategori_ad { get; set; }
7      }
8      public class KitapListeResultModel
9      {
10         public int id { get; set; }
11         public string ad { get; set; }
12         public decimal fiyat { get; set; }
13         public string yazar_ad { get; set; }
14         public string kategori_ad { get; set; }
15     }
```

Şekil 7: KitapResultModel Sınıfının Kodları

2.5 Controller

MVC'nin Controller bileşenidir ve kullanıcı isteklerini yönetir. Index metodu, uygulamanın ana sayfasını döndürür ve genellikle kullanıcıya HTML içeriği sunar. kitapListe metodu, bir HTTP POST isteği ile çağrıldığında, KitapListePostModel tipindeki veriyi alır, modelin geçerliliğini kontrol eder ve KitapModel üzerinden kitap listesini çeker. Daha sonra bu listeyi JSON formatında istemciye (web tarayıcısına) geri döndürür.


```

[HttpPost]
public JsonResult kitapListe(KitapListePostModel postModel)
{
    try
    {
        if (!ModelState.IsValid)
        {
            //var errorMessages = ModelState.Values.SelectMany(v => v.Errors)
            //                                     .Select(e => e.ErrorMessage)
            //                                     .ToList();

            //return Json(new { success = false, errors = errorMessages });
        }

        IKitapModel kitapModel = new KitapModel();
        List<KitapListeResultModel> kitapList = kitapModel.KitapListe(postModel);

        /*paging*/

        var queryResultPage = kitapList
            .Skip(postModel.length * postModel.start / postModel.length)
            .Take(postModel.length);

        var _return = new DataGridReturnModel()
        {
            recordsTotal = kitapList.Count,
            recordsFiltered = kitapList.Count,
            data = queryResultPage,
        };

        return Json(_return);
    }
    catch (Exception ex)
    {
        return Json(new { success = false, error = ex.Message });
    }
}

```

Şekil 8: Controller Sınıfının Kodları

3 Listeleme İşleminin Gerçekleştirilmesi

Listeleme işlemi yapılırken hazır grid kullanılmıştır. Grid bu websiteden alınmıştır (4). Bootstrap kütüphanesinden yararlanılmıştır. Bootstrap hakkındaki araştırmalar ve öğrenimlerde (3) sitesinden yararlanılmıştır. Listelenen gridin son hali şekil:9 da bulunmaktadır.

10 ▾

Kitap Adı	Yazar Adı	Kategori	Kitap Fiyatı
🔍 Hayvan Çiftliği	George Orwell	Roman	29.99
🔍 Beyaz Diş	Jack London	Roman	19.99
🔍 İnsan Geleceğini Nasıl Kurar	İlber Ortaylı	Kişisel Gelişim	9.99
🔍 Nutuk	Mustafa Kemal Atatürk	Söylev	14.99
🔍 Savaş ve Barış	Tolstoy	Roman	20
🔍 Hayvanlardan Tanrılara Sapiens	Yuval Noah Harari	Tarih	12
🔍 Utangaç Ayı Monti	Duncan Beedie	Çocuk Kitabı	14.99
🔍 Simyacı	Paulo Coelho	Kişisel Gelişim	29.99
🔍 Yakın Tarihin Gerçekleri	İlber Ortaylı	Tarih	29.99
🔍 Kumarcı	Dostoyevski	Roman	17.99

Showing 1 to 10 of 34 entries

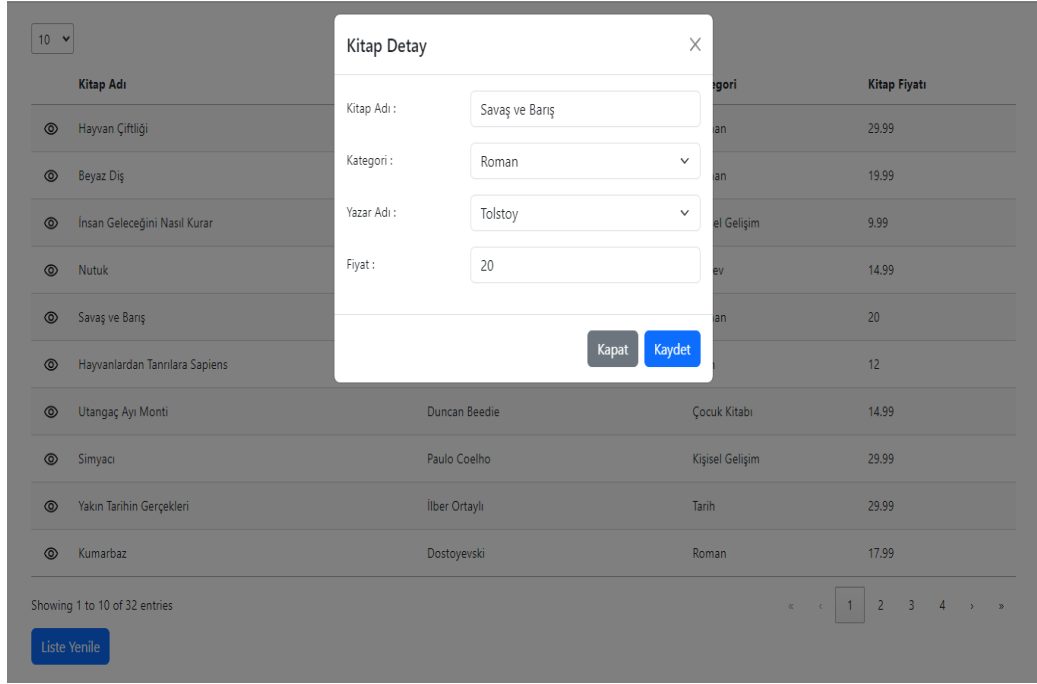
« ‹ 1 2 3 4 › »

Liste Yenile

Şekil 9: Listelenen Veriler

4 Detay Butonunun Eklenmesi

Gerçekleştirilen listeleme işlemi sonrasında bir detay butonu eklendi. Bu detay butonunun amacı aslında bu yapılan listemedeki sütun sayılarından çok daha fazlasına sahip tablolarda ana ekranda gösterilemeyen bilgileri detay ekranında göstererek düzen sağlamayı hedefler (6).



Şekil 10: Detay Popup

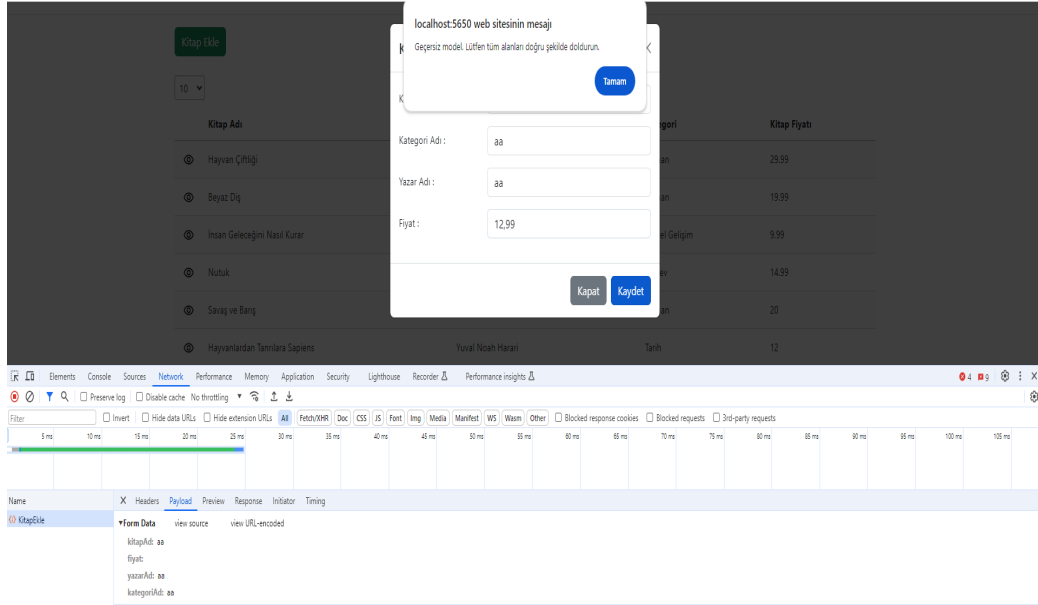
Bu Bootstrap modal örneği, kullanıcıların bir detayı incelemek veya düzenlemek için bir açılan pencere kullanmalarını sağlar. Bu modal, bir kitabın detaylarını görüntülemek için kullanılmıştır. Modelin sağladığı kolaylık sayesinde, kullanıcı işlem yaparken bu açılan pencere açıldığında yeni bir sayfaya geçiş yapmadan ve önceki sayfayı kapatmadan işlemini gerçekleştirebiliyor. Bu da kullanıcı için büyük bir kolaylık sağlar.

Kullanılan modelin özelliklerinden bazıları şunlardır: Modal, bir detay butonu öğesi tarafından tetiklenir. Bu, kullanıcıların sadece belirli bir detayı incelemek veya düzenlemek istediklerinde modala erişebilecekleri anlamına gelir.

Modal, belirlenen bir data-bs-backdrop özelliği sayesinde statik bir arka plana sahiptir, bu da kullanıcıların modal dışındaki alanlara tıklamalarının modali kapatmayacağı anlamına gelir.

5 Ekleme Olayında Alınan Hata

Önceki modelde kullanılan yapı, bu sefer açılan pencereden dört değer almak-tadır. Ancak, fiyat parametresinin değeri alınırken bir hata oluşmaktadır. Konsoldan yapılan kontroller sonucunda fonksiyonun doğru bir şekilde çalıştığı gözlemlenmiştir. Diğer parametreler sorunsuz bir şekilde alınırken, fiyat parametresinin değeri boş olarak gelmektedir. İlk olarak, hatanın giriş adları ile parametre adlarının uyumsuzluğundan kaynaklanabileceği düşünülmüştür. Ancak, gerekli kontroller yapılmış ve bu konuda bir hata bulunamamıştır. Veritabanında fiyat parametresinin tipi 'numeric' olarak tanımlanmıştır. Bu nedenle, JavaScript kodunda fiyat değişkeni üzerinde dönüşümler gerçekleştirilmiş, fakat bu da sorunu çözmemiştir. Backend kodundaki değişken isimlerinin uyumu kontrol edildiğinde, PostModel sınıfındaki değişken isimleri ile JavaScript kodundaki değişken isimlerinin aynı olduğu görülmüş ve bu durum da sorunu açıklamamıştır. Hatanın kaynağı tespit edilememiş olmasına rağmen, hatanın ortaya çıktığı noktanın backend kodunda uygulanan model doğrulama kod bloğunda meydana geldiği belirlenmiştir. Çünkü, ön tarafta kullanıcının girdiği değer, controller'daki post modele 0 olarak gelmektedir.



Şekil 11: Hata

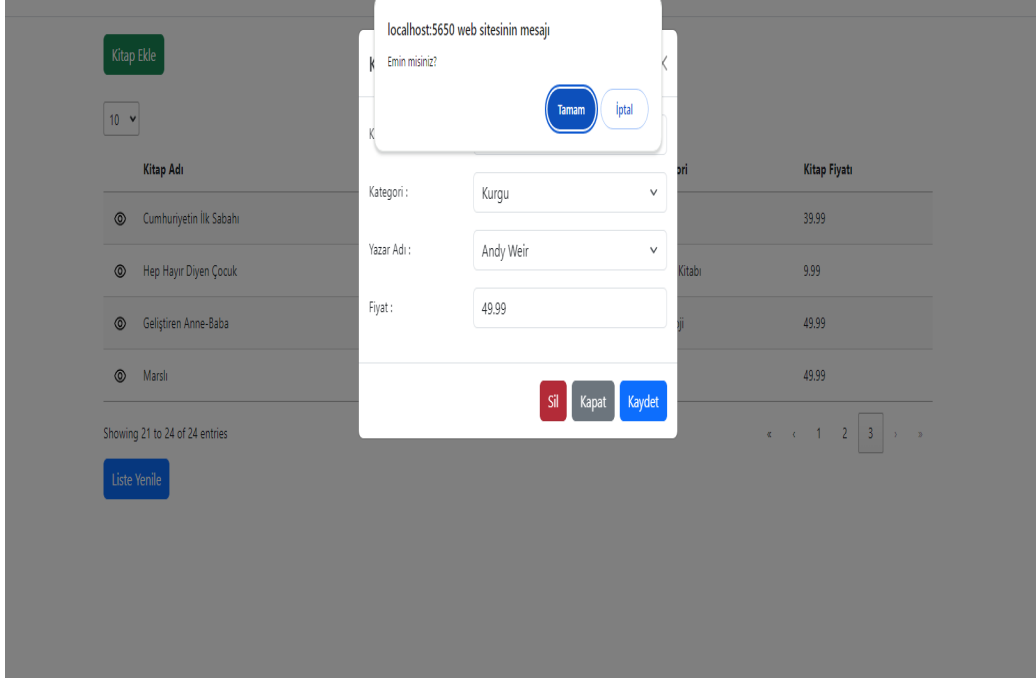
6 Ekleme İşleminin Gerçekleştirilmesi

Geçtiğimiz aylarda, Türkiye Cumhuriyeti hükümeti tarafından alınan karar doğrultusunda, uluslararası platformlarda "Turkey" yerine "Türkiye" isminin kullanılması kararlaştırılmıştır. Bu kararın ardından, teknoloji şirketleri de yazılımlarında bu değişikliği yansıtmak üzere güncellemeler yapmaya başlamıştır. Özellikle Microsoft, Windows işletim sisteminde bu değişikliği uygulamış ve "Turkey" ifadelerini "Türkiye" olarak güncelleştir. Bu kapsamda, 22H2 sürüm numarasına sahip Windows 11 güncellemesi (22621.2715), Türkiye'nin adının doğru bir şekilde yansıtılması amacıyla yayımlanmıştır. Ancak, bu güncelleme sonrası, bazı yazılım ve kodlarda "Turkey" ifadesine bağlı olarak çalışan sistemlerde hatalar meydana gelmiştir. Hatayı bulmama yardımcı olan adres (5).

Bu isim değişikliğinin ardından, yazılım projemde çeşitli hatalar meydana gelmiştir. Sorunun kaynağını araştırdığımda, PostgreSQL veritabanımda yer alan tablolarımın bazı parametrelerinin veri tiplerini yeniden tanımlamam gerektiğini fark ettim. Bu süreçte tüm verilerimi dışa aktarıp, PostgreSQL'i tamamen silip yeniden yükledim. Verilerimi tekrar yüklerken, bazı sütunların veri tiplerini "identity always" olarak tanımlamam gerektiğini unutmuşum. Bu eksiklik, yeni veri ekleme işlemleri sırasında her yeni kaydın 1'den başlamasına ve böylece aslında mevcut olan benzersiz değerlere çakışmasına neden olmuştur. Sonuç olarak, bu durum kodumda hatalara yol açmış ve projemin düzgün çalışmasını engellemiştir. Bu deneyim, veritabanı yapılandırmalarının ve veri tiplerinin doğru tanımlanmasının yazılım projelerinin sürdürülebilirliği açısından ne denli kritik olduğunu göstermiştir. Ekleme işlemini gerçekleştirirken (2) yararlanılmıştır.

7 Silme İşleminin Gerçekleştirilmesi

Kitap detay butonunda aktif hale gelen pencere içerisinde silme işlemi gerçekleştirildi. Bu işlem, kullanıcıların mevcut kitapları veritabanından silmelerine olanak tanır ve kullanıcı dostu bir arayüz sağlamak amacıyla dikkatlice ele alındı. Kullanıcı, kitap listesinde yer alan bir kitabın detaylarını görmek için ilgili kitap satırındaki detay butonuna tıklayarak kitap detay penceresini aktif hale getirir. Detay penceresinde bulunan sil butonuna tıklayarak kitap silme işlemi başlatılır ve onay verilirse arka planda silme işlemi başlar. Veritabanı katmanında, PostgreSQL’de gerekli fonksiyonlar yazıldı ve bu fonksiyonlar kitap id’sine göre kitabın güvenli bir şekilde silinmesini sağlar. İşlem başarılı bir şekilde tamamlandığında, kullanıcıya işlem sonucunu bildiren geri bildirim mesajı gösterilir ve bu mesaj, belirli bir süre ekranda kalıp otomatik olarak kaybolur, böylece arayüz daha temiz ve kullanıcı dostu hale gelir. Kullanıcının karşılaşabileceği hataları daha iyi anlaması için hata mesajları da detaylandırıldı ve çeşitli senaryolar altında test edilerek kullanıcı etkileşimlerinin düzgün çalıştığından ve geri bildirim mesajlarının doğru zamanda ve doğru şekilde gösterildiğinden emin olundu. Bu iyileştirmeler, kullanıcıların kitapları yönetirken daha sorunsuz bir deneyim yaşamalarını sağlar ve uygulamanın genel kullanıcı deneyimini iyileştirir.



Şekil 12: Silme İşlemi

Kaynakça

- [1] Murat Yücedağ, "Murat Yücedağ YouTube Kanalı", YouTube. <https://www.youtube.com/MurattYucedag>
- [2] ChatGPT, OpenAI. <https://openai.com/gpt>
- [3] The Bootstrap Authors. <https://getbootstrap.com/>
- [4] DataTables. <https://datatables.net/>
- [5] Halil Han Badem. *PostgreSQL'de Türkiye Sorunu ve Windows Güncellemesi*. 2024. <https://tr.linkedin.com/pulse/postgresqlde-t%C3%BCrkiye-sorunu-windows-g%C3%BCncellemesi-halil-han-badem-7kxhf>.
- [6] *Pop-up Nedir?*. 2024. <https://www.jivochat.com.tr/blog/araclar/popup-nedir.html>.