

MARMARA ÜNİVERSİTESİ
TEKNİK BİLİMLER MESLEK
YÜKSEKOKULU

MARMARA EĞİTİM PORTALI
BACKEND ÇALIŞMASI

Bitirme Projesi

360120049

Furkan Can İşci

Bölüm: Bilgisayar Teknolojileri
Program: Bilgisayar Programcılığı

Danışman :

Ercan ERKALKAN

Juri Üyeleri :

Vedat TOPUZ

Fatih KAZDAL

Zehra Aysun ALTIKARDEŞ

2022

Özgünlük Bildirisi

1. Bu çalışmada, başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini,
2. Alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımın benim tarafımdan yapıldığını bildiririm.

Tarih: 16 / 06 / 2022

Furkan Can İşci

MARMARA EĞİTİM PORTALI BACKEND TASARIMI

-ÖZET -

Proje bir eğitim içerikleri platformu olmasıyla birlikte aynı zamanda okul departmanı bazlı bir öğrenci yönetim sistemidir. Projeyi anlama ve yapılacakları belirleme aşamasında Danışmanım Ercan Hoca ile projenin başlanıç aşamasında yaptığımız konuşmalar sonucunda projenin hem online bir eğitim içerikleri portalı hem de bir öğrenci yönetim sistemi olmasına karar verdik. Daha sonra ilk iş olarak projenin ana şemasını hazırlamak için veritabanı modelini hazırladım. Bunun için bir kullanıcı üyelik sistemi ve kullanıcının kolayca kayıt olabileceği dersler, kurslar, eğitim dönemleri, eğitimciler gibi veritabanında gerekli olan tabloları oluşturdum ve modelini hazırladım. Projeyi Pycharm üzerinde geliştirebilmek için sanal ortamı kurdum. Projeyi çalışır hale getirdikten sonra veritabanına bağlama kısmında settings.py üzerinde bulunan 'databases' kısmında veritabanı adını belirttim. Model oluşturmaya başlayıp tabloları projeye aktarmaya başladım. Modelleri oluşturduktan sonra 'makemigrations' ve 'migrate' komutlarıyla modelleri tablolar şeklinde veritabanına kayıt ettim. Views kısmında tabloya veri ekleme tablodan veri çekme gibi işlemler için modelleri Django-ORM yapısıyla sağladım. Urls kısmında viewlerin hangi sayfada görüleceği açısından path kısmında kullandım. View aracılığıyla templates klasöründe veritabanından verileri gösterdim ya da verileri kaydetmek için butonlar oluşturdum. Üyelik için kendim bir üyelik sistemi kullandım. E-mail kısmında yer alan ifadeye göre üyeler sadece kendi url bağlantılarına göre sitede ilerleyebiliyorlar.

İÇİNDEKİLER

1.Giriş.....	5
2.Marmara Eğitim Portalı	6
2.1. Projenin Konusu.....	6
2.2 Projenin Amacı	6
3. Projenin Yazılımı	7
3.1.VeritabanıTasarımı.....	9
3.2.Pycharm Kurulumu ve Modellerin Hazırlanması	10
3.3.View Aracılığıyla Modellerin Kullanılması	11
3.4.Url Kısımında Route İşlemleri.....	15
3.5.Templates Klasörünün Ayarlanması	17
4. Marmara Eğitim Portalı Uygulaması	18
4.1.Proje Amacı ve Görselleri	18
4.2. Kullanıcı Üyelik Sistemi	18
4.3. Admin Paneli Tasarımı.....	20
4.4. Student ve Staff Panellerinin Tasarımı.....	21
3.5.Route Ayarlamaları ve Email-Backend.....	22
5. Çıkarımlarım	25
6. Kaynaklar	26

1.Giriş

Günümüzde çok fazla kullanılan eğitim içerikleri platformları bulunmaktadır.Bunlar dünyanın her yerinden insanları istedikleri alanda farklı kurslar ve derslere kayıt olup bu derslere ait içeriklerden faydalanmak veya kurslar oluşturup dersler ekleyerek insanlara ulaşmak istiyorlar.Bununla beraber bu eğitim platformlarının kullanımı çok yaygınlaştı.

Kullanıcıların bu platformda aradıkları en önemli özellikleri arasında aradığına hızlı ve kolayca istediği zaman ulaşabilmesi,faydalı eğitim içeriklerine sahip olabilmesi denilebilir. Bu konuda kullanıcılar sisteme üye olduktan sonra hesaplarına giriş yaptıklarında kullanıcı paneline ilerledikleri zaman ders görüntüleme, ders ekleme, mesaj göndermek gibi işlevleri rahat bir şekilde uygulamak istiyorlar.

Çağımızın en önemli farklarından biri bilgiye ulaşabilme hızıdır.Bu kapsamda kullanıcının panelin tasarımından işlevselliğine kadar memnun kalması önemli.Bu platformda en önem verdiğim hususlardan biri de kullanıcı dostu kullanıcı paneli tasarımıdır.Kullanıcı panel sistemine girdiği zaman sahip olduğu kursları,dersleri,öğrencileri,soru geçmişi gibi konularda rahat bir şekilde bilgiye ulaşabilmek istiyor.

Rapor boyunca bu kısmın ardına projenin amacı,projeyi tasarlarken bu konuları nasıl ele aldığımı adım adım anlattım.Ayrıca projede kullandığım teknolojiler ve projemdeki önemlerini,tasarım sürecini de anlattım.

2.Marmara Eğitim Portalı

2.1. Projenin Konusu

Projenin amacı kısaca dünyada halihazırda çok popüler olan eğitim içeriği platformları ile bir okul departmanına kayıtlı olan öğrencileri yönetim sistemini birleştirmektir.Projeyi planlama aşmasında sadece bir eğitim platformu olarak değil de aynı zamanda kursların kayıtlı olduğu bir departman yönetim sistemi hazırladım.Kullanıcıların sitesinde görüntülenecek olan kursların eklenmesi sisteme kayıtlı öğrenciler, öğretmenlerin rahat bir şekilde yönetilebilmesini hedefledim.

2.2 Projenin Amacı

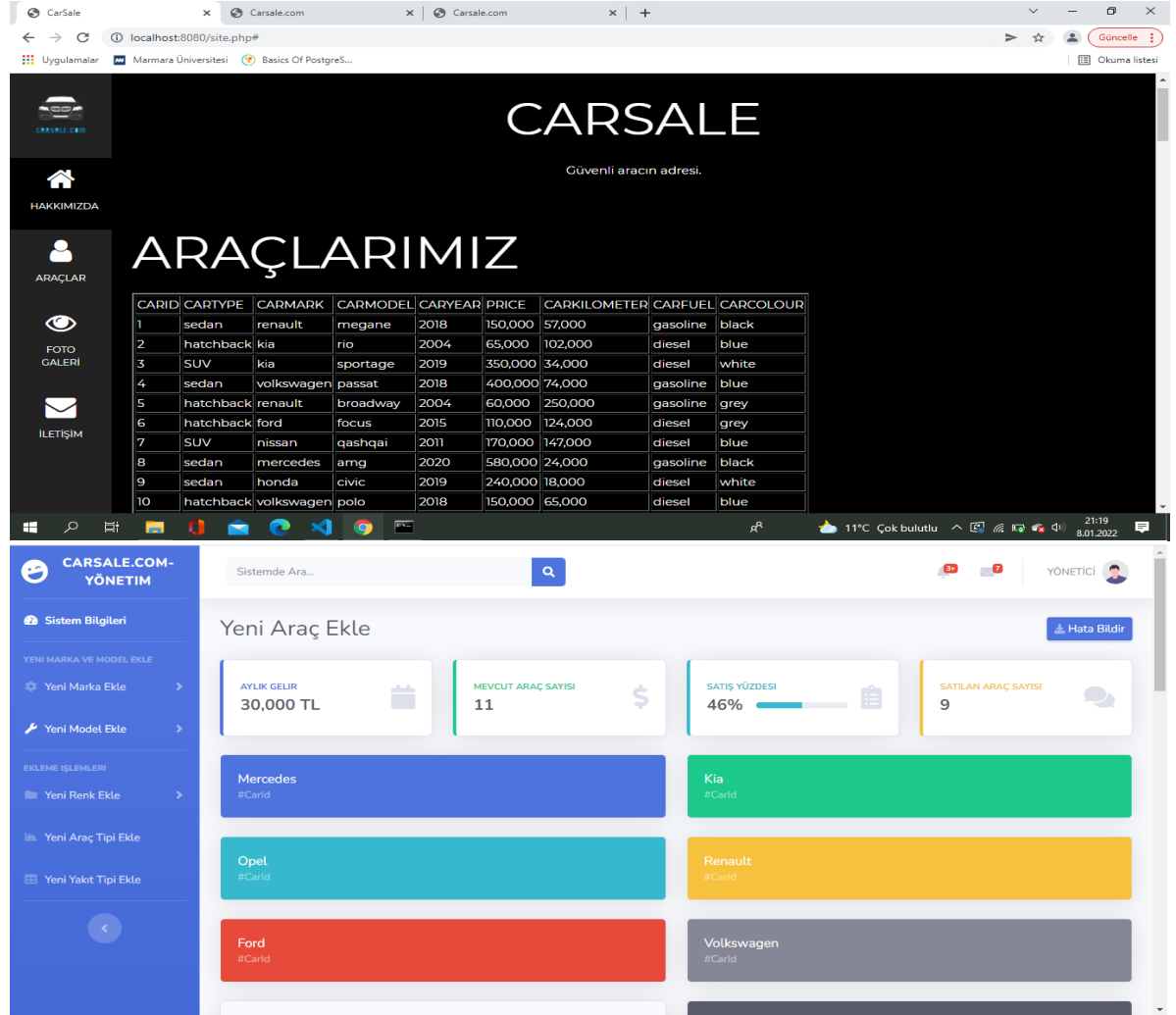
Proje kullanıcıların kayıtlı olduğu eğitim portalında görüntüleyecekleri içeriklerin yönetilmesi,kullanıcıların üyelik sisteminin rollerinin dağıtılması ve hangi haklara sahip olduğunu projenin arka planında tasarlanması.Ve bununla birlikte kullanıcıya bir eğitim platformu sistemi sunmak.

Kullanıcının sisteme rolüne uygun olarak kayıt olması ve eğitim içeriklerinden faydalanabilmesi için tasarladığım panellerde kullanıcının rolüne göre değişen özellikler ekledim.Admin panelinde sisteme zaten kayıtlı kurslar,öğrenciler,eğitmenleri görüntülemek ya da sisteme yeni öğrenciler,öğretmenler,kurslar,dersler gibi kısımları eklemek gibi özellikler vardır.Öğretmen panelinde kendisine bağlı öğrencileri,kursları,dersleri görüntüleyebilir ve öğrenciye kursla ilgili olarak soru sorabilir veya not bırakabilir.Öğrenci ise bu sorulara ve notlara mesaj kısmından cevap yazabilir veya departmana mesaj yazabilir.Bunun yanında sahip olduğu kursları ve dersleri görüntüleyebilir.

3. Projenin Yazılımı

Backend projesinde Python framework uygulaması olan Django kullandım Daha öncesinde backend alanında php(laravel),flask ile tecrübeye sahiptim.Backend mantığına ve çalışma şekline route yöntemlerine aşındıyım.Tabii her yazılım dilinde framework arasında bile ciddi farklılıklar olabiliyor.Bu noktada django alışılması kolay olan bir yazılım dili.Django projemi yönetebilme ve bir sunucu üzerinde çalıştırabilmek için Pycharm kullandım.Pycharm kullanırken herhangi bir sıkıntı yaşamadım.Kullanımı kolay tasarımı gayet rahat olan bir derleyici.

Daha öncesinde kendi yapmış olduğum php ile yaptığım araç satın alma sitesi.



CarSale

localhost:8080/site.php#

Uygulamalar Marmara Üniversitesi Basics Of PostgreS...

Okuma listesi

1 sedan renaul megane 2018 150,000 57,000 gasoline black

SATIN AL

CARID	CARTYPE	CARMARK	CARMODEL	CARYEAR	PRICE	CARKILOMETER	CARFUEL	CARCOLOUR
2	hatchback	kia	rio	2004	65,000	102,000	diesel	blue

SATIN AL

CARID	CARTYPE	CARMARK	CARMODEL	CARYEAR	PRICE	CARKILOMETER	CARFUEL	CARCOLOUR
3	SUV	kia	sportage	2019	350,000	34,000	diesel	white

DB Browser for SQLite - C:\SQL\connection.db

Dosya Düzenle Görünüm Araçlar Yardım

Yeni Veritabanı Veritabanı Aç Değişiklikleri Kaydet Değişiklikleri Geri Al Proje Aç Projeyi Kaydet Veritabanı Ekle Veritabanı Kapat

Pragmaları Düzenle SQL kodunu yürüt Veriyi Görüntüle Veritabanı Yapısı

Tablo: car

	id	caryear	carcolourid	carmarkid	carmodelid	carkilometer	carfuelid	cartypeid	price
	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre	Filtre
1	1	2018	1	1	1	57,000	1	1	150,000
2	2	2004	2	2	3	102,000	2	2	65,000
3	3	2019	3	2	5	34,000	2	3	350,000
4	4	2018	2	3	4	74,000	1	1	400,000
5	5	2004	4	1	2	250,000	1	2	60,000
6	6	2015	4	4	6	124,000	2	2	110,000
7	7	2011	2	5	7	147,000	2	3	170,000
8	8	2020	1	6	8	24,000	1	1	580,000
9	9	2019	3	7	9	18,000	2	1	240,000
10	10	2018	2	3	10	65,000	2	2	150,000
11	11	2020	4	4	11	32,000	2	3	290,000

1 - 11 / 11

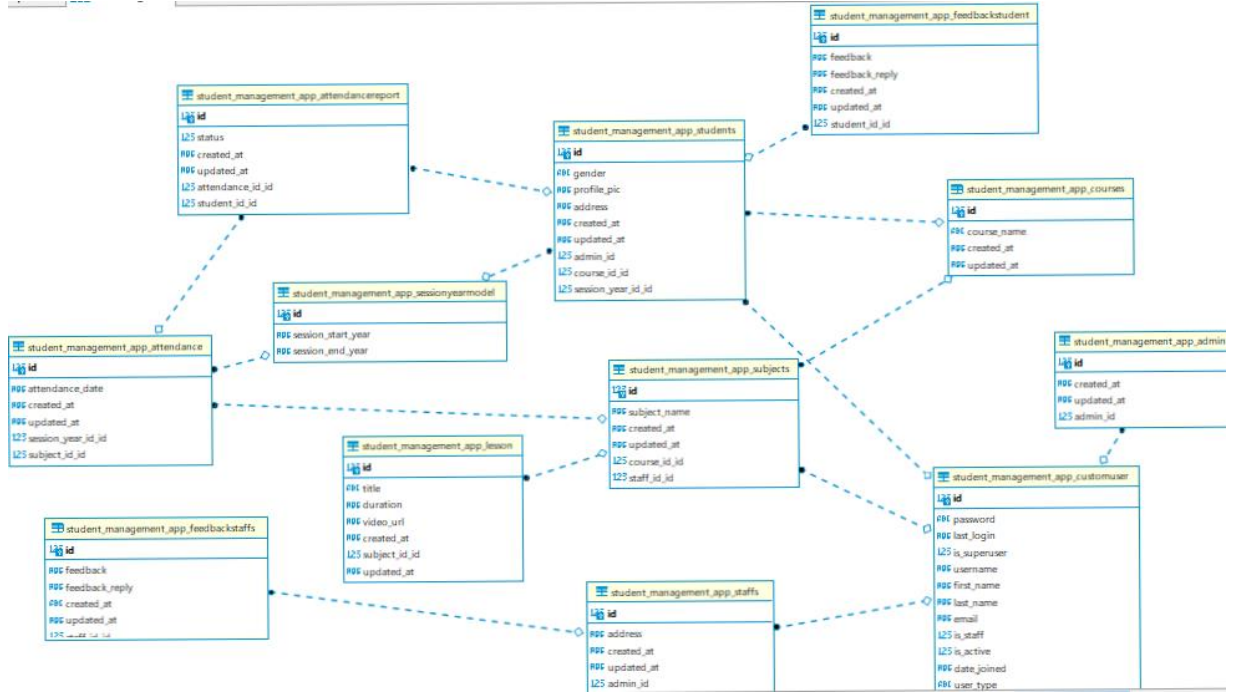
Bu kayda gidin: 1

UTF-8

3.1.Proje planlaması ve Veritabanı tasarlanması

Backend projemi veritabanını hazırlamadan önce projenin planlanması ve tasarımı çok önemliydi.Projenin tasarlanıp bunun için gerekli olan veritabanı modellerinin belirlenmesi aşamasından sonra uygun veritabanı tasarımına başladım.Bu projede veritabanı yönetim sistemi olarak SQLite kullandım.SQLite kullanımı gayet rahat olan bir veritabanı yönetim sistemi.Ayrıca tabloların modelini görebilmek için ise Dbeaver veritabanı yönetim aracını kullandım.

Veritabanı modelini bu şekilde oluşturdum.



Courses : Bu tabloda departmana kayıtlı olan kurslara ait bilgiler bulunmaktadır.Id,kurs adı,kurs oluşturulma tarihi,kurs güncellenme tarihi vb özellikler ekledim. Proje gerçek hayatta kullanılırsa kursa ait bilgiler daha fazla sütun şeklinde eklenilebilir.

Subject: Kurslara ait dersler için bir ana tablo özelliği taşıyor.Çünkü staff ,attendance,courses,lesson gibi tablolarla foreign key ilişkisi barındırıyor.Her ders bir eğitmen ve bir kursa bağlıdır.Aynı zamanda subject tablosunda kaydettiğim derslerin ayrıntılı halinin olarak kaydolması için subject tablosuyla lesson tablosunun foreign key ilişkisi önemli.

Lesson: Subject tablosunda kaydettiğimiz kurslara ait olan derslerin ayrıntılı olarak her bir derse ait bilgilerin(ait olduğu ders videosu dahil)tutulduğu bir tablodur.Eğitmen ve öğrenci tablolarında kullanılmak üzere tasarlanmıştır.

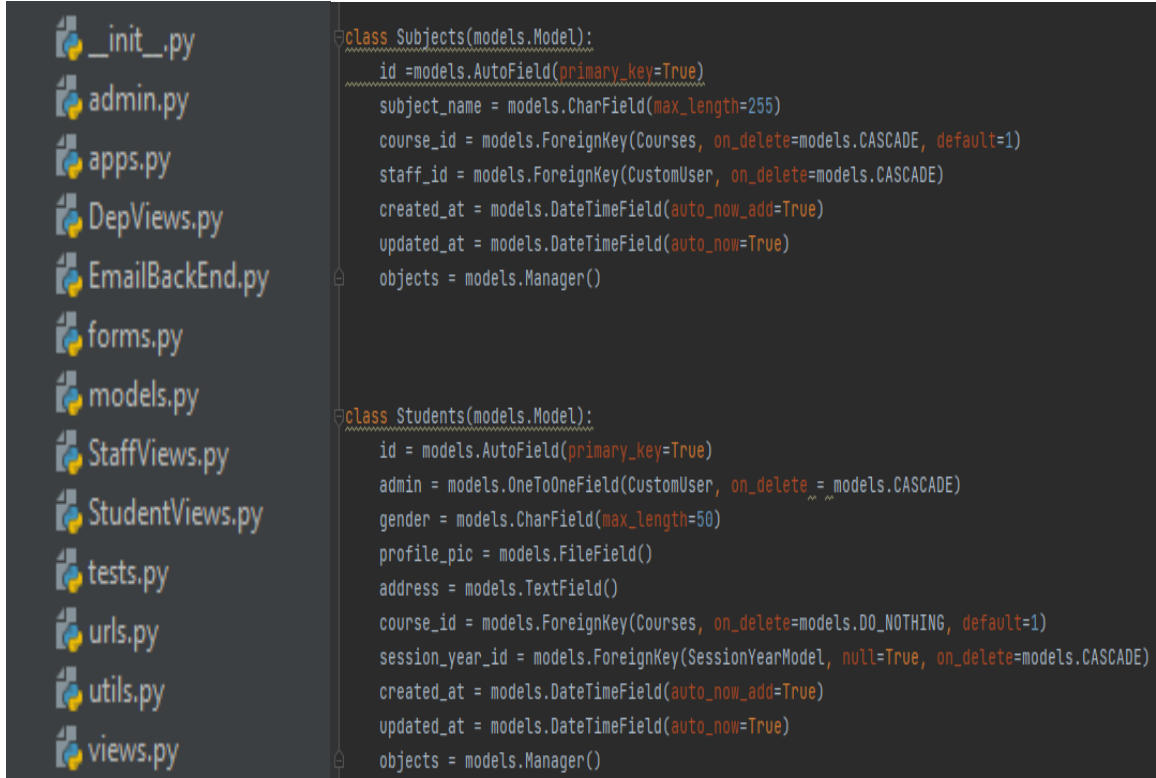
3.2.Pycharm kurulumu ve Modeller

Pycharm web geliştirme teknolojilerinde uygulamanın backend tarafında bir Python uygulaması geliştirme ortamı sağlar.Öncelikle Pycharm kendi sitesinden kurulum dosyasını indirdim ve Pycharm kurulumunu bitirdim.Gerekli olan paketleri(Django,setuptools vb) indirdim.Gerekli olan paketleri derleyiciye indirdikten sonra proje klasörünü komut isteminde oluşturdum ve sanal ortamı kurdum.Daha sonra projeyi derleyebilmek ve düzenleyebilmek için Pycharm ile proje dizinini kurdum.Ve ‘python manage.py runserver’ komutu ile uygulamayı çalıştırdım.

Projede settings.py kısmında databases kısmında veritabanı bağlantısı için veritabanı ismini tanıttım.Bu oluşturduğum veritabanı modelini projeye taşıyabilmek için modeller oluşturup migrate işlemini yapmam gerekiyordu.Modeller kısmında hazırladığım veritabanı şemasını projeye taşıyabilmek için bütün tabloları sütun alanlarıyla birlikte tanıttım.Bununla birlikte models.py kısmında tüm modelleri tek bir dosyada tuttum.

Models.py

Models.py ve örnek iki tane model



```
class Subjects(models.Model):
    id = models.AutoField(primary_key=True)
    subject_name = models.CharField(max_length=255)
    course_id = models.ForeignKey(Courses, on_delete=models.CASCADE, default=1)
    staff_id = models.ForeignKey(CustomUser, on_delete=models.CASCADE)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
    objects = models.Manager()

class Students(models.Model):
    id = models.AutoField(primary_key=True)
    admin = models.OneToOneField(CustomUser, on_delete=models.CASCADE)
    gender = models.CharField(max_length=50)
    profile_pic = models.FileField()
    address = models.TextField()
    course_id = models.ForeignKey(Courses, on_delete=models.DO_NOTHING, default=1)
    session_year_id = models.ForeignKey(SessionYearModel, null=True, on_delete=models.CASCADE)
    created_at = models.DateTimeField(auto_now_add=True)
    updated_at = models.DateTimeField(auto_now=True)
    objects = models.Manager()
```

3.3.Views ve modeller

Views ile modelleri kullanıp urls aracılığıyla web uygulaması geliştirilir.Bir fonksiyon gibi çalışırlar.Değerler alabilirler ve modelleri kullanıp sonuç üretirler.Modelden aldığımız verileri view ile kullanıp templates kısmına göndeririz ve bu sayede veriye views aracılığıyla ulaşabiliriz.

Views kısmını kullanıcı üyeliği için(StudentViews,StaffViews,DepViews)olarak üç ayrı başlıkta ele aldım.Öğrenci panelinde gözükecek olan viewler için StudentView,Eğitmen panelinde gözükecek olan viewler için StaffView, Admin panelinde gözükecek olan viewler için de DepView oluşturdum.

Ve urls kısmında da böyle route verdim.Ayrıca üyelik işlemleri için de views.py oluşturdum.İleride daha ayrıntılı olarak anlatacağım.Urlls yönlendirmelerinde herhangi bir yola gitmesi için View kullanılır.Urlls kısmında Student,Staff,Admin olarak üç kısımda incelenir.Admin tarafındaki yönlendirmelerde DepViews kullanılır.DepViews fonksiyonlarında ise Admin template dosyaları gösterilir.

3.3.1.StudentViews

Görmüş olduğunuz fotoğrafta da olduğu gibi öğrenci panelinde görünecek olan viewler StudentView üzerinden urls kısmına gidiyor.Bunun sebebi ise views.py kısmında belirttiğim kullanıcı üyelik için kullanıcı tipi alma kısmıyla birlikte kullanıcı mailinde student ifadesi geçtiğinde kullanıcı tipi 3(öğrenci) olarak veritabanına kaydoluyor.Ayrıca bu durumda route işlemi ile kullanıcı student urls kısmına gönderilir ve StudentView kısmına iletilir.

StudentViews’e ait lesson modelini kullandığımız viewler.

```
def student_view_lesson(request):
    student = Students.objects.get(admin=request.user.id)
    course = student.course_id

    subjects = Subjects.objects.filter(course_id=course)
    lessons = Lesson.objects.filter(subject_id=subjects)
    context = {
        "subjects": subjects,
        "lessons": lessons
    }
    return render(request, "student_template/student_view_lesson.html", context)

def student_view_lesson_post(request):
    if request.method != "POST":
        messages.error(request, "Gecersiz method")
        return redirect('student_view_lesson')
    else:
        subject_id = request.POST.get('subject')
        lesson_text = Lesson.objects.get(subject_id=subject_id)
        lesson_obj = Lesson.objects.get(subject_id=subject_id)

        url = lesson_obj.video_url
        url = url.replace("https://www.youtube.com/watch?v=", "https://www.youtube.com/embed/")
        lesson_obj.video_url = url
```

Bu ekran görüntüsünde StudentViews’e ait iki tane view örneği bulunmakta.Birincisi Öğrenci panelinde görüntülenen ders görüntüle sayfasına ait view.Reques.user.id ile kullanıcının öğrenci kimliği alınır.Daha sonra o öğrencinin hangi kurslara kayıtlı olduğu bulunur.Daha sonra kayıtlı olduğu kurslara ait olan dersler gelir.Subject tablosuyla lesson tablosu arasındaki ilişki lesson tablosundaki video url bilgisine ulaşabilmek için kurulur.Ders seçildikten sonra o dersin sayfası gelir ve eğitmen tarafından dersler (ilk ders,ikinci ders) gibi gözüktür ve ders videosu açılır.

3.3.2.StaffViews

StaffView ile öğretmen paneli dizaynını yaptım.Burada kurs,ders,soru,not ekleme gibi birbirinden farklı viewler staff kimliğine sahip kullanıcılar tarafından urls kısmında StaffViews tarafına iletilmesiyle kullanılır

StaffView'e ait örnek view fonksiyonları.

```
def staff_add_lesson(request):
    subjects = Subjects.objects.all()
    context = {
        "subjects": subjects
    }
    return render(request, 'staff_template/add_lesson_template.html', context)

def staff_add_lesson_save(request):
    if request.method != "POST":
        messages.error(request, "Geçersiz metod")
        return redirect('staff_add_lesson')
    else:
        title = request.POST.get('title')
        duration = request.POST.get('duration')
        video_url = request.POST.get('video_url')

        subject_id = request.POST.get('subject')
        subject = Subjects.objects.get(id=subject_id)

        try:
            lesson = Lesson(title=title, duration=duration, video_url=video_url, subject_id=subject)
            lesson.save()
            messages.success(request, "Lesson başarıyla eklendi")
```

3.3.3.DepViews

Panellerin bana göre en önemli kısmı admin paneli.Admin paneli için aslında bir veritabanı yönetim sistemi diyebilirimBunun zaten örneği phpMyAdmin.En çok ihtiyaç duyulan panel denilebilir.Siteyi yönetmek için tablolarda bulunan içerikler rahatça eklenebilir.Daha iyi bir tasarımı olması için Django'nun zaten olan admin panelini kendi admin paneli yapıma taşıdım.Ve projede kendi admin panelimi tasarladım.

DepView'e ait viewler.

```
def add_staff_save(request):
    if request.method != "POST":
        messages.error(request, "Geçersiz method ")
        return redirect('add_staff')
    else:
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        username = request.POST.get('username')
        email = request.POST.get('email')
        password = request.POST.get('password')
        address = request.POST.get('address')

        try:
            user = CustomUser.objects.create_user(username=username, password=password, email=email, first_name=first_name, last_name=last_name)
            user.staffs.address = address
            user.save()
            messages.success(request, "Başarıyla kaydedildi!")
            return redirect('add_staff')
        except:
            messages.error(request, "Başarısız kaydetme")
            return redirect('add_staff')
```

Bu view'de görüldüğü gibi sisteme kayıtlı eğitimcilerin dışında admin paneli ile tablolara register dışında admin paneliyle yeni kullanıcı kayıt etmeyi sağlayan bir view.Else kısmında olduğu gibi bir eğitime ait olan bilgiler request ile post yöntemi ile admin tarafından girilen değerler alınıyor.Daha sonra CustomUser modeline eklenmeye çalışıyor ve kaydediliyor.

3.4.URLS ve Route

Urls için bir güzergah ya da harita denilebilir.Route ise urls modelidir.Uygulamada viewler aracılığıyla urls üzerinden farklı görünümeler elde edilmesidir.Ben projemde viewde olduğu gibi urls kısmını da AdminUrls,StudentUrls,StaffUrls olarak üç kısma ayırdım.

Tanımlanan path aracılığıyla viewlere ulaşıyoruz viewler aracılığıyla da templates klasörüne ulaşıyoruz Burada urls bir köprü görevi görüyor.

Admin Urls kısmında route tanımlamaları.

```
path('admin_home/', DepViews.admin_home, name="admin_home"),
path('add_staff/', DepViews.add_staff, name="add_staff"),
path('add_staff_save/', DepViews.add_staff_save, name="add_staff_save"),
path('manage_staff/', DepViews.manage_staff, name="manage_staff"),
path('edit_staff/<staff_id>/', DepViews.edit_staff, name="edit_staff"),
path('edit_staff_save/', DepViews.edit_staff_save, name="edit_staff_save"),
path('delete_staff/<staff_id>/', DepViews.delete_staff, name="delete_staff"),
path('add_course/', DepViews.add_course, name="add_course"),
path('add_course_save/', DepViews.add_course_save, name="add_course_save"),
path('manage_course/', DepViews.manage_course, name="manage_course"),
path('edit_course/<course_id>/', DepViews.edit_course, name="edit_course"),
path('edit_course_save/', DepViews.edit_course_save, name="edit_course_save"),
path('delete_course/<course_id>/', DepViews.delete_course, name="delete_course"),
path('manage_session/', DepViews.manage_session, name="manage_session"),
path('add_session/', DepViews.add_session, name="add_session"),
path('add_session_save/', DepViews.add_session_save, name="add_session_save"),
path('edit_session/<session_id>', DepViews.edit_session, name="edit_session"),
path('edit_session_save/', DepViews.edit_session_save, name="edit_session_save"),
path('delete_session/<session_id>/', DepViews.delete_session, name="delete_session"),
path('add_student/', DepViews.add_student, name="add_student"),
path('add_student_save/', DepViews.add_student_save, name="add_student_save"),
path('edit_student/<student_id>', DepViews.edit_student, name="edit_student"),
path('edit_student_save/', DepViews.edit_student_save, name="edit_student_save"),
path('manage_student/', DepViews.manage_student, name="manage_student"),
path('delete_student/<student_id>/', DepViews.delete_student, name="delete_student"),
path('add_subject/', DepViews.add_subject, name="add_subject"),
```


Staff Urls kısmında route tanımlamaları.

```
path('staff_home/', StaffViews.staff_home, name="staff_home"),
path('staff_take_attendance/', StaffViews.staff_take_attendance, name="staff_take_attendance"),
path('get_students/', StaffViews.get_students, name="get_students"),
path('staff_add_lesson/', StaffViews.staff_add_lesson, name="staff_add_lesson"),
path('staff_add_lesson_save/', StaffViews.staff_add_lesson_save, name="staff_add_lesson_save"),
path('save_attendance_data/', StaffViews.save_attendance_data, name="save_attendance_data"),
path('staff_update_attendance/', StaffViews.staff_update_attendance, name="staff_update_attendance"),
path('get_attendance_dates/', StaffViews.get_attendance_dates, name="get_attendance_dates"),
path('get_attendance_student/', StaffViews.get_attendance_student, name="get_attendance_student"),
path('update_attendance_data/', StaffViews.update_attendance_data, name="update_attendance_data"),
path('staff_apply_leave/', StaffViews.staff_apply_leave, name="staff_apply_leave"),
path('staff_apply_leave_save/', StaffViews.staff_apply_leave_save, name="staff_apply_leave_save"),
path('staff_feedback/', StaffViews.staff_feedback, name="staff_feedback"),
path('staff_feedback_save/', StaffViews.staff_feedback_save, name="staff_feedback_save"),
path('staff_profile/', StaffViews.staff_profile, name="staff_profile"),
path('staff_profile_update/', StaffViews.staff_profile_update, name="staff_profile_update"),
path('staff_add_result/', StaffViews.staff_add_result, name="staff_add_result"),
path('staff_add_result_save/', StaffViews.staff_add_result_save, name="staff_add_result_save"),
```

Student Urls kısmında route tanımlamaları.

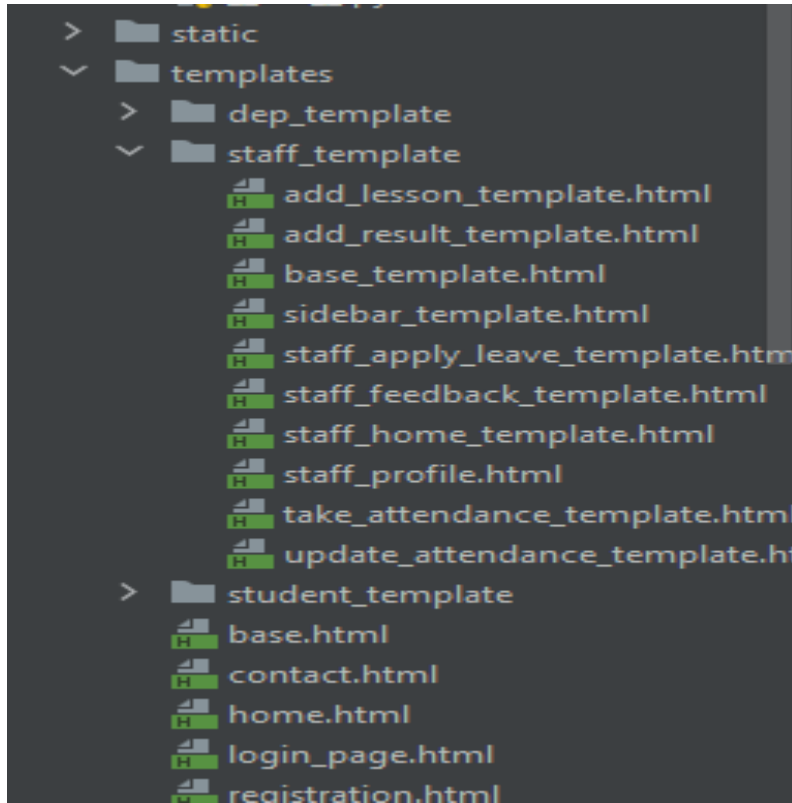
```
path('student_home/', StudentViews.student_home, name="student_home"),
path('student_view_lesson/', StudentViews.student_view_lesson, name="student_view_lesson"),
path('student_view_lesson_post/', StudentViews.student_view_lesson_post, name="student_view_lesson_post"),
path('student_view_attendance/', StudentViews.student_view_attendance, name="student_view_attendance"),
path('student_view_attendance_post/', StudentViews.student_view_attendance_post, name="student_view_attendance_post"),
path('student_apply_leave/', StudentViews.student_apply_leave, name="student_apply_leave"),
path('student_apply_leave_save/', StudentViews.student_apply_leave_save, name="student_apply_leave_save"),
path('student_feedback/', StudentViews.student_feedback, name="student_feedback"),
path('student_feedback_save/', StudentViews.student_feedback_save, name="student_feedback_save"),
path('student_profile/', StudentViews.student_profile, name="student_profile"),
path('student_profile_update/', StudentViews.student_profile_update, name="student_profile_update"),
path('student_view_result/', StudentViews.student_view_result, name="student_view_result"),
```


3.5.Templates Klasörü

Tüm buraya kadar olanlar projenin tam anlamıyla backend taffıydı.Fakat bu noktada backend tarafının frontend ayarlamalarını yapacağız.Admin panelinde gözüken şablonu oluşturan html dosyaları templates klasöründe tutulur ve viewlerde herhangi bir html dosyası kullanılacağı zaman templates klasörü içerisinde uygun olan html dosyası seçilir.

Templates klasöründeki CSS/Javascript kaynaklarını bootstrap üzerinden çektim.Halihazırda Django üzerinde url kısmında admin yazdığınızda karşınıza gelen admin paneli ekranını kendim oluşturduğum admin temalı template kısmına taşıdım.

Templates klasörü yapısı



4. Marmara Eğitim Portalı'nın Tasarımın

4.1. Kullanıcı anasayfası

Bu sayfa da sadelik ve rahat ulaşım çok önemli.Yukarıda görünen kısımda yer alan Giriş ve kayıt ol butonlarına kayıt olup sisteme giriş yapabilirsiniz.

Home.html dosyası

MARMARA EĞİTİM PORTALI BACKEND

Giriş Yap

Kayıt Ol

MARMARA

4.2. Kullanıcı Üyelik Sistemi


Projede önemli bir parça olan kullanıcı üyelik sistemini Backend tarafında olduğumuz için sadece paneller üzerinden gidiyoruz.Eğer ki backend ile frontend birleşseydi front end tarafından alınan hesaba giriş yaptıktan sonra bir buton ile ders vermek isteyenler ders verir(StudentView)aracılığıyla bir mail ile hem öğrenci hem de öğretmen olarak kullanılabilecekti.

Kayıt ol ekranı

ANASAYFA

Giriş yap

Kayıt ol




Kayıt ol

Kullanıcı giriş ekranı

ANASAYFA

Giriş

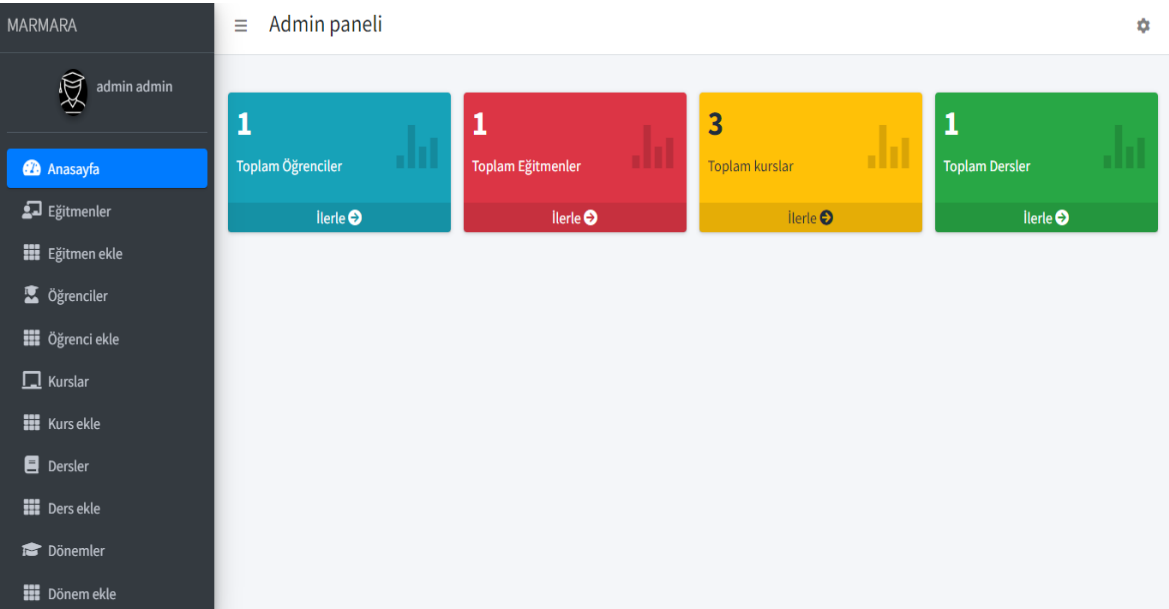


Gönder

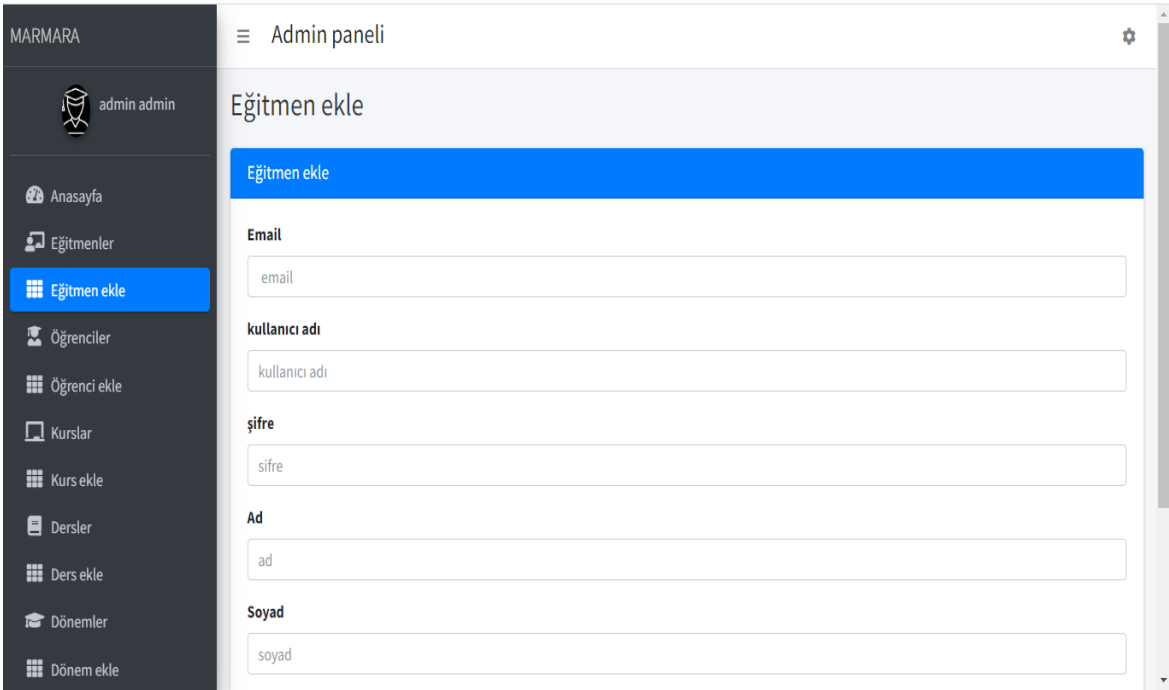
Kayıt ol

4.3. Admin paneli kullanımı

Admin panelinde bulunması gereken tablolara ait kısımlar var.Burada kurs ekleme,ders görüntüleme vb gibi durumları sağlıyor.Django kullanıcı yönetimini halihazırda bulunmasına karşın ben farklı bir admin teması kullandım.Admin panelini taşıdım.



Admin paneline ait görseller.




4.4. Student paneli ve Staff paneli

Öğrencilerin ve öğretmenlerin gördüğü studentview ve staffview dosyalarının urls kısmında staff_home ve student_home yönlendirmeleriyle view üzerinde bulunan view fonksiyonları çalışır.Studentview’de herhangi birşey eklemek yerine kayıtlı olduğu kurslar,eğitmenler veritabanından Django-ORM yapısıyla çekilir.Bunun dışında ekleme olarak mesaj gönderme vardır.Staffview buna karşılık olarak ders, soru,not ekleme gibi ekleme özelliklerine artı olarak kursları dersleri bağlı olan öğrencileri Django-ORM yapısıyla sağladık.

Student panelinden görüntüler.

MARMARA

ogrenci ogrenci

Anasayfa

Dersleri Görüntüle

Soru/not

Mesaj

Admin Paneli

Mesaj gönder

Mesaj gönder

Mesaj gönder

Mesaj gönder

Mesaj Geçmişi

#ID	Mesaj	Mesaj dönüşü
-----	-------	--------------

MARMARA

ogrenci ogrenci

Anasayfa

Dersleri Görüntüle

Soru/not

Mesaj

Admin Paneli

Dersleri İzle

Dersleri göster

Ders İsmi

sql

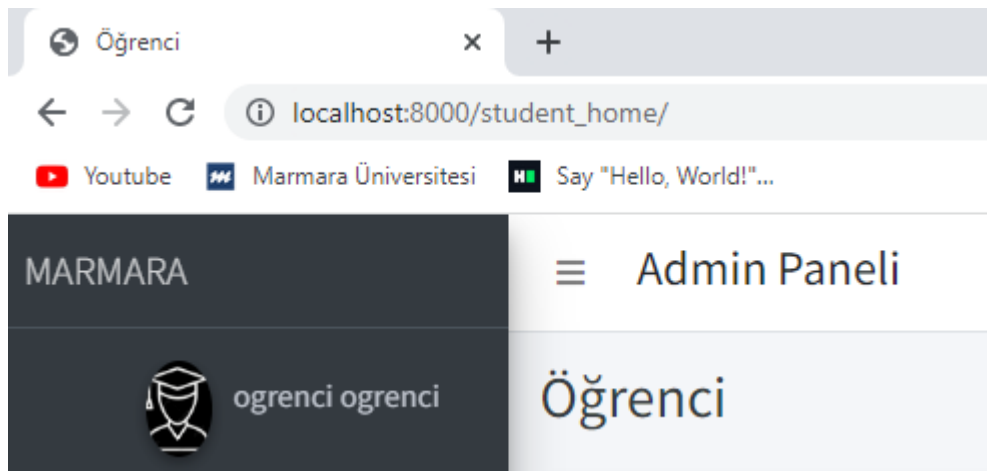
Ders bul

Staff panelinden görüntü.

The screenshot shows the 'Ders Ekle' (Add Course) form in the MARMARA Admin Panel. The left sidebar contains navigation links: 'ogretmen' (teacher), 'Anasayfa' (Home), 'Soru/not' (Question/Note), 'Soru güncelle' (Update Question), 'Ders videosu Ekle' (Add Course Video), and 'Mesaj' (Message). The main content area has a header 'Ders Ekle' and a sub-header 'Ders Ekle'. The form fields are: 'Ders İsmi' (Course Name) with a text input, 'Duration' with a text input, 'Video url girin' (Enter Video URL) with a text input, and 'Subject' with a dropdown menu showing 'sql'. A blue 'Ders Ekle' button is at the bottom.

4.5.Route-path, Not ve Email-Backend

Projede tüm bu viewlerin kayıtlı olduğu pathler bu view ve template dosyalarının yuvasıdır.View görünümü gibi path tanımlamaların doğru olması da önemlidir.Her view ve template için bir konumu bütün dizin içinden tanımlamaktır.



Bu sayfada da görüldüğü gibi kullanıcı login sayfasından sonra giriş yap düğmesine basınca eğer ki giriş bilgileri doğruysa urls ile path kısmında student_home şeklinde yönlendirilir.

StudentView tarafından görüntülenen kısım.

Oğrenci öğrenci

Anasayfa

Dersleri Görüntüle

Soru/not

Mesaj

Soru/not

Soru/not bildirimi

Ders

sql

Başlangıç tarihi

15.06.2022

Bitiş tarihi

23.06.2022

Soru/not

Eğitmen tarafından belli tarihleri içerisinde geçerli bir ders olup olmadığını gösteriyor. Ve buradaki yoklama sayesinde eğitmen bir öğrencinin derse girip girmediğini anlıyor.

MARMARA

Oğrenci öğrenci

Anasayfa

Dersleri Görüntüle

Soru/not

Mesaj

Admin Paneli

Yoklama Bilgisi

Yoklama Bilgisi: sql

Date : June 16, 2022
[Durum : Var]

Date : June 15, 2022
[Durum : Var]

Soru/not kısmında öncelikli olarak eğitmen tarafından istediği tarih aralığını belirtip bir mesaj yollamasıdır. Bunun amacı ise eğitmenin o tarih aralığında yüklediği videoya artık olarak soru,not,ders videosunu sisteme yükleyeceğini ya da sahip olduğu öğrenciye mesaj atacağını temsil eder.

Email-Backend kısmında ise amaç kullanıcı üyelik sistemini kullanıcı maili ile sağlamaktır. Frontend ile birleştirilmediği için tek bir hesap üzerinden hem öğrenci hem de öğretmen olabilme durumu söz konusu değil. Bundan dolayı projeyi öğrenci, öğretmen ve admin olarak üç farklı rolde inceledim.

Views.py dosyasında bu şekilde olan kod bloğuyla kayıt edilen kullanıcıyı kullanıcı tipine göre ayırıyor.

```
if user_type == CustomUser.STAFF:
    Staffs.objects.create(admin=user)
elif user_type == CustomUser.STUDENT:
    Students.objects.create(admin=user)
elif user_type == CustomUser.HOD:
    AdminHOD.objects.create(admin=user)
return render(request, 'login_page.html')
```

Bu kısımda ise kullanıcının üyelik tipine göre farklı url yönlendirme yapılıyor.

```
if user.user_type == CustomUser.STUDENT:
    return redirect('student_home/')
elif user.user_type == CustomUser.STAFF:
    return redirect('staff_home/')
elif user.user_type == CustomUser.HOD:
    return redirect('admin_home/')
```

Bu sayede kayıt olan bir kullanıcı mailindeki domaine göre belli bir kullanıcı tipi değeri alıyor(1=admin,2=staff,3=student). Daha sonra hesabına giriş yaparken mailinde üyelik tipine göre farklı url yönlendirmelerine gidiyor diğer üyelik tipinde bir hesap bir başka üyelik tipine ait panele gidemez.

5. Çıkarımlarım

Projenin sonucunda hem eğitim içeriği platformu özelliğini taşıyan,kullanıcı üyelik tipini ayıran,bir departman yönetim sistemi gibi özellikleri bir arada bulunduran kullanıcı dostu dizaynı ile rahat bir deneyim sunan tüm bunlara artı olarak soru/not,mesaj gibi öğretmenle öğrencinin birebir iletişim kurmasını sağlayan ve aynı zamanda frontend çalışmasıyla ve gerçek ödeme sistemiyle bir araya geldiğinde gerçek bir sistem oluşturacak bir backend sistemi geliştirdim.

Projemi geliştireceğim konular:

- Gerçekçi bir ödeme sistemi eklenerek gerçek hayatta kullanılabilir.
- Ödeme esnasında oluşabilecek güvenlik sorunlarının çözülmesi.
- Projenin bir okul için düşünürse farklı departmanların eklenmesi
- Frontend ile birleştirilmesi durumunda verilerin nesneler halinde templates klasöründe çekilip siteye kazandırılması.Ders verilerinin panelde değil de frontend yani sitenin kendisinde görünmesi.

6. Kaynaklar

- <https://www.djangoproject.com/start/>
- <https://www.tutorialspoint.com/django/index.html>
- <https://www.w3schools.com/django/>