**1.GİRİŞ**

Anaokulu projesinde ilk önce bu program için gerekli araştırmalar yapıldı. Veriler toplandıktan sonra bu veriler incelenerek bilgi oluşturuldu. Bu bilgiler ışığında tablolar oluşturuldu ve ER diyagramı çizildi. En son olarak gerekli trigger ve prosedürler yazılarak veri tabanı işlemleri gerçekleştirildi. Local hosta oluşturulan bu veri tabanı Microsoft Azure’den kiralanan sunucuya atandı. Bu bölümde yapılan gerekli işlemler bölüm ikide gösterilmiştir.

Veri tabanı oluşturulduktan sonra ara yüz tasarımı için Java Script dili kullanıldı. Ara yüzü tasarlarken Atom uygulaması kullanıldı. Ara yüz tasarımı oluşturulduktan sonra sunucuya SQL kurulumundan sonra Node. js kurularak ara yüz kodları sunucuya atıldı. Bu bölümde yapılan gerekli işlemler bölüm üçte gösterilmiştir.

Anaokulu projesinde yazılan bütün kodlar ekler bölümüne eklenmiştir.

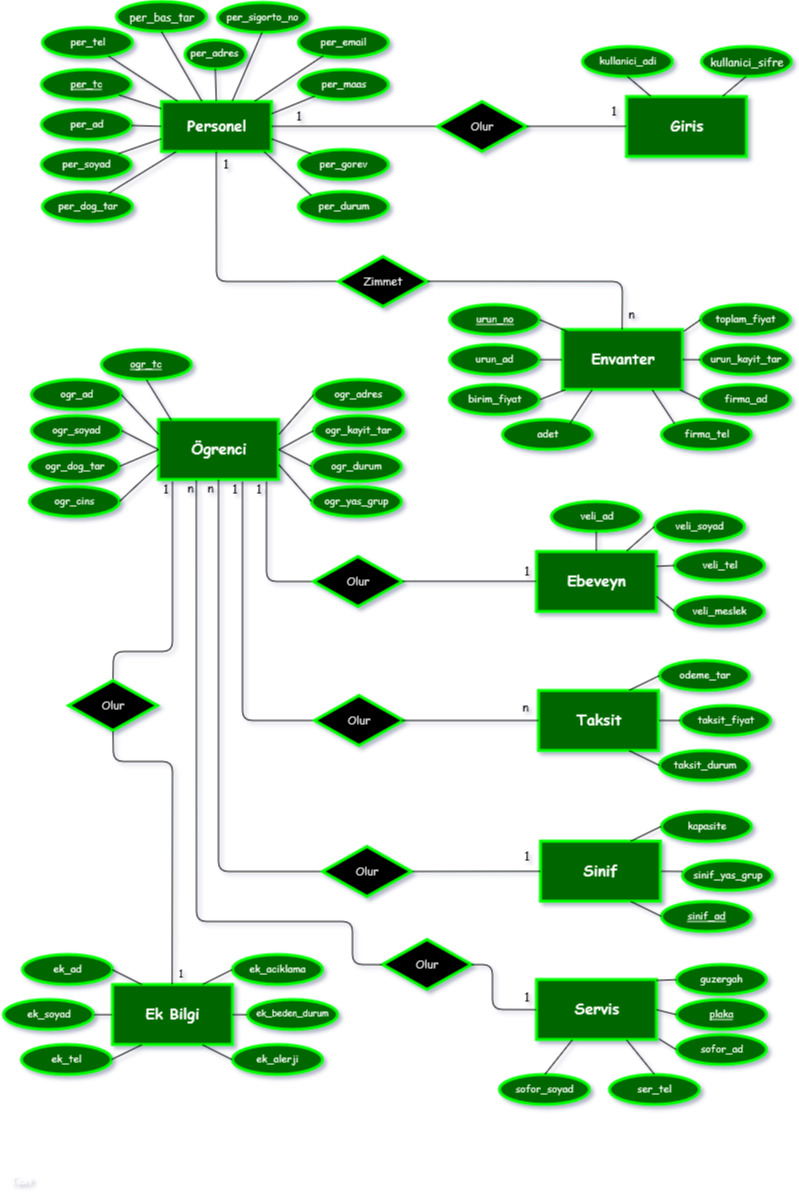
**2.ANAOKULU VERİTABANI**

Anaokulu veri tabanını oluşturmadan önce gerekli araştırma yapıldı. Anaokulu yazılımlarında kullanıcılar hangi verileri tutmak istiyor, ne şekilde ara yüzler tercih ediliyor gibi konular hakkında veriler toplandı. Daha sonra bu veriler analiz edilerek ER diyagramı Şekil 2.1.gösterildiği gibi çizildi. ER diyagramı oluşturulduktan sonra SQL Server 2014 Managment Studio programı yardımıyla aşağıda ismi verilen tablolar oluşturuldu.

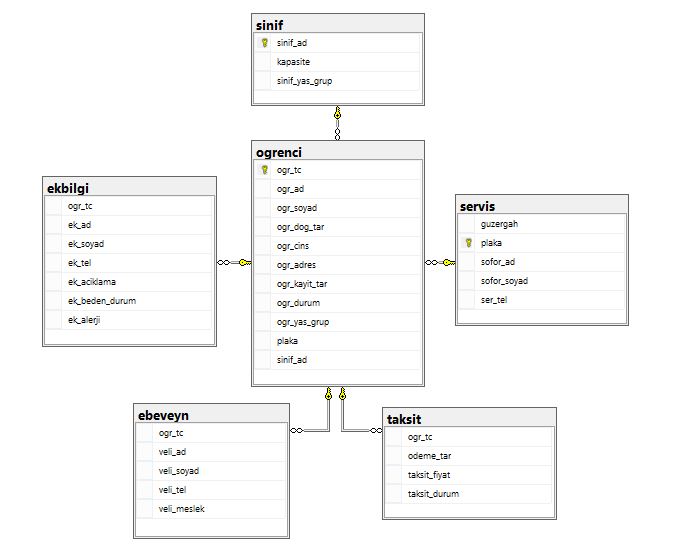
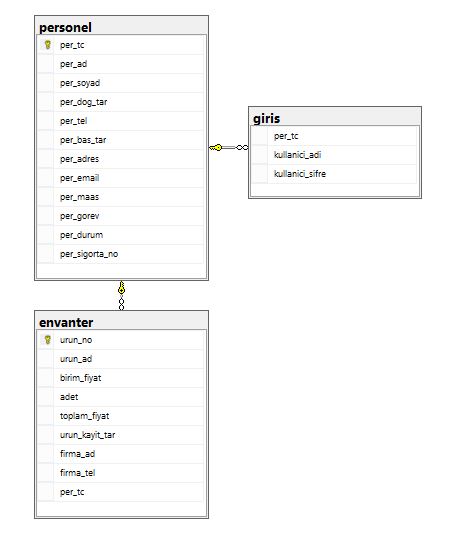
* Ebeveyn
* Ek Bilgi
* Envanter
* Giris
* Ogrenci
* Personel
* Servis
* Sinif
* Taksit
* Silinenler Tabloları

SQL Server 2014 Managment Studio programını açtıktan sonra New Query penceresine tıkladıktan sonra ilk önce ‘ CREATE DATABASE anaokulu’ komutu ile veri tabanı oluşturuldu. İkinci adım olarak ‘USE anaokulu ’komutu ile bu veri tabanını kullanıma açtık. Daha sonra tabloları ‘CREATE TABLE … ’ komutuyla tabloları oluşturmaya başladık. Tablo oluşturulurken gerekli kontroller constraint komutu ile gerçekleştirildi. En son olarak trigger ve procedurler yazıldı.

Ana tablolar oluşturulduktan sonra silme tabloları da oluşturuldu. Bu tablolara kullanıcı direk erişemediği için herhangi bir kontrol yazılmasına gerek duyulmadı. Silinen tablolarına trigger yardımı ile veriler silindiğinde direk olarak atanır. Bu kodların ayrıntılı kısmı ekler bölümünde gösterilmiştir.



**Şekil 2.1** Anaokulu ER Diyagramı

****

**Şekil 2.2** VeritabanıŞeması

**2.1.VERİ TABANI TABLOLARI**

**2.1.1.Ebeveyn**

Ebeveyn tablosunda 5 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; ogr\_tc, veli\_ad, veli\_soyad, veli\_tel, veli\_meslek dir. Bu alan adlarının birincisi ogr\_tc, türü varchar olarak tanımlanmıştır. Bu alan adı froeing key olarak tanımlandıktan sonra ogrenci tablosuna referans edilmiştir ve oradaki ogr\_tc’ler kullanılmıştır.

Ebeveyn tablosunda ikinci alan veli\_ad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Ebeveyn tablosunda üçüncü alan veli\_soyad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Ebeveyn tablosunda dördüncü alan veli\_tel dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına bir constraint yazılarak telefon numarası girişi ve en son olarak aynı telefon girişini engellemek için UNIQUE yazılarak gerekli kontroller yapılmıştır.

Ebeveyn tablosunda beşinci alan veli\_meslek dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir.

**2.1.2.Ek Bilgi**

Ekbilgi tablosunda 7 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; ogr\_tc, ek\_ad, ek\_soyad, ek\_tel, ek\_acıklama, ek\_beden\_durum, ek\_alerji dir. Bu alan adlarının birincisi ogr\_tc, türü varchar olarak tanımlanmıştır. Bu alan adı froeing key olarak tanımlandıktan sonra ogrenci tablosuna referans edilmiştir ve oradaki ogr\_tc’ler kullanılmıştır.

Ekbilgi tablosunda ikinci alan ek\_ad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Ekbilgi tablosunda üçüncü alan ek\_soyad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Ekbilgi tablosunda dördüncü alan ek\_tel dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına bir constraint yazılarak telefon numarası girişi ve en son olarak aynı telefon girişini engellemek için UNIQUE yazılarak gerekli kontroller yapılmıştır.

Ekbilgi tablosunda ikinci alan ek\_aciklama dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 50 karakter olarak seçilmiştir.

Ekbilgi tablosunda ikinci alan ek\_beden\_durum dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 50 karakter olarak seçilmiştir.

Ekbilgi tablosunda ikinci alan ek\_alerji dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 50 karakter olarak seçilmiştir.

**2.1.3.Envanter**

Envanter tablosunda 9 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; urun\_no, urun\_ad, birim\_fiyat, adet, toplam\_fiyat, urun\_kayit\_tar, firma\_ad, firma\_tel, per\_tc dir. Bu alan adlarının birincisi urun\_no, türü int olarak tanımlanmıştır. Bu alan adı primary key olarak tanımlandıktan sonra identity ile birden başlayıp birer birer artarak urun no oluşturulmuştur.

Envanter tablosunda ikinci alan urun\_ad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Envanter tablosunda üçüncü alan birim\_fiyat dır. Bu alan adının veri türü money olarak tanımlandı. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Envanter tablosunda dördüncü alan adet dır. Bu alan adının veri türü int olarak tanımlandı. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Envanter tablosunda beşinci alan toplam\_fiyat dır. Bu alan adına kullanıcı veri girişi yapmayarak arka planda ‘toplam\_fiyat as (birim\_fiyat\*adet)’ koduyla birim fiyat ve adet çarpma işleminin sonucu buraya yazıldı.

Envanter tablosunda altıncı alan urun\_kayıt\_tar dır. Bu alan adının veri türü date olarak tanımlandı. Daha sonra bu alan adına bir constraint yazılarak en fazla sistem tarihinin girişine izin verilmesini sağlar.

Envanter tablosunda yedinci alan firma\_ad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 20 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Envanter tablosunda sekizinci alan firma\_tel dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına bir constraint yazılarak telefon numarası girişi ve en son olarak aynı telefon girişini engellemek için UNIQUE yazılarak gerekli kontroller yapılmıştır.

Envanter tablosunda son alan adı per\_tc, türü varchar olarak tanımlanmıştır. Bu alan adı froeing key olarak tanımlandıktan sonra personel tablosuna referans edilmiştir ve oradaki per\_tc’ler kullanılmıştır. Bu alan adı ürünlerin zimmetlenmesi için kullanılmıştır.

**2.1.4.Giris**

Giris tablosunda 3 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; per\_tc, kullanici\_adi, kullanici\_sifre dir. Bu alan adlarının birincisi per\_tc, türü varchar olarak tanımlanmıştır. Bu alan adı froeing key olarak tanımlandıktan sonra personel tablosuna referans edilmiştir ve oradaki per\_tc’ler kullanılmıştır.

Giris tablosunda ikinci alan kullanıcı\_adi dır. Bu alan adının veri türü varchar olarak tanımlandı ve aynı kullanıcı adı girişini engellemek için UNIQUE yazılarak gerekli kontroller yapılmıştır. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Giris tablosunda üçüncü alan kullanıcı\_sifre dır. Bu alan adının veri türü varchar olarak tanımlandı. Bu alan adına şifreler ara yüzde hash olarak çevirme işlemi yapıldıktan sonra kaydedilir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.0

**2.1.5.Ogrenci**

Ogrenci tablosunda 11 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; ogr\_tc,ogr\_ad,ogr\_soyad,ogr\_dog\_tar,ogr\_cins,ogr\_adres,ogr\_kayıt\_tar,ogr\_durum,ogr\_yas\_grup,plaka,sinif\_ad dır. Bu alan adlarının birincisi ogr\_tc, türü varchar olarak tanımlanmıştır. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alana constraint yazılarak öğrenci T.C.’nin 11 haneli olması kontrol edilmiştir.

Ogrenci tablosunda ikinci alan ogr\_ad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Ogrenci tablosunda üçüncü alan ogr\_soyad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 20 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Ogrenci tablosunda dördüncü alan ogr\_dog\_tar dır. Bu alan adının veri türü date olarak tanımlandı. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Anaokulunda kullanılacak bu program en az 2, en fazla 6 yaşındaki çocukların kaydını aldığı için gerekli constraint yazıldı.

Ogrenci tablosunda beşinci alan ogr\_cins dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 1 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Kaydedilen öğrencilerin erkek olanlara ‘E’ kadın olanlara ‘K’ harfi kaydının yapılması için gerekli constraint yazıldı.

Ogrenci tablosunda altıncı alan ogr\_adres dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 200 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Ogrenci tablosunda yedinci alan ogr\_kayit\_tar dır. Bu alan adının veri türü date olarak tanımlandı. Bu alan adı boş geçilmesi halinde sistem tarihini atayan constraint yazıldı.

Ogrenci tablosunda sekizinci alan ogr\_durum ’dur. Bu alan adının veri türü varchar olarak tanımlandı. Daha sonra bu alan için contraint yazılarak gerekli kontroller yapıldı.

Ogrenci tablosunda dokuzuncu alan ogr\_yas\_grup dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 3 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan için constraint yazılarak yanlış yaş grubu girilmesi engellendi.

Ogrenci tablosunda onuncu alan plaka dır. Bu alan adının veri türü varchar olarak tanımlandı. Bu alan adı froeing key olarak tanımlandıktan sonra servis tablosuna referans edilmiştir ve oradaki plakalar kullanılmıştır.

Ogrenci tablosunda on birinci alan sinif\_ad dır. Bu alan adının veri türü varchar olarak tanımlandı. Bu alan adı froeing key olarak tanımlandıktan sonra sinif tablosuna referans edilmiştir ve oradaki sinif\_ad kullanılmıştır.

**2.1.6.Personel**

Personel tablosunda 12 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; per\_tc,per\_ad,per\_soyad,per\_dog\_tar,per\_tel,per\_bas\_tar,per\_adres,per\_email,per\_maas,per\_gorev,per\_durum,per\_sigorta\_no dır. Bu alan adlarının birincisi per\_tc, türü varchar olarak tanımlanmıştır. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alana constraint yazılarak personel T.C.’nin 11 haneli olması kontrol edilmiştir.

Personel tablosunda ikinci alan per\_ad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Personel tablosunda üçüncü alan per\_soyad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Personel tablosunda dördüncü alan per\_dog\_tar dır. Bu alan adının veri türü date olarak tanımlandı. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Personel kaydı yapılırken doğum tarihinin sistem saatinden büyük olmasını engellemek için constraint yazılmıştır.

Personel tablosunda beşinci alan per\_tel dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına bir constraint yazılarak telefon numarası girişi ve en son olarak aynı telefon girişini engellemek için UNIQUE yazılarak gerekli kontroller yapılmıştır.

Personel tablosunda altıncı alan personel\_bas\_tar dır. Bu alan adının veri türü date olarak tanımlandı. Bu alan adının boş geçilmesi halinde sistem tarihini kaydeden contraint yazıldı.

Personel tablosunda yedinci alan per\_adres dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 200 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Personel tablosunda sekizinci alan per\_email dır. Bu alan adının veri türü varchar olarak tanımlandı. Personel email girişi yapılırken gerekli @işareti, .com veya .org uzantılarının kontrolü için bir contraint yazıldı.

Personel tablosunda dokuzuncu alan per\_maas dır. Bu alan adının veri türü money olarak tanımlandı. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Personel tablosunda onuncu alan per\_gorev dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra kurumdaki çalışanların görevleri yanlış girilmemesi için gerekli contraint yazıldı.

Personel tablosunda on birinci alan per\_durum ’dur. Bu alan adının veri türü varchar olarak tanımlandı. Daha sonra bu alan için contraint yazılarak gerekli kontroller yapıldı.

Personel tablosunda on ikinci alan per\_sigorta\_no ’dur. Bu alan adının veri türü varchar olarak tanımlandı. Daha sonra bu alan için geçerli sigorta numarasının girilmesi için constraint yazıldı.

**2.1.7.Servis**

Servis tablosunda 5 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; guzergah, plaka, sofor\_ad, sofor\_soyad, ser\_tel dir. Bu alan adlarının birincisi güzergah, türü varchar olarak tanımlanmıştır. . Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Servis tablosunda ikinci alan plaka dır. Bu alan adının veri türü varchar ve primary key olarak tanımlandıktan sonra uzunluğu 9 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına constraint yazılarak plakanın doğru bir şekilde girilmesi sağlanmıştır.

Servis tablosunda üçüncü alan sofor\_ad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Servis tablosunda dördüncü alan sofor\_soyad dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Servis tablosunda beşinci alan ser\_tel dır. Bu alan adının veri türü varchar olarak tanımlandıktan sonra uzunluğu 10 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına bir constraint yazılarak telefon numarası girişi ve en son olarak aynı telefon girişini engellemek için UNIQUE yazılarak gerekli kontroller yapılmıştır.

**2.1.8.Sinif**

Sinif tablosunda 3 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; sinif\_ad, kapasite, sinif\_yas\_grup dir. Bu alan adlarının birincisi sinif\_ad, türü varchar ve primary key olarak tanımlandıktan sonra uzunluğu 15 karakter olarak seçilmiştir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Sinif tablosunda ikinci kapasite dır. Bu alan adının veri türü smallint dir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Sinif tablosunda üçüncü sinif\_yas\_grup dır. Bu alan adının veri türü varchar dir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına constraint yazılarak doğru yaş gruplarının girilmesi sağlanmıştır.

**2.1.9.Taksit**

Taksit tablosunda 4 tane alan adı oluşturuldu. Bu alan adların isimleri şu şekildedir; ogr\_tc, odeme\_tar, taksit\_fiyat, taksit\_durum dir. Bu alan adlarının birincisi ogr\_tc, türü varchar olarak tanımlanmıştır. . Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. . Bu alan adı froeing key olarak tanımlandıktan sonra ogrenci tablosuna referans edilmiştir ve oradaki ogr\_tc’ler kullanılmıştır.

Taksit tablosunda ikinci odeme\_tar dır. Bu alan adının veri türü date dir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Taksit tablosunda üçüncüsü fiyat dır. Bu alan adının veri türü int dir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir.

Taksit tablosunda dördüncüsü taksit\_durum dır. Bu alan adının veri türü varchar dir. Bu alan adı boş geçilmemesi için NOT NULL yazılarak bu işlem gerçekleşmiştir. Daha sonra bu alan adına constraint yazılarak iki farklı seçeneğin girilmesine olanak sağlamıştır.

**2.1.10.Silinenler Tabloları**

Bu tablolara ek olarak silinen \_envanter, silinen\_ögrenci, silinen\_personel adında üç tane tablo oluşturulmuştur. Bu tabloları oluştururken yukarıdaki envanter, öğrenci ve personel tablolarının veri tipleri kullanılmıştır. Ancak bu tablolara kullanıcı direk veri girmediği için veri kontrolü yapılmasına gerek duyulmadı.

**2.2.VERİ TABANI PROSEDÜRLER**

**2.2.1.sp\_anasayfa**

Sp\_anasayfa prosedürü öğrenci kayıt işlemi yapılırken gerekli olan sınıf adları, personel adları ve servislerin getirilmesi için yazılmış bir prosedürdür. Bu prosedür de veriler json formuna dönüştürerek gönderiyoruz. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.2.sp\_ogrenci\_kayit**

Sp\_ogrenci\_kayit prosedürü öğrenci kayıt işlemi yapılırken kullanılıyor. İlk önce öğrenci kayıt ederken ara yüzde alacağımız bilgiler için değişkenler tanımlandı. Kayıt işlemi başlamadan önce öğrenci TC si personel TC si ile kontrol edilip aynı olması engelleniyor. Bu kontrolü öğrenci tablosunda yapmamıza gerek yok çünkü ogr\_tc primary keydir. Eğer if döngüsüne girer ise insert işlemi gerçekleşir. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.3.sp\_personel\_kayit**

Sp\_personel\_kayit prosedürü personel kayıt işlemi yapılırken kullanılıyor. İlk önce personel kayıt ederken ara yüzde alacağımız bilgiler için değişkenler tanımlandı. Kayıt işlemi başlamadan önce personel TC si öğrenci TC si ile kontrol edilip aynı olması engelleniyor. Bu kontrolü personel tablosunda yapmamıza gerek yok çünkü per\_tc primary keydir. Eğer if döngüsüne girer ise insert işlemi gerçekleşir. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.4.sp\_gecmis\_taksit**

Sp\_gecmis\_taksit prosedürü ara yüzde bütçe kısmında kullanılıyor. Adından da anlaşılacağı gibi bir öğrencinin ödenmiş taksitlerini gösteriyoruz. Bu prosedürü yazarken o yıl ödenmiş ve geçen yıllar ki ödenmiş taksitlerin kontrolü yapılarak veriler getiriliyor. Veriler hazırlandıktan sonra en son order by ile öğrenci adlarına göre sıralıyoruz. Daha sonra bu veriler json formatına dönüştürülerek ara yüze gönderiyoruz. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.5.sp\_gelecek\_taksit**

Sp\_gelecek\_taksit prosedürü ara yüzde bütçe kısmında kullanılıyor. Adından da anlaşılacağı gibi bir öğrencinin gelecek taksitlerini gösteriyoruz. Bu prosedürü yazarken o yıl ki taksitlerini ve gelecek yıldaki taksitlerini gösteriyoruz. Veriler hazırlandıktan sonra en son order by ile öğrenci adlarına göre sıralıyoruz. Daha sonra bu veriler json formatına dönüştürülerek ara yüze gönderiyoruz. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.6.sp\_odenmemis\_taksit**

Sp\_odenmemis\_taksit prosedürü ara yüzde bütçe kısmında kullanılıyor. Adından da anlaşılacağı gibi bir öğrencinin gelecek taksitlerini gösteriyoruz. Bu prosedürü yazarken o yıl ki ödenmemiş taksitlerini gösteriyoruz. Veriler hazırlandıktan sonra en son order by ile öğrenci adlarına göre sıralıyoruz. Daha sonra bu veriler json formatına dönüştürülerek ara yüze gönderiyoruz. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.7.sp\_odenmis\_taksit**

Sp\_odenmis\_taksit prosedürü ara yüzde bütçe kısmında kullanılıyor. Adından da anlaşılacağı gibi bir öğrencinin ödenmiş taksitlerini gösteriyoruz. Bu prosedürü yazarken o yıl ki ödenmiş taksitlerini gösteriyoruz. Veriler hazırlandıktan sonra en son order by ile öğrenci adlarına göre sıralıyoruz. Daha sonra bu veriler json formatına dönüştürülerek ara yüze gönderiyoruz. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.8.sp\_taksit\_toplam**

Sp\_taksit \_toplam prosedürü ara yüzde bütçe kısmında kullanılıyor. Bu prosedür veriler grafik çiziminde kullanılmıştır. Öğrencilerin sitem ayındaki taksitlerini topladı. Veriler hazırlandıktan sonra en son order by ile aylara göre sıralıyoruz. Daha sonra bu veriler json formatına dönüştürülerek ara yüze gönderiyoruz. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.2.9.sp\_yedekten\_geri\_yukle**

Sp\_yedekten\_geri\_yukle prosedürü ara yüzde ayarlar kısmında kullanılmıştır. Daha önceden yapılan veri tabanı yedekleme işlemini geri yükler. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.3.VERİ TABANI TRİGGER’LARI**

**2.3.1.trg\_silinen\_ogrenci**

Trg\_silinen\_ogrenci trigere öğrenci tablosuna güncelleme işlemi yapıldığında devreye girer. Trigger devre girdikten sonra ilk önce silinecek öğrencinin bilgileri silinen öğrenci tablosuna insert işlemi gerçekleşir. Daha sonra öğrencinin bilgileri sınıftan silinir. Bu işlemlerde öğrenciler veri tabanında aktif pasif şekilde tutulur. . Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.3.2.trg\_personel**

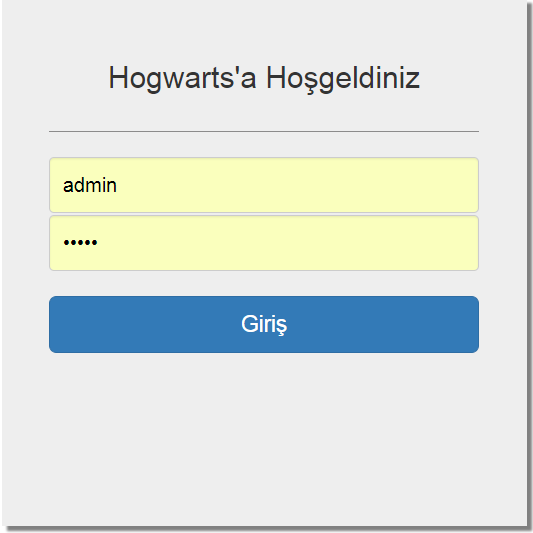
Trg\_personel trigere personel tablosuna güncelleme işlemi yapıldığında devreye girer. Trigger devre girdikten sonra ilk önce silinecek personelin bilgileri silinen personel tablosuna insert işlemi gerçekleşir. Daha sonra personelin bilgileri silinir. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

**2.3.3.trg\_silinen\_envanter**

Trg\_silinen\_envanter trigere envanter tablosuna delete işlemi yapıldığında devreye girer. Bu triggerda instead of kullanılarak delete işlemi yapılmadan önce veriler silinen tablosuna kaydedilir. Bu işlem sanal tabloda tutulan veriler sayesinde yapılır. Daha sonra envanterin bilgileri silinir. Bu kodların ayrıntılı şekli ekler bölümünde gösterilmiştir.

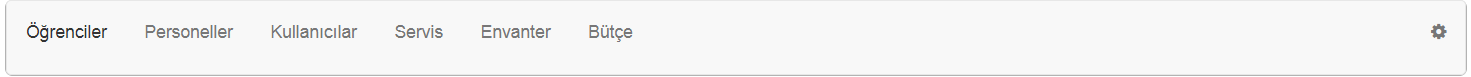
**3.ANAOKULU ARAYÜZ TASARIMI**

Ara yüzü tanıtacak olursak uygulamamız 52.164.200.179:3000 adresinde bulunur. Adrese ilk girişte karşımıza kullanıcı giriş menüsü geliyor. Şekil 3.1 de gösterilmiştir Kullanıcı adı admin olan personel ile giriş yapalım.



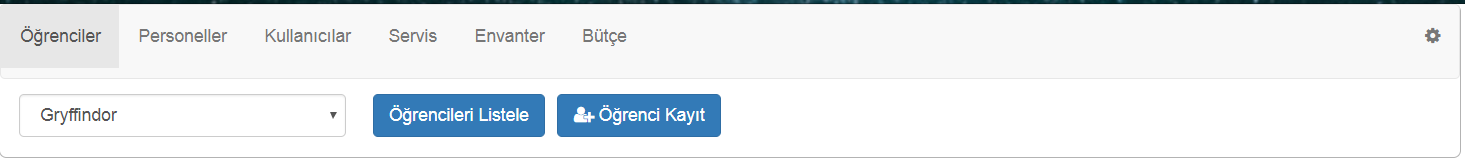
**Şekil 3.1.** Kullanıcı Giriş Menüsü

Açılan sayfada “Öğrenciler”, ”Personeller”, ”Kullanıcılar”, ”Servis”, ”Envanter”, ”Bütçe” ve “Ayarlar” menüleri bulunmaktadır. Bu kısım Şekil 3.2 gösrilmiştir.



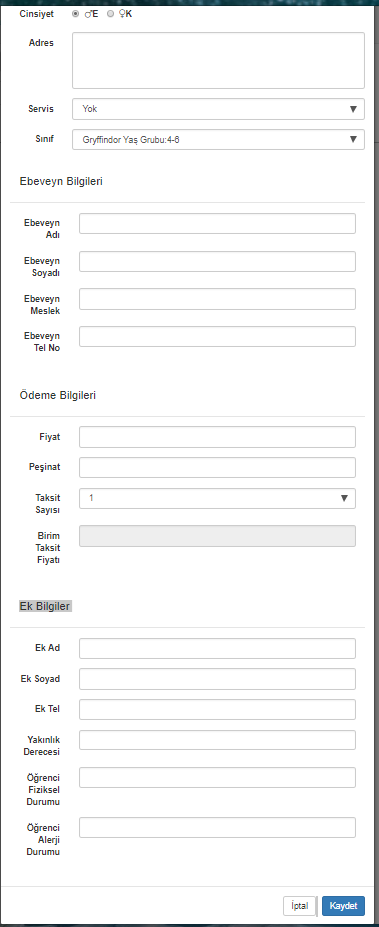
**Şekil 3.2.** Arayüz Menüleri

Öğrenciler menüsüne tıkladığımızda “Öğrencileri Listele” butonu, “Öğrenci Kayıt” butonu ve sınıf isimlerinin bulunduğu drop-dawn list bulunur.



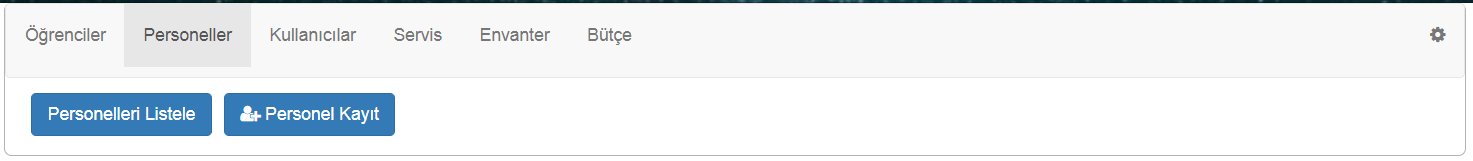
**Şekil 3.3.**Öğrenciler Menüsüne tıklandığında açılan sayfa

Listede bulunana sınıflardan biri seçildikten sonra “Öğrencileri Listele” butonuna tıklarsak seçilen sınıftaki öğrencileri tablo biçiminde listeleyebiliyoruz. Herhangi bir sınıfı listeledikten sonra gelen tabloda öğrencinin “T.C”, ”Ad” ve ”Soyad” bilgilerine ulaşabiliriz. Eğer istersek bu tablonun altında bulunan “Export” listesinden tabloyu .pdf veya .json uzantısıyla bilgisayarımıza indirebiliyoruz.Eğer bu tabloda bulunan bir öğrencinin bilgilerinin bulunduğu satıra tıklarsak “Öğrenci Detayı” bölümü açılır ve bu bölümde seçilen öğrencinin detaylı bilgileri bulunur. Eğer tabloda öğrencinin bilgilerinin bulunduğu satıra çift tıklarsak ekrana o öğrenciyi silebileceğimiz bir “Sil” butonu gelir ve bu butonla seçilen öğrenciyi tablodan silebiliriz. “Öğrenci Kayıt” butonuna bastığımızda ise yeni öğrenci ekleyebileceğimiz bir menü açılıyor. Açılan menüye yeni kayıt edeceğimiz öğrencinin bilgilerini girebileceğimiz kısımları doldurup yeni öğrenci kayıt edebiliyoruz. Bu bilgiler “Öğrenci Bilgileri” , “Ebeveyn Bilgileri ” , “Ödeme Bilgileri” ve “Ek Bilgiler” olarak dört kısımdan oluşur. “Öğrenci Bilgileri” kısmında öğrencinin “T.C. No” , ”Ad” , ”Soyad” , ”Doğum Tarihi” , ”Cinsiyet” , ”Adres“ , “Servis” ve ”Sınıf” bilgileri istenmektedir. “Ebeveyn Bilgileri” kısmında “Ebeveyn Adı” , ”Ebeveyn Soyadı” , “Ebeveyn Meslek” ve “Ebeveyn Tel No” bilgileri istenmektedir. “Ödeme Bilgileri” kısmında “Fiyat” , “Peşinat” , “Taksit Sayısı” ve “Birim Taksit Fiyatı” bilgileri istenmektedir. “Ek Bilgiler” kısmında ise “Ek Ad” , “Ek Soyad” , “Ek Tel” , “Yakınlık Derecesi” , “Öğrenci Fiziksel Durumu” ve “Öğrenci Alerji Durumu” bilgileri istenmektedir. Bu bilgiler girildikten sonra “Kaydet” veya “İptal” butonlarıyla kayıt işlemini tamamlayabiliriz veya da iptal edebiliriz.



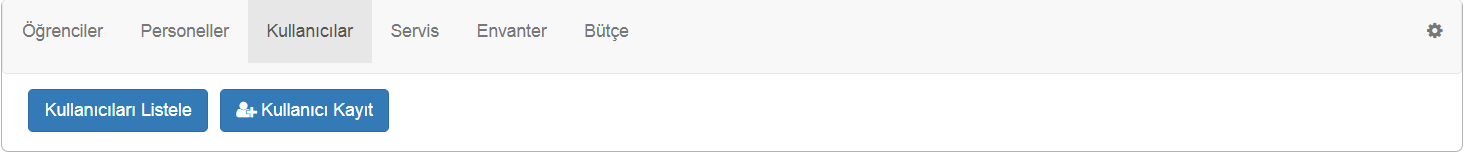
**Şekil 3.4.** Öğrenci Kayıt Menüsü

Personeller menüsüne tıkladığımızda “Personelleri Listele” ve “Personel Kayıt” butonları bulunur. “Personelleri Listele” butonuna tıklarsak sistemde kayıtlı olan personelleri tablo biçiminde listeleyebiliyoruz. Personeller listeledikten sonra gelen tabloda bir personelin “T.C”, ”Ad” , ”Soyad” , “Doğum Tarihi” , “Telefon” , “Maaş” , “Görev” , “Eposta” ve “Sigorta No.” bilgilerine ulaşabiliriz. Eğer istersek bu tablonun altında bulunan “Export” listesinden tabloyu .pdf veya .json uzantısıyla bilgisayarımıza indirebiliyoruz. Eğer bu tabloda bulunan bir personelin bilgilerinin bulunduğu satıra tıklarsak ekrana o personeli silebileceğimiz bir “Sil” butonu gelir ve bu butonla seçilen personeli tablodan silebiliriz. “Personel Kayıt” butonuna bastığımızda ise yeni personel ekleyebileceğimiz bir menü açılıyor. Açılan menüye yeni kayıt edeceğimiz personelin bilgilerini girebileceğimiz kısımları doldurup yeni personel kayıt edebiliyoruz. Bu bilgiler “Personel Bilgileri” olarak bir kısımdan oluşur. “Personel Bilgileri” kısmında personelin “T.C. No” , ”Ad” , ”Soyad” , ”Doğum Tarihi” , ”Telefon” , ”Maaş“ , “Görev” , “Eposta” ve ”Sigorta No.” bilgileri istenmektedir. Bu bilgiler girildikten sonra “Kaydet” veya “İptal” butonlarıyla kayıt işlemini tamamlayabiliriz veya da iptal edebiliriz.



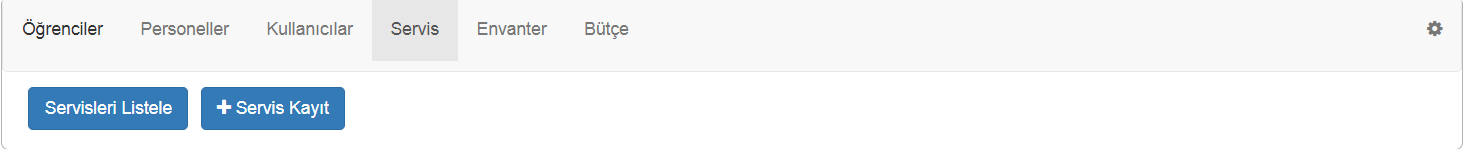
**3.5.** Personel menüsü

Kullanıcılar menüsüne tıkladığımızda “Kullanıcıları Listele” ve “Kullanıcı Kayıt” butonları bulunur. “Kullanıcıları Listele” butonuna tıklarsak sistemde kayıtlı olan personellerin kullanıcı bilgilerini tablo biçiminde listeleyebiliyoruz. Personellerin kullanıcı bilgileri listeledikten sonra gelen tabloda bir personelin ”Ad-Soyad” , “Kullanıcı Adı” ve “Hash’lenmiş şifre” bilgilerine ulaşabiliriz. Eğer bu tabloda bulunan bir personelin bilgilerinin bulunduğu satıra tıklarsak ekrana o personeli ve kullanıcı bilgilerini silebileceğimiz bir “Sil” butonu gelir ve bu butonla seçilen personeli ve kullanıcı bilgilerini tablodan silebiliriz. “Kullanıcı Kayıt” butonuna bastığımızda ise yeni personel kullanıcı bilgileri ekleyebileceğimiz bir menü açılıyor. Açılan menüye yeni kayıt edeceğimiz personelin bilgilerini girebileceğimiz kısımları doldurup yeni personel kullanıcı bilgileri kayıt edebiliyoruz. Bu bilgiler “Kullanıcı Bilgileri” olarak bir kısımdan oluşur. “Kullanıcı Bilgileri” kısmında personelin “Kullanıcı Adı” , ”Şifre” ve ”Personel” bilgileri istenmektedir. Bu bilgiler girildikten sonra “Kaydet” veya “İptal” butonlarıyla kayıt işlemini tamamlayabiliriz veya da iptal edebiliriz.



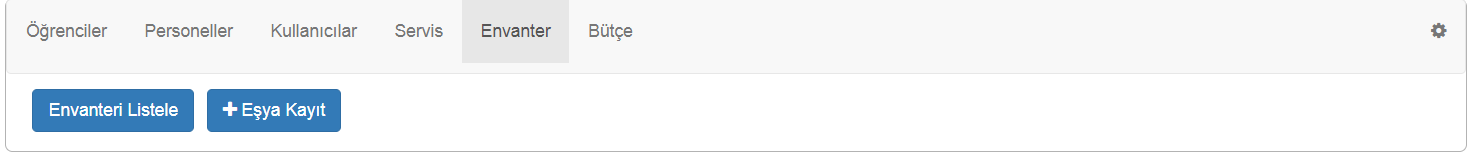
**3.6.** Kullanıcı Menüsü

Servis menüsüne tıkladığımızda “Servisleri Listele” ve “Servis Kayıt” butonları bulunur. “Servisleri Listele” butonuna tıklarsak sistemde kayıtlı olan servislerin bilgilerini tablo biçiminde listeleyebiliyoruz. Servisler listeledikten sonra gelen tabloda bir servisin ”Şöför Ad” , “Şöför Soyadı” , “Şöför Telefonu” , “Güzergah” ve “Plaka” bilgilerine ulaşabiliriz. Eğer bu tabloda bulunan bir servis bilgilerinin bulunduğu satıra tıklarsak ekrana o servis bilgilerini silebileceğimiz bir “Sil” butonu gelir ve bu butonla seçilen servis bilgilerini tablodan silebiliriz. “Servis Kayıt” butonuna bastığımızda ise yeni servis bilgileri ekleyebileceğimiz bir menü açılıyor. Açılan menüye yeni kayıt edeceğimiz servis bilgilerini girebileceğimiz kısımları doldurup yeni servis bilgileri kayıt edebiliyoruz. Bu bilgiler “Servis Bilgileri” olarak bir kısımdan oluşur. “Servis Bilgileri” kısmında servisin “Şöför Adı” , ” Şöför Soyadı” , “Şöför Telefonu” , “Güzergah” ve ”Plaka” bilgileri istenmektedir. Bu bilgiler girildikten sonra “Kaydet” veya “İptal” butonlarıyla kayıt işlemini tamamlayabiliriz veya da iptal edebiliriz.



**3.7.** Servis Menüsü

Envanter menüsüne tıkladığımızda “Envanterleri Listele” ve “Eşya Kayıt” butonları bulunur. “Envanterleri Listele” butonuna tıklarsak sistemde kayıtlı olan envanter bilgilerini tablo biçiminde listeleyebiliyoruz. Envanterleri listeledikten sonra gelen tabloda bir eşyanın ”Ürün No.”, “Ürün Adı”, “Birim Fiyat”, “Adet”, “Toplam Fiyat”, “Ürün Kayıt Tarihi”, “Firma Ad”, “Firma Tel” ve “Personel” bilgilerine ulaşabiliriz. Eğer bu tabloda bulunan bir eşya bilgilerinin bulunduğu satıra tıklarsak ekrana o eşya bilgilerini silebileceğimiz bir “Sil” butonu gelir ve bu butonla seçilen eşya bilgilerini tablodan silebiliriz. “Eşya Kayıt” butonuna bastığımızda ise yeni eşya bilgileri ekleyebileceğimiz bir menü açılıyor. Açılan menüye yeni kayıt edeceğimiz eşya bilgilerini girebileceğimiz kısımları doldurup yeni eşya bilgileri kayıt edebiliyoruz. Bu bilgiler “Eşya Bilgileri” olarak bir kısımdan oluşur. “Eşya Bilgileri” kısmında servisin “Eşya Adı” , ” Birim Fiyat” , “Adet” , “Firma Adı”, “Firma Telefonu” ve ”Personel” bilgileri istenmektedir. Bu bilgiler girildikten sonra “Kaydet” veya “İptal” butonlarıyla kayıt işlemini tamamlayabiliriz veya da iptal edebiliriz.



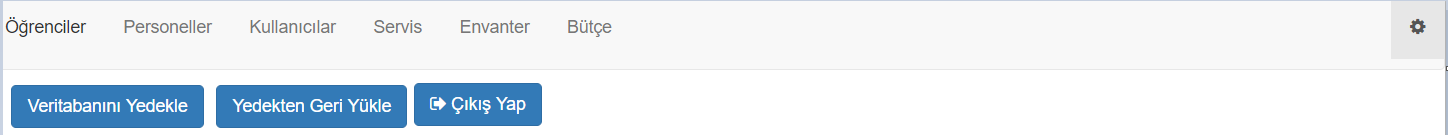
**3.8.** Envanter Menüsü

Ara yüzdeki bütçe kısmının görüntüsü Şekil 3.9 de gösterilmiştir. Bütçe kısmında anaokulu muhasebesini girdi ve çıktıları bir grafikte gösterilmiştir. Grafikteki değerler, prosedürün çalışması sonucu gelen verilerdir. Grafiğin altında 4 tane buton bulunmaktadır. Bu butonların isimleri sırası ile şu şekildedir; ödeme tarihi geçmiş ve ödenmemiş taksitleri listele, bu ayki ödenmemiş taksitleri listele, bu ayki ödenen taksitleri listele, gelecek taksitleri listele’dir. Buradaki butonlara SELECT işlemi prosedürler yardımı ile gösterilmiştir.



**Şekil 3.10** Bütçe Ara Yüzü

Ara yüzde son kısım olan ayarlar bölümü Şekil 3.2 de gösterildiği gibidir. Bu bölümde 3 tane buton vardır ve bunların isimleri şu şekildedir; veritabanı yedekle, yedekten geri yükle ve çıkış butonları vardır. Veritabanı yedekle butonu bütün veri tabanını prosedür yardımıyla yedekleme işlemini gerçekleştirir. İkinci buton olan yedekten geri yükle ise daha önceden yedekleme işlemi yapılan dosyayı tekrar import eder.



**Şekil 3.10** Ayarlar Ara Yüzü

**4.EKLER**

**4.1.VERİTABANI TABLOLARI**

**4.1.1.Ebeveyn**

Create Table ebeveyn

(

ogr\_tc varchar(11) Foreign Key References ogrenci(ogr\_tc),

veli\_ad varchar(15) Not Null,

veli\_soyad varchar(15) Not Null,

veli\_tel varchar(10) Unique Not Null

constraint ck\_veli\_tel

check(veli\_tel Like '[5][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

veli\_meslek varchar(15),

)

**4.1.2.Ek Bilgi**

Create Table ekbilgi

(

ogr\_tc varchar(11) Foreign Key References ogrenci(ogr\_tc) ,

ek\_ad varchar(15) Not Null,

ek\_soyad varchar(15) Not Null,

ek\_tel varchar(10) Unique Not Null

constraint ck\_ek\_tel

check(ek\_tel Like '[5][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

ek\_aciklama varchar(50),

ek\_beden\_durum varchar(50),

ek\_alerji varchar(25)

)

**4.1.3.Envanter**

Create Table envanter

(

urun\_no int identity(1,1) primary key,

urun\_ad varchar(10) Not Null,

birim\_fiyat money Not Null,

adet int Not Null,

toplam\_fiyat as (birim\_fiyat\*adet),

urun\_kayit\_tar date

constraint ck\_urun\_kayit\_tar

check(urun\_kayit\_tar<=getdate()),

firma\_ad varchar(20) Not Null,

firma\_tel varchar(10) UNIQUE Not Null

constraint ck\_firma\_tel

check(firma\_tel Like '[5][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

per\_tc varchar(11) Foreign Key References personel(per\_tc)

)

**4.1.4.Giris**

CREATE TABLE giris

(

per\_tc varchar(11) Foreign Key References personel(per\_tc) ,

kullanici\_adi varchar(30) UNIQUE Not Null,

kullanici\_sifre varchar(64) Not Null

)

**4.1.5.Ogrenci**

CREATE TABLE ogrenci

(

ogr\_tc varchar(11) Primary Key

constraint ck\_ogr\_tc

check(len(ogr\_tc)=11) Not Null,

ogr\_ad varchar(15) Not Null,

ogr\_soyad varchar(20) Not Null,

ogr\_dog\_tar date Not Null

constraint ck\_ogr\_dog\_tar

check(ogr\_dog\_tar<getdate() And (YEAR(GETDATE())-YEAR(ogr\_dog\_tar))>=2 And (YEAR(GETDATE())-YEAR(ogr\_dog\_tar))<=6),

ogr\_cins varchar(1) Not Null

constraint ck\_ogr\_cins

check(ogr\_cins in('E','K')),

ogr\_adres varchar(200) Not Null,

ogr\_kayit\_tar date

constraint ck\_ogr\_kayit\_tar

default(getdate()),

ogr\_durum varchar(5)

constraint ck\_ogr\_durum

check(ogr\_durum in('Aktif','Pasif','Mezun')),

ogr\_yas\_grup varchar(3) Not Null

constraint ck\_ogr\_yas\_grup

check(ogr\_yas\_grup in('2-4','4-6')),

plaka varchar(9) Foreign Key References servis(plaka) ,

sinif\_ad varchar(15) Foreign Key References sinif(sinif\_ad)

)

**4.1.6.Personel**

CREATE TABLE personel

(

per\_tc varchar(11) Primary Key Not Null

constraint ck\_per\_tc

check(len(per\_tc)=11),

per\_ad varchar(15) Not Null,

per\_soyad varchar(15) Not Null,

per\_dog\_tar date Not Null

constraint ck\_per\_dog\_tar

check(per\_dog\_tar<getdate()),

per\_tel varchar(10) Unique Not Null

constraint ck\_per\_tel

check(per\_tel Like '[5][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

per\_bas\_tar date

constraint ck\_per\_bas\_tar

default(getdate()),

per\_adres varchar(200) Not Null,

per\_email varchar(30)

constraint ck\_per\_email

check(per\_email like '%\_@\_\_%.com' or per\_email like '%\_@\_\_%.org' and charindex(' ',per\_email,1)=0),

per\_maas money Not Null,

per\_gorev varchar(10) Not Null

constraint ck\_per\_\_gorev

check(per\_gorev in ('Öğretmen','Müdür','Hademe','Aşçı')),

per\_durum varchar(5)

constraint ck\_per\_durum

check(per\_durum in('Aktif','Pasif')),

per\_sigorta\_no varchar(13)

constraint ck\_per\_sigorta\_no

check(per\_sigorta\_no Like '[5][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]')

)

**4.1.7.Servis**

Create Table servis

(

guzergah varchar(20) Not Null,

plaka varchar(9) primary key Not Null

constraint ck\_plaka

check( plaka Like '[0-7][0-9][A-Z][A-Z][0-9][0-9][0-9]'

or plaka Like '[8][0][A-Z][A-Z][0-9][0-9][0-9]'

or plaka Like '[8][1][A-Z][A-Z][0-9][0-9][0-9]'

or plaka Like '[0-7][0-9][A-Z][A-Z][0-9][0-9][0-9][0-9]'

or plaka Like '[8][0][A-Z][A-Z][0-9][0-9][0-9][0-9]'

or plaka Like '[8][1][A-Z][A-Z][0-9][0-9][0-9][0-9]'

or plaka Like '[0-7][0-9][A-Z][A-Z][0-9][0-9]'

or plaka Like '[8][0][A-Z][A-Z][0-9][0-9]'

or plaka Like '[8][1][A-Z][A-Z][0-9][0-9]'

or plaka Like '[0-7][0-9][A-Z][A-Z][A-Z][0-9][0-9]'

or plaka Like '[8][0][A-Z][A-Z][A-Z][0-9][0-9]'

or plaka Like '[8][1][A-Z][A-Z][A-Z][0-9][0-9]'

or plaka Like '[0-7][0-9][A-Z][A-Z][A-Z][0-9][0-9][0-9]'

or plaka Like '[8][0][A-Z][A-Z][A-Z][0-9][0-9][0-9]'

or plaka Like '[8][1][A-Z][A-Z][A-Z][0-9][0-9][0-9]'),

sofor\_ad varchar(10) Not Null,

sofor\_soyad varchar(15) Not Null,

ser\_tel varchar(10) Unique Not Null

constraint ck\_ser\_tel

check(ser\_tel Like '[5][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]'),

)

**4.1.8.Sinif**

Create Table sinif

(

sinif\_ad varchar(15) primary key Not Null,

kapasite smallint Not Null,

sinif\_yas\_grup varchar(3)

constraint ck\_sinif\_yas\_grup

check(sinif\_yas\_grup in('2-4','4-6')),

)

**4.1.9.Taksit**

Create Table taksit

(

ogr\_tc varchar(11) Foreign Key References ogrenci(ogr\_tc) ,

odeme\_tar date Not Null,

taksit\_fiyat money Not Null,

taksit\_durum varchar(10) Not Null

constraint ck\_taksit\_durum

check(taksit\_durum in ('Ödendi','Ödenmedi')),

)

**4.1.10.Silinenler Tabloları**

CREATE TABLE silinen\_ogrenci

(

ogr\_tc varchar(11),

ogr\_ad varchar(15) ,

ogr\_soyad varchar(20) ,

ogr\_dog\_tar date,

ogr\_cins varchar(1),

ogr\_adres varchar(200) ,

ogr\_kayit\_tar date,

ogr\_durum varchar(5),

ogr\_yas\_grup varchar(3),

plaka varchar(9),

sinif\_ad varchar(15)

)

CREATE TABLE silinen\_personel

(

per\_tc varchar(11),

per\_ad varchar(15) ,

per\_soyad varchar(15),

per\_dog\_tar date,

per\_tel varchar(10),

per\_bas\_tar date,

per\_adres varchar(200),

per\_email varchar(30),

per\_maas money ,

per\_gorev varchar(10),

per\_durum varchar(5),

per\_sigorta\_no varchar(13)

)

Create Table silinen\_envanter

(

urun\_no int,

urun\_ad varchar(10),

birim\_fiyat money,

adet int,

toplam\_fiyat as (birim\_fiyat\*adet),

urun\_kayit\_tar date,

firma\_ad varchar(20),

firma\_tel varchar(10),

per\_tc varchar(11)

)

**4.2.VERİ TABANI PROSEDÜRLER**

**4.2.1.sp\_anasayfa**

Create Proc sp\_anasayfa

As

Select \* From sinif for json path

Select \* From personel for json path

Select \* From servis for json path

Go

**4.2.2.sp\_ogrenci\_kayit**

Create Procedure sp\_ogrenci\_kayit

@tc varchar(11),@ad varchar(15),@soyad varchar(20),@dogtar date,@cins char(1),@adres varchar(200),@kayittar date,

@durum varchar(5),@yasgrup varchar(3),@plaka varchar(9),@sinif varchar(15),@veli\_ad varchar(15),

@veli\_soyad varchar(15),@veli\_tel varchar(10),@veli\_meslek varchar(15),@ek\_ad varchar(15),@ek\_soyad varchar(15),

@ek\_tel varchar(10),@aciklama varchar(50),@beden\_durum varchar(50),@alerji varchar(25)

As

IF(Select Count(\*) From personel where per\_tc = @tc)=0

Begin

Insert Into ogrenci Values(@tc,@ad,@soyad,@dogtar,@cins,@adres,@kayittar,@durum,@yasgrup,@plaka,@sinif)

IF(Select Count(\*) From ogrenci where ogr\_tc = @tc)>0

Begin

Insert Into ebeveyn Values(@tc,@veli\_ad,@veli\_soyad,@veli\_tel,@veli\_meslek)

Insert Into ekbilgi Values(@tc,@ek\_ad,@ek\_soyad,@ek\_tel,@aciklama,@beden\_durum,@alerji)

End

else

begin

return 2

end

Return 1

End

Else

Begin

Return 2

End

Go

**4.2.3.sp\_personel\_kayit**

Create Proc sp\_personel\_kayit

@per\_tc varchar(11),@per\_ad varchar(15),@per\_soyad varchar(15),@per\_dog\_tar date

,@per\_tel varchar(10),@per\_bas\_tar date,@per\_adres varchar(200),@per\_email varchar(30),

@per\_maas money,@per\_gorev varchar(10),@per\_durum varchar(5),@per\_sigorta\_no varchar(13)

As

IF(Select Count(\*) From ogrenci where ogr\_tc = @per\_tc)=0

Begin

Insert Into personel Values(@per\_tc,@per\_ad,@per\_soyad,@per\_dog\_tar,@per\_tel

,@per\_bas\_tar,@per\_adres,@per\_email,@per\_maas,@per\_gorev,@per\_durum,@per\_sigorta\_no)

Return 1

End

Else

Begin

Return 2

End

Go

**4.2.4.sp\_gecmis\_taksit**

Create Proc sp\_gecmis\_taksit

As

Select \*

From taksit inner join ogrenci

on ogrenci.ogr\_tc=taksit.ogr\_tc

where ((datepart(month,odeme\_tar)<datepart(month,getdate()))

and (datepart(year,odeme\_tar)=datepart(year,getdate())))

or (datepart(year,odeme\_tar)<datepart(year,getdate()))

ORDER BY odeme\_tar DESC for json auto

Go

**4.2.5.sp\_gelecek\_taksit**

Create Proc sp\_gelecek\_taksit

As

Select \* From taksit inner join ogrenci

on ogrenci.ogr\_tc=taksit.ogr\_tc

where

((datepart(month,odeme\_tar)>datepart(month,getdate()))

and

(datepart(year,odeme\_tar)=datepart(year,getdate())))

or (datepart(year,odeme\_tar)>datepart(year,getdate()))

ORDER BY ogr\_ad ASC,odeme\_tar ASC

for json auto

Go

**4.2.6.sp\_odenmemis\_taksit**

Create Proc sp\_odenmemis\_taksit

As

Select \* From taksit inner join ogrenci

on ogrenci.ogr\_tc=taksit.ogr\_tc

where (datepart(month,odeme\_tar)=datepart(month,getdate())

and datepart(year,odeme\_tar)=datepart(year,getdate())

and taksit\_durum='Ödenmedi')

ORDER BY odeme\_tar ASC,ogr\_ad ASC

for json auto

Go

**4.2.7.sp\_odenmis\_taksit**

Create Proc sp\_odenmis\_taksit

Select \* From taksit inner join ogrenci

on ogrenci.ogr\_tc=taksit.ogr\_tc

where (datepart(month,odeme\_tar)=datepart(month,getdate())

and datepart(year,odeme\_tar)=datepart(year,getdate())

and taksit\_durum='Ödendi' )

ORDER BY odeme\_tar ASC,ogr\_ad ASC

for json auto

Go

**4.2.8.sp\_taksit\_toplam**

Create Proc sp\_taksit\_toplam

@yil smallint

As

Select Month(odeme\_tar) AS ay,sum(taksit\_fiyat) AS toplam

From taksit

where Year(odeme\_tar)=@yil and taksit\_durum='Ödendi' Group By MONTH(odeme\_tar) for json auto

Go

**4.2.9.sp\_yedekten\_geri\_yukle**

Create Proc sp\_yedekten\_geri\_yukle

AS

RESTORE DATABASE [anaokulu]

FROM DISK = N'/home/alp/anaokulu\_yedek.bak'

WITH FILE = 1, REPLACE

GO

**4.3.VERİ TABANI TRİGGER’LARI**

**4.3.1.trg\_silinen\_ogrenci**

Create Trigger trg\_silinen\_ogrenci

On ogrenci

After Update

As

IF update(ogr\_durum)

Begin

Insert Into silinen\_ogrenci

(ogr\_tc,ogr\_ad,ogr\_soyad,ogr\_dog\_tar,ogr\_cins,ogr\_adres,

ogr\_kayit\_tar,ogr\_durum,ogr\_yas\_grup,plaka,sinif\_ad)

Select ogr\_tc,ogr\_ad,ogr\_soyad,ogr\_dog\_tar,ogr\_cins,ogr\_adres,

ogr\_kayit\_tar,ogr\_durum,ogr\_yas\_grup,plaka,sinif\_ad From ogrenci where ogr\_durum='Pasif'

IF(Select Count(\*) From ekbilgi where ogr\_tc = (Select ogr\_tc From ogrenci where ogr\_durum='Pasif'))>0

Begin

Delete From ekbilgi Where ogr\_tc=(Select ogr\_tc From ogrenci where ogr\_durum='Pasif')

End

IF(Select Count(\*) From taksit where ogr\_tc = (Select ogr\_tc From ogrenci where ogr\_durum='Pasif'))>0

Begin

Delete From taksit Where ogr\_tc=(Select ogr\_tc From ogrenci where ogr\_durum='Pasif')

End

Delete From ebeveyn Where ogr\_tc=(Select ogr\_tc From ogrenci where ogr\_durum='Pasif')

Delete From ogrenci Where ogr\_durum='Pasif'

End

**4.3.2.trg\_personel**

Create Trigger trg\_personel

On personel

After Update

As

IF update(per\_durum)

Begin

Insert Into silinen\_personel

(per\_tc,per\_ad,per\_soyad,per\_dog\_tar,per\_tel,per\_bas\_tar,

per\_adres,per\_email,per\_maas,per\_gorev,per\_durum,per\_sigorta\_no)

Select per\_tc,per\_ad,per\_soyad,per\_dog\_tar,per\_tel,per\_bas\_tar,

per\_adres,per\_email,per\_maas,per\_gorev,per\_durum,per\_sigorta\_no From personel where per\_durum='Pasif'

IF(Select Count(\*) from envanter where per\_tc=(Select per\_tc From personel where per\_durum='Pasif'))>0

Begin

Execute geribil

End

IF(Select Count(\*) From giris where per\_tc = (Select per\_tc From personel where per\_durum='Pasif'))>0

Begin

Delete From giris where per\_tc = (Select per\_tc From personel where per\_durum='Pasif')

End

Delete From personel Where per\_durum='Pasif'

End

**4.3.3.trg\_silinen\_envanter**

Create Trigger trg\_silinen\_envanter

on envanter

instead of delete

as

declare @a varchar(11)

select @a = deleted.urun\_no from deleted

insert into silinen\_envanter(urun\_no,urun\_ad,birim\_fiyat,adet,urun\_kayit\_tar,firma\_ad,firma\_tel,per\_tc)

select urun\_no,urun\_ad,birim\_fiyat,adet,urun\_kayit\_tar,firma\_ad,firma\_tel,per\_tc From envanter where urun\_no=@a

Delete From envanter where urun\_no=@a

**4.1.ARAYÜZ KODLARI**

**4.1.1.Functions**

/\*addBus\*/

const sql = require('mssql');

exports.attempt = async (bus) => {

try {

const query = await sql.query`

insert into servis

values(

${bus.path},

${bus.plate},

${bus.name},

${bus.surname},

${bus.phone}

)`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*addStaff\*/

const sql = require('mssql');

exports.attempt = async (staff) => {

try {

const query = await sql.query`

insert into personel

values(

${staff.tc},

${staff.name},

${staff.surname},

${staff.bDay},

${staff.phone},

${new Date().toISOString().slice(0, 10).replace('T', ' ')},

${null},

${staff.email},

${staff.salary},

${staff.position},

'Aktif',

${staff.insurance}

)`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*addStudent\*/

const sql = require('mssql');

exports.attempt = async (student) => {

let {tc, name, surname, bDay, ageGroup, gender, address,

status, plate, sClass, parentName,

parentPhone, parentJob, parentSurname, extraName, extraSurname,

extraPhone, extraState, extraPhysical, extraAllergic, price,

advancePayment, payNum, payUnitPrice} = student;

const request = new sql.Request()

.input('tc', tc)

.input('ad', name)

.input('soyad', surname)

.input('dogtar', bDay)

.input('cins', gender)

.input('adres', address)

.input('kayittar', new Date().toISOString().slice(0, 10).replace('T', ' '))

.input('durum', 'Aktif')

.input('yasgrup', ageGroup)

.input('plaka', plate)

.input('sinif', sClass)

.input('veli\_ad', parentName)

.input('veli\_soyad', parentSurname)

.input('veli\_tel', parentPhone)

.input('veli\_meslek', parentJob)

.input('ek\_ad', extraName)

.input('ek\_soyad', extraSurname)

.input('ek\_tel', extraPhone)

.input('aciklama', extraState)

.input('beden\_durum', extraPhysical)

.input('alerji', extraAllergic);

try {

const exec = await request.execute('sp\_ogrenci\_kayit');

if(exec.returnValue != 1) return {status: 500};

if (advancePayment > 0)

await sql.query`

insert into taksit

values(

${tc},

${new Date().toISOString().slice(0, 10).replace('T', ' ')},

${advancePayment},

'Ödendi'

)`;

for (let i = 0; i < payNum; i++) {

let paymentDay = new Date();

paymentDay.setMonth(new Date().getMonth() + i + 1);

paymentDay = paymentDay.toISOString().slice(0, 10).replace('T', ' ');

await sql.query`

insert into taksit

values(

${tc},

${paymentDay},

${payUnitPrice},

'Ödenmedi'

)`;

}

return {status: 200};

} catch (err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*addStuff\*/

const sql = require('mssql');

exports.attempt = async (stuff) => {

try {

const query = await sql.query`

insert into envanter

values(

${stuff.name},

${stuff.price},

${stuff.number},

${new Date().toISOString().slice(0, 10).replace('T', ' ')},

${stuff.compName},

${stuff.compPhone},

${stuff.staffTC}

)`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*addUser\*/

const crypto = require('crypto');

const sql = require('mssql');

exports.attempt = async (user) => {

try {

const query = await sql.query`

insert into giris

values(${user.tc}, ${user.username}, ${hash(user.pass)})`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

function hash(text) {

return crypto.createHash('sha256').update(text).digest('hex');

}

/\*deleteBus\*/

const sql = require('mssql');

exports.attempt = async (bus) => {

try {

const query = await sql.query`

insert into servis

values(

${bus.path},

${bus.plate},

${bus.name},

${bus.surname},

${bus.phone}

)`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*deleteStaff\*/

const sql = require('mssql');

exports.attempt = async (tc) => {

try {

const query = await sql.query`

update personel

set per\_durum='Pasif'

where per\_tc=${tc}`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*deleteStudent\*/

const sql = require('mssql');

exports.attempt = async (tc) => {

try {

const query = await sql.query`

update ogrenci

set ogr\_durum='Pasif'

where ogr\_tc=${tc}`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*deleteStuff\*/

const sql = require('mssql');

exports.attempt = async (no) => {

try {

const query = await sql.query`

delete from envanter

where urun\_no=${no}`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*deleteUser\*/

const sql = require('mssql');

exports.attempt = async (tc) => {

try {

const query = await sql.query`

delete from giris

where per\_tc=${tc}`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*getBackup\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`

BACKUP DATABASE [anaokulu]

TO DISK = N'/home/alp/anaokulu\_yedek.bak'

WITH NOFORMAT, INIT, NAME = 'anaokulu\_yedek', NOREWIND`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*importBus\*/

const sql = require('mssql');

exports.attempt = async (buses) => {

try {

await buses.forEach(async (bus) => {

const query = await sql.query`

insert into servis

values(

${bus.guzergah},

${bus.plaka},

${bus.sofor\_ad},

${bus.sofor\_soyad},

${bus.ser\_tel}

)`;

})

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listBuses\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`

select \*

from servis

for json auto`;

return query.recordset[0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listInventory\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`

select \*

from envanter

inner join personel on envanter.per\_tc = personel.per\_tc

for json auto`;

return query.recordset[0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listNextPay\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`exec sp\_gelecek\_taksit`;

return query.recordsets[0][0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listNotPay\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`exec sp\_odenmemis\_taksit`;

return query.recordsets[0][0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listOldPay\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`exec sp\_gecmis\_taksit`;

return query.recordsets[0][0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listPaid\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`exec sp\_odenmis\_taksit`;

return query.recordsets[0][0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listStaffs\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`select \* from personel for json path`;

return query.recordset[0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listStudents\*/

const sql = require('mssql');

exports.attempt = async studentClass => {

try {

const query = await sql.query`

select \*

from ogrenci

inner join ebeveyn on ogrenci.ogr\_tc = ebeveyn.ogr\_tc

inner join ekbilgi on ogrenci.ogr\_tc = ekbilgi.ogr\_tc

where sinif\_ad=${studentClass}

for json auto`;

return query.recordset[0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*listUsers\*/

const sql = require('mssql');

exports.attempt = async () => {

try {

const query = await sql.query`

select \*

from giris

inner join personel on giris.per\_tc=personel.per\_tc

for json auto`;

return query.recordset[0];

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*login\*/

const crypto = require('crypto');

const sql = require('mssql');

exports.attempt = async (username, password) => {

try {

const userQuery = await sql.query`select \* from giris where kullanici\_adi=${username} for json path`;

const user = userQuery.recordset[0][0];

if(user)

if(compareHash(password, user.kullanici\_sifre)) return {status:200};

else return {status:401};

else return {status:404};

} catch (err) {

return {status:404};

}

}

function compareHash(text, hash) {

return crypto.createHash('sha256').update(text).digest('hex') == hash;

}

/\*pay\*/

const sql = require('mssql');

exports.attempt = async (tc,date) => {

try {

const query = await sql.query`

update taksit

set taksit\_durum='Ödendi'

where ogr\_tc=${tc}

and odeme\_tar=${date}`;

return {status:200};

} catch(err) {

console.error(err);

return {status:err.number ? err.number : 500};

}

}

/\*restoreBackup\*/

const sql = require('mssql');

exports.attempt = async () =>

new sql.ConnectionPool({

"server" : "52.164.200.179",

"user": "SA",

"password": "anaokulu\_123",

"database": "master",

}).connect().then(pool => {

return pool.query`exec sp\_yedekten\_geri\_yukle`;

}).then(result => {return {status:200}}).catch(err => {

console.error(err);

return {status:err.number ? err.number : 500};

});

**4.1.1.Public(css)**

/\*style.css\*/

html {

font-size: 62.5%;

}

body {

font-family: 'Open Sans', sans-serif;

font-size: 1.6rem;

margin: 20px;

background-image: url("/logo.jpg");

background-repeat: no-repeat;

background-attachment: fixed;

background-position: right top;

}

@media only screen and (max-width: 1555px) {

body {

background-image: url("/small\_logo.jpg");

}

}

nav.navbar {

margin: 0 auto !important;

padding: 0 !important;

height: 60px !important;

}

ul.nav.navbar-nav {

width: 100%;

}

li:nth-last-child(1) {

float: right !important;

display: inline-block;

}

input::-webkit-outer-spin-button,

input::-webkit-inner-spin-button {

-webkit-appearance: none;

margin: 0;

}

tr.student {

cursor:pointer;

-moz-user-select: none;

-webkit-user-select: none;

-ms-user-select:none;

-o-user-select:none;

user-select:none;

}

tr.staff {

cursor:pointer;

-moz-user-select: none;

-webkit-user-select: none;

-ms-user-select:none;

-o-user-select:none;

user-select:none;

}

textarea {

resize: none;

}

label {

-moz-user-select: none;

-webkit-user-select: none;

-ms-user-select:none;

-o-user-select:none;

user-select:none;

}

.container {

border: 1px solid rgba(0, 0, 0, .3);

border-radius: 6px;

margin: 0 auto;

padding: 0 !important;

border-top: 0px;

min-height: 60px;

}

.btn-primary {

margin-left: 6px;

}

.dropdown {

margin-bottom: 6px !important;

}

#staffTable {

margin-top:4px !important;

}

#usersTable {

margin-top:4px !important;

}

#busesTable {

margin-top:4px !important;

}

**4.1.2.Public(scripts)**

/\*login\*/

/\*Hatayi gizle eger acik kaldiysa\*/

hideErr();

/\*HTML'den formu bulup formu gonderme eventini yakala\*/

document.getElementById('signin-form').onsubmit = event => {

/\*normal form olarak calismasini engele\*/

event.preventDefault();

/\*

Login olmayi dene

(parametre olarak yollanan nesnede html'den alinmis kullanici adi ve sifre degerleri var)

\*/

attemptLogin({

username: document.getElementById('username').value,

pass: document.getElementById('password').value

});

}

/\*Giris Yapmayi Dene\*/

function attemptLogin(data) {

/\*Animasyon\*/

document.getElementById('wrapper').className = 'wrapper animated fadeOutRightBig';

/\*Hata gizle eger acik kaldiysa\*/

hideErr();

/\*Api'ye istek yolla\*/

return fetch('/auth', {

method: 'POST',

credentials: 'include',

headers: {

'Accept': 'application/json',

'Content-Type': 'application/json'

},

body: JSON.stringify({username:data.username, password:data.pass})

})

.then(async (response) => {

/\*Gelen cevabi kontrol et(eger hatali islem yapildi ise http kodu 400'un ustundedir)\*/

if(400 <= response.status) throw response.status;

else {

data = await response.json(); //Burada async/await kullandik cunku eger apiden cevap gelmeden bu islem yapilirsa bos degere esitler ve buyuk sicariz

document.cookie = 'username=' + data.username + ';'

window.location.href = '/';

}

})

.catch(err => showErr(err));/\*Bu islemlerde hata olursa yakala ve hatayi goster\*/

}

/\*Hatayi Gosterme Fonksiyonu\*/

function showErr(code) {

/\*Animasyon\*/

document.getElementById('wrapper').className = 'wrapper animated fadeInRightBig';

/\*HTML'deki hata div'ini bul\*/

let errAlert = document.getElementById('err');

/\*Hata koduna gore farkli hatayi yazdir(innerHTML ile div'in icine)\*/

if (code === 401)

errAlert.innerHTML = '<a href="javascript:hideErr()" class="close" aria-label="close">&tim'+

'es;</a><strong>Error!</strong> Yanlış Şifre!'+

'<img src="/400.gif" width="270" height="103" id="ticket" alt="Your Ticket">'

else if (code === 404)

errAlert.innerHTML = '<a href="javascript:hideErr()" class="close" aria-label="close">&tim'+

'es;</a><strong>Error!</strong> Kullanıcı Bulunamadı!'+

'<img src="/400.gif" width="270" height="103" id="ticket" alt="Your Ticket">'

else if (code === 500)

errAlert.innerHTML = '<a href="javascript:hideErr()" class="close" aria-label="close">&tim'+

'es;</a><strong>Error!</strong> Server Bozuldu!'+

'<img src="/400.gif" width="270" height="103" id="ticket" alt="Your Ticket">'

else if(code === undefined || code === 0)

return false

else

errAlert.innerHTML = '<a href="javascript:hideErr()" class="close" aria-label="close">&tim'+

'es;</a><strong>Unknown Error!</strong> ' + code

/\*hata div'ini gorunur yap(css ile)\*/

errAlert.style.visibility = 'visible'

}

/\*Hata gizleme Fonksiyonu\*/

function hideErr() {

/\*HTML'den hata div'ini bulup gorunmez yap(css ile)\*/

document.getElementById('err').style.visibility = 'hidden'

}

/\*main\*/

let studentsCache = [];

let busesCache = [];

let inventoryCache = [];

let usersCache = [];

let staffCache = [];

let paysCache = [];

let studentTC = document.getElementById('studentTC');

let studentName = document.getElementById('studentName');

let studentSurname = document.getElementById('studentSurname');

let studentBday = document.getElementById('studentBday');

let studentAddress = document.getElementById('studentAddress');

let studentSchoolBus = document.getElementById('studentSchoolBus');

let studentClass = document.getElementById('studentClass');

let studentParentName = document.getElementById('studentParentName');

let studentParentSurname = document.getElementById('studentParentSurname');

let studentParentJob = document.getElementById('studentParentJob');

let studentParentPhone = document.getElementById('studentParentPhone');

let studentPrice = document.getElementById('studentPrice');

let studentAdvancePayment = document.getElementById('studentAdvancePayment');

let studentPayNum = document.getElementById('studentPayNum');

let studentPayUnitPrice = document.getElementById('studentPayUnitPrice');

let studentExtraName = document.getElementById('studentExtraName');

let studentExtraSurname = document.getElementById('studentExtraSurname');

let studentExtraPhone = document.getElementById('studentExtraPhone');

let studentExtraState = document.getElementById('studentExtraState');

let studentExtraPhysical = document.getElementById('studentExtraPhysical');

let studentExtraAllergic = document.getElementById('studentExtraAllergic');

let classToList = document.getElementById('classToList');

let studentList = document.getElementById('studentList');

let schoolBus = document.getElementById('schoolbus');

let inventory = document.getElementById('inventory');

let \_users = document.getElementById('users');

let \_staffs = document.getElementById('staff');

let \_pays = document.getElementById('\_budget');

let ctx = document.getElementById("chart").getContext('2d');

let staffBday = document.getElementById('staffBday');

let backup = false;

function karakterKontrol(olay){

var tusKodu;

if(window.event){ // IE

tusKodu = olay.keyCode

}else if(olay.which){ // Netscape/Firefox/Opera

tusKodu = olay.which;

}

//alert(tusKodu)

if(tusKodu == 8){ // backspace tuşuna da izin vermek istiyorsak

return true;

}

if ((tusKodu > 64 && tusKodu < 91)

|| (tusKodu > 96 && tusKodu < 123)

|| tusKodu == 32

|| tusKodu == 105

|| tusKodu == 305

|| tusKodu == 287

|| tusKodu == 252

|| tusKodu == 351

|| tusKodu == 246

|| tusKodu == 231)

{

return true;

}

else{

tusKodu.keyCode = 0;

return false;

}

}

function rakamKontrol(olay){

var tusKodu;

if(window.event){ // IE

tusKodu = olay.keyCode

}else if(olay.which){ // Netscape/Firefox/Opera

tusKodu = olay.which;

}

//alert(tusKodu)

if(tusKodu == 8){ // backspace tuşuna da izin vermek istiyorsak

return true;

}

if (tusKodu < 48 || tusKodu > 57){

tusKodu.keyCode = 0;

return false;

}

else{

return true;

}

}

/\*DOGUM TARIHI SINIRLARI\*/

let minDate = new Date(new Date().setFullYear(new Date().getFullYear() - 6)).toJSON().split('T')[0];

let maxDate = new Date(new Date().setFullYear(new Date().getFullYear() - 2)).toJSON().split('T')[0];

studentBday.setAttribute('min', minDate);

studentBday.setAttribute('max', maxDate);

staffBday.setAttribute(

'max',

new Date(new Date().setFullYear(new Date().getFullYear() - 18)).toJSON().split('T')[0]

);

/\*Birim Taksit Hesabi\*/

function calUnitPrice() {

if (studentPrice.value.length > 5)

studentPrice.value = studentPrice.value.slice(0,5);

if (studentAdvancePayment.value.length > 5)

studentAdvancePayment.value = studentAdvancePayment.value.slice(0,5);

let price = parseInt(studentPrice.value);

let advancePayment = parseInt(studentAdvancePayment.value);

let unitSize = parseInt(studentPayNum.value);

if(price && advancePayment && unitSize && price > advancePayment)

studentPayUnitPrice.value = Math.round(((price - advancePayment)/unitSize) \* 100) / 100;

else if (advancePayment == 0) {

studentPayUnitPrice.value = Math.round((price /unitSize) \* 100) / 100;

}

else

studentPayUnitPrice.value = null;

}

/\*Ogrenci ekleme\*/

function addStudent() {

let student = {};

student.tc = studentTC.value;

student.name = studentName.value;

student.surname = studentSurname.value;

student.bDay = new Date(studentBday.value).toISOString().slice(0, 10).replace('T', ' ');

student.ageGroup = new Date().getFullYear() - new Date(student.bDay).getFullYear() < 4 ?

'2-4' : '4-6';

student.gender = document.getElementById('male').checked ? 'E' : 'K';

student.address = studentAddress.value;

student.sClass = studentClass.value;

student.bus = studentSchoolBus.value == 'none' ?

null : studentSchoolBus.value;

student.parentName = studentParentName.value;

student.parentSurname = studentParentSurname.value;

student.parentPhone = studentParentPhone.value;

student.parentJob = studentParentJob.value;

student.price = studentPrice.value;

student.advancePayment = studentAdvancePayment.value;

student.payNum = studentPayNum.value;

student.payUnitPrice = studentPayUnitPrice.value;

student.extraName = studentExtraName.value;

student.extraSurname = studentExtraSurname.value;

student.extraPhone = studentExtraPhone.value;

student.extraState = studentExtraState.value;

student.extraPhysical = studentExtraPhysical.value;

student.extraAllergic = studentExtraAllergic.value;

return fetch('/student', {

method: 'POST',

credentials: 'include',

headers: {

'Accept': 'application/json',

'Content-Type': 'application/json'

},

body: JSON.stringify(student)

})

.then(response => {

$('#addStudent').modal('hide');

if(response.status < 400) {

$.notify({

message: `${studentName.value} adlı öğrenci kaydedildi.`

},{

type: 'success'

});

} else {

$.notify({

message: `${studentName.value} adlı öğrenci kaydedilemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

}

/\*Ogrenci listeleme\*/

function listStudents() {

let first = true;

if($('#studentTable')[0]) {

first = false;

$('#studentTable').animateCss('flipOutX', function(){

$("#studentTable").css('opacity', '0');

});

}

let studentTableContent = ``;

fetch(`/student/${classToList.value}`,{credentials: 'include'})

.then(response => response.json()).then(students => {

studentsCache = students;

for (let i in students) {

let student = students[i];

if(student.ogr\_durum == 'Aktif')

studentTableContent += `

<tr class="student" onclick="studentDetail('${i}')" ondblclick="studentAlert('${i}')">

<td class="class="col-md-2">${student.ogr\_tc}</td>

<td class="class="col-md-2">${student.ogr\_ad}</td>

<td class="class="col-md-2">${student.ogr\_soyad}</td>

</tr>`;

}

studentList.innerHTML = `

<table class="table table-striped table-bordered table-hover" id="studentTable">

<thead>

<tr>

<th class="class="col-md-2">T.C</th>

<th class="class="col-md-2">Ad</th>

<th class="class="col-md-2">Soyad</th>

</tr>

</thead>

${studentTableContent}

</table>

`;

studentList.innerHTML += `

<div class="dropdown">

<button

class="btn btn-default dropdown-toggle"

type="button"

id="export"

data-toggle="dropdown"

aria-haspopup="true"

aria-expanded="true">

Export

<span class="caret"></span>

</button>

<ul class="dropdown-menu" aria-labelledby="export">

<li>

<a

download="${classToList.value}.json"

href='data:text/json;charset=utf-8,${JSON.stringify(studentsCache)}'>

JSON

</a>

</li>

<li><a href="javascript:studentsToPDF()">PDF</a></li>

<li><a href="#"></a></li>

</ul>

</div>`;

if(!first) {

$("#studentTable").css('opacity', '1');

$('#studentTable').animateCss('flipInX');

} else {

$('#studentTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

/\*Ogrenci'ye cift tiklayinca cikan alert\*/

function studentAlert(i) {

let student = studentsCache[i];

swal({

title: `${student.ogr\_ad} ${student.ogr\_soyad}`,

showConfirmButton: false,

showCancelButton: true,

cancelButtonColor: '#d33',

confirmButtonText: 'Düzenle',

cancelButtonText: 'Sil'

}).then((result) => {

if (result.dismiss === 'cancel') {

return fetch(`/student/${student.ogr\_tc}`, {

method: 'DELETE',

credentials: 'include',

})

.then(response => {

if(response.status < 400) {

$.notify({

message: `${student.ogr\_ad} adlı öğrenci silindi.`

},{

type: 'success'

});

listStudents();

} else {

$.notify({

message: `${student.ogr\_ad} adlı öğrenci silinemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err));

}

})

}

/\*Envantere cift tiklayinca cikan alert\*/

function stuffAlert(i) {

let stuff = inventoryCache[i];

swal({

title: `${stuff.urun\_ad}`,

showConfirmButton: false,

showCancelButton: true,

cancelButtonColor: '#d33',

cancelButtonText: 'Sil'

}).then((result) => {

if (result.dismiss === 'cancel') {

return fetch(`/inventory/${stuff.urun\_no}`, {

method: 'DELETE',

credentials: 'include',

})

.then(response => {

if(response.status < 400) {

$.notify({

message: `${stuff.urun\_ad} silindi.`

},{

type: 'success'

});

listInventory();

} else {

$.notify({

message: `${stuff.urun\_ad} silinemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err));

}

})

}

/\*Personele cift tiklayinca cikan alert\*/

function staffAlert(i) {

let staff = staffCache[i];

swal({

title: `${staff.per\_ad} ${staff.per\_soyad}`,

showConfirmButton: false,

showCancelButton: true,

cancelButtonColor: '#d33',

cancelButtonText: 'Sil'

}).then((result) => {

if (result.dismiss === 'cancel') {

return fetch(`/staff/${staff.per\_tc}`, {

method: 'DELETE',

credentials: 'include',

})

.then(response => {

if(response.status < 400) {

$.notify({

message: `${staff.per\_ad} silindi.`

},{

type: 'success'

});

listStaffs();

} else {

$.notify({

message: `${staff.per\_ad} silinemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err));

}

})

}

/\*Kullanıcıya cift tiklayinca cikan alert\*/

function userAlert(i) {

let user = usersCache[i];

swal({

title: `${user.kullanici\_adi}`,

showConfirmButton: false,

showCancelButton: true,

cancelButtonColor: '#d33',

cancelButtonText: 'Sil'

}).then((result) => {

if (result.dismiss === 'cancel') {

return fetch(`/user/${user.per\_tc}`, {

method: 'DELETE',

credentials: 'include',

})

.then(response => {

if(response.status < 400) {

$.notify({

message: `${user.kullanici\_adi} silindi.`

},{

type: 'success'

});

listUsers();

} else {

$.notify({

message: `${user.kullanici\_adi} silinemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err));

}

})

}

/\*Kullanıcıya cift tiklayinca cikan alert\*/

function busAlert(i) {

let bus = busesCache[i];

swal({

title: `${bus.guzergah}`,

showConfirmButton: false,

showCancelButton: true,

cancelButtonColor: '#d33',

cancelButtonText: 'Sil'

}).then((result) => {

if (result.dismiss === 'cancel') {

return fetch(`/bus/${bus.plaka}`, {

method: 'DELETE',

credentials: 'include',

})

.then(response => {

if(response.status < 400) {

$.notify({

message: `${bus.guzergah} silindi.`

},{

type: 'success'

});

listSchoolBuses();

} else {

$.notify({

message: `${bus.guzergah} silinemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err));

}

})

}

/\*OGRENCI DETAY\*/

function studentDetail(i) {

$('#studentDetail').animateCss('flash');

let student = studentsCache[i];

let parent = student.ebeveyn[0];

let extra = student.ebeveyn[0].ekbilgi[0];

document.getElementById('studentDetail').innerHTML = `

<h4 id="studentDetailH">Öğrenci Detayı</h4>

<label class="col-sm-6 control-label">Öğrenci Adı:</label>

<p class="col-sm-6">${student.ogr\_ad}</p>

<label class="col-sm-6 control-label">Öğrenci Soyadı:</label>

<p class="col-sm-6">${student.ogr\_soyad}</p>

<label class="col-sm-6 control-label">Öğrenci T.C:</label>

<p class="col-sm-6">${student.ogr\_tc}</p>

<label class="col-sm-6 control-label">Öğrenci Cinsiyeti:</label>

<p class="col-sm-6">${student.ogr\_cins == 'E' ? 'Erkek' : 'Kız'}</p>

<label class="col-sm-6 control-label">Öğrenci Adresi:</label>

<p class="col-sm-6">${student.ogr\_adres}</p>

<label class="col-sm-6 control-label">Öğrenci Doğum Tarihi:</label>

<p class="col-sm-6">${student.ogr\_dog\_tar}</p>

<label class="col-sm-6 control-label">Öğrenci Yaş Grubu:</label>

<p class="col-sm-6">${student.ogr\_yas\_grup}</p>

<label class="col-sm-6 control-label">Öğrenci Kayıt Tarihi:</label>

<p class="col-sm-6">${student.ogr\_adres}</p>

<label class="col-sm-6 control-label">Veli Adı:</label>

<p class="col-sm-6">${parent.veli\_ad}</p>

<label class="col-sm-6 control-label">Veli Soyadı:</label>

<p class="col-sm-6">${parent.veli\_soyad}</p>

<label class="col-sm-6 control-label">Veli Telefonu:</label>

<p class="col-sm-6">${parent.veli\_tel}</p>

<label class="col-sm-6 control-label">Veli Mesleği:</label>

<p class="col-sm-6">${parent.veli\_meslek}</p>

<label class="col-sm-6 control-label">Öğrencinin Servisi:</label>

<p class="col-sm-6">${student.plaka ? student.plaka : 'Servis Kullanmıyor'}</p>

<label class="col-sm-6 control-label">Ek Ad:</label>

<p class="col-sm-6">${extra.ek\_ad}</p>

<label class="col-sm-6 control-label">Ek Soyad:</label>

<p class="col-sm-6">${extra.ek\_soyad}</p>

<label class="col-sm-6 control-label">Ek Açıklama:</label>

<p class="col-sm-6">${extra.ek\_aciklama}</p>

<label class="col-sm-6 control-label">Ek Tel:</label>

<p class="col-sm-6">${extra.ek\_tel}</p>`;

if(extra.ek\_beden\_durum)

document.getElementById('studentDetail').innerHTML += `

<label class="col-sm-6 control-label">Ek Beden Durum:</label>

<p class="col-sm-6">${extra.ek\_beden\_durum}</p>`;

if(extra.ek\_alerji)

document.getElementById('studentDetail').innerHTML += `

<label class="col-sm-6 control-label">Ek Alerji:</label>

<p class="col-sm-6">${extra.ek\_alerji}</p>`;

}

/\*Servisleri listeleme\*/

function listSchoolBuses() {

let first = true;

if($('#busesTable')[0]) {

first = false;

$('#busesTable').animateCss('flipOutX', function(){

$("#busesTable").css('opacity', '0');

});

}

let busTableContent = ``;

fetch(`/bus`,{credentials: 'include'})

.then(response => response.json()).then(buses => {

busesCache = buses;

for (let i in buses) {

let bus = buses[i];

busTableContent += `

<tr class="staff" onclick="busAlert('${i}')">

<td>${bus.sofor\_ad}</td>

<td>${bus.sofor\_soyad}</td>

<td>${bus.ser\_tel}</td>

<td>${bus.guzergah}</td>

<td>${bus.plaka}</td>

</tr>`;

}

schoolBus.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listSchoolBuses()">Servisleri Listele</a>

<button class="btn btn-primary" data-toggle="modal" data-target="#addBus">

Servis Kayıt

</button>

</div>

<table class="table table-striped table-bordered table-hover" id="busesTable">

<thead>

<tr>

<th>Şöför Adı</th>

<th>Şöför Soyadı</th>

<th>Şöför Telefonu</th>

<th>Güzergah</th>

<th>Plaka</th>

</tr>

</thead>

${busTableContent}

</table>

`;

schoolBus.innerHTML += `

<div class="dropdown">

<button

class="btn btn-default dropdown-toggle"

type="button"

id="export"

data-toggle="dropdown"

aria-haspopup="true"

aria-expanded="true">

Export

<span class="caret"></span>

</button>

<ul class="dropdown-menu" aria-labelledby="export">

<li>

<a

download="Servisler.json"

href='data:text/json;charset=utf-8,${JSON.stringify(busesCache)}'>

JSON

</a>

</li>

<li><a href="javascript:busesToPDF()">PDF</a></li>

<li><a href="#"></a></li>

</ul>

</div>

<input id="upload" type="file" style="display:none" onchange="importBus()"/>

<a class="btn btn-default" href="" id="upload\_link">import JSON</a>​`;

if(!first) {

$("#busesTable").css('opacity', '1');

$('#busesTable').animateCss('flipInX');

} else {

$('#busesTable').animateCss('zoomIn');

}

$("#upload\_link").on('click', function(e){

e.preventDefault();

$("#upload:hidden").trigger('click');

});

})

.catch(err => console.error(err));

}

/\*Envanteri listeleme\*/

function listInventory() {

let first = true;

if($('#inventoryTable')[0]) {

first = false;

$('#inventoryTable').animateCss('flipOutX', function(){

$("#inventoryTable").css('opacity', '0');

});

}

let inventoryTableContent = ``;

fetch(`/inventory`,{credentials: 'include'})

.then(response => response.json()).then(stuffs => {

inventoryCache = stuffs;

for (let i in stuffs) {

let stuff = stuffs[i];

inventoryTableContent += `

<tr class="staff" onclick="stuffAlert('${i}')">

<td>${stuff.urun\_no}</td>

<td>${stuff.urun\_ad}</td>

<td>${stuff.birim\_fiyat}</td>

<td>${stuff.adet}</td>

<td>${stuff.toplam\_fiyat}</td>

<td>${stuff.urun\_kayit\_tar}</td>

<td>${stuff.firma\_ad}</td>

<td>${stuff.firma\_tel}</td>

<td>${stuff.personel[0].per\_ad} ${stuff.personel[0].per\_soyad}</td>

</tr>`;

}

inventory.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listInventory()">Envanteri Listele</a>

<button class="btn btn-primary" data-toggle="modal" data-target="#addStuff">

Eşya Kayıt

</button>

</div>

<table class="table table-striped table-bordered table-hover" id="inventoryTable">

<thead>

<tr>

<th>Ürün No.</th>

<th>Ürün Adı</th>

<th>Birim Fiyat</th>

<th>Adet</th>

<th>Toplam Fiyat</th>

<th>Ürün Kayıt Tarihi</th>

<th>Firma Ad</th>

<th>Firma Tel</th>

<th>Personel</th>

</tr>

</thead>

${inventoryTableContent}

</table>

`;

if(!first) {

$("#inventoryTable").css('opacity', '1');

$('#inventoryTable').animateCss('flipInX');

} else {

$('#inventoryTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

/\*Personelleri listeleme\*/

function listStaffs() {

let first = true;

if($('#staffTable')[0]) {

first = false;

$('#staffTable').animateCss('flipOutX', function(){

$("#staffTable").css('opacity', '0');

});

}

let staffTableContent = ``;

fetch(`/staff`,{credentials: 'include'})

.then(response => response.json()).then(staffs => {

staffCache = staffs;

for (let i in staffs) {

let staff = staffs[i];

staffTableContent += `

<tr class="staff" onclick="staffAlert('${i}')">

<td>${staff.per\_tc}</td>

<td>${staff.per\_ad}</td>

<td>${staff.per\_soyad}</td>

<td>${staff.per\_dog\_tar}</td>

<td>${staff.per\_tel}</td>

<td>${staff.per\_maas}</td>

<td>${staff.per\_gorev}</td>

<td>${staff.per\_email}</td>

<td>${staff.per\_sigorta\_no}</td>

</tr>`;

}

\_staffs.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listStaffs()">Personelleri Listele</a>

<button class="btn btn-primary" data-toggle="modal" data-target="#addStaff">

<i class="fa fa-user-plus"></i> Personel Kayıt

</button>

</div>

<table class="table table-bordered table-hover table-responsive table-striped" id="staffTable">

<thead>

<tr>

<th>T.C</th>

<th>Ad</th>

<th>Soyad</th>

<th>Doğum Tarihi</th>

<th>Telefon</th>

<th>Maaş</th>

<th>Görev</th>

<th>Eposta</th>

<th>Sigorta No.</th>

</tr>

</thead>

${staffTableContent}

</table>

<div class="dropdown">

<button

class="btn btn-default dropdown-toggle"

type="button"

id="exportStaff"

data-toggle="dropdown"

>

Export

<span class="caret"></span>

</button>

<ul class="dropdown-menu">

<li><a

download="Personeller.json"

href="data:text/json;charset=utf-8,${JSON.stringify(staffs)}">JSON</a></li>

<li><a href="javascript:staffToPDF()">PDF</a></li>

<li><a href="#"></a></li>

</ul>

</div>

`;

if(!first) {

$("#staffTable").css('opacity', '1');

$('#staffTable').animateCss('flipInX');

} else {

$('#staffTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

/\*Kullanıcıları listeleme\*/

function listUsers() {

let first = true;

if($('#usersTable')[0]) {

first = false;

$('#usersTable').animateCss('flipOutX', function(){

$("#usersTable").css('opacity', '0');

});

}

let usersTableContent = ``;

fetch(`/user`,{credentials: 'include'})

.then(response => response.json()).then(users => {

usersCache = users;

for (let i in users) {

let user = users[i];

usersTableContent += `

<tr class="staff" onclick="userAlert('${i}')">

<td>${user.personel[0].per\_ad} ${user.personel[0].per\_soyad}</td>

<td>${user.kullanici\_adi}</td>

<td>${user.kullanici\_sifre}</td>

</tr>`;

}

\_users.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listUsers()">Kullanıcıları Listele</a>

<button class="btn btn-primary" data-toggle="modal" data-target="#addUser">

<i class="fa fa-user-plus" aria-hidden="true"></i> Kullanıcı Kayıt

</button>

</div>

<table class="table table-bordered table-hover table-striped" id="usersTable">

<thead>

<tr>

<th>Personel</th>

<th>Kullanıcı Adı</th>

<th>Hash'lenmiş şifre</th>

</tr>

</thead>

${usersTableContent}

</table>`;

\_users.innerHTML += `

<div class="dropdown">

<button

class="btn btn-default dropdown-toggle"

type="button"

id="export"

data-toggle="dropdown"

aria-haspopup="true"

aria-expanded="true">

Export

<span class="caret"></span>

</button>

<ul class="dropdown-menu" aria-labelledby="export">

<li>

<a

download="Kullanıcılar.json"

href='data:text/json;charset=utf-8,${JSON.stringify(usersCache)}'>

JSON

</a>

</li>

<li><a href="javascript:usersToPDF()">PDF</a></li>

<li><a href="#"></a></li>

</ul>

</div>`;

if(!first) {

$("#usersTable").css('opacity', '1');

$('#usersTable').animateCss('flipInX');

} else {

$('#usersTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

/\*Kullanıcı ekleme\*/

function addUser() {

let user = {

username: document.getElementById('userName').value,

pass: document.getElementById('userPassword').value,

tc: document.getElementById('userStaff').value

};

return fetch('/user', {

method: 'POST',

credentials: 'include',

headers: {

'Accept': 'application/json',

'Content-Type': 'application/json'

},

body: JSON.stringify(user)

})

.then(response => {

$('#addUser').modal('hide');

if(response.status < 400) {

$.notify({

message: `${user.username} kaydedildi.`

},{

type: 'success'

});

} else {

$.notify({

message: `${user.username} kaydedilemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err))

}

/\*Personel ekleme\*/

function addStaff() {

let staff = {

tc: document.getElementById('staffTC').value,

name: document.getElementById('staffName').value,

surname: document.getElementById('staffSurname').value,

bDay: staffBday.value,

phone: document.getElementById('staffPhone').value,

position: document.getElementById('staffPosition').value,

email: document.getElementById('staffEmail').value,

salary: document.getElementById('staffSalary').value,

insurance: document.getElementById('staffInsurance').value,

};

return fetch('/staff', {

method: 'POST',

credentials: 'include',

headers: {

'Accept': 'application/json',

'Content-Type': 'application/json'

},

body: JSON.stringify(staff)

})

.then(response => {

$('#addStaff').modal('hide');

if(response.status < 400) {

$.notify({

message: `${staff.name} kaydedildi.`

},{

type: 'success'

});

} else {

$.notify({

message: `${staff.name} kaydedilemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err))

}

/\*Esya ekleme\*/

function addStuff() {

let stuff = {

name: document.getElementById('stuffName').value,

price: document.getElementById('stuffPrice').value,

number: document.getElementById('stuffNum').value,

compName: document.getElementById('stuffCompName').value,

compPhone: document.getElementById('stuffCompPhone').value,

staffTC: document.getElementById('stuffStaff').value,

};

return fetch('/inventory', {

method: 'POST',

credentials: 'include',

headers: {

'Accept': 'application/json',

'Content-Type': 'application/json'

},

body: JSON.stringify(stuff)

})

.then(response => {

$('#addStuff').modal('hide');

if(response.status < 400) {

$.notify({

message: `${stuff.name} kaydedildi.`

},{

type: 'success'

});

} else {

$.notify({

message: `${stuff.name} kaydedilemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err))

}

/\*Servis ekleme\*/

function addBus() {

let bus = {

name: document.getElementById('busName').value,

surname: document.getElementById('busSurname').value,

phone: document.getElementById('busPhone').value,

path: document.getElementById('busPath').value,

plate: document.getElementById('busPlate').value,

};

return fetch('/bus', {

method: 'POST',

credentials: 'include',

headers: {

'Accept': 'application/json',

'Content-Type': 'application/json'

},

body: JSON.stringify(bus)

})

.then(response => {

$('#addBus').modal('hide');

if(response.status < 400) {

$.notify({

message: `${bus.path} kaydedildi.`

},{

type: 'success'

});

} else {

$.notify({

message: `${bus.path} kaydedilemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err))

}

/\*Butce listelemeleri\*/

function listOldPays() {

let first = true;

if($('#paysTable')[0]) {

first = false;

$('#paysTable').animateCss('flipOutX', function(){

$("#paysTable").css('opacity', '0');

});

}

let paysTableContent = ``;

fetch(`/oldpays`,{credentials: 'include'})

.then(response => response.json()).then(pays => {

for (let i in pays) {

let pay = pays[i];

paysTableContent += `

<tr class="staff" onclick="payAlert('${i}')">

<td>${pay.ogrenci[0].ogr\_ad} ${pay.ogrenci[0].ogr\_soyad}</td>

<td>${pay.taksit\_fiyat}</td>

<td>${pay.odeme\_tar}</td>

<td>${pay.taksit\_durum}</td>

</tr>`;

}

\_pays.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listOldPays()">Ödeme Tarihi Geçmiş ve Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPays()">Bu Ayki Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPaid()">Bu Ayki Ödenen Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listNextPays()">Gelecek Taksitleri Listele</a>

</div>

<table class="table table-bordered table-hover table-striped" id="paysTable">

<thead>

<tr>

<th>Öğrenci</th>

<th>Tutar</th>

<th>Ödeme Tarihi</th>

<th>Durum</th>

</tr>

</thead>

${paysTableContent}

</table>`;

if(!first) {

$("#paysTable").css('opacity', '1');

$('#paysTable').animateCss('flipInX');

} else {

$('#paysTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

function listPays() {

let first = true;

if($('#paysTable')[0]) {

first = false;

$('#paysTable').animateCss('flipOutX', function(){

$("#paysTable").css('opacity', '0');

});

}

let paysTableContent = ``;

fetch(`/newpays`,{credentials: 'include'})

.then(response => response.json()).then(pays => {

for (let i in pays) {

let pay = pays[i];

paysTableContent += `

<tr class="staff" onclick="payAlert('${i}')">

<td>${pay.ogrenci[0].ogr\_ad} ${pay.ogrenci[0].ogr\_soyad}</td>

<td>${pay.taksit\_fiyat}</td>

<td>${pay.odeme\_tar}</td>

<td>${pay.taksit\_durum}</td>

</tr>`;

}

\_pays.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listOldPays()">Ödeme Tarihi Geçmiş ve Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPays()">Bu Ayki Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPaid()">Bu Ayki Ödenen Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listNextPays()">Gelecek Taksitleri Listele</a>

</div>

<table class="table table-bordered table-hover table-striped" id="paysTable">

<thead>

<tr>

<th>Öğrenci</th>

<th>Tutar</th>

<th>Ödeme Tarihi</th>

<th>Durum</th>

</tr>

</thead>

${paysTableContent}

</table>`;

if(!first) {

$("#paysTable").css('opacity', '1');

$('#paysTable').animateCss('flipInX');

} else {

$('#paysTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

function listPaid() {

let first = true;

if($('#paysTable')[0]) {

first = false;

$('#paysTable').animateCss('flipOutX', function(){

$("#paysTable").css('opacity', '0');

});

}

let paysTableContent = ``;

fetch(`/newpaid`,{credentials: 'include'})

.then(response => response.json()).then(pays => {

for (let i in pays) {

let pay = pays[i];

paysTableContent += `

<tr class="staff">

<td>${pay.ogrenci[0].ogr\_ad} ${pay.ogrenci[0].ogr\_soyad}</td>

<td>${pay.taksit\_fiyat}</td>

<td>${pay.odeme\_tar}</td>

<td>${pay.taksit\_durum}</td>

</tr>`;

}

\_pays.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listOldPays()">Ödeme Tarihi Geçmiş ve Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPays()">Bu Ayki Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPaid()">Bu Ayki Ödenen Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listNextPays()">Gelecek Taksitleri Listele</a>

</div>

<table class="table table-bordered table-hover table-striped" id="paysTable">

<thead>

<tr>

<th>Öğrenci</th>

<th>Tutar</th>

<th>Ödeme Tarihi</th>

<th>Durum</th>

</tr>

</thead>

${paysTableContent}

</table>`;

if(!first) {

$("#paysTable").css('opacity', '1');

$('#paysTable').animateCss('flipInX');

} else {

$('#paysTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

function listNextPays() {

let first = true;

if($('#paysTable')[0]) {

first = false;

$('#paysTable').animateCss('flipOutX', function(){

$("#paysTable").css('opacity', '0');

});

}

let paysTableContent = ``;

fetch(`/nextpays`,{credentials: 'include'})

.then(response => response.json()).then(pays => {

paysCache = pays;

for (let i in pays) {

let pay = pays[i];

paysTableContent += `

<tr class="staff" onclick="payAlert('${i}')">

<td>${pay.ogrenci[0].ogr\_ad} ${pay.ogrenci[0].ogr\_soyad}</td>

<td>${pay.taksit\_fiyat}</td>

<td>${pay.odeme\_tar}</td>

<td>${pay.taksit\_durum}</td>

</tr>`;

}

\_pays.innerHTML = `

<div class="form-group" style="margin-top:12px;">

<a class="btn btn-primary" href="javascript:listOldPays()">Ödeme Tarihi Geçmiş ve Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPays()">Bu Ayki Ödenmemiş Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listPaid()">Bu Ayki Ödenen Taksitleri Listele</a>

<a class="btn btn-primary" href="javascript:listNextPays()">Gelecek Taksitleri Listele</a>

</div>

<table class="table table-bordered table-hover table-striped" id="paysTable">

<thead>

<tr>

<th>Öğrenci</th>

<th>Tutar</th>

<th>Ödeme Tarihi</th>

<th>Durum</th>

</tr>

</thead>

${paysTableContent}

</table>`;

if(!first) {

$("#paysTable").css('opacity', '1');

$('#paysTable').animateCss('flipInX');

} else {

$('#paysTable').animateCss('zoomIn');

}

})

.catch(err => console.error(err));

}

function payAlert(i) {

let pay = paysCache[i];

swal({

title: `${pay.ogrenci[0].ogr\_ad} ${pay.ogrenci[0].ogr\_soyad} ${pay.odeme\_tar}`,

showConfirmButton: true,

showCancelButton: false,

cancelButtonColor: '#d33',

confirmButtonText: 'Ödendi'

}).then((result) => {

if (result.value) {

return fetch(`/pay/${pay.ogrenci[0].ogr\_tc}/${pay.odeme\_tar}`, {

method: 'PATCH',

credentials: 'include',

})

.then(response => {

if(response.status < 400) {

$.notify({

message: `Ödendi.`

},{

type: 'success'

});

} else {

$.notify({

message: `Ödenemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err));

}

})

}

function getBackup() {

fetch(`/getbackup`,{credentials: 'include'})

.then(response => {

if(response.status < 400) {

$.notify({

message: `Yedek kaydedildi.`

},{

type: 'success'

});

} else {

$.notify({

message: `Yedek kaydedilemedi.`

},{

type: 'danger'

});

}

});

}

function restoreBackup() {

fetch(`/restorebackup`,{credentials: 'include'})

.then(response => {

if(response.status < 400) {

$.notify({

message: `Yedekden geri dönüldü.`

},{

type: 'success'

});

} else {

$.notify({

message: `Yedekden geri dönülemedi.`

},{

type: 'danger'

});

}

});

}

/\*PDF CIKTILARI\*/

function staffToPDF() {

let staffs = document.getElementById('staffTable');

staffs.style.fontFamily = 'Helvetica,Arial,sans-serif';

html2pdf(

staffs,

{

margin:4,

filename:'Personeller.pdf',

html2canvas:{ dpi: 192 },

jsPDF:{ orientation: 'landscape' }

}

);

staffs.style.fontFamily = '';

}

function studentsToPDF() {

let students = document.getElementById('studentTable');

students.style.fontFamily = 'Helvetica,Arial,sans-serif';

html2pdf(

students,

{

margin:4,

filename:`${classToList.value}.pdf`,

html2canvas:{ dpi: 192 },

}

);

students.style.fontFamily = '';

}

function usersToPDF() {

let users = document.getElementById('usersTable');

users.style.fontFamily = 'Helvetica,Arial,sans-serif';

html2pdf(

users,

{

margin:4,

filename:`Kullanıcılar.pdf`,

html2canvas:{ dpi: 192 },

}

);

users.style.fontFamily = '';

}

function busesToPDF() {

let buses = document.getElementById('busesTable');

buses.style.fontFamily = 'Helvetica,Arial,sans-serif';

html2pdf(

buses,

{

margin:4,

filename:`Servisler.pdf`,

html2canvas:{ dpi: 192 },

}

);

buses.style.fontFamily = '';

}

/\*Cikis Yapma\*/

function logout() {

document.cookie = 'username=;expires=Thu, 01 Jan 1970 00:00:01 GMT;';

location.reload();

}

let myChart = new Chart(ctx, {

type: 'line',

data: {

labels: ['Ocak', 'Şubat', 'Mart', 'Nisan', 'Mayıs', 'Haziran', 'Temmuz', 'Ağustos', 'Eylül', 'Ekim', 'Kasım', 'Aralık'],

datasets: [{

label: '# Giderler',

data: [\_sum,\_sum,\_sum,\_sum,\_sum,\_sum,\_sum,\_sum,\_sum,\_sum,\_sum,\_sum],

backgroundColor: 'rgba(211,47,47,.5)',

borderColor: 'rgba(211,47,47,.5)',

fill:false

},

{

label: '# Gelirler',

data: [55000,65000,76000,64000,59000,44000,37000,45000,58000,62000,64000,69000],

backgroundColor: 'rgba(67,160,71,.5)',

borderColor: 'rgba(67,160,71,.5)',

fill:false

}]

},

options: {

scales: {

yAxes: [{

ticks: { beginAtZero:true }

}]

}

}

});

function importBus() {

let file = document.querySelector('input[type=file]').files[0];

let reader = new FileReader();

reader.readAsText(file, "UTF-8");

reader.onload = (evt) => {

data = JSON.parse(evt.target.result);

return fetch('/importbus', {

method: 'POST',

credentials: 'include',

headers: {

'Accept': 'application/json',

'Content-Type': 'application/json'

},

body: JSON.stringify(data)

})

.then(response => {

$('#addBus').modal('hide');

if(response.status < 400) {

$.notify({

message: `Veriler kaydedildi.`

},{

type: 'success'

});

} else {

$.notify({

message: `Veriler kaydedilemedi! HATA:(${response.status})`

},{

type: 'danger'

});

}

})

.catch(err => console.error(err))

}

console.log(FileReader.result);

}

**4.1.1.Router**

/\*KUTUPHANELER\*/

const router = require('express').Router();

const sql = require('mssql');

/\*FONKSIYONLAR\*/

const login = require('./functions/login');

const pay = require('./functions/pay');

const importBus = require('./functions/importBus');

const addBus = require('./functions/addBus');

const addStaff = require('./functions/addStaff');

const addStudent = require('./functions/addStudent');

const addStuff = require('./functions/addStuff');

const addUser = require('./functions/addUser');

const deleteBus = require('./functions/deleteBus');

const deleteStaff = require('./functions/deleteStaff');

const deleteStudent = require('./functions/deleteStudent');

const deleteStuff = require('./functions/deleteStuff');

const deleteUser = require('./functions/deleteUser');

const listBuses = require('./functions/listBuses');

const listInventory = require('./functions/listInventory');

const listStaffs = require('./functions/listStaffs');

const listStudents = require('./functions/listStudents');

const listUsers = require('./functions/listUsers');

const listOldPays = require('./functions/listOldPays');

const listPaid = require('./functions/listPaid');

const listNotPaid = require('./functions/listNotPaid');

const listNextPays = require('./functions/listNextPays');

const getBackup = require('./functions/getBackup');

const restoreBackup = require('./functions/restoreBackup');

/\*Anasayfayi getirme\*/

router.get('/', async (req, res) => {

try {

let paySum = 0;

const mainPageData = await sql.query`exec sp\_anasayfa`;

const \_data = await sql.query`exec sp\_taksit\_toplam`;

mainPageData.recordsets[1][0].forEach(per => {

paySum += per.per\_maas;

});

res.render('index', {

Admin:req.cookies.username == 'admin',

Classes:mainPageData.recordsets[0][0],

Staffs:mainPageData.recordsets[1][0],

SchoolBuses:mainPageData.recordsets[2][0],

StaffSum: paySum,

\_sum:\_data.recordsets[0][0],

helpers: {

ifEquals: function(arg1, arg2, options) {

return (arg1 == arg2) ? options.fn(this) : options.inverse(this);

},

json: function(context) {

return JSON.stringify(context);

}

}

});

} catch (err) {

console.error(err);

}

});

/\*API ISTEKLERINI YONLENDIRME\*/

/\*Login Sayfasini getirme\*/

router.get('/auth', async(req, res) => {

res.render('login');

});

/\*Login islemi\*/

router.post('/auth', async(req, res) => {

login.attempt(req.body.username,req.body.password)

.then(data => res.status(data.status).json({username:req.body.username}))

.catch(err => console.error(err));

});

/\*Kullanicilari getirme\*/

router.get('/user', async(req, res) => {

listUsers.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Kullanici ekleme\*/

router.post('/user', (req, res) => {

addUser.attempt(req.body)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Kullanici silme\*/

router.delete('/user/:tc', (req, res) => {

deleteUser.attempt(req.params.tc)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Personelleri listeleme\*/

router.get('/staff', (req, res) => {

listStaffs.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Personel ekleme\*/

router.post('/staff', (req, res) => {

addStaff.attempt(req.body)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Personel silme\*/

router.delete('/staff/:tc', (req, res) => {

deleteStaff.attempt(req.params.tc)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Envanteri listeleme\*/

router.get('/inventory', (req, res) => {

listInventory.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Esya ekleme\*/

router.post('/inventory', (req, res) => {

addStuff.attempt(req.body)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Esya silme\*/

router.delete('/inventory/:no', (req, res) => {

deleteStuff.attempt(req.params.no)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Sinifa gore ogrencileri listele\*/

router.get('/student/:class', (req, res) => {

listStudents.attempt(req.params.class)

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Ogrenci ekleme\*/

router.post('/student', (req, res) => {

addStudent.attempt(req.body)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Ogrenci silme\*/

router.delete('/student/:tc', (req, res) => {

deleteStudent.attempt(req.params.tc)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Servisleri listeleme\*/

router.get('/bus', (req, res) => {

listBuses.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Servis ekleme\*/

router.post('/bus', (req, res) => {

addBus.attempt(req.body)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Servis silme\*/

router.delete('/bus/:plate', (req, res) => {

deleteBus.attempt(req.params.plate)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Servis import\*/

router.post('/importbus', (req, res) => {

importBus.attempt(req.body)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Eski taksitleri listeleme\*/

router.get('/oldpays', (req, res) => {

listOldPays.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Bu ayki odenen taksitleri listeleme\*/

router.get('/newpaid', (req, res) => {

listPaid.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Bu aki odenmemis taksitleri listeleme\*/

router.get('/newpays', (req, res) => {

listNotPaid.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

/\*Gelecek taksitleri listeleme\*/

router.get('/nextpays', (req, res) => {

listNextPays.attempt()

.then(data => res.status(200).json(data))

.catch(err => console.error(err));

});

router.patch('/pay/:tc/:date', (req, res) => {

pay.attempt(req.params.tc, req.params.date)

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Yedek alma\*/

router.get('/getbackup', (req, res) => {

getBackup.attempt()

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

/\*Yedekten geri yükleme\*/

router.get('/restorebackup', (req, res) => {

restoreBackup.attempt()

.then(data => res.status(data.status).send())

.catch(err => console.error(err));

});

**5.KAYNAKLAR**

[2] VTYS dersi notları

[2] Adar, İ., SQL *Server,* abaküs, Nisan 2016

[3] Seker, S., “Bilgisayar kavramları”, Haziran 2017. <http://bilgisayarkavramlari.sadievrenseker.com/category/veri-tabani-database/>

[4] w3scool.com. <https://www.w3schools.com/sql/>

[5] Yücedağ, M.,” Murat Yücedağ Youtube Sayfası SQL Dersleri”, Şubat 2017. <https://youtu.be/9ai0Rc-4LKY?list=PLKnjBHu2xXNP6Qa6u8GLawPnzo1brHZPP/>

[6] Mastar, M., “muhammedmastar.com”, Ocak2017. <http://www.muhammedmastar.com/csharp-dersleri>

[7] Yücedağ, M.,” Murat Yücedağ Youtube C# Dersleri”, Ağustos 2017. <https://youtu.be/9ai0Rc-4LKY?list=PLKnjBHu2xXNP6Qa6u8GLawPnzo1brHZPP/>

[8] Cansever, A., “ahmetcansever.com”, Mart 2017. http://www.ahmetcansever.com/programlama/c-sql-server-veritabani-baglantisi-select-insert-update-delete/