



T.C

İSTANBUL AREL ÜNİVERSİTESİ

MESLEK YÜKSEK OKULU

**Mesleki Proje Dersi için Web Tabanlı Proje Seçim Otomasyon
Sistemi (ASP.NET MVC)+ Mobil Uygulama**

Fatih ŞENTÜRK

Ahmet TÜRKMEN

Yunus ŞEKER

BİLGİSAYAR TEKNOLOJİLERİ BÖLÜMÜ

BİLGİSAYAR PROGRAMCILIĞI PROGRAMI

MESLEKİ PROJE REPORU

DANIŞMAN

Öğr.Gör.Ebru İDMAN

İSTANBUL 2020

T.C.

İSTANBUL AREL ÜNİVERSİTESİ

MESLEK YÜKSEK OKULU

**Mesleki Proje Dersi için Web Tabanlı Proje Seçim Otomasyon
Sistemi (ASP.NET MVC)+ Mobil Uygulama**

Fatih ŞENTÜRK - 19103022

Ahmet TÜRKMEN - 191030036

Yunus ŞEKER – 191030010

BİLGİSAYAR TEKNOLOJİLERİ BÖLÜMÜ

BİLGİSAYAR PROGRAMCILIĞI PROGRAMI

MESLEKİ PROJE RAPORU

DANIŞMAN

Öğr.Gör.Ebru İDMAN

İSTANBUL 2020

ÖNSÖZ

İnsanlık; bilimin, teknolojinin atası, tanrısı. Yüzbinlerce yıldır var olan ve varlığını devam ettiren bizler. Her dem de daha kolayını elde etmek için, zaman için uğraşlar vermişizdir. Var olan ihtiyaçtan doğmuştur felsefesi büyük ölçüde doğrudur. İnsanlığın tüm keşifleri icatları da bir nevi bundan kaynaklı değil midir? ve açıkçası her zaman daha kolayını tercih etme isteği bizleri pek terk edecek gibi görünmüyor. 21. Yüzyıl teknolojinin, bilimin yükselişi ve altın çağı artık teknoloji bütünüyle hayatımızın bir parçası ve sıkı bir şekilde bağlı ve bu bağ gittikçe sıkılaşmak da. Teknolojinin daha da ilerlemesiyle birlikte farklı dallarda daha çeşitli ihtiyaçlarımız doğmaya başladı. Bu tez de ise ASP.NET MVC ile geliştirdiğimiz Mesleki Proje Seçim Otomasyonunu anlatmaya çalıştık. Teknolojinin ve bilimin ışığında sonsuza dek ilerlemek ümidiyle...

İÇİNDEKİLER

ÖNSÖZ	i
İÇİNDEKİLER	ii
KISALTMALAR	v
BÖLÜM 1. GİRİŞ VE AMAÇ	1
BÖLÜM 2. GENEL BİLGİLER	3
2.1 Literatür Çalışması	3
BÖLÜM 3. ÇALIŞMALAR.....	6
3.1 Veritabanı Tasarımı.....	6
3.1.1 Tablolar Arası İlişkiler	6
3.2 WEB PROGRAMLAMA	7
3.2.1 Öğrenci.....	10
3.2.1.1 Öğrenci Menü	11
3.2.1.2 Profil Düzenle.....	11
3.2.1.3 Proje Seç	12
3.2.1.4 Öğrenci Akademisyenler	14
3.2.1.5 Öğrenci Bölümler.....	14
3.2.1.6 Çıkış Yap	14

3.2.2 Akademisyen	14
3.2.2.1 Profil Düzenleme	14
3.2.2.2 Akademisyen Projelerim	15
3.2.2.2.1 Akademisyen Proje Ekle.....	16
3.2.2.2.2 Akademisyen Proje Güncelle	18
3.2.2.2.3 Akademisyen Proje Sil.....	18
3.2.2.3 Çıkış Yap	19
3.2.3 Admin	20
3.2.3.1 Admin Öğrenciler.....	21
3.2.3.1.1 Admin Öğrenci Ekle	21
3.2.3.1.2 Admin Öğrenci Güncelle	23
3.2.3.1.3 Admin Öğrenci Sil	24
3.2.3.2 Admin Bölümler.....	24
3.2.3.2.1 Admin Bölüm Ekle	25
3.2.3.2.2 Admin Bölüm Güncelle	26
3.2.3.2.3 Admin Bölüm Sil	27
3.2.3.3 Admin Akademisyenler	27
3.2.3.3.1 Admin Akademisyen Ekle.....	28
3.2.3.3.2 Admin Akademisyen Güncelle	29
3.2.3.3.3 Admin Akademisyen Sil.....	29

3.2.3.4 Admin Projeler	30
3.2.3.4.1 Admin Proje Ekle	31
3.2.3.4.2 Admin Proje Güncelleme	32
3.2.3.4.3 Admin Proje Sil	33
3.3 MOBİL UYGULAMA	35
BÖLÜM 4. SONUÇLAR	37
4.1 PROGRAMLAMANIN SONUÇLARI	37
KAYNAKLAR	39
EKLER	41

KISALTMALAR

ASP: Active Server Page

MVC: Model, View, Controller

SQL: Structured Query Language

ADO: ActiveX Data Objects

MAX: Maximum

ID: Identification (Numara)

URL: Uniform Resource Loader

BÖLÜM 1. GİRİŞ VE AMAÇ

Mesleki proje seçim otomasyonu adlı projemizde öncelikli olarak amacımız, mesleki proje seçiminde yaşanan sorunları, aksaklıkları gidermek ve kullanıcıyı yormayan basit ve anlaşılır ara yüze sahip bir otomasyon geliştirmek. Öğrenciler projelerini seçerken akıllarında hiçbir sorun kalmaması karışıklık yaşanmaması için proje açıklaması, projenin gereksinimleri gibi birçok bilgiye yer verilmesini hedefledik. Projemizi oluştururken, öncelikle farklı otomasyon sistemlerine bakıp nasıl işlediği ve arka planda nasıl bir yapı olduğunu öğrenmeye çalıştık. Projemizin web ayağında ASP.NET MVC Framework kullanıldı. Otomasyonu model, view ve controller rollerine bölerek geliştirdik. Projenin tasarım kısmında bootstrap frameworkü kullanıldı. Kullanıcının ilk karşılaşacağı ara yüz de akademisyen girişi öğrenci girişi ve öğrenci kayıt gibi kısımlar böldük kısımda kayıtlı olmayan kullanıcılar kayıt olabilecek kayıtlı olanlar ise giriş yapabilecektir. Kullanıcı kayıt olduğunda verileri veritabanına alınacaktır ve böylelikle sorun yaşamadan sağlıklı bir şekilde otomasyona tekrar giriş yapabilecektir. Kullanıcı giriş yaptıktan sonra karşılaşacağı ara yüzde çeşitli bilgilerini görüntüleyebilecek ama burada asıl amaç proje seçimi olacaktır. Proje seçim bölümünden projeleri, projenin hangi akademisyene ait olduğu, istenilenler, proje adı, proje açıklaması, içinde kullanılması gereken teknolojiler, araçlar gibi çeşitli bilgileri görüntüleyebilecektir. Kullanıcı buradan bir proje seçtikten sonra seçtiği proje kapanır ve o proje bir daha seçilemez ve kullanıcı da bir projeye sahipken başka proje alamaz. Ayrıca seçilen projeleri farklı kullanıcıların istekte bulunmaması için projeyi seçtiği zaman bu proje seçilmiştir gibi bir hata ile farklı projelere yöneltilecektir. Bununla birlikte otomasyonda genel ayarların yapıldığı, akademisyen ekleme-çıkarma, öğrenci ekleme-çıkarma, proje ismi değiştirme, açıklamaları güncelleme gibi ayarların yapıldığı sisteme müdahale gereken durumlarda sisteme müdahale edebilecek bir admin paneli de olacaktır. Admin paneli ile birlikte sistem daha sağlıklı ve etkin kullanılacaktır. Öncesine taslak oluşturuldu ve sitenin hangi sayfasının nasıl görüneceği kararlaştırıldı. Sonra grup içinde kabul ettiğimiz taslağı html ve css kodlar ile sitemizin görseli oluşturuldu.

Controller bölümünde her sayfanın ayrı ayrı controller oluşturup c# ile yazılan yerler olan Kayıt, Giriş yap, proje vs. bu tip fonksiyonları kodladıktan sonra SQL veri tabanı bağlantısı yapıldı.

SQL üzerinden oluşturduğumuz veritabanımızı projemize model kısmından bağlayacağız. SQL kodları ile akademisyenlerimizin bilgileri, öğrencilerin bilgileri ve projeleri vs. tutmayı planlıyoruz.

Mobil teknolojinin yayılmasıyla ve çok fazla kullanılması nedeniyle projemizin mobil ayağı da olacaktır. Mobil uygulamamız da web otomasyonumuz ile entegre olacak ve tamamıyla yukarda web otomasyonunda hedeflenenler mobilde de olacaktır. Mobil programlamada ise android işletim sisteminde çalışan bir otomasyon olacaktır. Web sitemizin her şeyini bitirdikten sonra mobil kısmına geçeceğiz. Mobil de ise android webview kullanılarak geliştirme yapılacaktır. Web sitemizi mobile entegre ederken siteye girecek kullanıcıların cihazlarının ekran boyutu farklı farklı olacağını göz önünde bulundurarak tasarımı daha esnek tutmayı ve karışık arka plandan kaçınmayı hedefliyoruz. Daha iyi performans verebilmek adına kullanıcıların çoğunlukla tıkladıkları veya tıklayabilecekleri kısımları menü bar açarak menüye eklemeyi hedefliyoruz.. sitemizin mobil bölümünde bunları göz önünde bulundurarak daha akıcı olacak ve kullanıcıları sıkmayacak şekilde yapıldı.

BÖLÜM 2. GENEL BİLGİLER

2.1 Literatür Çalışması

Uğur TALAŞ (2019) ÖĞRENCİ BİLGİ SİSTEMİ DERS SEÇME MODÜLÜ ÖRNEĞİ

Bilecik Şeyh Edebali Üniversitesinin kullanmış olduğu öğrenci bilgi sistemi yazılımı incelenip, yazılım mimarisinde kullanılan sorgulama teknolojilerinde ve sunucu yapılandırmasında geliştirmeler yapılarak bu problemin önüne geçilmiştir. Yeni geliştirilen sistemde ASP.Net mimarisi yerine MVC mimarisi kullanılmıştır. Bu mimariyle birlikte tasarım kısmında Bootstrap kullanılmıştır. Tasarım farklı boyutlardaki ekranlarda uyumlu çalışması sağlanmıştır. Eski sistem incelendiğinde birçok hesaplama C#'ta, yani sunucu tarafında yapıyorken sunucuya binen yükü istemcilere taşımak amacıyla hesaplamaların birçoğu JavaScript'te alınmıştır. Bu sayede yapılacak hesaplamaların her birini sunucu değil öğrencinin kullandığı bilgisayarın yapması sağlanmıştır .

Mustafa Zahit GÜRBÜZ, Onur AKÇİN (2020) – Optimizasyon Otomasyonu için Cplex Çözücüsünün Web Tabanlı Sunucu Üzerinden Çalıştırılması

Bu çalışmada, web arayüzünden girilen parametreler ile IBM ILOG CPLEX' e bağlanıp sonuçlar gösterilmiştir. Uygulama olarak optimizasyon problemlerinde bilinen reklam örneği kullanılmıştır. Günümüzde reklamcılık konusunda internet ve televizyon reklamcılığı ön plana çıkmaktadır. Uygulamada, forma üreten bir şirketin televizyonlara spot reklamlar vermesi ele alınmıştır. . Uygulama masaüstü ve web uygulamalarından oluşmaktadır. Bu uygulamalar Visual Studio 2017 'de C# programlama dili kullanarak yapılmıştır. Veritabanı olarak SQL Server 2012 kullanılmıştır. Masaüstü uygulamasının görevi, görev zamanlayıcı ile her dakika çalışıp, web uygulamasından gelen reklam parametrelerini SQL Server tablolarından okumaktır. CPLEX bağlantı kodlarının karmaşık yapısı ve web uyumlu olmamasından dolayı bağlantı kodları masaüstü programda yazılmıştır. Web uygulaması ise aynı şekilde visual studio geliştirme aracı olan ASP.NET MVC 4 ile yazılmıştır.

Muammer AKÇAY, Ömer KASIM , Zekiye TAŞDELEN (2020) - ASP.NET Ve MVC Temelli Esnek (Responsive) Web Uygulaması

Web site tasarımında kullanılan ve Microsoft tarafından geliştirilen C sharp, ASP, MVC gibi teknolojilerin gelişimi E-ticaret sitelerinin daha üst seviyede güvenli ve daha rahat tasarlanıp kodlanmasını sağlamaktadır. Bu çalışmada geliştirilen dinamik web içeriklerinin kullanılmasıyla MVC teknolojisini temel alan bir E-ticaret sayfası geliştirilmiştir. Model, görünüm ve kontrol bölümlerinin ayrı ayrı gerçekleştirilmesi ile ön taraf (Frontend) ve arka tarafa (Backend) yapılar birbirinden ayrı ayrı olarak geliştirilmiştir. Ön taraf bölümü ASP kaynak kodları ile geliştirilen çalışmada arka taraf bölümünde kontrol ve veri tabanı yer almaktadır.

Cihat ERDOĞAN, Ahmet SAYGILI (2015) - AKILLI DERS YÖNETİM SİSTEMİ İLE PROGRAMLAMA ÖDEVLERİ İÇİN İNTİHAL TESPİTİ UYGULAMASI

Bu projenin amacı, hiyerarşik kümeleme yöntemlerini kullanarak ödev değerlendirme sürecine katkıda bulunacak bir makine öğrenmesi sistemini geliştirmektir. Bu uygulamada öğretim üyesi ders notlarını, uygulama notlarını ve ödevleri internet üzerinden düzenleyebilmektedir. Öğrencilerde öğretim üyesinin verdiği notları görebilmelerin yanı sıra kendine ait ödevi bu sistem üzerinden dersin öğretim üyesine gönderebilmektedir. Bu projede öğretim üyesi için orijinal ödev kontrolünü kolayca sağlayabileceği bir sistem tasarlanmıştır. Böylece, bu uygulama üzerinden verilen ödevlerin öğrencileri orijinal ödev hazırlamaya teşvik edecektir.

Atilla ERGÜZEN , Halil İbrahim LÜY (2017) - Maden Ocaklarını Gerçek Zamanlı İzleme ve Görüntüleme Yazılım Uygulaması

Tasarlanan sistemin yazılım uygulaması yapılmış ve donanımsal hiç bir cihaz kullanılmamıştır. Sistem gerekli olan donanım verilerini sanal olarak oluşturulan rasgele veri kümesinden almıştır. Bu yazılım uygulaması her tipten kullanıcının kullanıp, anlayabileceği şekilde basit olarak tasarlanmıştır. Kullanıcılara web ara yüzü üzerinden maden için yeni hücreler ekleyip silme imkanı verilmiştir. Bu sayede kullanıcılar maden haritalarını istedikleri şekilde oluşturabilmektedirler. Aynı zamanda yazılım uygulaması sayesinde kullanıcılar tasarladıkları maden üzerinden madencilerin konumunu anlık olarak 3 boyutlu harita üzerinde görebilmektedirler.

Ahmet Gürkan YÜKSEK - Halil ARSLAN - Yavuz TÜRKEY - Oğuz KAYNAR - Yunis TORUN (2016) : Çevik Yazılım Geliştirme Yaklaşımı İle Beden Eğitimi Özel Yetenek Sınav Otomasyon Sisteminin Tasarımı

Bu çalışma da MVC ile beden eğitimi özel yetenek sınavları için BT teknolojileri kullanılarak bütünleşik bir Sınav sistemi önerilmiştir. Sistem yazılım ve donanım olmak üzere iki bölümden oluşmaktadır. Sistemin donanım kısmı, sporcuların test sahasındaki performanslarını sensörler yardımıyla toplamakta, daha sonra toplanan bu verileri kablosuz haberleşme yardımıyla yazılım sistemine aktarmaktadır. Veri toplama ve aktarma işlemleri mikro denetleyiciye sahip bir elektronik kart yardımıyla gerçekleştirilmektedir. Sistemin yazılım kısmı ise Masaüstü ve Web platformlarında gerçekleştirilen iki yazılım modülünden oluşmaktadır.

Hamdi DEMİREL - Barış ÖZKAN (2015) : Üç Katmanlı Nesne-İlişkisel Eşleme Mimarisi İçin Otomatik Fonksiyonel Büyüklük Ölçümü

Fonksiyonel Büyüklük Ölçümü'nde, çeşitli fonksiyonel büyüklük ölçüm metodları kullanılarak objektiflik ve tutarlılığın artırılması için standartlaşmaya ihtiyaç duyulmaktadır. Bazı mimari stiller için kullanımda olan çeşitli standardizasyon tanımları yapılmış olsa da, objektiflik ve tutarlılığı artırmak için hala yapılabilecek iyileştirmeler bulunmaktadır. Bu iyileştirmeler, otomasyon methodu kullanılarak ölçümcü hatalarının, zaman/efor miktarının ve yazılım geliştirme ortamına bağımlılığın azaltılması ile sağlanabilir. Bu çalışmada; Üç-Katmanlı Nesne İlişkisel Eşleme Mimarisi kullanan iş yazılımları için bir yaklaşım oluşturulmuş, prototip bir araç tasarlanmış ve bu araç yardımı ile bulunan sonuçlar, yaklaşımın doğrulanması için deneysel bir çalışma ile sunulmuştur.

Adnan KAVAK and A. Burak İNNER (2018) : ALTHIS: Uçtan Uca Bütünleşik Bir Uzaktan Hasta Takip Sistemi Tasarımı ve Diyabetik Hastalar için Durum Çalışması

Bu çalışmada, hastaların vital ölçüm değerlerinin gerçek zamanlı olarak hastane bilgi sistemine aktarılması, efektif hasta doktor etkileşiminin sağlanması, kritik durumlarda gerekli bildirimlerin yapılması (hasta acil çağırısı, hastaya veya yakınına anormal vital değer tespitinde sistem uyarısı, vb.) amacıyla geliştirdiğimiz uçtan uca uzaktan hasta takip sisteminin (ALTHIS) tasarımı, geliştirilmesi ve bileşenleri açıklanmaktadır. web programlama için Visual Studio MVC, veri tabanı için Microsoft SQL Server kullanılmıştır.

BÖLÜM 3. ÇALIŞMALAR

3.1 Veritabanı Tasarımı

Projeye ilk olarak veritabanı tasarımı ile başlandı. veritabanı MSSQL server üzerinden geliştirildi. Veritabanının da otomasyon adında (adını bilmiyoz) oluşturuldu ve bu veritabanına Admin, Akademisyenler, Bölümler, Öğrenciler ve Projeler adında beş adet tablo eklendi. Öncelikle Admin tablosunda id, kullanıcı adı , şifre kolonları eklendi ardından id primary key olarak ayarlandı ve diğer sütunlar ise nvarchar(max) olarak belirlendi EK-Şekil 3.1’de gösterildiği gibi. Akademisyen tablosuna id, ad, soyad, kullanıcı adı, şifre bölüm id kolonları eklendi. Bu eklemeyen sonra akademisyen id not null, primary key ve int değer olarak ayarlandı. Diğer sütunlarda nvarchar(max) ve not null olarak belirlendi EK-Şekil 3.2. Bölümler tablosunda id ve ad kolonları eklendi id kolonu not null, int ve primary key olarak ayarlandı. Ad kolonu ise not null olarak ayarlandı EK-Şekil 3.3. Öğrenciler tablosunda id, ad, soyad, kullanıcı adı, şifre, bölüm id ve proje id sütunları eklendi. Id sütunu not null ve primary key olarak ayarlandı, diğer sütunlarda not null olarak belirlendi EK-Şekil 3.4. Projeler tablosuna id, proje adı, proje tanımı, bölüm id ve akademisyen id sütunları eklendi. Proje id sütünü primary key olarak ayarlandı diğer sütunlar ise not null olarak belirlendi EK-Şekil 3.5.

3.1.1 Tablolar Arası İlişkiler

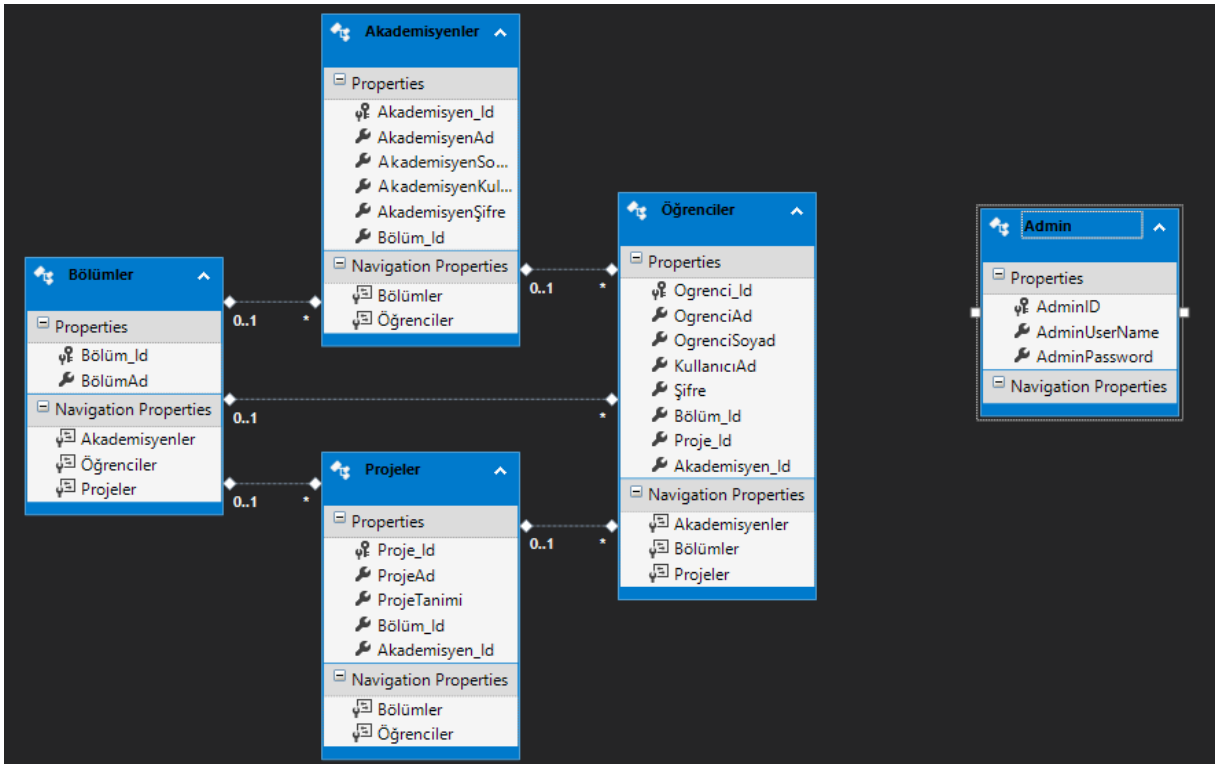
Oluşturulan tablolar birbiri ile veri alışverişinde bulunduğundan dolayı otomasyonun doğru şekilde çalışabilmesi için tablolar birbirleri ile ilişkili olmak zorundadır. Bundan dolayı MSSQL server üzerinden database diagram adlı klasörün altında diyagram oluşturulup, ilişkili olması gereken kolonlar ilişkilendirilmiştir. Kurulan ilişkiler gerekli sorgular ile test edilip doğru şekilde çalışıp çalışmadıklarına bakılmıştır. EK - Şekil 3.6’da Use Case şeması,EK-Şekil 3.7’de Class şeması ve EK-Şekil 3.8’de E-R şeması gösterilmiştir. Görüldüğü üzere her tablonun Bölümler tablosu ile ilişkisi bulunmaktadır. Şekilde ki ilişki basitçe şudur, proje tablosu bölümler tablosu hariç diğer tablolarla ilişkilidir çünkü her akademisyen bir projeye sahip olabilir ve bu yüzden akademisyen tablosuna proje id kolonu verilir, her öğrenci bir projeye sahip olabilir ve bu yüzden projeler tablosuna proje id kolonu verilir.

3.2 WEB PROGRAMLAMA

Web otomasyonu ASP.NET MVC teknolojisi kullanılarak programlandı. ASP.NET web veya masaüstü uygulamaları geliştirmek için kullanılan bir çerçevedir. MVC ise özellikle web uygulaması geliştirmekte önemli yere sahip mimari desenlerden birisidir. EK-Şekil 3.9’da gösterildiği gibi MVC 3 katmandan oluşur bunlar model, view ve controllerdır. Model, uygulamada kullanılacak verilerin bulunduğu veritabanı ile ilgili bağlantıların yapıldığı ve classların oluşturulduğu katmandır. View, model içerisindeki verilerin görselleştirilmesinden sorumlu yani ara yüz tasarımının yapıldığı katmandır. Controller katmanı ise model ve view katmanı arasında köprü görevi görür programlamanın yapıldığı katmandır.

İlk olarak projemizin database bağlantısını ADO.Net ile gerçekleştirdik. Elle kod yazmadan mvc’nin size sunduğu otomatik ilerleyen bir sistem olan ADO.Net ile oluşturduğumuz databaseyi projemize ekledik. Şekil 3.1’de ki gibi

Şekil 3.1 Model



Otomasyon içi programlamaya ilk olarak home controlller adında bir controller oluşturularak başlanmıştır, ardından bu controller'ın index action'ına bağlı index adında bir view oluşturulmuştur. Şekil 3.2'de görüldüğü üzere bu view da akademisyen girişi, öğrenci girişi ve öğrenci kayıt gibi butonlar oluşturulmuştur ve bu butonların linklerine gitmeleri gereken viewlar tanımlanmıştır.

Şekil 3.2



Ardından Şekil 3.4'de görüldüğü üzere öğrencilerin kayıt olması için create controllerı oluşturulmuştur. Create controllerına bağlı 2 action yazılmıştır. İlk action da, kayıt kısmında bulunan bölüm seçim menüsü için SQL'den bölümler listendi, diğer action da ise kullanıcının girdiği bilgileri SQL komutları ile veritabanı'na kayıt işlemi gerçekleştirildi. Create actionına view açıldıktan sonra view içinde asp.net mvc de yer alan bir özellik olan textboxfor ile bu text box lara yazılan değerler otomasyon veritabanına kayıt edildi ve Şekil 3.3'de kayıt aşamasında girilen alanların boş bırakılmaması veya kullanıcı adı eksik hataları için FluentValidation kütüphanesi projeye eklendi. View kısmında EK-Şekil 3.10'de görüldüğü üzere gerekli validation komutları yazıldıktan sonra modelde bulunan otomasyon veritabanı altındaki EK -Şekil 3.11'de görülen öğrenciler classına sorgular yazıldı.

Şekil 3.3

Şekil 3.4

```
0 bayyuru
public ActionResult Create()
{
    DbOtomasyon entitiess = new DbOtomasyon();
    var bölümgetir = entitiess.Bölümler.ToList();
    ViewBag.bölümlist = new SelectList(bölümgetir, "Bölüm_Id", "BölümAd");

    return View();
}

[HttpPost]
0 bayyuru
public ActionResult Create(Öğrenciler ogr)
{
    DbOtomasyon entitiess = new DbOtomasyon();

    var bölümgetir = entitiess.Bölümler.ToList();
    ViewBag.bölümlist = new SelectList(bölümgetir, "Bölüm_Id", "BölümAd");

    if (ModelState.IsValid)
    {
        entitiess.Öğrenciler.Add(new Öğrenciler() { ÖğrenciAd=ogr.ÖğrenciAd, ÖğrenciSoyad=ogr.ÖğrenciSoyad, KullanıcıAd=ogr.KullanıcıAd, Şifre=ogr.Şifre, Bölüm_Id=ogr.Bölüm_Id, } );
        var Kullanici = entitiess.Öğrenciler.Where(a => a.KullanıcıAd == ogr.KullanıcıAd).FirstOrDefault();

        if(Kullanici != null)
        {
            Response.Write("<script lang='JavaScript'>alert('Kullanıcı Adı Başkasına Ait. Lütfen Tekrar Deneyiniz.');
```

Yukarıda ki Şekil 3.4’de gösterilen kod parçacığında ToList metoduyla veritabanında bölümler çekilmiştir. Create viewına bu bölümler dropdown olarak eklenmiştir. Oluşturulan kontrol yapısı ile view de ye alan text boxlara girilen değerler eğer kullanıcı adı aynı değilse kayıt edilmiştir. Kullanıcı adının aynı olması durumunda hata verilmesi ve tekrar değerlerin girilmesi sağlanmıştır. Kayıt olduktan sonra öğrenci return ile loginin viewine yönlendirme yapılmıştır. Tamam eklerde kalsın onlar kodu burada anlattım

Bu kodda projeye eklenen FluentValidation kütüphanesi ile öğrenci kayıt olurken belli kontrollerin yapılması sağlanmıştır. Modelin altında bulunan öğrenci classında boş girilmemesi gereken alanlara Required hata mesajları tanımlanarak o alanların boş geçildiği zaman hata mesajı verilmesi sağlanmıştır. MaxLength ve MinLength mesajları ile şifre ve kullanıcı adı gibi alanların yeterli uzunlukta olmadığı zaman hata mesajları verilmesi sağlanmıştır.

Create viewında ise EK-Şekil 3.10’da gösterilen @Html.ValidationMessageFor ile özelliklerin alanları belirtilerek ve hata mesajlarının renkleri de belirtilerek bu mesajların öğrenci classında istenilenler sağlanmadığı zamanlar viewda hata mesajları verilmesi sağlanmıştır.

3.2.1 Öğrenci

Öğrenci kaydının ardından login controllerı oluşturuldu bu controllerın içinde giriş yapmaya çalışan kullanıcıların bilgileri FirstOrDefault ile kontrol edildi. Ve giriş yapan kullanıcının anlık bilgilerini almak için kullanıcıya ait her bilgi için Şekil 3.5’de gösterilen Session oluşturuldu. Hatalı giriş yapıldığında hata mesajı verilmesi için kontrol yapısı kullanıldı. Akademisyen girişi için controller oluşturulduktan sonra login controllerın da yapılan işlemler akademisyen içinde yazılmıştır. Kullanıcı sisteme giriş yapmadan URL üzerinden siteye giriş yapmak isterse kullandığımız “Authorize” güvenlik kodları ile girdiği adrese gidemeden EK-Şekil 3.12’de gösterilen link ile bizim yönlendirdiğimiz adrese yani home view’ına yönlendirilecektir.

Şekil 3.5

```
[HttpPost]
0 bagvuru
public ActionResult Login(Öğrenciler ogrenci)
{
    var ogrencis = db.Öğrenciler.FirstOrDefault(x => x.KullanıcıAd == ogrenci.KullanıcıAd && x.Şifre == ogrenci.Şifre);

    if (ogrencis != null)
    {
        FormsAuthentication.SetAuthCookie(ogrencis.KullanıcıAd, false);

        Session["KullaniciAdi"] = ogrencis.OgrenciAd;
        Session["KullaniciSoyad"] = ogrencis.OgrenciSoyad;
        Session["KullaniciId"] = ogrencis.Ogrenci_Id;
        Session["BolumId"] = ogrencis.Bölüm_Id;
        Session["BolumAdi"] = ogrencis.Bölümler.BölümAd;
        if(ogrencis.Proje_Id >= 1)
        {
            Session["ProjeId"] = ogrencis.Proje_Id;
            Session["ProjeAdi"] = ogrencis.Projeler.ProjeAd;
        }

        return RedirectToAction("OgrenciProfil", "Profil");
    }
    else
    {
        ViewBag.Hata = "Kullanıcı adı veya Şifre Yanlış";
        return View();
    }
}

0 bagvuru
public ActionResult Logout()
{
    FormsAuthentication.SignOut();
    return RedirectToAction("Index", "Home");
}
```

Şekil 3.5’de gösterilen kod da öğrenci sisteme giriş yapmaya çalıştığı zaman FirstOrDefault ile giriş yapan öğrencinin bilgileri öğrencis değerine atanmıştır oluşturulan kontrol yapısı ile öğrencis değerinin boş olup olmadığı kontrol edilmiştir. Sessionlar ile giriş yapan kullanıcının bilgileri alınmıştır ve öncesinde oluşturulan kontrol yapısında öğrencis değeri null olunca ViewBag ile hata verilmesi sağlanmıştır. Logout acitoni ile de kullanıcının menüde yer alan çıkış yap butonu ile sistemden çıkması sağlanmıştır.

3.2.1.1 Öğrenci Menü

Öğrenci girişinin ardından kullanıcının karşılaşacağı sayfa için EK-Şekil 3.13’de gösterilen bootstrap ile bir menü layout oluşturuldu. Otomasyonun mobil uyumlu olması açısından oluşturduğumuz menüyü Slide menü olarak tasarladık çünkü öğrenci veya akademisyen sisteme mobil üzerinden giriş yaparsa içeriği rahat görebilmesi için fazla yer kaplamamış olacaktır. Öğrenci menüsü içerisinde Profil Düzenleme, Proje Seç, Akademisyenler, Bölümler ve Çıkış Yap butonları bulunmaktadır.

3.2.1.2 Profil Düzenle

Profil Düzenleme bölümünde, Şekil 3.7’de gösterildiği gibi birinci action da hangi öğrencinin bilgilerini getirmek istiyorsak, sisteme giriş yapan öğrenci id’si ile öğrenciler tablosundan o id’ye ait öğrencinin Şekil 3.6’te gösterildiği gibi bilgilerini getiriyoruz . Diğer action da ise seçilen öğrenci bilgilerini eğer değiştirmek istiyorsa yeni bilgileri ile eşitleyip değiştirmek istediği bilgiyi view da yazılan metin kutularına yenisini yazarak SQL’e kayıt ettirebiliyor.

Şekil 3.6

The image shows a web form titled "Profil Güncelle" (Update Profile). It contains four input fields, each with a label to its left: "Öğrenci Ad" (Student Name) with the value "Yılmaz", "Öğrenci Soyad" (Student Surname) with the value "Öztürk", "Öğrenci Şifre" (Student Password) with the value "123456789", and "Öğrenci ID" (Student ID) with the value "47". Below these fields is a yellow button labeled "Güncelle" (Update).

Şekil 3.7

```
0 başvuru
public class OgranciEditController : Controller
{
    DbOtomasyon entitiess = new DbOtomasyon();

    // GET: OgranciEdit
    [Authorize]
    0 başvuru
    public ActionResult ogranciEdit(int id)
    {
        var og = entitiess.Oğrenciler.Find(id);
        return View("ogranciEdit", og);
    }
    [HttpPost]
    0 başvuru
    public ActionResult ogranciEdit(Oğrenciler ogr)
    {
        var ogranci = entitiess.Oğrenciler.Find(ogr.Ogranci_Id);
        if (ogranci != null)
        {
            ogranci.OgranciAd = ogr.OgranciAd;
            ogranci.OgranciSoyad = ogr.OgranciSoyad;
            ogranci.Şifre = ogr.Şifre;
            entitiess.SaveChanges();
            return RedirectToAction("ogranciEdit");
        }
        else
        {
            ViewBag.Hata = "ÖĞRENCİ ID'Yİ DEĞİŞTİREMEZSİNİZ!!!";
            return View();
        }
    }
}
```

Yukarı da ki Şekil 3.7’de gösterilen kod da birinci actionresult parametrelili olarak tanımlanmıştır, tanımlanan id’yi Find komutu ile öğrenciyi buluyoruz. İkinci action result da ise textbox da yazılan id değeri find metodu ile bulunup öğrenci değişkenine atanmıştır. Oluşturulan karar yapısı ile id nin null olup olmadığı kontrol edilmiştir. İf kontrol yapısı içerisinde textboxlara girilen değerler savechanges metodu ile kayıt edilmiştir ve bu işlemin ardından ogranciedit viewı tekrar dönürülmüştür. Öğrenci id değerinin değiştirilmeye çalışılması durumunda viewbag ile hata mesajı verilmiştir.

3.2.1.3 Proje Seç

Öğrenci menüden Proje Seç bölümüne girdiğinde karşısına kendi bölümüne ait bütün projeler çıkmaktadır. Proje seç için açtığımız view’a EK-Şekil 3.14’de ki gibi foreach ile her bir proje için Şekil 3.8’de gösterilen proje numarası, adı, açıklaması ve Seç butonu tanımlandı

Şekil 3.8

Akademisyen ID	Projeler	Proje Tanımı	Proje ID	Seç
2	Cosmic veri tabanından veri toplama ve örüntü bulma, arayüz hazırlama:	Kansere neden olan mutasyonların konum bilgilerinin bulunduğu Cosmic sitesinden Gigabyte boyutunda verilerin indirilmesi ve gereklidir. Bu verilerin bilgisayarla analiz sonucunda gen-mutasyon-hastalık ilişkisine dair örüntüler yakalanması hedeflenmektedir.	1	Seç
1	Gen-Hastalık – İlaç etkileşimini yapan veri kaynaklarını birleştirme ve elde edilen verilerden yapıy öğrenme arayüz uygulaması geliştirme		2	Seç
1	DNA sekans veri tabanı dizinleme ve sorgulama yapan arayüzü geliştirme		3	Seç
1	Medikal sekans veya görüntülerde derin öğrenme uygulaması		4	Seç
1	miRNA - gen veri etkileşimi ve gen regülasyon ağı oluşturma		5	Seç

Şekil 3.9

```
0 bayıru  
public ActionResult Sec(int id)  
{  
  
    var prjsec = entitiess.Projeler.FirstOrDefault(x => x.Proje_Id == id);  
  
    int gelenKullaniciId = (int)Session["KullaniciId"];  
  
    var dbResult = entitiess.Öğrenciler.SingleOrDefault(Öğrenciler => Öğrenciler.Oğrenci_Id == gelenKullaniciId);  
    if (dbResult.Proje_Id == null)  
    {  
        if (dbResult != null) // Kayıt bulundu  
        {  
            dbResult.Proje_Id = id;  
            Response.Write("<script lang='JavaScript'>alert('Proje Başarıyla Seçildi');</script>");  
            entitiess.SaveChanges();  
        }  
    }  
    else  
    {  
        Response.Write("<script lang='JavaScript'>alert('Projeniz Varken Başka Proje Seçemezsiniz!!!');</script>");  
    }  
    return View(prjsec);  
}
```

Yukarı da ki Şekil 3.9’da Öncelikle Seç butonuna bastığında hangi projeyi seçtiğini anlamamız için id tanımlandı. Tanımlanmış olan id’ye seçilmek istenen projenin SQL’de kayıtlı olan id ile eşitledik. Daha sonra, önce ki öğrencilerin seçtiği projeyi başka bir öğrenci tekrar seçmemesi için kullandığımız butonu tıklanamaz hale getirmemiz gerekiyor, bunun için EK-Şekil 3.14’de ki Foreach’ e ButtonStatus adında boş bir string tanımladık. Daha sonra önceden seçilen projeleri bulmak için EK-Şekil 3.14’da gösterilen SQL sorgusu ile öğrenci tablosunda bulunan kayıtları çektik ve if içerisine çekilen öğrenciler arasından eğer proje id si dolu olan var ise o proje id’sini tanımladığımız ButtonStatus’ e disabled verdik ve Şekil 3.8’de ki gibi önceden seçilmiş projelerin butonlarını tıklanamaz hale getirdik.

Öğrenci porje seçiminde eğer önceden proje seçmiş ve başka bir proje seçmeye çalıştığında ise buna izin vermememiz gerekiyor bunun için Şekil 3.9’da bulunan action içine seçim işlemini yapmak isteyen öğrencinin id’sini loginde açtığımız session kullanıcı ID sini gelenkullaniciId adında bir değişkene atıyoruz, daha sonra bu değişkeni SQL sorgusu ile öğrenciler tablosunda var olup olmadığını buluyoruz. İf içerisinde gelenkullaniciId’nin bir projesi var ise şekilde gösterildiği gibi script kodları ile hata mesajı veriyoruz. Öğrencinin önceden projesi yoksa ve yeni bir proje seçiyorsa Proje seçimi başarılı şeklinde mesaj ile seçtiği projeyi veritabanı’na kayıt ettirildi.

3.2.1.4 Öğrenci Akademisyenler

Öğrenci menüden akademisyenler sayfasına girdiğinde öğrencinin bölüm id'si ile akademisyenler tablosunda bulunan akademisyenlerin bölüm id'si aynı olanları controller da listeleterek karşısına kendi bölümüne ait akademisyenler gözükecektir. Akademisyenler sayfasında akademisyenin okul numarası, ad, soyad ve e-posta bilgilerini görüntüleyebilir bu sayede istediği akademisyen ile iletişime geçebilmektedir. Site görseli EK-Şekil 3.15'de gösterilmiştir.

3.2.1.5 Öğrenci Bölümler

Bölümler controllerında ToList metoduyla veritabanında bulunan bölümler çekilmiştir bolumler acitonı üzerinden açılan viewde oluşturulan tabloya bölüm numaraları ve bölüm isimleri yazdırılmıştır. Site görseli EK-Şekil 3.16'de gösterilmiştir.

3.2.1.6 Çıkış Yap

Menüde çıkış yap kısmına basıldığı zaman çalışan LogOut action' da çalışan metodun amacı çok basittir. Çıkış yap kısmına bastığında EK-Şekil 3.17'te gösterildiği gibi kullanıcıyı Home controllerin içinde ki Index'e yani kullanıcı karşılayan ilk sayfaya atmaktadır.

3.2.2 Akademisyen

Akademisyen'in sisteme kayıt edilme işlemi öğrenciden farklı olarak kendi kendini kayıt edemiyor sadece Admin gerçekleştirebiliyor. Akademisyen'in menüsünde ise EK-Şekil 3.18'te gösterildiği gibi Profil Düzenleme, Projelerim ve Çıkış Yap butonu bulunmaktadır.

3.2.2.1 Profil Düzenleme

Öğrenci de olduğu gibi akademisyenin Profil ayar bölümü aynı şekilde işliyor. Profil Düzenleme bölümüne ait site görseli EK-Şekil 3.19'da gösterilmiştir.

Şekil 3.10

```
0 bagyuru
public class OgretmenEditController : Controller
{
    DbOtomasyon entitiess = new DbOtomasyon();

    // GET: OgretmenEdit
    [Authorize]
    0 bagyuru
    public ActionResult OgretmenEdit(int id)
    {
        var og = entitiess.Akademisyenler.Find(id);
        return View("OgretmenEdit", og);
    }

    [HttpPost]
    0 bagyuru
    public ActionResult OgretmenEdit(Akademisyenler akd)
    {
        var akademisyen = entitiess.Akademisyenler.Find(akd.Akademisyen_Id);
        if (akademisyen != null)
        {
            akademisyen.AkademisyenAd = akd.AkademisyenAd;
            akademisyen.AkademisyenSoyad = akd.AkademisyenSoyad;
            akademisyen.AkademisyenŞifre = akd.AkademisyenŞifre;
            entitiess.SaveChanges();
            return RedirectToAction("OgretmenEdit");
        }
        else
        {
            ViewBag.Hata = "ÖĞRENCİ ID'Yİ DEĞİŞTİREMEZSİNİZ!!!";
            return View();
        }
    }
}
```

Yukarı da ki Şekil 3.10'de gösterildiği gibi birinci action da hangi akademisyenin bilgilerini getirmek istiyorsak, sisteme giriş yapan akademisyenin id'si ile Akademisyenler tablosundan o id'ye ait akademisyenin bilgilerini getiriyoruz. Diğer action da ise seçilen akademisyen bilgilerini eğer değiştirmek istiyorsa yeni bilgileri ile eşitleyip değiştirmek istediği bilgiyi view da yazılan metin kutuları sayesinde akademisyen yenisini yazarak SQL'e kayıt ettirebiliyor.

3.2.2.2 Akademisyen Projelerim

Akademisyen Projelerim bölümüne girdiğinde ise karşısına kendisine ait olan projeler çıkmaktadır. Karşısına çıkan tabloda Proje Numarası,Proje Adı Proje Tanımı ve eğer seçilmiş ise öğrencinin adı, soyadı gösterilmektedir. Aynı zamanda tabloda 3 farklı buton bulunmaktadır, bunlar Şekil 3.11 gösterilen Proje Ekle, Proje Güncelle ve Sil butonlarıdır. Proje Ekle butonu sayfa da sadece bir adet bulunurken Güncelleme ve Silme butonları foreach ile listelediğimiz projelerin her birinde bulunur.

Şekil 3.11

Projelerim

Yeni Proje Ekle						
ID	Projelerim	Açıklamalar	Oğrenci Ad	Oğrenci Soyad	Sil	Güncelle
16	Akıllı tanımlama için mobil uygulama geliştirimi/ sistem tasarımı				Sil	Güncelle
17	Döner Kanat Hava Araçları için Gerçek Zamanlı Tel Algılama Sistemi				Sil	Güncelle
18	Dağıtık Fiber Optik Akustik Algılama Uygulamaları				Sil	Güncelle
19	Helikopter Sağlık ve Kullanım İzleme Sistemlerinde (HUMS) Kalman Filtrelerinin Uygulanması				Sil	Güncelle
20	Otonom Araçlarda Sensör ve Sensör Füzyonu Kullanılarak Yol Kenarındaki Şeritlerin Algılanması XX	TEST DENEME			Sil	Güncelle

3.2.2.2.1 Akademisyen Proje Ekle

Proje Ekle butonuna bastığında karşısına Şekil 3.12’de gösterildiği gibi iki adet TextBox ve iki adet ise DropDownList çıkmaktadır. TextBoxlar Proje Adı ve Proje Tanımı alanları içinde, DropDownList’ler ise projenin kayıtlı olacağı bölüm ve kayıt ettirmek isteyen akademisyenin akademisyenin Kullanıcı adı alanlarıdır.

Şekil 3.12



Proje Ekle

Proje Adı:

Proje Tanımı:

Projenin Bölümü:

Projenin Akademisyeni:

Proje Kaydet

Şekil 3.13

```

[Authorize]
0 başvuru
public ActionResult AkdProjeEkle()
{

    var akdgetir = db.Akademisyenler.ToList();
    ViewBag.akdlist = new SelectList(akdgetir, "Akademisyen_Id", "AkademisyenKullanıcı");

    var blmgetir = db.Bölümler.ToList();
    ViewBag.blmlist = new SelectList(blmgetir, "Bölüm_Id", "BölümAd");

    return View("AkdProjeEkle");
}

[HttpPost]
0 başvuru
public ActionResult AkdProjeEkle(Projeler prj)
{
    if (prj.Proje_Id == 0)
    {
        var SonKayit = db.Projeler.ToList().Last();

        prj.Proje_Id = SonKayit.Proje_Id + 1;

        db.Projeler.Add(prj);
    }
    else
    {
        var guncelle = db.Projeler.Find(prj.Proje_Id);
        if (guncelle == null)
        {
            return HttpNotFound();
        }
        guncelle.ProjeAd = prj.ProjeAd;
        guncelle.ProjeTanimi = prj.ProjeTanimi;
        guncelle.Bölüm_Id = prj.Bölüm_Id;
        guncelle.Akademisyen_Id = prj.Akademisyen_Id;
    }
    db.SaveChanges();

    return RedirectToAction("AkdProfil");
}

```

Şekil 3.13’de ki ilk action’da proje eklemek için ihtiyacımız olan akademisyen ve bölümleri dropdown’a getirmek ve bunun için akademisyen ile bölümler tablosunda ki verileri ViewBag’e kayıtlı ediyoruz. Sonra ki action da ise ekleme işlemi için proje id’yi if içerisinde 0 olup olmadığını anlıyoruz eğer 0 ise yeni bir kayıt olduğunu anlıyor ve yeni proje için önce projeler tablosunda bulunan en son kayıtlı olan projenin id’yi Last komutu ile alıyor +1 ekledikten sonra yeni kayıt edilmek istenen proje id’ye ekliyor, bu şekilde proje SQL’e başarılı bir şekilde kayıtlı ediliyor.

3.2.2.2 Akademisyen Proje Güncelle

Proje Güncelleme butonuna basıldığında karşısına Proje Ekle sayfasında bulunan alanların aynısı çıkacaktır fakat güncelleme yapacağı için bu sefer bu alanlar seçtiği projeye ait bilgiler ile otomatik doldurulmalıdır, bu doldurma işlemini ise EK-Şekil 3.20’de gösterildiği gibi öncelikle ProjeGüncelle adında bir action oluşturduktan sonra if içerisinde seçilen projenin id’nin null olup olmadığını buluyoruz ve eğer id’si null ise sayfa NotFound hatası ile kapanıyor, eğer null değil ise bu demektir ki var olan bir proje seçildi ve güncelleme olacaktır, anlayıp return ile Proje Ekle view’ına gönderiliyor. Gönderilen proje’nin bilgileri sayfada bulunan alanlara otomatik şekilde dolduruluyor. Akademisyen bu sayfa da bulunan Proje’nin adı, tanımı, akademisyeni ve bölümünü istediği şekilde değiştirebilmektedir.

3.2.2.3 Akademisyen Proje Sil

Proje Sil butonuna basıldığında, akademisyen kendi projelerinden istediğini silebilmektedir. Silme işlemi EK-Şekil 3.21’de görüldüğü üzere öncelikle id tanımlıyoruz, daha sonra tanımladığımız id’yi projeler tablosunda buluyoruz. If içerisinde seçilen proje id boş ise hata sayfası döndürüyor else içinde ise seçilen proje id’nin boş olmadığını anladıktan sonra seçilen proje id’yi SQL’den kalıcı bir şekilde siliyoruz.

Şekil 3.14’de ki sil butonuna silme işlemi yapılmak istenen projeyi öncesinden bir öğrenci seçmiş ise seçili olan projenin butonu tıklanamaz hale getiriliyor. Bunun asıl nedeni ise Öğrencinin seçmiş olduğu projeyi akademisyenin silememesinin asıl nedeni ise eğer akademisyen seçilmiş bir projeyi silerse ve o projeyi alan öğrencinin haberi olmaz ise öğrenci için geri dönüşü olmayan bir sorun oluşmuş olur.

Şekil 3.14

```
<table class="table">
  <tr class="thead-dark">
    <th scope="col"> ID </th>
    <th scope="col"> Projelerim </th>
    <th scope="col"> Açıklamalar </th>
    <th scope="col"> Öğrenci Adı </th>
    <th scope="col"> Öğrenci Soyadı </th>
    <th scope="col"> Sil </th>
    <th scope="col"> Güncelle </th>
  </tr>
  <tbody>
    <foreach (var prj in Model)>
      <tr>
        <td>@prj.Proje_Id</td>
        <td>@prj.ProjeAd</td>
        <td>@prj.ProjeTanımı</td>
        <td>@prj.OğrAd</td>
        <td>@prj.OğrSoyad</td>
        <td><a href="/Akademisyenler/AkdProjeSil/@prj.Proje_Id" class="btn btn-dark font-italic @ButtonStatus">Sil</a>
        <td><a href="/Akademisyenler/AkdProjeGüncelle/@prj.Proje_Id" class="btn btn-danger font-italic">Güncelle</a>
      </tr>
    </foreach>
  </tbody>
</table>
```

Bu işlem ise Şekil 3.14’de gösterilen öğrenci sayfasında olduğu gibi öncelikle ButtonStatus adında boş bir stringtanımlıyoruz. Kontrol adında bir değişken oluşturup bu değişkene SQL sorguları ile öğrenciler tablosunda kayıtlı olan öğrencilerden proje seçmiş olanları buluyoruz , daha sonra if içerisinde kontrol değişkeni boş değilse boş atadığımız string’e disabled atayarak o akademisyenin proje sil butonunu tıklanamaz hale getiriyoruz.

3.2.2.3 Çıkış Yap

Menüde çıkış yap kısmına basıldığı zaman çalışan LogOut action’ da çalışan metodun amacı çok basittir. Çıkış yap kısmına bastığında EK-Şekil 3.22’de gösterildiği gibi kullanıcıyı Home controllerin içinde ki Index’e yani kullanıcı karşılayan ilk sayfaya atmaktadır.

3.2.3 Admin

Adminin siteye giriş yapabilmesi için admin controllerı oluşturuldu. Şekil 3.15’de görüldüğü üzere Admin controllerı altında oluşturulan adminLogin acitonında FirstOrDefeult kullanılarak adminin siteye girişinde admin kullanıcı adı ve admin şifre kontrolü yapıldı, admin login görüntüsü EK-Şekil3.23’de gösterilmiştir. Kontrol yapısı ile bu değerler yanlış girildiği vakit sistemin hata mesajı döndürmesi sağlandı.Admin sayfasında; Öğrenciler, Akademisyenler, Projeler, Bölümler ve Çıkış Yap butonu menüye eklendi. Site görseli EK-Şekil 3.24’de gösteriştir.

Şekil 3.15

```
[HttpPost]
0 references
public ActionResult AdminLogin(Admin admin)
{
    var adminl = db.Admin.FirstOrDefault(a => a.AdminUserName == admin.AdminUserName && a.AdminPassword == admin.AdminPassword);

    if (adminl != null)
    {
        return RedirectToAction("AdminProfil", "Admin");
    }
    else
    {
        ViewBag.Hata = "Kullanıcı adı veya şifre Yanlış";
        return View();
    }
}
```

Yukarı da ki kod resminde gösterilen admin sistem de bulunan metin alanlarına girdiği ile veri tabanında ki girdiği kullanıcı adı ve şifresi eşleşiyor ise if içinde gösterilen AdminProfil sayfasına yönlendirilir. Eğer girdiği bilgiler veritabanının da ki bilgiler ile eşlemiyor ise hata mesahı vererek tekrardan girmesi istenir.

3.2.3.1 Admin Öğrenciler

Admin öğrenciler bölümüne girdiğinde ise karşısına okulda kayıtlı olan öğrenciler çıkmaktadır. Çıkan tabloda öğrenci numarası, öğrenci adı, soyadı, kullanıcı adı, şifre, bölüm ID ve proje ID gösterilmektedir. Aynı zamanda tabloda 3 farklı buton bulunmaktadır, bunlar Öğrenci Ekle, Öğrenci Güncelle ve Sil butonlarıdır. Öğrenci Ekle butonu sayfa da sadece bir adet bulunurken Güncelleme ve Silme butonları Şekil 3.16’da gösterilen öğrencilerin her birinde bulunur.

Şekil 3.16

Öğrenci Ekle							
ID	Ad	Soyad	Kullanıcı Adı	Şifre	Bölüm ID	Proje ID	Durum
16	Ahmet	Türkmen	123123	123123	3	105	Sil Güncelle
17	Tarik	Aslan	1910300	123456.	1	4	Sil Güncelle
18	ahmet	türkmen	asd	asd	4	119	Sil Güncelle
19	123	123	123	123	4	123	Sil Güncelle
21	asdasd	asdasd	asdasd	asdasd	3		Sil Güncelle

3.2.3.1.1 Admin Öğrenci Ekle

Admin, Öğrenci Ekle butonuna tıkladığı zaman Şekil 3.17’de gösterilen sayfada öğrenci adı, soyadı, kullanıcı adı, şifre ve bölümü alanlarını girecektir. Admin bu alanları istenilen şekilde doldurarak sisteme yeni bir öğrenci kaydı yapacaktır. Öğrenci kaydı yapabilmek için Şekil 3.18’de görüldüğü üzere öncelikle if içerisinde öğrenci id’nin 0 a eşit mi olduğunu buluyoruz. Öğrenci id 0 a eşit ise bu demektir ki sisteme yeni bir öğrenci eklenmek isteniyor, yeni öğrenci ekleneceğini anlayınca da Şekil 3.17’de ki site görselinde, boş alanları doldurarak girilen değerleri SQL’e kayıt ettiriliyor.

Şekil 3.17

Öğrenciler
Akademisyenler
Projeler
Bölümler
Çıkış Yap

Oğrenci Adı :

Oğrenci Soyadı :

Oğrenci Kull. Adı :

Oğrenci Şifre :

Oğrenci Bölümü :

Öğrenci Kaydet

Aşağıda ki Şekil 3.18’de ki kod resminde gösterildiği üzere ilk action da ekleme sayfasında ki dropdown listesi için sistemde kayıtlı olan bütün bölümleri getirdik çünkü kayıt edilen öğrenci istenilen bölüme kayıt olabilmesi için ekleme admin ekle butonuna bastığı anda sistem if else içerisine giriyor ve öğrenci id’ye bakıyor. Öğrenci id 0 ise yani sistemde öyle bir öğrenci yok ise Add komutu ile sisteme öğrenciyi ekliyor. Sistemde o id’ye sahip bir öğrenci var ise else içerisine girip öğrenci bilgilerini getiriyor eğer değiştirilirse veritabanının da ki bilgileri değiştirmektedir.

Şekil 3.18

```
0 bapıru  
public ActionResult AdminOğrenciEkle()  
{  
    var bölümgetir = db.Bölümler.ToList();  
    ViewBag.bölümlist = new SelectList(bölümgetir, "Bölüm_Id", "BölümAd");  
  
    return View("AdminOğrenciEkle");  
}  
  
[HttpPost]  
0 bapıru  
public ActionResult AdminOğrenciEkle(Öğrenciler ogr)  
{  
    if (ogr.Oğrenci_Id == 0)  
    {  
        db.Öğrenciler.Add(ogr);  
    }  
    else  
    {  
        var guncelle = db.Öğrenciler.Find(ogr.Oğrenci_Id);  
        if (guncelle == null)  
        {  
            return HttpNotFound();  
        }  
        guncelle.OğrenciAd = ogr.OğrenciAd;  
        guncelle.OğrenciSoyad = ogr.OğrenciSoyad;  
        guncelle.KullanıcıAd = ogr.KullanıcıAd;  
        guncelle.Şifre = ogr.Şifre;  
        guncelle.Bölüm_Id = ogr.Bölüm_Id;  
    }  
}
```

3.2.3.1.2 Admin Öğrenci Güncelle

Öğrenci güncelle butonu ile admin, öğrencilerin listelendiği ve sisteme kayıtlı tüm öğrencileri gördüğü, öğrenciler sayfasında, istediği öğrencinin bilgilerini belirlenen şartlar çerçevesinde güncelleyebilecektir. Admin herhangi bir öğrenciyi güncellemek isteyeceği karşısına öğrencinin sisteme kayıtlı olduğu tüm bilgileri gelecektir ve öğrenciye ait istediği bilgiyi güncelleyebilecektir. Tasarlanılan view sisteminde, 1 view de 2 iş yapılması planladı ve doğru bir şekilde sisteme geçirildi.

Şekil 3.19

```
0 bapuru
public ActionResult AdminOgrenciGuncelle(int id)
{
    var bölümgetir = db.Bölümler.ToList();
    ViewBag.bölümlist = new SelectList(bölümgetir, "Bölüm_Id", "BölümAd");

    var ogr = db.Öğrenciler.Find(id);
    if (ogr == null)
    {
        return HttpNotFound();
    }
    return View("AdminOgrenciEkle", ogr);
}
```

Yukarı da ki Şekil 3.19’da gösterilen öğrenci güncelleme controller’da önce işlem yapılan öğrencinin id’yi 0 a eşit olup olmadığı sorgulanıyor, eğer id 0 a eşit ise güncelleme işlemi yapılmadığı anlamına gelir ve böyle bir durumda sistem hata sayfası döndürür. Eğer butona basılan öğrencinin id’si 0 a eşit değilse bu demektir ki var olan bir öğrenci üzerinden işlem yapılmak isteniyor ve kaydı return ile öğrenci ekle view’ına göndererek işlemlere devam ediyor. Bunu anladıktan sonra sistem seçilen öğrencinin bilgilerini öğrenci kayıt kısmında ki boş alanlara öğrencinin bilgilerini tek tek doldurarak admin’in önüne getiriyor gerekli işlemi yaptırmasına izin veriyor.

Admin bundan sonra anlatacağımız her güncelleme ve ekleme işleminde Öğrenci ekle ve güncelleme de anlatıldığı gibi 1 view de 2 iş yapacaktır. Action a ekle viewına yönlendiriyoruz. Örneğin Akademisyen Ekle view’i aynı zamanda adminin akademisyen güncelleme işlemini de gerçekleştireceği yerdir. Örneği EK-Şekil 3.25’de. Yani akademisyen ekle butonuna tıklanıldığı zaman değer girilecek alanlar boş gelirken, akademisyen güncelle butonuna tıklanıldığı zaman gözükecek olan yer akademisyen ekle viewi olurken ayrıyeten, değer girilecek alanlar seçilen akademisyenin bilgileri ile dolmuş olmaktadır.

3.2.3.1.3 Admin Öğrenci Sil

Öğrenci sil butonu ile admin, öğrencilerin listelendiği ve sisteme kayıtlı tüm öğrencileri gördüğü öğrenciler sayfasında istediği kayıtlı öğrenciyi sistemden silebilmektedir.

Şekil 3.20

```
0 başvuru
public ActionResult AdminOgrenciSil(int id)
{
    var ogrsil = db.Öğrenciler.Find(id);

    if (ogrsil == null)
    {
        return HttpNotFound();
    }
    db.Öğrenciler.Remove(ogrsil);

    db.SaveChanges();
    return RedirectToAction("AdminOgrenci");
}
```

Yukarıda ki Şekil 3.20’de gösterilen action ile sil butonuna bir id verdik, ardından ogrsil adında oluşturulan değişkene öğrencinin id’sini tanımladık. If içerisinde oluşturulan değişken boş ise öyle bir kayıt olmadığını gösterip hata sayfası döndürüldü. Else ise atanılan değişkenin null olmadığını gösterince Remove komutu ile öğrenciyi veritabanı kaydından silip kayıtları kaydettik ve bu sayede öğrenci başarılı bir şekilde silinmiş oldu.

3.2.3.2 Admin Bölümler

Admin bölümler sayfasına girdiğinde ise Şekil 3.21’de gösterilen okulda kayıtlı tüm bölümler çıkmaktadır. Karşısına çıkan tabloda bölümlerin adını ve bu bölümlerin her birinde foreach ile listelediğimiz Sil ve Güncelle butonları bulunmaktadır, sayfada sadece bir adet Bölüm Ekle butonu bulunmaktadır ve tablonun alt tarafında gösterilmektedir.

Şekil 3.21

ID	Bölümler	Durum
1	Bilgisayar Mühendisliği	Sil Güncelle
2	Elektrik Elektronik Mühendisliği	Sil Güncelle
3	Endüstri Mühendisliği	Sil Güncelle
4	Mekatronik Mühendisliği	Sil Güncelle
5	Makina Mühendisliği	Sil Güncelle

[Bölüm Ekle](#)

3.2.3.2.1 Admin Bölüm Ekle

Admin, bölümler sayfasında bölüm ekle butonuna tıkladığı zaman tıpkı öğrenci ekle de olduğu gibi önce Bölüm id'si 0 mı ona bakılıyor, eğer 0 ise yeni bir bölüm eklemeye geçiyor. Ekleme sayfasına girdiğinde Şekil 3.22'de gösterilen gibi karşısına çıkan sayfada kendisinden istenen sadece bir bölüm adı girmesidir. Eklemek istediği bölümün adını girip kaydet butonuna tıkladığı zaman sisteme yeni id'ye sahip bir bölüm başarılı bir şekilde eklenmektedir.

Şekil 3.22

Bölüm Adı:

Şekil 3.23

```
[HttpPost]
0 başvuru
public ActionResult AdminBölümEkle(Bölümler blm)
{
    if (blm.Bölüm_Id == 0)
    {
        db.Bölümler.Add(blm);
    }
    else
    {
        var guncelle = db.Bölümler.Find(blm.Bölüm_Id);
        if (guncelle == null)
        {
            return HttpNotFound();
        }
        guncelle.BölümAd = blm.BölümAd;
    }
}
```

Yukarı da ki Şekil 3.23'de bulunan kod görselinde ekle butonuna basıldığında sistem tıklanan butona ait bir bölüm id kayıtlı mı onu kontrol ediyor, eğer kayıtlı değil ise yani 0'a eşit ise Add komutu ile sisteme kayıt ediyor. Eğer sistemde öyle bir bölüm var ise yani bölüm numarası 0'a eşit değil ise güncelleme sayfasını gösteriyor.

3.2.3.2.2 Admin Bölüm Güncelle

Bölüm güncelle butonu ile admin, bölümler sayfasında hangi bölümü güncellemek istiyorsa o bölümün karşısında ki güncelle butonuna tıkladığında, Bölümün adını istediği şekilde değiştiren admin güncelle butonuna tıklayınca bölümü başarı ile güncellemektedir.

Şekil 3.24

```
0 başvuru
public ActionResult AdminBölümGüncelle(int id)
{
    var blm = db.Bölümler.Find(id);
    if (blm == null)
    {
        return HttpNotFound();
    }
    return View("AdminBölümEkle", blm);
}
```

Yukarı da gösterilen Şekil 3.24’de her güncelle butonuna bir id değişkeni atıyoruz, daha sonra atadığımız id değişkenini seçilmek istenen bölümün id’si ile eşitliyoruz. Seçilen bölüm id’yi if else içerisinde alarak önce id’nin var olup olmadığına bakıyoruz yani bölüm id boş ise güncelleme işlemi değil ekleme işlemi olduğunu anlayıp sayfayı hata döndürüyor, eğer bölüm id null değil ise else içerisinde return ile AdminBölümEkle view’ına atıyor ve bölüm ekle de bulunan boş alanları güncelle olduğu için seçilen bölümün bilgilerini getiriyor ve adminde güncelleme imkanı sunuyor.

3.2.3.2.3 Admin Bölüm Sil

Bölüm sil butonu ile admin, bölümler sayfasında hangi bölümü silmek istiyorsa o bölümün karşısında ki sil butonuna tıkladığı zaman bölüm başarılı bir şekilde sistemden silinmektedir. Eğer silinmek istenilen bölüm yeni değil ise yani o bölüme ait akademisyenler, öğrenciler ve projeler var ise bölüm silindiği vakit akademisyenler veya öğrencilerin otomatik bir şekilde silinmesine izin verilmemiştir.

Şekil 3.25

```
0 başvuru
public ActionResult AdminBölümSil(int id)
{
    var blmsil = db.Bölümler.Find(id);
    if (blmsil == null)
    {
        return HttpNotFound();
    }
    db.Bölümler.Remove(blmsil);
    db.SaveChanges();
    return RedirectToAction("AdminBölüm");
}
```

Yukarı da ki Şekil 3.25’de önce her bir sil butonuna id değişkeni tanımlıyoruz, tanımladığımız buton id ye seçilen bölümün id sini verip if else içine gönderiyoruz. If içerisinde id’nin boş olup olmadığına bakıyoruz ve eğer boş ise böyle bir bölüm olmadığı anlamına geldiği için sayfayı hata döndürür. Else içerisinde id’nin boş olmadığını anlayınca Remove komutu ile sistemden kalıcı bir şekilde siliyor

3.2.3.3 Admin Akademisyenler

Admin akademisyenler bölümüne girdiğinde ise karşısına sistemde kayıtlı olan akademisyenler çıkmaktadır. Karşısına çıkan tabloda akademisyen numarası, akademisyen adı, soyadı, kullanıcı adı, şifre ve bölüm gösterilmektedir. Aynı zamanda tabloda 3 farklı buton bulunmaktadır, bunlar Akademisyen Ekle, Akademisyen Güncelle ve Akademisyen Sil butonlarıdır. Akademisyen Ekle butonu sayfa da sadece bir adet bulunurken Güncelleme ve Silme butonları Şekil 3.26’da gösterilen foreach ile listelediğimiz akademisyenlerin her birinde bulunur.

Şekil 3.26

Akademisyen Ekle						
ID	Ad	Soyad	E-posta	Şifre	Bölüm ID	Durum
2	Ramiz	Karaeski	ramizkaraeski@areLedu.tr	12345a.	1	<button>Sil</button> <button>Güncelle</button>
3	Kemal	Fidanboyu	kemalfidanboyu@areLedu.tr	12345a.	1	<button>Sil</button> <button>Güncelle</button>
4	Pınar	Kırcı	pinarkirici@areLedu.tr	12345a.	1	<button>Sil</button> <button>Güncelle</button>
5	Metin	Bilgin	metinbilgin@areLedu.tr	12345a.	1	<button>Sil</button> <button>Güncelle</button>

3.2.3.3.1 Admin Akademisyen Ekle

Admin, akademisyenler bölümünde akademisyen ekle butonuna tıkladığı zaman karşısına gelen sayfa da akademisyen adı, soyadı, kullanıcı adı, şifre, e-mail ve dropdownlistfor kullanılarak getirilen sistemde ki tüm bölümlerden herhangi birini seçip kaydet butonuna tıkladığı zaman sisteme başarılı bir şekilde akademisyen eklemektedir.

Şekil 3.27

```

0 bagvuru
public ActionResult AdminAkdEkle()
{
    var bölümgetir = db.Bölümler.ToList();
    ViewBag.bölümlist = new SelectList(bölümgetir, "Bölüm_Id", "BölümAd");

    return View("AdminAkdEkle");
}

[HttpPost]
0 bagvuru
public ActionResult AdminAkdEkle(Akademisyenler akd)
{
    if (akd.Akademisyen_Id == 0)
    {
        db.Akademisyenler.Add(akd);
    }
    else
    {
        var guncelle = db.Akademisyenler.Find(akd.Akademisyen_Id);
        if (guncelle == null)
        {
            return HttpNotFound();
        }
        guncelle.AkademisyenAd = akd.AkademisyenAd;
        guncelle.AkademisyenSoyad = akd.AkademisyenSoyad;
        guncelle.AkademisyenKullanıcı = akd.AkademisyenKullanıcı;
        guncelle.AkademisyenŞifre = akd.AkademisyenŞifre;
        guncelle.Bölüm_Id = akd.Bölüm_Id;
    }

    db.SaveChanges();

    return RedirectToAction("AdminAkademisyen");
}

```

Bu kayıt işlemini ise Yukarıda ki Şekil 3.27’de gösterildiği gibi önce yapılacak işlemin ekleme mi yoksa güncelleme işlemi mi olduğunu anlatıyoruz. Bunun için işlem yapılmak istenilen akademisyenin id’yi alıyoruz, if içerisinde eğer seçilen akademisyen id’si boş ise yeni bir kayıt olduğunu anlıyor ve Add komutu ile girilecek değerleri6 sisteme kayıt ediyor.

3.2.3.3.2 Admin Akademisyen Güncelle

Admin, akademisyenlerin listelendiği akademisyen sayfasında güncellemek istediği akademisyeni güncelle butonuna basarak güncelleyebilir. Güncelle butonuna basıldığında akademisyenin ad, soyad, şifre, kullanıcı adı ve e-mail textboxlara, ait olduğu bölüm ise dropdownlist'e hazır bir şekilde gelmektedir. Admin bu alanlarda akademisyenin istediği bilgisini başarılı bir şekilde güncellemektedir. Bu işlem önceden anlatılan gibi önce id'nin 0 olmadığını kontrol ediyor ve 0 değil ise ekleme view'ına gönderiyor. Site görseli EK-Şekil 3.26'de gösterilmiştir.

Şekil 3.28

```
0 başvuru
public ActionResult AdminAkdGüncelle(int id)
{
    var bölümgetir = db.Bölümler.ToList();
    ViewBag.bölümlist = new SelectList(bölümgetir, "Bölüm_Id", "BölümAd");

    var akd = db.Akademisyenler.Find(id);
    if (akd == null)
    {
        return HttpNotFound();
    }
    return View("AdminAkdEkle", akd);
}
```

Yukarı da ki Şekil 3.28'de gösterilen akademisyen güncelleme action da önce id değişkeni tanımlıyoruz, tanımladığımız id'yi akademisyenler tablosunda Find komutu ile o id ye sahip akademisyeni SQL'den buluyoruz. Bulunan akademisyenin id'si if içerisinde boş ise hata döndürüyor eğer boş değilse güncelleme işlemi olduğunu anlayıp admin içerisinde ki akademisyen ekleme view'a gönderip orada bulunan boş alanlara seçilen akademisyenin bilgileri sistem tarafından dolduruluyor.

3.2.3.3.3 Admin Akademisyen Sil

Admin, akademisyenler sayfasında hangi akademisyeni silmek istiyorsa karşısında bulunan sil butonuna tıkladığı zaman akademisyeni başarılı bir şekilde sistemden silmektedir. Eğer silinen akademisyenin sistemde kayıtlı bir projesi var ise bu durumda öğrenci mağduriyetini

azaltmak adına admin kontrolü ile silinen akademisyenin projelerini diğer akademisyenlere dağıtılmaktadır.

3.2.3.4 Admin Projeler

Admin projeler bölümüne girdiğinde ise karşısına diğer sayfalarda olduğu gibi direkt olarak projeler gösterilmemektedir bunun nedeni sistemde çok fazla proje olacağından bir karışıklık ihtimalini engellemek içindir. Bu nedenle projeler sayfasına girdiğinde EK-Şekil 3.27’de gösterilen öncelikle karşısına okulun bölümleri çıkmaktadır ve bu bölümlerin yanında Seç butonu bulunmaktadır. Admin hangi bölümde olan projeye işlem yapmak istiyorsa Şekil o bölüme ait projeleri seç butonu ile listelemektedir.

Şekil 3.29

```
<div class="adminicerik">
  <table class="table">
    <tr class="thead-dark">
      <th scope="col">ID</th>
      <th scope="col">Bölümler</th>
      <th>İşlem yapmak istediğiniz Bölümü Seçin</th>
    </tr>
    <tbody>
      <foreach (var blm in Model)>
        <tr>
          <td class="font-italic">@blm.Bölüm_Id</td>
          <td class="font-italic">@blm.BölümAd</td>
          <td><a href="/AdminProje/AdminProjeSec/@blm.Bölüm_Id" class="btn btn-success font-italic">Seç</a></td>
        </tr>
      </foreach>
    </tbody>
  </table>
</div>
```

Yukarı da ki Şekil 3.29’da gösterildiği gibi bölümleri listelemek için foreach ile bölüm ID ve bölüm adları bulunmaktadır. Hangi bölümden proje listelemek istiyor ise admin, o bölümün yanında bulunan Seç butonuna tıklayınca o bölüme ait projeler listelenmektedir.

Listelenen proje sayfasında Şekil 3.18’de gösterildiği gibi proje numarası, Proje Adı, proje tanımı, eğer seçilmiş ise öğrenci adı soyadı ve son olarak sil-güncelle butonları bulunmaktadır.

Şekil 3.30

Proje Ekleme için Tıklayınız

ID	Proje Ad	Proje Tanımı	Öğrenci Adı	Öğrenci Soyadı	Durum
1	Cosmic veri tabanından veri toplama ve görüntü bulma, arayüz hazırlama:	Kansere neden olan mutasyonların konum bilgilerinin bulunduğu Cosmic sitesinden Gigabyte boyutunda verilerin indirilmesi ve gereklidir. Bu verilerin bilgisayarla analizi sonucunda gen-mutasyon-hastalık ilişkisine dair görüntüler yakalanması hedeflenmektedir.			Sil Güncelle
2	Gen - Hastalık - İlaç etkileşimini yapan veri kaynaklarını birleştirme ve elde edilen verilerden yapay öğrenme arayüz uygulaması geliştirmek		Yılmaz	Öztürk	Sil Güncelle
3	DNA sekans veri tabanı dizinleme ve sorgulama yapan arayüzü geliştirme				Sil Güncelle
4	Medikal sekans veya görüntülerde derin öğrenme uygulaması		Tarık	Aslan	Sil Güncelle

Yukarı Şekil 3.30’da gösterilen proje sayfasının view kodlarıdır. Bu sayfada bir adet proje ekleme butonu ve foreach ile listelediğimiz projeler, açıklamalar gibi bilgileri, her bir projenin yanında bulunan sil ve güncelle butonları gösterilmektedir. Proje sayfasında eğer bir öğrenci öncesinden proje seçmişse o projeye ait sil butonu tıklanamaz hale getirilmiştir ve bu sayede öğrencinin aldığı proje silinemeyecektir.

3.2.3.4.1 Admin Proje Ekle

Admin, projeler sayfasında proje ekle butonuna tıkladığı zaman karşısına EK-Şekil 3.28’de gösterilen proje adının girileceği textbox, projenin tanımının girileceği textarea ve projenin hangi bölüme ait olacağını seçmesi için tüm bölümleri listeleyen bir dropdownlist ve projenin hangi akademisyene ait olacağını seçmesi için tüm akademisyenlerin kullanıcı adını listeleyen bir dropdownlist getirilmektedir, EK-Şekil 3.29’de ki view’da tanımlanan textbox ve dropdown list alanlarını doldurarak kaydet butonuna bastığında sisteme başarılı bir şekilde proje kaydedilmektedir.

Şekil 3.31

```
0 basyuru
public ActionResult AdminProjeEkle(Projeler prj)
{
    if (prj.Proje_Id == 0)
    {
        var SonKayit = db.Projeler.ToList().Last();

        prj.Proje_Id = SonKayit.Proje_Id + 1;

        db.Projeler.Add(prj);
    }
    else // eğer proje_id gelmişse bul ve güncelle.
    {
        var guncelle = db.Projeler.Find(prj.Proje_Id);
        if (guncelle == null)
        {
            return HttpNotFound();
        }
        guncelle.ProjeAd = prj.ProjeAd;
        guncelle.ProjeTanimi = prj.ProjeTanimi;
        guncelle.Bölüm_Id = prj.Bölüm_Id;
        guncelle.Akademisyen_Id = prj.Akademisyen_Id;
    }
    db.SaveChanges();

    return RedirectToAction("AdminProje");
}
```

Yukarı da gösterilen Şekil 3.31’da admin proje ekle action da öncelikle projenin id’si 0’a eşit mi ona bakıyoruz eğer sıfıra eşit ise, projeler tablosunda bulunan en son proje kaydının ne olduğunu buluyor ve son projenin id’ye 1 ekliyor ve girilen değerleri SQL’e yeni kayıt olarak kaydetmektedir.

3.2.3.4.2 Admin Proje Güncelleme

Admin, projeler sayfasında güncellemek istediği projenin karşısında ki güncelle butonuna tıkladığı zaman diğer güncelleme işlemlerinde anlatıldığı gibi aynı zamanda EK-Şekil 3.30’da gösterildiği gibi var olan proje üstünden işlem yapılmaktadır ve yine aynı şekilde tek bir view altında yazdırılmaktadır. Karşısına gelen sayfasda proje adı textboxa,proje tanımı textarea’ya ,projenin ait olduğu bölüm ve akademisyen dropdownlistten seçilmiş şekilde gelmektedir. Admin bu alanlarda istediği güncellemeyi yaptıktan sonra güncelle butonuna tıkladığında sistemde proje başarılı bir şekilde güncellenmektedir.

Şekil 3.32

```
0 başvuru
public ActionResult AdminProjeGüncelle(int id)
{
    var blmgetir = db.Bölümler.ToList();
    ViewBag.blmlist = new SelectList(blmgetir, "Bölüm_Id", "BölümAd");

    var akdgetir = db.Akademisyenler.ToList();
    ViewBag.akdlist = new SelectList(akdgetir, "Akademisyen_Id", "AkademisyenKullanıcı");

    var prj = db.Projeler.Find(id);
    if (prj == null)
    {
        return HttpNotFound();
    }
    return View("AdminProjeEkle", prj);
}
```

Yukarı da ki Şekil 3.32’de gösterilen admin proje güncelle actionun da önce dropdown listler için bölümler ve akademisyenleri ViewBag’e aktarıyoruz. Güncelleme işlemi için id değişkeni atıyoruz, atanılan id’yi projeler tablosunda bulunuyor ve if else içerisine giriyor. If içerisinde bulunan proje id’nin boş olup olmadığına bakılıyor. Proje id boş ise güncelleme işlemi olmadığını anlayıp sayfayı hata döndürüyor, eğer proje id boş değil ise öyle bir kayıt sistemde var ise güncellenmek istenen projenin bilgilerini return ile proje ekle view’a gönderiyor, ekle view da ki boş textbox ve dropdownları seçilen proje id’nin bilgileri ile doldurup adminde değiştirme imkanı sunuyor. Admin, istediği bilgiyi güncelleyerek SQL’e kayıt edebiliyor.

3.2.3.4.3 Admin Proje Sil

Admin, projeler sayfasında silmek istediği projenin karşısında ki sil butonuna tıkladığı zaman proje başarılı bir şekilde silinmektedir. Akademisyen sayfasında olduğu gibi eğer herhangi bir öğrenci tarafından o proje alınmış ise o proje silinemez hale getiriliyorç

Proje silme işlemi EK-Şekil 3.31’de gösterildiği gibi işlem yapılmak istenen proje id’si ile öncesinde tanımladığımız boş id’ye atıyoruz, if içerisinde eğer id boş ise sayfayı hata mesajı

döndürüyor else içerisinde ise eğer seçilen id'ye kayıtlı bir proje var ise Remove komutu ile silme işlemini gerçekleştiriyor.

Şekil 3.33

```
<a href="/AdminProje/AdminProjeEkle" class="btn btn-info">Proje Ekleme İçin Tıklayınız </a>
<br /><br />

<table class="table">
  <tr class="thead-dark">
    <th scope="col">ID</th>
    <th scope="col">Proje Ad</th>
    <th scope="col">Proje Tanımı</th>
    <th scope="col">Öğrenci Adı</th>
    <th scope="col">Öğrenci Soyadı</th>
    <th scope="col">Durum</th>
  </tr>
  <tr>
    <th></th>
  </tr>
  <tbody>
    <foreach (var prj in Model)
    <{
      string ButtonStatus = ""; // disabled
      string ograd = "";
      string ogrsoyad = "";

      var kontrol = db.Öğrenciler.SingleOrDefault(Öğrenciler => Öğrenciler.Proje_Id == prj.Proje_Id);

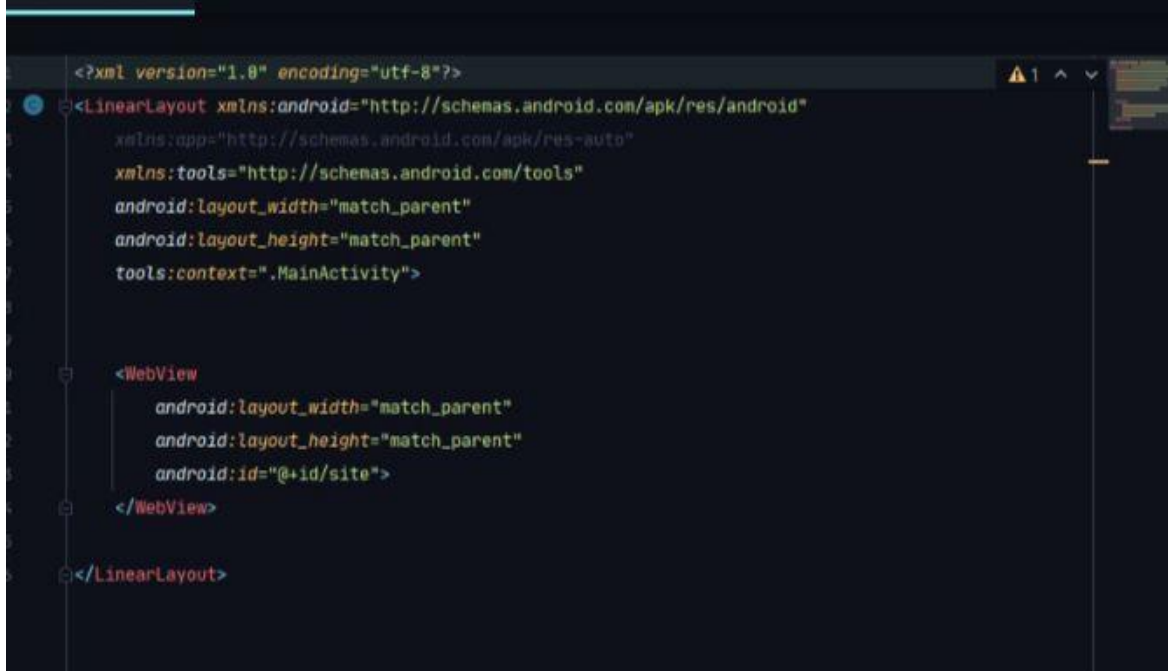
      if (kontrol != null)
      <{
        ButtonStatus = "disabled";
        ograd = kontrol.ÖğrenciAd;
        ogrsoyad = kontrol.ÖğrenciSoyad;
      <

      <tr>
        <td class="font-italic">@prj.Proje_Id</td>
        <td class="font-italic">@prj.ProjeAd</td>
        <td class="font-italic">@prj.ProjeTanımı</td>
        <td class="font-italic">@ograd</td>
        <td class="font-italic">@ogrsoyad</td>
        <td><a href="/AdminProje/AdminProjeSil/@prj.Proje_Id" class="btn btn-dark font-italic @ButtonStatus">Sil</a>
        <td><a href="/AdminProje/AdminProjeGüncelle/@prj.Proje_Id" class="btn btn-danger font-italic">Güncelle</a>
      </tr>
    </tbody>
  </table>
```

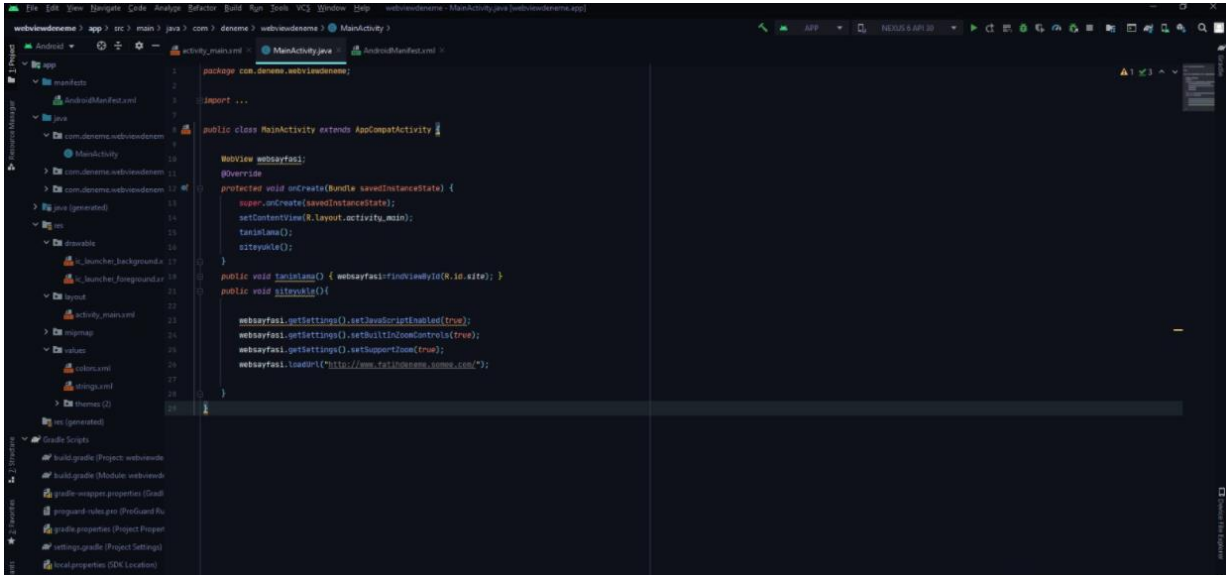
Yukarıda ki Şekil 3.33’de gösterildiği gibi boş bir ButtonStatus adında değişken tanımlıyoruz, daha sonra akademisyen de yapıldığı gibi kontrol adında değişkene öğrenciler tablosun da proje semiş öğrencileri atayıp seçili projelerin sadece sil butonlarını boş atadığımız ButtonStatus ile disabled tanımlayıp tıklanamaz hale getiriyoruz.

3.3 MOBİL UYGULAMA

Mobil uygulama için webview yöntemi kullanıldı. Webview uzak bir sunucudaki websitesinin. Mobil uygulamada gösterilmesini sağlayan bir android studio bileşenidir.



Webview uygulaması için ilk olarak android studio da oluşturulan dosyada activity_main dosyasının webview etiketleri açıldı, bu etiketlerin içinde layoutun genişliği ve uzunluğu verildi ve id belirlendi.



Bunun ardından MainActivity sınıfımızın içerisinde ilk olarak uygulamanın açıldığında çalışması için onCreate metodu oluşturulur. Bu metodun ardından tanımlama ve siteyükle metodları oluşturulur. Tanımlama metodunda Layoutda ki webview ile ilişkili olması için yukarıda oluşturulan webview nesnesi findViewById metodu ile bağlanır. Siteyükle

metodunda ise Web sitemizde yer alan JavaScript kodlarını başarılı bir şekilde çalıştırmak için oluşturulan webview nesnesi `getSettings().setJavaScriptEnabled(true)` şeklinde set edilmiştir. Telefonda zoom özelliğini açabilmek için siteyi yüklediği metodunda oluşturulan webview nesnesi üzerinden `setBuiltInZoomControls(true)` denilerek bu özellik aktif hale getirilir. Uygulamada zoom kontrolünün aktif olması için de `setSupportZoom(true)` yapılarak gerçekleştirilir. `loadUrl` metoduyla uygulamanın açması gereken websitesi belirtilir.

```
package="com.deneme.webviewdeneme">  
  
<uses-permission android:name="android.permission.INTERNET"></uses-permission> //internet  
<application  
    android:allowBackup="true"  
    android:icon="@mipmap/ic_launcher"
```

Bu kod da uygulamanın açılabilmesi içinde internet bağlantısı `AndroidManifest` dosyasında `uses.permission` etiketinde `android name android.permission.INTERNET` verilerek sağlanır.

BÖLÜM 4. SONUÇLAR

4.1 PROGRAMLAMANIN SONUÇLARI

Yapmış olduğumuz çalışma ile Mesleki Proje seçme otomasyonunu baştan aşağı: veritabanı tasarımı, ara yüz tasarımı ve kodlama olarak gerçekleştirmiş bulunmaktayız. Mesleki proje seçim otomasyonunda temel amaç öğrenci ve akademisyenin etkili bir şekilde iletişim kurabileceği ve mesleki proje seçiminde yaşanan sorunları gidermek olduğu tespit edilmiştir. Bu bağlamda öğrenci bilgilerinin, akademisyen bilgilerinin ve proje bilgilerinin depolanabilmesi için veritabanı tasarımı yapılmıştır. Veritabanında çeşitli sorgular ile veritabanı test edilmiştir. Veritabanı ADO.NET ile projeye bağlanmıştır. ASP.NET bünyesinde ADO.NET'in Data Adapter ve Data Set gibi gelişmiş veritabanı erişim objeleri bulunmaktadır. Bunlarda etkinlik, güvenilirlik ve kodlamada hız sağladığı gözlemlenmiştir. Oluşturulan login controllerları ile akademisyen ve öğrencinin otomasyona girişleri sağlanmıştır. Ayrıca create controller ile sistemde mevcut olmayan öğrencilerin sisteme kayıt olabilmeleri sağlanmıştır. OTOMASYON sisteminin daha interaktif bir sistem olabilmesi adına bootstrap frameworkü kullanılarak menü oluşturulmuştur.

Öğrenci ekranında profil düzenleme, proje seçme, akademisyenler ve bölümler gibi seçenekler menüye eklenmiştir. Profil düzenleme ekranında öğrencinin kullanıcı adı, isim, soy isim ve şifre gibi seçenekleri değiştirebilmesi güncellenebilmesi sağlanmıştır.

Projenin ana noktası olan mesleki proje seçme işlemi, proje seç kısmında tanımlanmıştır. Proje seç kısmında öğrencilerin bölümlerine ait projeleri görebilmesi, projelere ait açıklamaları görüntüleyebilmesi sağlanmıştır. Öğrencinin Proje seçmesi, seçilen projeleri görüntüleyebilmesi ve proje seçtikten sonra başka proje seçmemesi sağlanmıştır.

Öğrencinin kendi bölümüne ait akademisyenleri ve akademisyenlere ait bilgileri görüntüleyebilmesi sağlanmıştır ve son olarak oluşturulan bölümler kısmı ile öğrencinin sistemde yer alan bölümleri görüntüleyebilmesi sağlanmıştır.

Güvenlik için öğrencinin sistemden çıkış yapması menüye eklenen buton ile sağlanmıştır. Akademisyenin de siteyi etkileşimli bir şekilde kullanabilmesi adına akademisyene ait menüye projelerim, profil güncelle ve çıkış butonları eklenmiştir. Akademisyenin projelerim kısmından

sistemde olan projeleri görüntüleyebilmesi sağlanmıştır ayrıca her projenin sağına güncelle ve sil butonları eklenmiştir. Böylelikle akademisyen istediği projeyi silebilecek veya istediği şekilde görüntüleyebilecektir. Ayrıca projelerim kısmında yer alan proje ekle butonu ile akademisyen buradan proje ekleme işlemini gerçekleştirmesi sağlanmıştır. Profil güncelle kısmından ise akademisyenin isim, soy isim, şifre, kullanıcı adı gibi özelliklerin değiştirebilmesi sağlanmıştır. Eklenen çıkış yap butonu ile akademisyenin sistemden sağlıklı bir şekilde çıkış yapabilmesi sağlanmıştır.

Otomasyon sisteminin sağlıklı bir şekilde işleyebilmesi için sistemde admin paneli oluşturulmuştur. Admin bilgileri veritabanına eklenmiştir. Adminin sağlıklı bir şekilde giriş yapabilmesi için giriş paneli oluşturulmuştur admin giriş yaptıktan sonra sitenin daha işlevsel olması adına menü eklenmiştir. Bu menüye öğrenciler, akademisyenler, projeler, bölümler ve çıkış yap seçenekleri eklenmiştir.

Menüde bulunan öğrenciler kısmında adminin sistemde yer alan öğrencilere ait bilgilere erişebilmesi sağlanmıştır. Her öğrenciye eklenen sil ve güncelle butonları ile adminin öğrenciler sistemden silebilmesi ve öğrenciye ait bilgileri güncelleyebilmesi sağlanmıştır. Ayrıca üstte yer alan öğrenci ekle butonu ile adminin sisteme başarılı bir şekilde öğrenci ekleyebilmesi sağlanmıştır.

Akademisyenler bölümünde de öğrencilerde olduğu gibi adminin sistemde yer alan akademisyenler görebilmesi sağlanmıştır ve ayrıca her akademisyene ait sil ve güncelle butonları oluşturulmuştur. Adminin sil ve güncelle butonları ile akademisyeni silebilmesi ve akademisyene ait bilgileri güncelleyebilmesi sağlanmıştır. Sayfada yer alan akademisyen ekle kısmı ile de adminin sisteme akademisyen eklenmesi sağlanmıştır.

Projeler kısmında adminin bölümleri görebilmesi sağlanmıştır. Her bölüme eklenen seç butonu ile adminin değişiklik yapmak istediği bölümün projelerini görebilmesi sağlanmıştır. Adminin, her projenin yanına eklenen sil ve güncelle butonları ile istediği projeyi silebilmesi ve projeye ait bilgileri güncelleyebilmesi sağlanmıştır ayrıca sitede yer alan proje ekle butonu ile de adminin sistemde sorun çıkması dahilinde istenilen akademisyen adına proje ekleyebilmesi sağlanmıştır.

KAYNAKLAR

Muammer AKÇAY , Ömer KASIM , Zekiye TAŞDELEN (2021) ASP.NET Ve MVC Temelli Esnek (Responsive) Web Uygulaması, ESTUDAM Bilişim Dergisi.

<https://dergipark.org.tr/en/download/article-file/1468404>

Cihat ERDOĞAN, Ahmet SAYGILI (2015) AKILLI DERS YÖNETİM SİSTEMİ İLE PROGRAMLAMA ÖDEVLERİ İÇİN İNTİHAL TESPİTİ UYGULAMASI, T.C NAMIK KEMAL ÜNİVERSİTESİ BİLİMSEL ARAŞTIRMA PROJELERİ KOORDİNASYON BİRİMİ.

<http://acikerisim.nku.edu.tr:8080/xmlui/handle/20.500.11776/3238>

Uğur TALAŞ (2019), DÜŞÜK BAŞARIMLI WEB UYGULAMASI PROBLEM ANALİZİ VE İYİLEŞTİRME ÇALIŞMASI: ÖĞRENCİ BİLGİ SİSTEMİ DERS SEÇME MODÜLÜ ÖRNEĞİ, BİLECİK ŞEYH EDEBALI ÜNİVERSİTESİ.

<http://acikkaynak.bilecik.edu.tr/xmlui/handle/11552/1442>

Faik TURAN (2018), ASP.NET MVC 4 TEKNOLOJİLERİNİ KULLANARAK BİR E-TİCARET SİTESİ UYGULAMASININ GELİŞTİRİLMESİ, T.C. BEYKENT ÜNİVERSİTESİ SOSYAL BİLİMLER ENSTİTÜSÜ İŞLETME YÖNETİMİ ANABİLİM DALI YÖNETİM BİLİŞİM SİSTEMLERİ BİLİM DALI

https://tez.yok.gov.tr/UlusalTezMerkezi/tezDetay.jsp?id=6B1D0_F3v6Ti9-ruoZP_Xg&no=w0xhyitakQV4xUateczdUA

Ahmet Gürkan YÜKSEK - Halil ARSLAN - Yavuz TÜRKAY - Oğuz KAYNAR - Yunis TORUN (2016) : Çevik Yazılım Geliştirme Yaklaşımı İle Beden Eğitimi Özel Yetenek Sınav Otomasyon Sisteminin Tasarımı

<https://dergipark.org.tr/tr/download/article-file/319469>

Ibrahim AĞLAYAN İSTANBUL KÜLTÜR ÜNİVERSİTESİ-FEN BİLİMLERİ
ENSTİTÜSÜ, YENİ WEB TEKNOLOJİLERİ VE WEB UYGULAMALARI

<https://acikerisim.iku.edu.tr/bitstream/handle/11413/477/ibrahim%20caglayan.pdf?sequence=1&isAllowed=y>

DEMİRKOL Zafer 2010, C# ile ASP.NET [Kitap]. - İstanbul : Pusula Yayıncılık

Ping, Y., Kontogiannis, K., & Lau, T. C. (2003, September). Transforming Legacy Web Applications to The MVC Architecture. In Eleventh Annual International Workshop on Software Technology and Engineering Practice

https://www.researchgate.net/publication/4112359_Transforming_legacy_Web_applications_to_the_MVC_architecture

LU Jun-wei, CHANG Lin, CHEN Yun-kun (Orient Science & Technology College of Hunan Agricultural University, Changsha 410128, China) ,MVC Design Pattern and ASP.NET MVC Framework Research

https://en.cnki.com.cn/Article_en/CJFDTotol-DNZS201019015.htm

EKLER

EK-Şekil 3.1

	Column Name	Data Type	Allow Nulls
▶	AdminID	int	<input type="checkbox"/>
	AdminUserName	nvarchar(MAX)	<input type="checkbox"/>
	AdminPassword	nvarchar(MAX)	<input type="checkbox"/>
			<input type="checkbox"/>

EK-Şekil 3.2

	Column Name	Data Type	Allow Nulls
▶	Akademisyen_Id	int	<input type="checkbox"/>
	AkademisyenAd	nvarchar(50)	<input checked="" type="checkbox"/>
	AkademisyenSoyad	nvarchar(50)	<input checked="" type="checkbox"/>
	AkademisyenKullanıcı	nvarchar(50)	<input checked="" type="checkbox"/>
	AkademisyenŞifre	varchar(50)	<input checked="" type="checkbox"/>
	Bölüm_Id	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

EK-Şekil 3.3

	Column Name	Data Type	Allow Nulls
PK	Bölüm_Id	int	<input type="checkbox"/>
	BölümAd	nvarchar(50)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

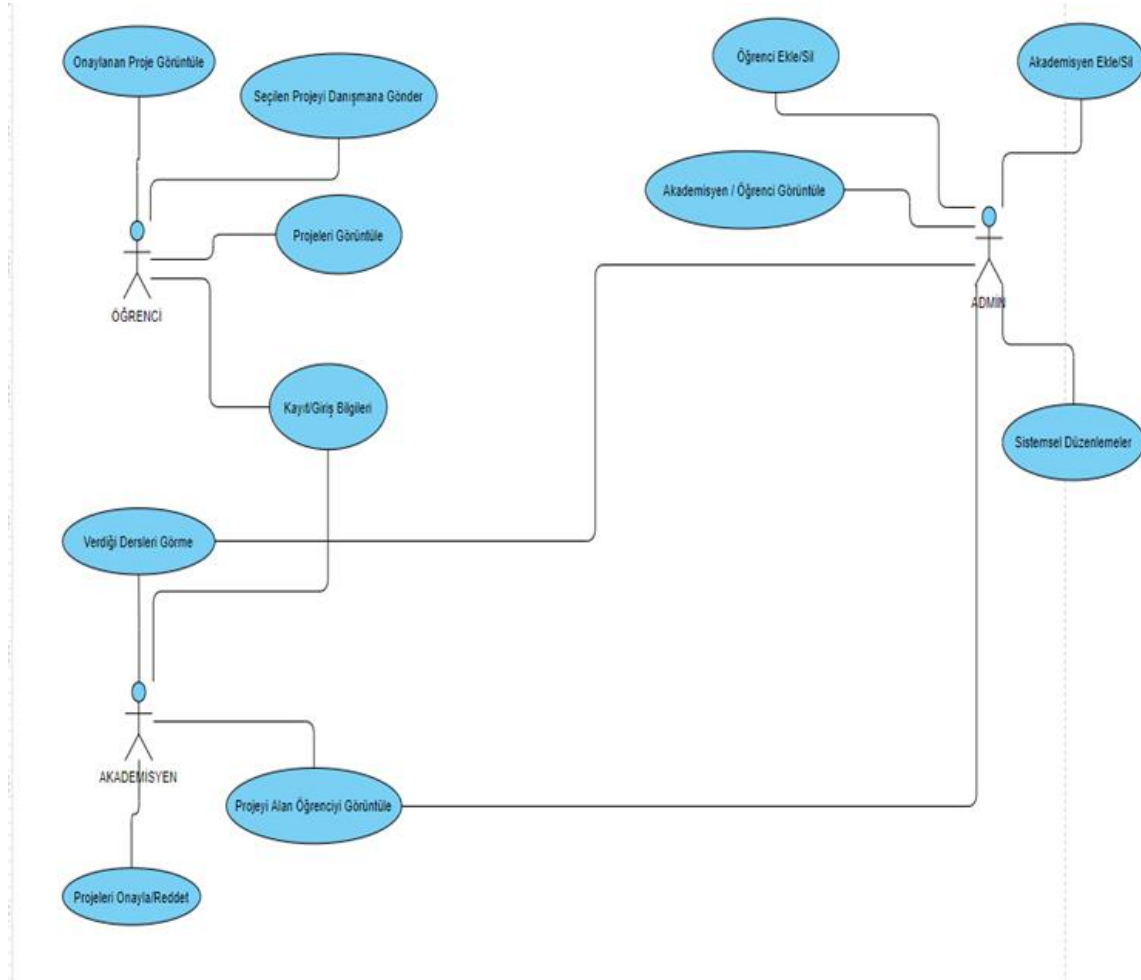
EK-Şekil 3.4

	Column Name	Data Type	Allow Nulls
PK	Ogrenci_Id	int	<input type="checkbox"/>
	OgrenciAd	nvarchar(50)	<input checked="" type="checkbox"/>
	OgrenciSoyad	nvarchar(50)	<input checked="" type="checkbox"/>
	KullaniciAd	nvarchar(50)	<input checked="" type="checkbox"/>
	Şifre	nvarchar(50)	<input checked="" type="checkbox"/>
	Bölüm_Id	int	<input checked="" type="checkbox"/>
	Proje_Id	int	<input checked="" type="checkbox"/>
	Akademisyen_Id	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

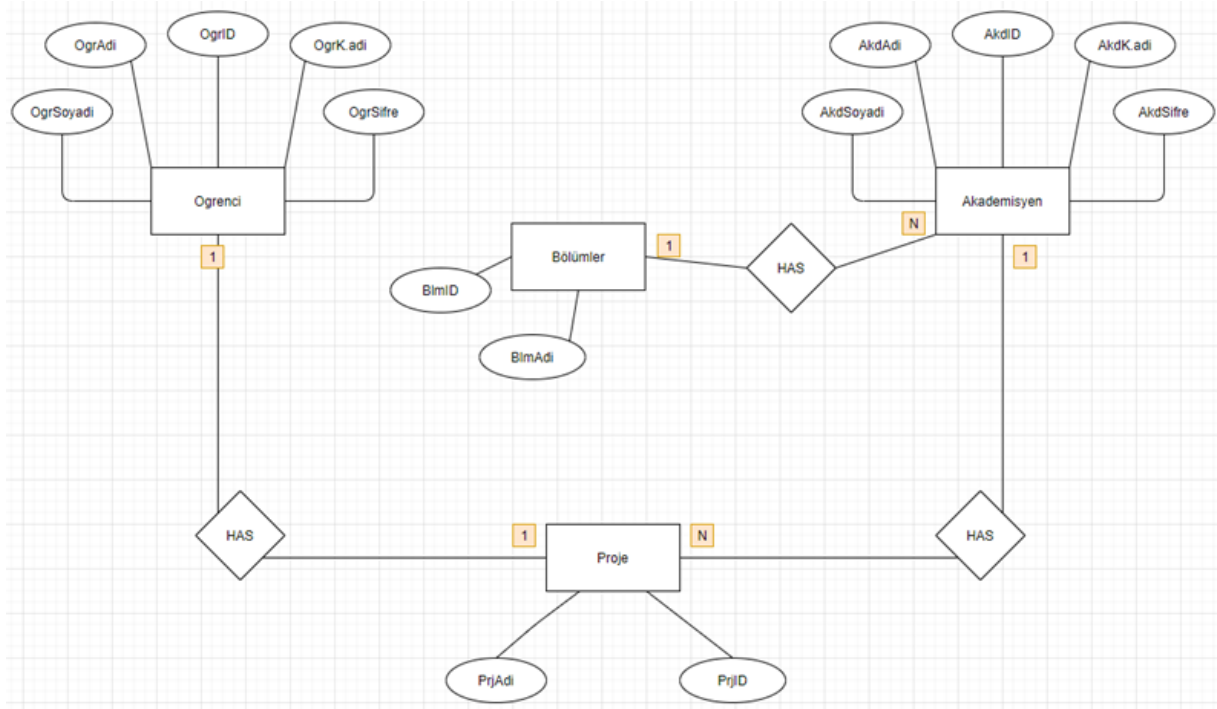
EK-Şekil 3.4

	Column Name	Data Type	Allow Nulls
PK	Proje_Id	int	<input type="checkbox"/>
	ProjeAd	nvarchar(MAX)	<input checked="" type="checkbox"/>
	ProjeTanimi	nvarchar(MAX)	<input checked="" type="checkbox"/>
	Bölüm_Id	int	<input checked="" type="checkbox"/>
	Akademisyen_Id	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

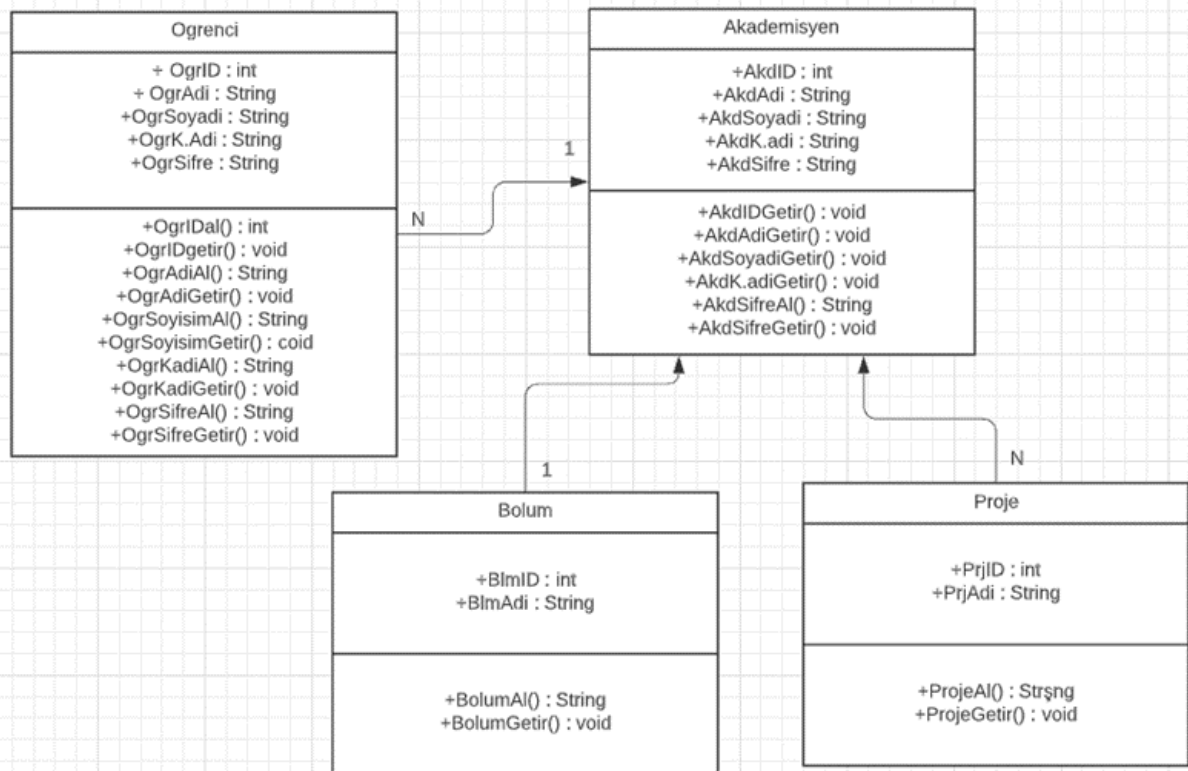
EK-Şekil 3.6



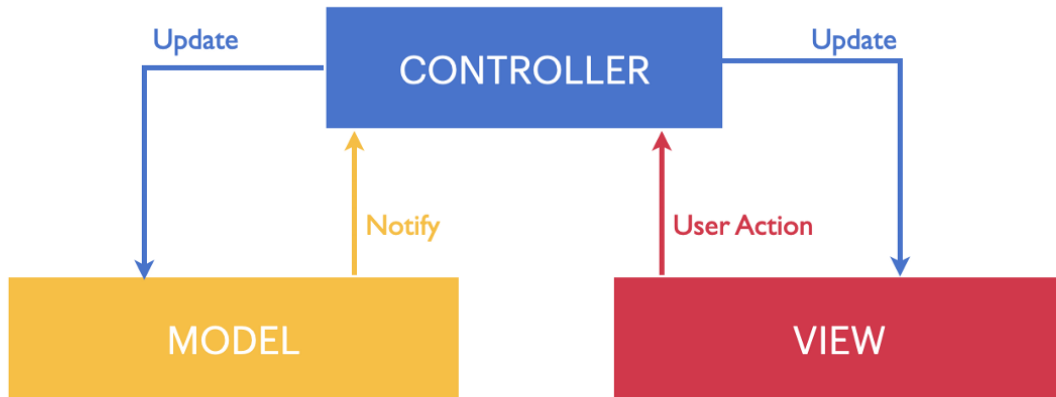
EK-Şekil 3.7



EK-Şekil 3.8



EK-Şekil 3.9



EK-Şekil 3.10

[illegible]

EK-Şekil 3.11

```

14 başvuru
public partial class Öğrenciler
{
    5 başvuru
    public int Oğrenci_Id { get; set; }

    [Required(ErrorMessage = "Adınızı Giriniz !!!")]
    9 başvuru
    public string OğrenciAd { get; set; }

    [Required(ErrorMessage = "Soyadınızı Giriniz!!!")]
    9 başvuru
    public string OğrenciSoyad { get; set; }

    [Required(ErrorMessage = "Kullanıcı Adınızı Giriniz !!!")]
    [MaxLength(9, ErrorMessage = "Kullanıcı Adı En Fazla 9 Karakterden Oluşmalıdır..")]
    [MinLength(9, ErrorMessage = "Kullanıcı Adı En Az 9 Karakterden Oluşmalıdır..")]
    11 başvuru
    public string KullanıcıAd { get; set; }

    [Required(ErrorMessage = "Şifrenizi Giriniz !!!")]
    [MaxLength(15, ErrorMessage = "Şifreniz En Fazla 15 Karakterden Oluşmalıdır..")]
    [MinLength(6, ErrorMessage = "Şifreniz En Az 6 Karakterden Oluşmalıdır..")]
    10 başvuru
    public string Şifre { get; set; }

    6 başvuru
    public Nullable<int> Bölüm_Id { get; set; }
    4 başvuru
    public Nullable<int> Proje_Id { get; set; }
    0 başvuru
    public Nullable<int> Akademisyen_Id { get; set; }

    0 başvuru
    public virtual Akademisyenler Akademisyenler { get; set; }

    1 başvuru
    public virtual Bölümler Bölümler { get; set; }
    1 başvuru
    public virtual Projeler Projeler { get; set; }
}

```

EK-Şekil 3.12

```

<authentication mode="Forms">
  <forms loginUrl="Home/Index"></forms>
</authentication>

```

EK-Şekil-13



EK-Şekil 3.14

```

<table class="table" style="position:relative; bottom:50px;">
  <tr class="thead-dark">
    <th scope="col">Akademisyen ID</th>
    <th scope="col">Projeler</th>
    <th scope="col">Proje Tanımı</th>
    <th scope="col">Proje ID </th>
    <th scope="col">Seç</th>
  </tr>
  <tbody>
    @foreach (var prj in Model)
    {
      string ButtonStatus = ""; // disabled
      string TextColor = "black";
      bool DelText = false;

      var dbResult = entitiess.Öğrenciler.SingleOrDefault(Öğrenciler => Öğrenciler.Proje_Id == prj.Proje_Id);

      if (dbResult != null)
      {
        ButtonStatus = "disabled";
        TextColor = "red";
        DelText = true;
      }

      <tr>
        <td style="color: @TextColor" class="font-italic">@prj.Akademisyen_Id</td>
        <td style="color: @TextColor" class="font-italic">
          @if (DelText == true)
          {
            <del class="font-italic"> @prj.ProjeAd </del>
          }
          else
          {
            @prj.ProjeAd
          }
        </td>
        <td style="color: @TextColor" class="font-italic">@prj.ProjeTanımı</td>
        <td style="color: @TextColor" class="font-italic"> @prj.Proje_Id </td>
        <td class="font-italic"> <a href="/ProjeSec/Sec/@prj.Proje_Id" class="btn btn-dark font-italic @ButtonStatus"> Seç </a></td>
      </tr>
    }
  </tbody>
</table>

```

EK-Şekil 3.15

≡ Akademisyenler

ID	Akademisyen Adı	Akademisyen Soyadı	Akademisyen E-posta
2	Ramiz	Karaeski	ramizkaraeski@arel.edu.tr
3	Kemal	Fidanboyu	kemalfidanboyu@arel.edu.tr
4	Pınar	Kırcı	pinarkirici@arel.edu.tr
5	Metin	Bilgin	metinbilgin@arel.edu.tr
6	Emre	Dirik	emredirik@arel.edu.tr
7	Ceyda	Öztürk	ceydaozturk@arel.edu.tr
8	Savaş	Takan	savastakan@arel.edu.tr
10	Selin	Evsen	selinevsen@arel.edu.tr

EK-Şekil 3.16

≡ Bölümler

ID	Bölümler
1	Bilgisayar Mühendisliği
2	Elektrik Elektronik Mühendisliği
3	Endüstri Mühendisliği
4	Mekatronik Mühendisliği
5	Makina Mühendisliği

EK-Şekil 3.17

<pre>Çıkış Yap</pre>
--

EK-Şekil 3.18



EK-Şekil 3.19

A screenshot of a web form titled 'Profil Güncelle'. The form is divided into four sections by horizontal lines. Each section has a label on the left and a text input field on the right. The first section is labeled 'Akademisyen Adı' with the input field containing 'Ramiz'. The second section is labeled 'Akademisyen Soyadı' with the input field containing 'Karaeski'. The third section is labeled 'Akademisyen Şifre' with the input field containing '12345a.'. The fourth section is labeled 'Akademisyen ID' with the input field containing '2'. Below the fourth section, there is a yellow button with the text 'Güncelle'.

EK-Şekil 3.20

```
O başvuru
public ActionResult AkdProjeGüncelle(int id)
{
    var blmgetir = db.Bölümler.ToList();
    ViewBag.blmlist = new SelectList(blmgetir, "Bölüm_Id", "BölümAd");

    var akdgetir = db.Akademisyenler.ToList();
    ViewBag.akdlist = new SelectList(akdgetir, "Akademisyen_Id", "AkademisyenKullanıcı");

    var ogrgetir = db.Öğrenciler.ToList();
    ViewBag.ogrlist = new SelectList(ogrgetir, "Ogrenci_Id", "OgrenciAd");

    var prj = db.Projeler.Find(id);
    if (prj == null)
    {
        return HttpNotFound();
    }
    return View("AkdProjeEkle", prj);
}
```

EK-Şekil 3.21

```
O başvuru
public ActionResult AkdProjeSil(int id)
{
    var prjsil = db.Projeler.Find(id);

    if (prjsil == null)
    {
        return HttpNotFound();
    }


    db.Projeler.Remove(prjsil);

    db.SaveChanges();
    return View("AkdProfil");
}
```

EK-Şekil 3.22

```
<authentication mode="Forms">  
  <forms loginUrl="Home/Index"></forms>  
</authentication>
```

EK-Şekil 3.23



A login form is displayed on a dark gray background. The form consists of two input fields and a button. The first input field is labeled 'Kullanıcı Adı Girin' (Enter Username) and the second is labeled 'Şifre Girin' (Enter Password). Below these fields is a blue button with the text 'Giriş Yap' (Login).

EK-Şekil 3.24

Öğrenciler

Akademisyenler

Projeler

Bölümler

Çıkış Yap

EK-Şekil 3.25

```
0 başvuru
public ActionResult AdminAkdGüncelle(int id)
{
    var bölümgetir = db.Bölümler.ToList();
    ViewBag.bölümlist = new SelectList(bölümgetir, "Bölüm_Id", "BölümAd");

    var akd = db.Akademisyenler.Find(id);
    if (akd == null)
    {
        return HttpNotFound();
    }
    return View("AdminAkdEkle", akd);
}
```

EK-Şekil 3.26

Öğrenciler	Akademisyen Adı :	<input type="text" value="Kemal"/>
Akademisyenler		
Projeler		
Bölümler	Akademisyen Soyadı :	<input type="text" value="Fidanboyu"/>
Çıkış Yap		
	Akademisyen E-posta :	<input type="text" value="kemalfidanboyu@arele.e"/>
	Akademisyen Şifre :	<input type="text" value="12345a."/>
	Akademisyen Bölümü :	<input type="text" value="Bilgisayar Mühendisliği"/>
		<input type="button" value="Akademisyen Kaydet"/>

EK-Şekil 3.27

ID	Bölümler	İşlem Yapmak İstediğiniz Bölümü Seçin
1	Bilgisayar Mühendisliği	<input type="button" value="Seç"/>
2	Elektrik Elektronik Mühendisliği	<input type="button" value="Seç"/>
3	Endüstri Mühendisliği	<input type="button" value="Seç"/>
25	Mekanik Mühendisliği	<input type="button" value="Seç"/>

EK-Şekil 3.28

Öğrenciler

Akademisyenler

Projeler

Bölümler

Çıkış Yap

Proje Adı:

Proje Tanımı Giriniz:

Bölüm Seçiniz:

Bölüm Seçiniz

Akademisyenin Seçiniz:

Akademisyen Seçiniz

Projeyi Kaydet

EK-Şekil 3.29

```
<div class="adminicerik">

    <using (Html.BeginForm("AdminProjeEkle", "AdminProje", FormMethod.Post))>

    <span style="margin-left:auto; font-size:20px;" class="font-italic">Proje Adı:</span><input type="text" value="@Model.ProjeAd" /> <br /> </div> </div>

    <span style="margin-left:auto; font-size:20px;" class="font-italic">Proje Tanımı Giriniz: </span><input type="text" value="@Model.ProjeTanimi" /> <br /> </div> </div>

    <span style="margin-left:auto; font-size:20px;" class="font-italic">Bölüm Seçiniz:</span> <input type="text" value="@Model.BolumId" /> <br /> </div> </div>

    <span style="margin-left:auto; font-size:20px;" class="font-italic">Akademisyenin Seçiniz: </span><input type="text" value="@Model.AkademisyenId" /> <br /> </div> </div>

    <input type="hidden" value="@Model.ProjeId" />
    <input type="hidden" value="@Model.BolumId" />

    <input type="submit" value="Projeyi Kaydet" class="btn btn-danger font-italic" style="margin-left:170px" />

</div>
```

EK-Şekil 3.30

Öğrenciler

Akademisyenler

Projeler

Bölümler

Çıkış Yap

Proje Adı:

GSM Tabanlı Akıllı Dijital

Proje Tanımı Giriniz:

aMikrodenetleyici
ile haberleşebilen
GSM modül
kullanılarak akıllı bir

Bölüm Seçiniz:

Elektrik Elektronik Mühendisliği

Akademisyenin Seçiniz:

sukruulker@arel.edu.tr

Projeyi Kaydet

EK-Şekil 3.31

```
Obaşvuru
public ActionResult AdminProjeSil(int id)
{
    var prjsil = db.Projeler.Find(id);

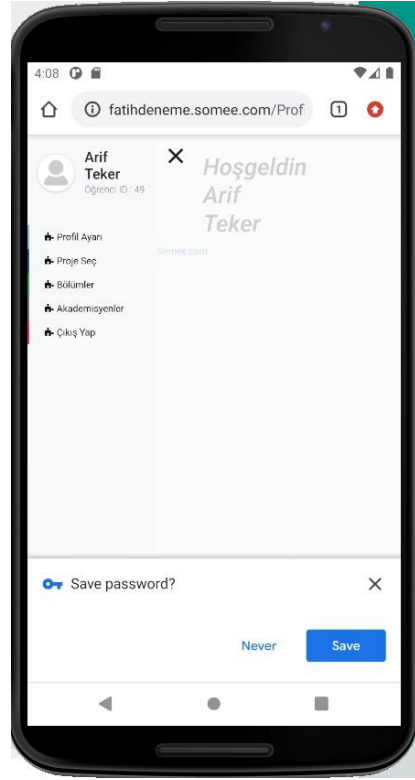
    if (prjsil == null)
    {
        return HttpNotFound();
    }
    db.Projeler.Remove(prjsil);

    db.SaveChanges();
    return RedirectToAction("AdminProje");
}
}
```

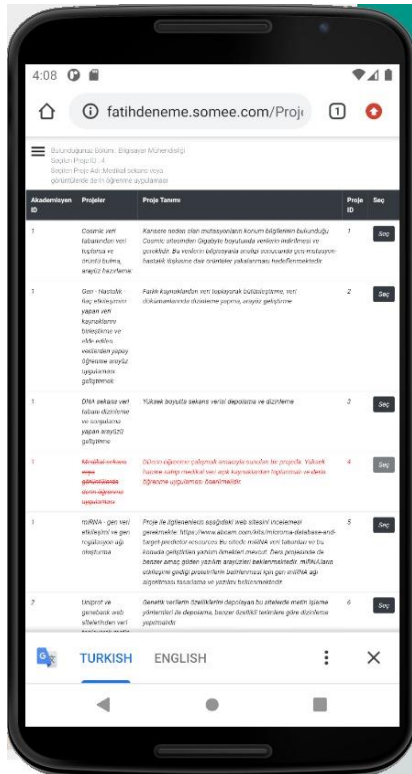
MOBİL



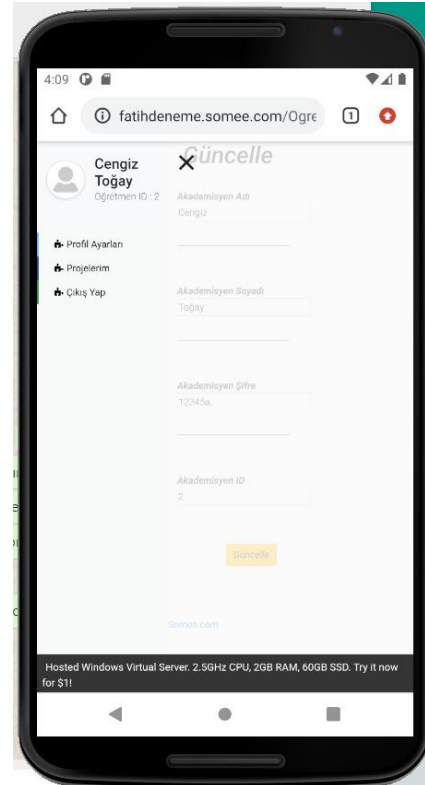
EK.Şekil 3.32



EK.Şekil 3.33

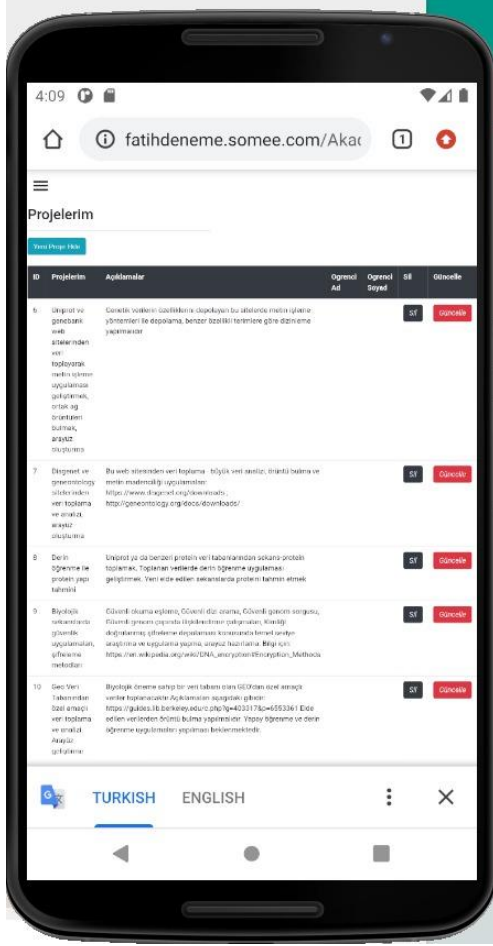


EK.Şekil 3.34

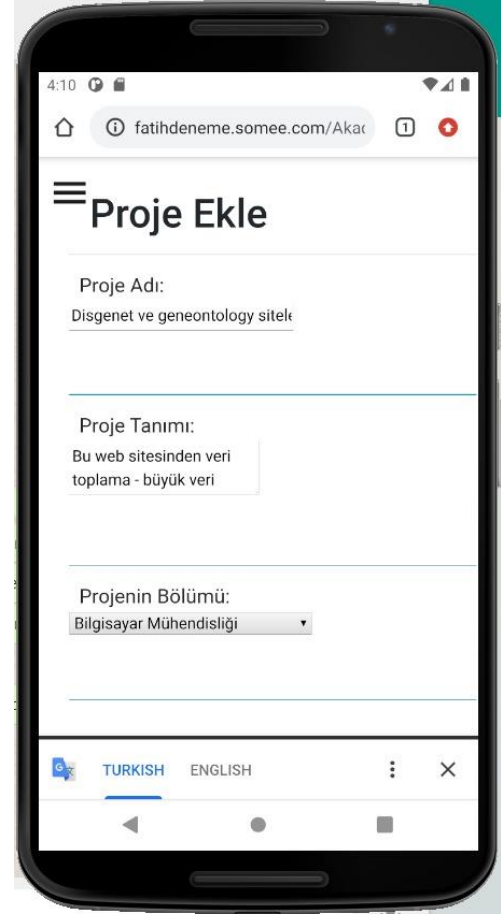


EK.Şekil.3.35

MOBİL



EK.Şekil 3.36



EK.Şekil 3.37

ÖZGEÇMİŞ

Ad : Fatih

Soyad : Şentürk

E-Posta : senturkfatih03@outlook.com

Staj ve İş Deneyimleri : Vektör Bilişim, uzun dönem stajyer.

Gerçekleştirdiği Projeler: Yok

Ad : Ahmet

Soyad : Türkmen

E-Posta : ahmet03_trkmn@hotmail.com

Staj ve İş Deneyimleri : Yok

Gerçekleştirdiği Projeler: Yok

Ad : Yunus

Soyad : Şeker

E-Posta : yunusseker32@gmail.com

Staj ve İş Deneyimleri : Yok

Gerçekleştirdiği Projeler: Yok