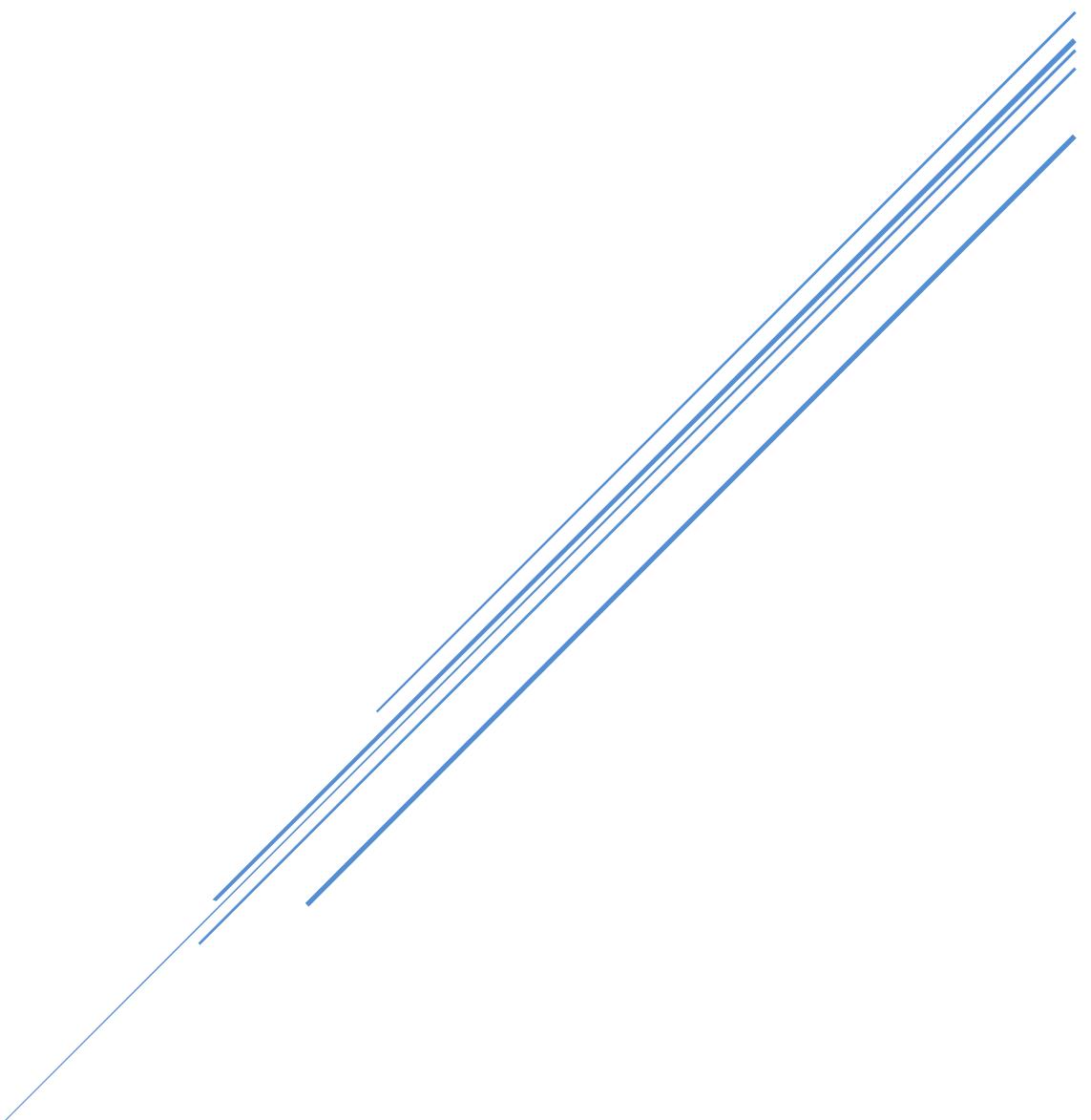


YAZILIM GELİŞTİRME 2

Bölüm Yönetim Yazılımı



Buse Nur Baş - 220229015
İbrahim Biner- 220229049

1. Proje Amacı ve İçeriği

Bu proje, üniversitelerin akademik bölmelerine yönelik olarak geliştirilen kapsamlı bir yönetim yazılımıdır. Yazılım, bölüm başkanları, öğretim elemanları ve idari personelin kullanıcı giriş sistemleri aracılığıyla güvenli şekilde sisteme erişmesini sağlamaktadır. Geliştirilen sistem sayesinde derslik ve ders programlarının planlaması merkezi bir yapı üzerinden gerçekleştirilebilmekte; haftalık ders çizelgeleri, öğretim elemanlarının uygunluk durumuna göre otomatik olarak oluşturulmaktadır.

Ayrıca sınav dönemlerinde karşılaşılan en büyük zorluklardan biri olan sınav oturma düzenlerinin oluşturulması da sistemin sunduğu rastgele atama algoritması ile kolaylaştırılmıştır. Bu özellik, sınıf kapasitelerini göz önünde bulundurarak öğrencilerin sınav salonlarına eşit ve adil biçimde dağıtılmmasını sağlar. Aynı zamanda gözetmen atamaları ve sınav listeleri de sistem üzerinden otomatik olarak belirlenebilir ve çıktı olarak alınabilir.

Proje, öğretim elemanlarına özel olarak bireysel ders programı oluşturulması, bu programların kapı isimliği formatında çıktı alınabilmesi gibi işlevler de sunarak kullanıcı deneyimini artırmaktadır. Tüm bu modüller sayesinde, akademik ve idari süreçler dijital ortamda etkin şekilde yönetilmekte, manuel iş yükü azaltılmakta ve kurum içi operasyonel verimlilik anlamlı ölçüde artırılmaktadır.

2. Materyal ve Yöntem

Proje genelinde Python programlama dili ve Django frameworkü kullanılmıştır. Veritabanı olarak django'ya yüksek uyumu nedeniyle SQLite tercih edilmiştir. Django içerisinde bulunan user ve form özellikleri genişletilerek kullanılmıştır.

2.1 Kullanıcı Giriş ve Yetkilendirme

Bu modül, sistemin güvenli ve verimli şekilde kullanılmasını sağlamak amacıyla giriş ve yetkilendirme mekanizması üzerine inşa edilmiştir. Sistem, farklı rollerle giriş yapan kullanıcıların (bölüm sekreteri, bölüm başkanı ve öğretim elemanı) sadece kendi görev ve yetki alanlarına uygun ekranlara ve işlemelere erişebilmesini garanti altına almaktadır. Bu amaçla Django web geliştirme çerçevesi kullanılarak role dayalı yetkilendirme sistemi entegre edilmiştir.

Kullanıcılar sisteme giriş yaptıklarında, sistem otomatik olarak kullanıcının rolünü algılamakta ve ona uygun bir panel sunmaktadır. Örneğin, bölüm sekreteri ders programlarını düzenleyebilir ve sınav oturma düzeni oluşturabilirken, öğretim elemanı yalnızca kendi ders programını görüntüleyebilir. Bölüm başkanı ise tüm sistem üzerinde yetkiye sahip olup genel planlama ve denetleme işlemlerini yürütebilmektedir.

Bu yapı sayesinde hem sistemin güvenliği artmakta hem de kullanıcı deneyimi kişilere özel ekranlar ile zenginleştirilmektedir. Her kullanıcı yalnızca ihtiyaç duyduğu bilgiye ve işlevlere ulaşarak daha hızlı ve odaklı bir şekilde çalışabilmektedir.

2.1.1 Özellik ve Fonksiyonlar



1. Login Fonksiyonu

Projede kullanılan login fonksiyonu django CustomUser modelini özelleştirilerek oluşturulan Bu fonksiyon, sistemde kullanıcıların giriş işlemlerini gerçekleştirmek amacıyla oluşturulmuştur. Django'nun AuthenticationForm sınıfı kullanılarak, kullanıcıdan gelen giriş bilgileri (kullanıcı adı ve parola) doğrulanmaktadır.

Fonksiyonun işleyişi şu adımlardan oluşmaktadır:

1. İstek Türü Kontrolü:

Fonksiyon ilk olarak, HTTP isteğinin POST olup olmadığını kontrol eder. Eğer istek POST ise, bu durum kullanıcının giriş formunu doldurup gönderdiğini ifade eder.

2. Form Nesnesinin Oluşturulması:

AuthenticationForm, gelen kullanıcı giriş verileri (request.POST) ile birlikte çağrılr ve bir form nesnesi oluşturulur. Bu form, Django'nun yerleşik kullanıcı doğrulama formudur.

3. Formun Geçerliliği Kontrolü:

form.is_valid() metodu ile formun geçerli olup olmadığı kontrol edilir. Yani kullanıcı adı ve parola doğru girilmiş mi, sistemde böyle bir kullanıcı var mı gibi doğrulamalar yapılır.

4. Kullanıcının Sisteme Giriş'i:

Eğer form geçerliyse, form.get_user() metodu kullanılarak giriş yapan kullanıcı nesnesi elde edilir ve login() fonksiyonu ile kullanıcı oturumu başlatılır. Bu işlem, kullanıcının sisteme başarılı bir şekilde giriş yaptığı gösterir.

5. Yönlendirme İşlemi:

Giriş başarılı olduktan sonra kullanıcı, sistemde tanımlı olan home adlı sayfaya yönlendirilir. Bu genellikle ana kontrol paneli veya kullanıcının rolüne özel gösterilecek ilk sayfadır.

6. GET İsteği Durumu:

Eğer HTTP isteği POST değilse (yani kullanıcı formu henüz göndermemişse), boş bir AuthenticationForm nesnesi oluşturularak giriş sayfası (login.html) form ile birlikte kullanıcıya sunulur.

The screenshot shows a user registration form titled "Kullanıcı Kaydı". The form includes fields for "Kullanıcı Adı" (Username), "E-posta" (Email), "Şifre" (Password), and "Şifre (Tekrar)" (Password Confirmation). A dropdown menu for "Rol" (Role) is set to "Seçiniz" (Select). A large green button at the bottom is labeled "Kullanıcıyı Kaydet" (Register User). Below the form, a link says "Hesabı Test Et ... Giriş Ekranı" (Test Account ... Login Screen).

Kullanıcı Adı:	
E-posta:	
Şifre:	
Şifre (Tekrar):	
Rol: Seçiniz	
Kullanıcıyı Kaydet	

Hesabı Test Et ... [Giriş Ekranı](#)

2. Kullanıcı Ekleme Fonksiyonu

user_register fonksiyonu, sistemde yalnızca belirli yetkiye sahip kullanıcıların (Bölüm Başkanı ve Bölüm Sekreteri) yeni kullanıcı kaydı oluşturabilmesini sağlamak amacıyla geliştirilmiştir. Bu yapı, sistemin güvenliğini artırırken, kullanıcıların sadece kendi yetki alanları dahilinde işlem yapmalarını garanti eder.

Fonksiyonun adım adım işleyişi aşağıdaki gibidir:

1. Giriş Zorunluluğu (login_required)

Fonksiyonun başında @login_required dekoratörü kullanılarak, bu işlemin sadece sisteme giriş yapmış kullanıcılar tarafından erişilebilir olması sağlanmıştır. Yani oturum açmamış kullanıcılar bu fonksiyona erişemez.

2. Rol Doğrulaması

Giriş yapan kullanıcının rolü kontrol edilir. Eğer kullanıcı, "bolum_baskani" veya "bolum_sekreteri" rollerinden birine sahip değilse, HttpResponseRedirectForbidden ile işlem engellenir ve kullanıcıya bu işlemi yapmaya yetkisi olmadığını bildiren bir mesaj gösterilir.

3. POST İsteği Durumu (Form Gönderilmişse)

Eğer gelen istek POST tipindeyse, bu durum formun doldurularak gönderildiğini gösterir. Bu durumda:

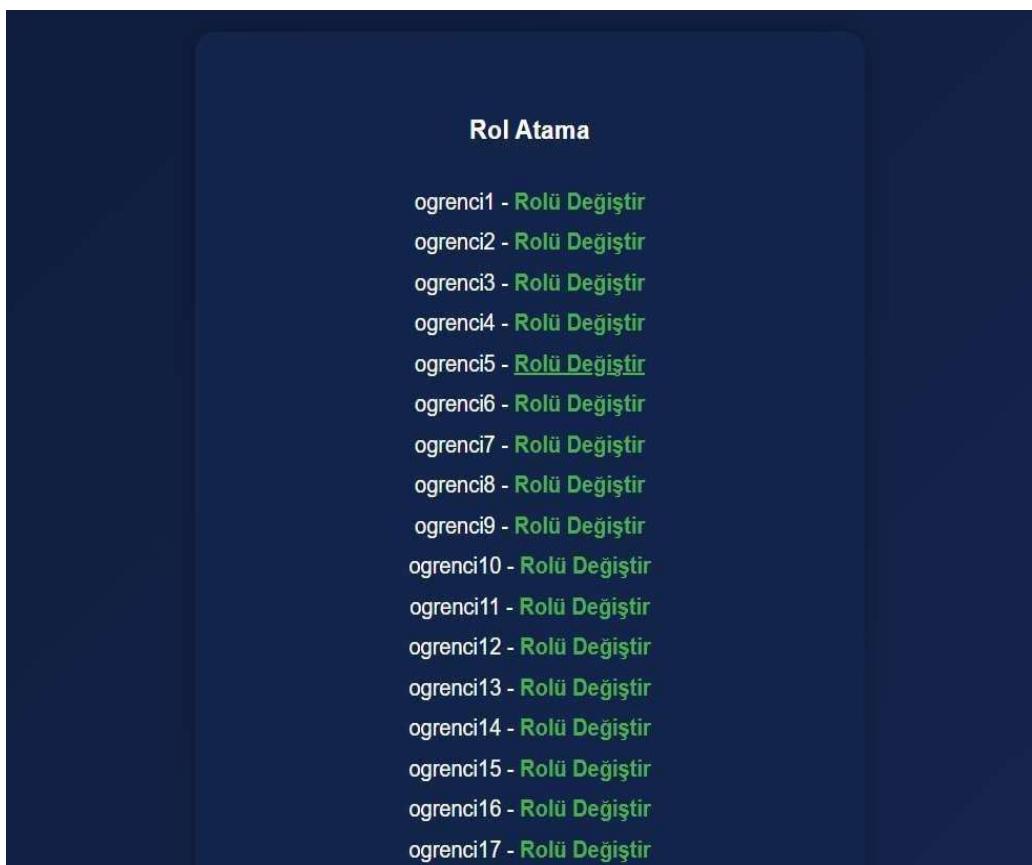
- CustomUserCreationForm adlı özel kullanıcı kayıt formu, request.POST verileriyle ve giriş yapan kullanıcının bilgisiyle oluşturulur.
- form.is_valid() ile formun doğruluğu kontrol edilir (tüm alanlar doğru doldurulmuş mu, kullanıcı adı daha önce alınmış mı vs.).
- Form geçerliyse, form.save(commit=False) ile kullanıcı nesnesi oluşturulur ancak henüz veritabanına kaydedilmez.
- Giriş yapan kullanıcı eğer Bölüm Sekreteri ise, sistem güvenliği gereği yeni kullanıcıya yalnızca "ogrenci" rolü atamasına izin verilir. Böylece sekreter rol seçemez, sadece öğrenci ekleyebilir.
- Kullanıcı kaydedildikten sonra sistemin ana sayfasına (home) yönlendirme yapılır.

4. GET İsteği Durumu (Form Henüz Gönderilmemişse)

Eğer sayfa ilk defa açılıyorsa (GET isteği), boş bir form oluşturulur. Burada request_user=request.user parametresi ile, formun nasıl görüntüleneceği kullanıcının rolüne göre ayarlanır. Örneğin, Bölüm Başkanı farklı roller atayabilecekken, Sekreter yalnızca öğrenci ekleyebilecektir.

5. Formun Sayfada Gösterilmesi

Son adımda register.html adlı şablon sayfasına form nesnesi gönderilir. Böylece kullanıcı formu doldurarak yeni kullanıcı ekleme işlemini gerçekleştirebilir.



3. Kullanıcı Yetkilendirme Fonksiyonu

assign_role_to_user fonksiyonu, sistemde yalnızca bölüm başkanının başka bir kullanıcıya rol atayabilmesini sağlamak amacıyla geliştirilmiştir. Bu yapı, yönetimsel işlemlerin sadece yetkili kişiler tarafından yapılmasını sağlayarak sistem güvenliğini artırır.

Fonksiyonun işleyişi adım adım şu şekildedir:

1. Yetki Kontrolü

Fonksiyonun başında, sisteme giriş yapmış kullanıcının rolü kontrol edilir. Eğer kullanıcı bolum_baskani değilse, sistem bu sayfaya erişimini engeller ve HttpResponseRedirectForbidden ile kullanıcıya "Bu sayfaya sadece bölüm başkanı erişebilir." uyarısı gösterilir.

2. Kullanıcı Bilgilerinin Getirilmesi

Rol atanmak istenen kullanıcıyı sistemdeki veritabanından çekmek için get_object_or_404 fonksiyonu kullanılır. Böylece eğer user_id ile belirtilen kullanıcı bulunamazsa, sistem otomatik olarak 404 (Bulunamadı) hatası döndürür.

3. Formun Gönderilmesi Durumu (POST)

Eğer sayfaya gelen istek POST tipindeyse, bu formun doldurularak gönderildiği anlamına gelir:

- RoleSelectionForm adlı form, POST verileri ile ve seçilen kullanıcıya (instance=user) bağlı olarak oluşturulur.
- form.is_valid() kontrolü ile formun kurallara uygun şekilde doldurulup doldurulmadığı kontrol edilir.
- Form geçerliyse form.save() ile kullanıcıya seçilen yeni rol atanır ve bilgiler veritabanına kaydedilir.
- İşlem tamamlandıktan sonra sistem, tekrar rol atama sayfasına (assign_roles) yönlendirir.

4. Sayfa İlk Açıldığında (GET)

Eğer istek GET tipindeyse (yani sayfa ilk kez açılmışsa), kullanıcıya ait mevcut bilgilerle RoleSelectionForm oluşturulur. Böylece sistem yönetici mevcut rolü görebilir ve değişiklik yapabilir.

5. Şablonun Gösterilmesi

Son olarak role_selection.html adlı HTML şablon dosyası üzerinden form ve ilgili kullanıcı bilgileri sayfaya gönderilir. Böylece bölüm başkanı kullanıcıya ait rolü bir arayüz üzerinden seçip kaydedebilir.

2.2 Ders Programı Hazırlama

Bu modül, bölümdeki derslerin haftanın günlerine ve saat aralıklarına göre uygun dersliklere atanmasını sağlamak amacıyla geliştirilmiştir. Sistem, öğretim elemanlarına derslerin atanması, derslik uygunluklarının kontrolü ve çakışmaların önlenmesi gibi işlevleri yerine getirmektedir.

Modülün altyapısında Django web geliştirme çatısı ve SQLite veritabanı kullanılmıştır. Bu sayede hem veri bütünlüğü korunmuş hem de işlemler hızlı ve güvenli bir şekilde gerçekleştirilmiştir.

Kullanıcılar (bölüm sekreteri veya bölüm başkanı), kullanıcı dostu arayızlar sayesinde dersleri seçerek uygun zaman dilimlerine ve dersliklere kolayca atayabilmektedir. Aynı zamanda sistem, girilen bilgiler doğrultusunda otomatik çakışma kontrolü yaparak aynı anda aynı sınıfta birden fazla ders atamasını engellemektedir. Öğretim elemanlarının haftalık ders yükleri ve uygunluklarına göre sistem önerilerde bulunabilmektedir.

Modül ayrıca:

- Aynı anda birden fazla öğretim elemanına ait derslerin çakışmaması için kontrol sağlar,
- Sınıfların kapasite ve uygunluk bilgilerini dikkate alır,
- Haftalık ders programının tablo şeklinde görsel olarak sunulmasına olanak tanır,
- Programların gerektiğinde güncellenebilmesi için esnek veri yönetimi sunar.

2.2.1 Fonksiyonlar ve Özellikler

Yeni Ders Ekle

Kod:

Ad:

Ogrenci sayısı:

Kredi:

Bolum:

Kaydet

[← Ders Listesine Geri Dön](#)

4. Ders Ekleme Fonksiyonu

ders_ekle fonksiyonu, sistemde yeni bir ders eklemek amacıyla kullanılan bir fonksiyondur. Bu işlem, bölüm başkanı ve bölüm sekreterinin ders programına yeni dersler eklemelerini sağlar.

1. Form Gönderimi (POST)

Fonksiyon, gelen isteğin POST olup olmadığını kontrol eder. Eğer istek POST tipindeyse, bu kullanıcı tarafından form verilerinin gönderildiği anlamına gelir. Bu durumda, fonksiyon şu adımları izler:

- DersForm adlı form, kullanıcının gönderdiği POST verileriyle oluşturulur.
- form.is_valid() kontrolü ile formun kurallara uygun şekilde doldurulup doldurulmadığı kontrol edilir.

Eğer form geçerli ise:

- form.save() ile ders verileri veritabanına kaydedilir.
- Başarılı bir şekilde ders eklenmesinin ardından, kullanıcı ders listesi sayfasına (ders_listesi) yönlendirilir.

2. Sayfa İlk Açıldığında (GET)

Eğer gelen istek GET tipindeyse (yani sayfa ilk kez açıldığında), fonksiyon şu adımları izler:

- Boş bir DersForm oluşturulur. Bu form, kullanıcıya ders eklemek için gerekli alanları sunar.

Bu durumda, kullanıcının veri girmesi için form boş olarak gösterilir.

3. Şablonun Gösterilmesi

Form, ders_ekle.html adlı HTML şablon dosyasına gönderilir. Şablon, formu kullanıcıya gösterir ve kullanıcıdan ders bilgilerini toplar. Formdaki bilgilerin doğruluğu kontrol edilip ders ekleme işlemi yapılacaktır.

Yeni Derslik Ekle

Ad:

Kapasite:

Kaydet

[← Derslik Listesine Geri Dön](#)

5. Derslik Ekleme Fonksiyonu

derslik_ekle fonksiyonu, sistemde yeni bir derslik eklemek amacıyla kullanılan bir fonksiyondur. Bu işlem, bölüm başkanı ve bölüm sekreterinin dersliklere yeni derslikler eklemelerini sağlar.

Fonksiyonun işleyişi adım adım şu şekildedir:

1. Form Gönderimi (POST)

Fonksiyon, gelen isteğin POST olup olmadığını kontrol eder. Eğer istek POST tipindeyse, bu kullanıcı tarafından form verilerinin gönderildiği anlamına gelir. Bu durumda, fonksiyon şu adımları izler:

- DerslikForm adlı form, kullanıcının gönderdiği POST verileriyle oluşturulur.
- `form.is_valid()` kontrolü ile formun kurallara uygun şekilde doldurulup doldurulmadığı kontrol edilir.

Eğer form geçerli ise:

- `form.save()` ile derslik verileri veritabanına kaydedilir.
- Başarılı bir şekilde derslik eklenmesinin ardından, kullanıcı derslik listesi sayfasına (`derslik_listesi`) yönlendirilir.

2. Sayfa İlk Açıldığında (GET)

Eğer gelen istek GET tipindeyse (yani sayfa ilk kez açıldığında), fonksiyon şu adımları izler:

- Boş bir DerslikForm oluşturulur. Bu form, kullanıcıya derslik eklemek için gerekli alanları sunar.

Bu durumda, kullanıcının veri girmesi için form boş olarak gösterilir.

3. Şablonun Gösterilmesi

Form, `derslik_ekle.html` adlı HTML şablon dosyasına gönderilir. Şablon, formu kullanıcıya gösterir ve kullanıcıdan derslik bilgilerini toplar. Formdaki bilgilerin doğruluğu kontrol edilip derslik ekleme işlemi yapılacaktır.

Yeni Ders Programı Ekle

Ders

Gün

Başlangıç Saati

Bitiş Saati

Öğretim Elemanı

Derslik

Kaydet

← Ders Programına Geri Dön

6. Ders Programı Ekleme Fonksiyonu

ders_programi_ekle fonksiyonu, kullanıcıların ders programına yeni bir ders eklemelerine olanak tanır. Bu fonksiyon, derslik ve akademisyen çalışmaları kontrol ederek sistemdeki hatalı veri girişlerini engeller ve dersin doğru bir şekilde programa eklenmesini sağlar.

Fonksiyonun işleyişi adım adım şu şekildedir:

1. Form Gönderimi (POST)

Fonksiyon, gelen isteğin POST olup olmadığını kontrol eder. Eğer istek POST tipindeyse, bu kullanıcı tarafından form verilerinin gönderildiği anlamına gelir. Bu durumda, fonksiyon şu adımları izler:

- DersProgramiForm adlı form, kullanıcının gönderdiği POST verileriyle oluşturulur.
- form.is_valid() kontrolü ile formun kurallara uygun şekilde doldurulup doldurulmadığı kontrol edilir.

Eğer form geçerli ise, formun içeriği veriler şu şekilde işlenir:

- **Derslik:** Kullanıcının seçtiği derslik verisi form.cleaned_data['derslik'] ile alınır.
- **Ders:** Seçilen ders bilgisi form.cleaned_data['ders'] ile alınır.
- **Öğretim Elemanı:** form.cleaned_data['ogretim_elemani'] ile seçilen öğretim elemanı alınır.
- **Gün:** form.cleaned_data['gun'] ile dersin hangi günde yapılacağı belirlenir.
- **Başlangıç ve Bitiş Saati:** form.cleaned_data['baslangic_saati'] ve form.cleaned_data['bitis_saati'] ile dersin başlama ve bitiş saatleri alınır.

2. Çakışma Kontrolleri

Fonksiyon, üç farklı çakışma kontrolünü yapar:

1. Derslik Çakışması Kontrolü

- **Amaç:** Aynı derslikte aynı gün ve saatte başka bir ders olup olmadığını kontrol etmek.

- DersProgrami.objects.filter(gun=gun, derslik=derslik, baslangic_saati_lt=bitis_saati, bitis_saati_gt=baslangic_saati) sorgusu ile, belirtilen gün ve saatte bu derslikte başka bir dersin olup olmadığı kontrol edilir.
- Eğer çakışma varsa, formda bir hata mesajı eklenir: "Seçilen derslikte belirtilen gün ve saatlerde başka bir ders bulunmaktadır."

2. Akademisyen Çakışması Kontrolü

- **Amaç:** Aynı öğretim elemanın belirtilen gün ve saatte başka bir dersi olup olmadığını kontrol etmek.
- DersProgrami.objects.filter(gun=gun, ogretim_elemani=ogretim_elemani, baslangic_saati_lt=bitis_saati, bitis_saati_gt=baslangic_saati) sorgusu ile, seçilen öğretim elemanın belirtilen saat aralığında başka bir dersi olup olmadığı kontrol edilir.
- Eğer çakışma varsa, formda bir hata mesajı eklenir: "Bu akademisyenin belirtilen gün ve saatlerde başka bir dersi bulunmaktadır."

3. Kapasite Kontrolü

- **Amaç:** Dersin öğrenci sayısının seçilen derslik kapasitesini aşması durumunda bir hata mesajı göstermek.
- kapasite_asimi = ders.ogrenci_sayisi > derslik.kapasite kontrolü ile, dersin öğrenci sayısının derslik kapasitesini aşması durumu kontrol edilir.
- Eğer kapasite aşılmışsa, formda bir hata mesajı eklenir: "Bu dersi alan öğrenci sayısı (X), seçilen dersliğin kapasitesini (Y) aşıyor."

3. Hata Durumu

Eğer yukarıdaki çakışma kontrolleri sonucu herhangi bir hata tespit edilirse, formun ilgili alanına hata mesajı eklenir ve form tekrar kullanıcıya gösterilir. Kullanıcı, hatalarını düzelterek formu yeniden gönderebilir.

4. Başarılı Durumda Ders Kaydetme

Eğer çakışma veya kapasite aşımı hatası yoksa, form geçerli sayılır ve form.save() ile ders programı veritabanına kaydedilir. Bu işlem tamamlandıktan sonra kullanıcı, ders programı listesi sayfasına (ders_programi_list) yönlendirilir.

5. Sayfa İlk Açıldığında (GET)

Eğer gelen istek GET tipindeyse, fonksiyon şu adımları izler:

- Boş bir DersProgramiForm oluşturulur. Bu form, kullanıcıya ders programı eklemek için gerekli alanları sunar.

Bu durumda, kullanıcının veri girmesi için form boş olarak gösterilir.

6. Şablonun Gösterilmesi

Form, ders_programi_ekle.html adlı HTML şablon dosyasına gönderilir. Şablon, formu kullanıcıya gösterir ve kullanıcıdan ders programı bilgilerini toplar. Formdaki bilgilerin doğruluğu kontrol edilip ders programı ekleme işlemi yapılacaktır.

7. Ders Programı Düzenleme Fonksiyonu

ders_programi_duzenle fonksiyonu, var olan bir ders programını düzenlemeye olanak tanır. Bu fonksiyon, kullanıcıların mevcut bir dersin bilgilerini güncelllemelerine olanak sağlar. Derslik ve öğretim elemanı çakışmaları kontrol edilerek, dersin doğru bir şekilde programdan güncellenmesini sağlar.

1. Ders Programının Getirilmesi

Fonksiyon, gelen pk (primary key) parametresi ile belirtilen ders programını veritabanından çeker. Bu işlem, get_object_or_404(DersProgrami, pk=pk) fonksiyonu ile yapılır. Eğer verilen pk ile bir ders programı bulunamazsa, Django otomatik olarak 404 hatası döndürecektir.

2. Form Gönderimi (POST)

Fonksiyon, gelen isteğin POST olup olmadığını kontrol eder. Eğer istek POST tipindeyse, bu kullanıcı tarafından form verilerinin gönderildiği anlamına gelir. Bu durumda, fonksiyon şu adımları izler:

- DersProgramiForm, kullanıcının gönderdiği POST verileri ile oluşturulur. Bu form ayrıca instance=program parametresi ile, var olan ders programının mevcut verileriyle doldurulur.

- `form.is_valid()` kontrolü ile formun kurallara uygun şekilde doldurulup doldurulmadığı kontrol edilir.

Eğer form geçerli ise, formun içeriği veriler şu şekilde işlenir:

- **Derslik:** Kullanıcının seçtiği derslik verisi `form.cleaned_data['derslik']` ile alınır.
- **Ders:** Seçilen ders bilgisi `form.cleaned_data['ders']` ile alınır.
- **Öğretim Elemanı:** `form.cleaned_data['ogretim_elemani']` ile seçilen öğretim elemanı alınır.
- **Gün:** `form.cleaned_data['gun']` ile dersin hangi günde yapılacağı belirlenir.
- **Başlangıç ve Bitiş Saati:** `form.cleaned_data['baslangic_saati']` ve `form.cleaned_data['bitis_saati']` ile dersin başlama ve bitiş saatleri alınır.

3. Çakışma Kontrolleri

Fonksiyon, üç farklı çakışma kontrolünü yapar:

1. Derslik Çakışması Kontrolü (Kendisi Hariç)

- **Amaç:** Aynı derslikte, aynı gün ve saatte başka bir dersin olup olmadığını kontrol etmek.
- `DersProgrami.objects.filter(gun=gun, derslik=derslik, baslangic_saati_lt=bitis_saati, bitis_saati_gt=baslangic_saati)` sorgusu ile, belirtilen gün ve saatte derslikte başka bir dersin olup olmadığı kontrol edilir.
- `exclude(pk=program.pk)` ile, düzenlemeyi yapan ders programının çakışma kontrolünden hariç tutulması sağlanır.
- Eğer çakışma varsa, formda bir hata mesajı eklenir: "Seçilen derslikte belirtilen gün ve saatlerde başka bir ders bulunmaktadır."

2. Akademisyen Çakışması Kontrolü (Kendisi Hariç)

- **Amaç:** Aynı öğretim elemanın belirtilen gün ve saatte başka bir dersi olup olmadığını kontrol etmek.
- `DersProgrami.objects.filter(gun=gun, ogretim_elemani=ogretim_elemani, baslangic_saati_lt=bitis_saati, bitis_saati_gt=baslangic_saati)` sorgusu ile, seçilen öğretim elemanın belirtilen saat aralığında başka bir dersi olup olmadığı kontrol edilir.

- `exclude(pk=program.pk)` ile, düzenlemeyi yapan ders programının çakışma kontrolünden hariç tutulması sağlanır.
- Eğer çakışma varsa, formda bir hata mesajı eklenir: "Bu akademisyenin belirtilen gün ve saatlerde başka bir dersi bulunmaktadır."

3. Kapasite Kontrolü

- **Amaç:** Dersin öğrenci sayısının seçilen derslik kapasitesini aşması durumunda bir hata mesajı göstermek.
- `kapasite_asimi = ders.ogrenci_sayisi > derslik.kapasite_kontrolü` ile, dersin öğrenci sayısının derslik kapasitesini aşması durumu kontrol edilir.
- Eğer kapasite aşılmışsa, formda bir hata mesajı eklenir: "Bu dersi alan öğrenci sayısı (X), seçilen dersliğin kapasitesini (Y) aşıyor."

4. Hata Durumu

Eğer yukarıdaki çakışma kontrolleri sonucu herhangi bir hata tespit edilirse, formun ilgili alanına hata mesajı eklenir ve form tekrar kullanıcıya gösterilir. Kullanıcı, hatalarını düzelterek formu yeniden gönderebilir.

5. Başarılı Durumda Ders Güncelleme

Eğer çakışma veya kapasite aşımı hatası yoksa, form geçerli sayılır ve `form.save()` ile ders programı güncellenir. Bu işlem tamamlandıktan sonra kullanıcı, ders programı listesi sayfasına (`ders_programi_list`) yönlendirilir.

Ders Programı						
Ders	Gün	Saat	Öğretim Elemanı	Derslik	Sil	
dhm - deneme Düzen	Pazartesi	9 a.m. - 11 a.m.	sondenemeuser	Z054	<button>Düzenle</button>	<button>Sil</button>
YZM 101 - Yazılım Mühendisliğine Giriş	Pazartesi	10 a.m. - midnight	ibrahim	108	<button>Düzenle</button>	<button>Sil</button>
YZM 623 - Bilgisayar Organizasyonu	Pazartesi	noon - 1 p.m.	sondenemeuser	Z054	<button>Düzenle</button>	<button>Sil</button>
YZM205 - Veri Yapıları	Pazartesi	1 p.m. - 3 p.m.	kullanıcı99	108	<button>Düzenle</button>	<button>Sil</button>
deneme - deneme1	Pazartesi	1 p.m. - 4 p.m.	ibrahim	Z054	<button>Düzenle</button>	<button>Sil</button>
YZM205 - Veri Yapıları	Çarşamba	10 a.m. - 11 a.m.	ibrahim	Z054	<button>Düzenle</button>	<button>Sil</button>
YZM 623 - Bilgisayar Organizasyonu	Perşembe	10 a.m. - noon	ibrahim	Z054	<button>Düzenle</button>	<button>Sil</button>
CENG 205 - Mikro İşlemci Tasarımı	Cuma	8 a.m. - 9 a.m.	kullanıcı99	Z054	<button>Düzenle</button>	<button>Sil</button>

[Yeni Ders Ekle](#)

8. Ders Programı Listeleme Fonksiyonu

ders_programi_list fonksiyonu, veritabanındaki ders programlarını alır, bu programları gün sırasına ve saat sırasına göre sıralar, ve kullanıcıya gösterir. Fonksiyon, derslerin haftanın hangi gününde ve hangi saatte olduğunu düzenli bir şekilde sunar.

1. Gün Sırasının Tanımlanması

Fonksiyonun başında, haftanın günleri için bir sıralama yapabilmek amacıyla bir gun_sirasi sözlüğü tanımlanır. Bu sözlük, günlerin sayısal karşılıklarını tutar. Bu sıralama sayesinde, ders programları sırasıyla gösterilebilir.

- **Pazartesi = 0**
- **Salı = 1**
- **Çarşamba = 2**
- **Perşembe = 3**
- **Cuma = 4**

Bu sıralama, ders programlarının gün sırasına göre düzenlenmesini sağlar. Bu sıralama yapılırken, haftanın hangi günü olduğunu belirlemek için sayısal değerler kullanılır.

2. Ders Programlarının Alınması

Fonksiyon, veritabanındaki tüm ders programlarını almak için DersProgrami.objects.all() fonksiyonunu kullanır. Bu, veritabanındaki tüm derslerin bilgilerini içeren bir queryset döndürür.

3. Ders Programlarının Sıralanması

Ders programları sıralanırken, gün sırasına ve başlangıç saatine göre sıralama yapılır. Gün sırası, gun_sirasi sözlüğünden alınarak sayısal bir değere dönüştürülür.

- **gun_sirasi.get(x.gun, 5)**: Bu ifade, her bir dersin hangi günde olduğunu belirler ve o günü sayısal değere dönüştürür. Eğer veritabanında tanımlı olmayan bir gün varsa, 5 değeri kullanılarak bu dersler listenin sonuna yerleştirilir.
- **x.baslangic_saati**: Bu ifade, her dersin başlangıç saatine göre sıralama yapar.

Bu sıralama için sorted() fonksiyonu kullanılır. Böylece ders programları önce gün sırasına, sonra ise başlangıç saatine göre sıralanmış olur.

9. Sınav Programı Hazırlama

Bu modül, sınav takvimini ve saatlerini planlar, gözetmen atamasını gerçekleştirir, akademisyenlerin taslak programa erişimini sağlar ve sisteme not girişi yetkisi verir.

Sınav Programı Ekle

Ders:

Tarih:

Saat:

Derslik:

Gözetmen:

Kaydet

← Sınav Listesine Geri Dön

Sınav programı ekleme ekranı

Sınav Programı Listesi					
Ders	Derslik	Tarih	Saat	Gözetmen	İşlem
denem 2	Z054	Oct. 2, 2012	10:20 a.m.	ibrahim	<button>Ölurma Düzeni</button>
denem 2	Z054	Oct. 2, 2012	10:20 a.m.	ibrahim	<button>Ölurma Düzeni</button>
deneme Düzen	Z054	Oct. 2, 2012	3:20 p.m.	ibrahim	<button>Ölurma Düzeni</button>
Veri Yapıları	108	Dec. 4, 2025	noon	ibrahim	<button>Ölurma Düzeni</button>
deneme Düzen	1044 Lab	Dec. 4, 2025	10 a.m.	ibrahim	<button>Ölurma Düzeni</button>

+ Yeni Sınav Ekle

Sınav programı listesi

1. Sınav Tarihlerinin Belirlenmesi

- SınavProgramıForm üzerinden sınav için tarih seçimi yapılır.
- GET isteğinde boş form yüklenir; POST'ta form.is_valid() kontrolünden geçtikten sonra form.save() ile tarih veritabanına kaydedilir.

2. Derslere Göre Sınav Saatlerinin Ayarlanması

- Formdan seçilen her ders için uygun saat aralığı girilir.
- bitis_saati > baslangic_saati kısıtı, form validasyonu sayesinde sağlanır.

3. Sınav Gözetmenlerinin Atanması

- Formun “Gözetmen” alanı yalnızca role='akademisyen' kullanıcıları listeler.
- Seçilen gözetmen, SınavProgramı.gozetmen alanına bağlanır ve kaydedilir.

4. Akademisyenlerin Taslak Programa Erişim ve Not Yazma Yetkisi

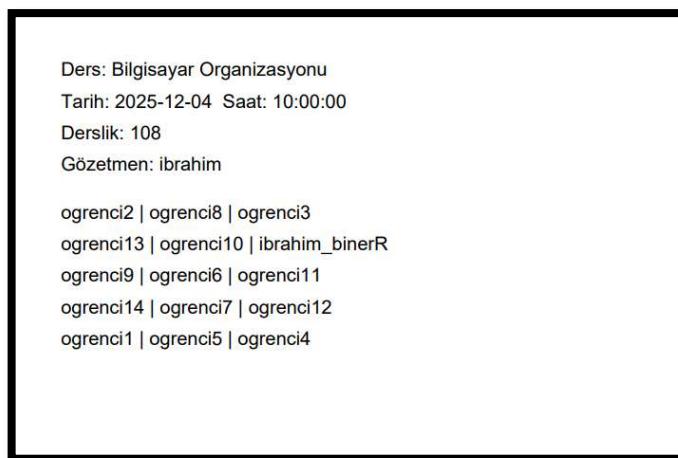
- sınav_listesi görünümünde akademisyenler sınav programı listesine erişebilir; düzenleme hakları kapalı tutulur.
- Sistemde her satırın yanında “Not Gir” işlevi için read-only giriş alanı veya AJAX endpoint tasarılanır; girilen not, sınavla ilişkilendirilir.

10. Dersliklere Göre Sınav Oturma Düzeni (Rastgele) Oluşturma

Bu modül, derslik kapasitesine göre rastgele oturma planı oluşturur, öğrenci ve gözetmen atamasını gösterir, tablo halinde listeler ve PDF raporu sunar.



Sınav oturma düzeni ekranı



Sınav oturma düzeni pdf olarak indirilmiş hali

1. Rastgele Oturma Düzeni Hesaplaması

- get_object_or_404(SinavProgrami, id=sinav_id) ile sınav nesnesi alınır.
- öğrenciler = CustomUser.objects.filter(role='ogrenci')[:ders.ogrenci_sayisi] sorgusuyla öğrenci listesi oluşturulur.
- random.shuffle(ogrenciler) ile liste karıştırılır.

- max_rows = derslik.kapasite // 3, required_rows = ceil(len(ogrenciler)/3),
row_count = min(required_rows, max_rows) formülleriyle sıra sayısı bulunur.
- İki boyutlu liste (oturma_duzeni) içinde her satırda üç koltuk atanır; boşlar None olur.

2. Gözetmen Bilgisinin Entegrasyonu

- sınav.gozetmen alanı, tablo başlığında {{ sınav.gozetmen.username }} olarak gösterilir.

3. Tablo Şablonunda Listeleme

- render(request, 'oturma_plani.html', context) ile oturma düzeni, ders adı, tarih, saat, derslik ve gözetmen bilgileri şablona aktarılır.
- <table> içinde her sıra ve koltuk, doluya username, boşsa “Boş” etiketile yazdırılır.

4. PDF Raporu Oluşturma

- HttpResponseRedirect(content_type='application/pdf') ve canvas.Canvas(...) ile yeni bir PDF nesnesi başlatılır.
- drawString çağrılarıyla başlık bilgileri eklenir.
- Oturma düzeni satır satır " | ".join(row) biçiminde PDF'e yazdırılır; boş hücreler "Bos" etiketi taşır.
- response['Content-Disposition'] = 'attachment';
filename="oturma_plani_<sınav_id>.pdf" başlığıyla indirme sağlanır.

5. Kullanıcıya Sunum

- PDF indirme düğmesi href="{% url 'oturma_plani_pdf' sınav.id %}" ile şablonda yer alır.
- Hem web tabanlı tablo hem de PDF çıktısı, sınavın fizikal sınıf düzenine uygun referans alınarak (Amfi-1, 108 vb.) oluşturulur.

11. Öğretim Elemanı Ders Programı (Kapı İsimliği) Oluşturma

Bu modül, her akademisyen için bireysel ders programı üretir, ders ve derslik eşleştirmesini listeler ve kapı isimliği formatında çıktı sağlar.

ibrahim - Haftalık Ders Programı			
Gün	Saat	Ders	Derslik
Pazartesi	9 a.m. - 11 a.m.	Veri Yapıları	108
Pazartesi	10 a.m. - midnight	Yazılım Mühendisliğine Giriş	1044 Lab
Çarşamba	10 a.m. - 11 a.m.	Veri Yapıları	Z054
Perşembe	9 a.m. - 10 a.m.	Mikro İşlemci Tasarımı	Z054
Perşembe	10 a.m. - noon	Bilgisayar Organizasyonu	Z054

[PDF Olarak İndir](#)

1. Ders Programı Görüntüleme

1. Giriş Zorunluluğu (login_required)

Yalnızca oturum açmış akademisyenler erişebilir.

2. Program Verilerinin Filtrelenmesi

– program = DersProgrami.objects.filter(ogretim_elemani=request.user) ile oturum açan kullanıcıya ait kayıtlar alınır.

3. Şablona Gönderme

– render(request, 'ogretim_elemani_programi.html', {'program': program}) ile ogretim_elemani_programi.html şablonuna liste aktarılır.

– Şablonda her kayıt için gün, saat, ders kod-adı ve derslik adı tablo veya kart formatında sunulur.

2. Kapı İsimliği (Door Plate) Çıktısı

ibrahim - Haftalık Ders Programı			
Gün	Saat	Ders	Derslik
Pazartesi	9 a.m. - 11 a.m.	Veri Yapılları	108
Pazartesi	10 a.m. - midnight	Yazılım Mühendisline Giri	1044 Lab
Çarşamba	10 a.m. - 11 a.m.	Veri Yapılları	Z054
Perşembe	9 a.m. - 10 a.m.	Mikro Sistemci Tasarımı	Z054
Perşembe	10 a.m. - noon	Bilgisayar Organizasyonu	Z054

1. PDF Başlatma

– response = HttpResponse(content_type='application/pdf') ve canvas.Canvas(response, pagesize=letter) ile yeni PDF nesnesi oluşturulur.
– response['Content-Disposition'] = 'attachment';
filename="kapı_isimliği_<username>.pdf"' ile indirme başlığı atanır.

2. Formatlama ve Veri Yazdırma

– Akademisyenin adı ve unvanı başlık olarak drawString ile eklendikten sonra her

ders için:

- p.drawString çağrılarıyla “Gün – Saat – Ders Adı – Derslik” formatı yazdırılır.
- Kart biçimli yerleşim için sayfaya birden fazla sütun veya satır düzeni kullanılabilir.

3. PDF'in Tamamlanması

- p.showPage() ve p.save() ile belge sonlandırılır.
- return response ile PDF kullanıcıya sunulur.

12. Teknik Gereksinimler

Bu başlık, sistemin altyapı ve işleyişini destekleyecek zorunlu bileşenleri tanımlar.

1. Veritabanı

- Uygulama, ilişkisel bir veritabanı kullanır (SQLite).
- Django ORM ile tablo ve ilişki tanımları (CustomUser, Ders, Derslik, SinavProgrami, DersProgrami, OturmaPlani) doğrudan modele dayalı olarak yönetilir.

2. Kullanıcı Dostu Arayüz

- Django şablon motoru ve Bootstrap veya benzeri CSS kütüphaneleriyle responsive tasarım sağlanır.
- Formlar, hatalı girişlerde anında uyarı veren validasyon mesajları içerir; net butonlar ve gezinme akışı kullanıcı deneyimini ön planda tutar.

3. Yetkilendirme ve Kimlik Doğrulama

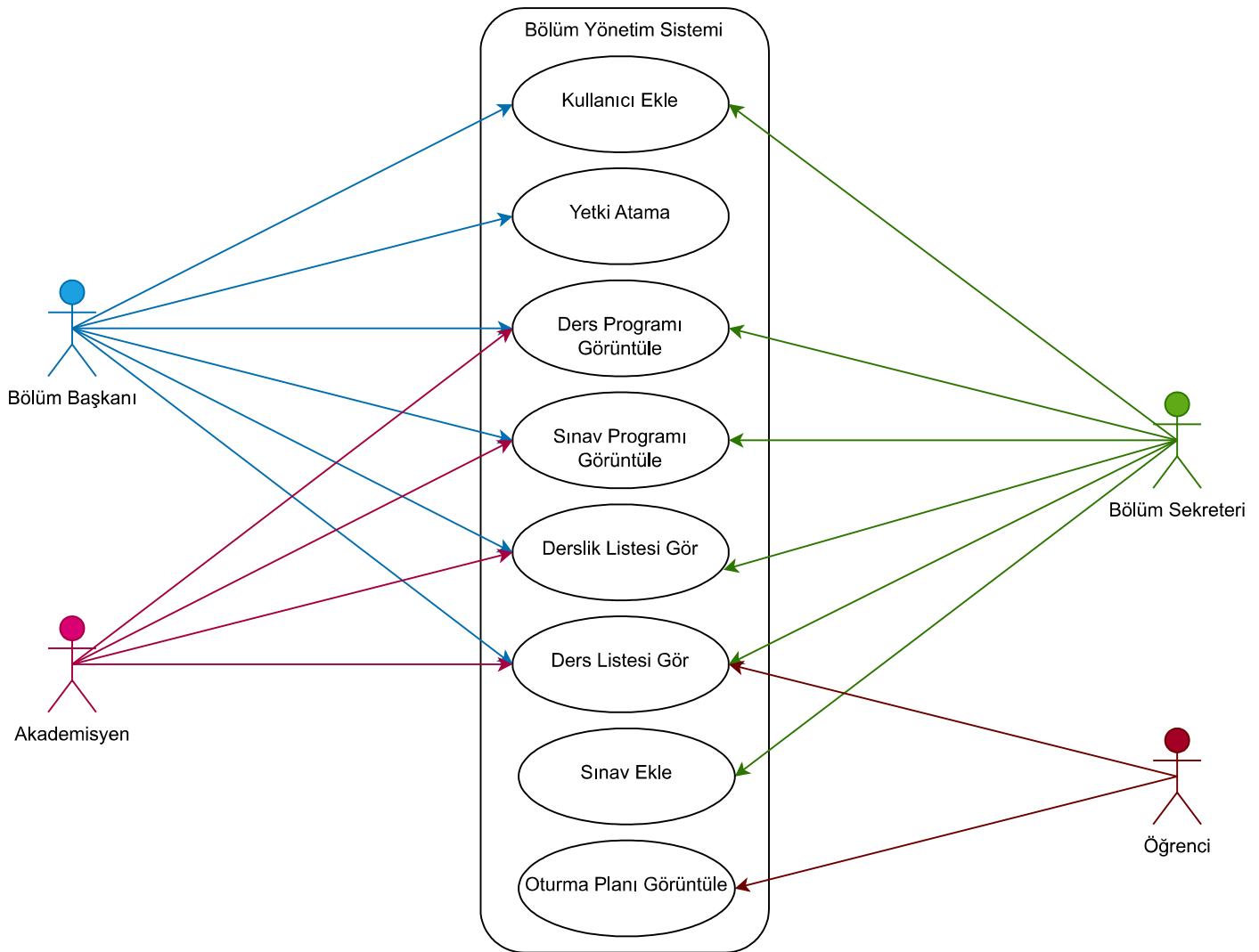
- Django'nun yerleşik AuthenticationForm, login_required decorator'ı ve Group/rol tabanlı erişim kontrolü kullanılır.
- CustomUser modelindeki role alanı, “bolum_baskani”, “akademisyen”, “ogrenci” gibi rolleri tanımlar; görünüm fonksiyonlarında HttpResponseRedirectForbidden ile yetkisiz erişim engellenir.

4. Rastgele Oturma Düzeni Algoritmaları

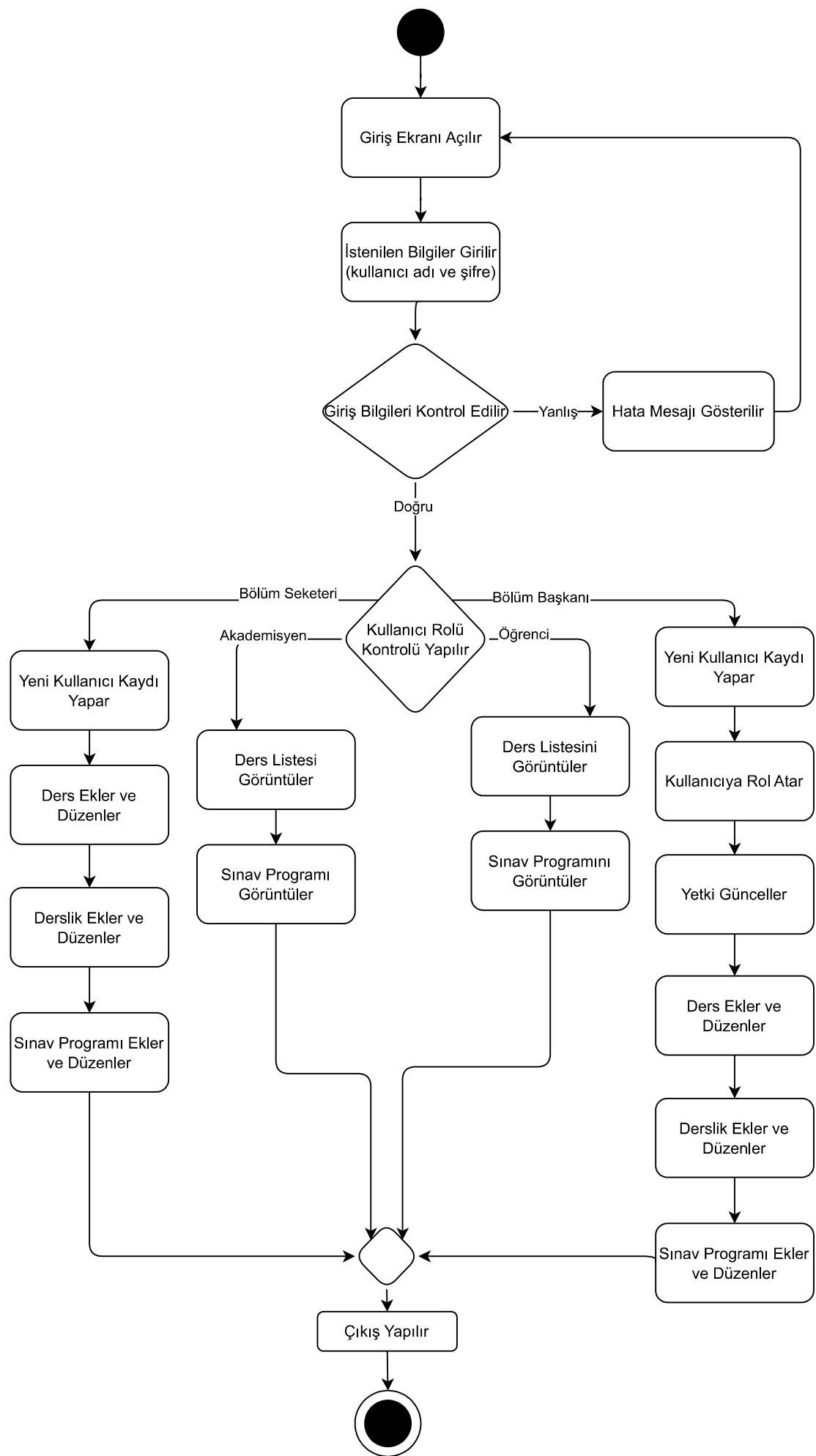
- Python'un random.shuffle fonksiyonu ile öğrenci listesi karıştırılır.
- Derslik kapasitesi ve ders kayıtlarındaki öğrenci sayısı göz önünde bulundurularak max_rows = kapasite//3, required_rows = ceil(n/3) formülleriyle sıra sayısı belirlenir.
- İki boyutlu liste yapısı kullanılarak her sıra üç koltuklu matris içinde atamalar rastgele dağıtilır; bu algoritma hem performans hem de çeşitlilik sunar.

13. Diyagramlar

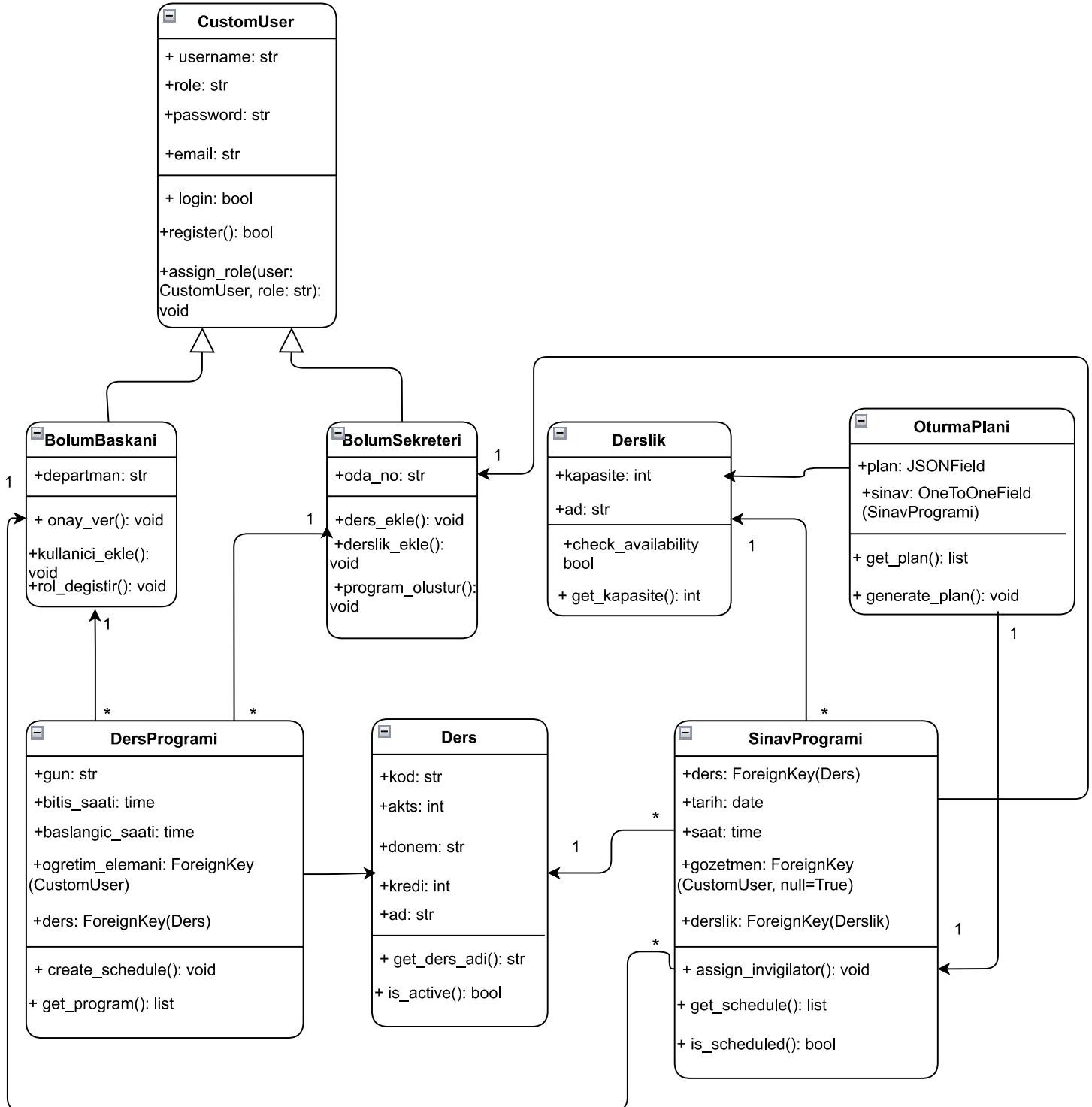
Kullanım Senaryosu Diyagramı



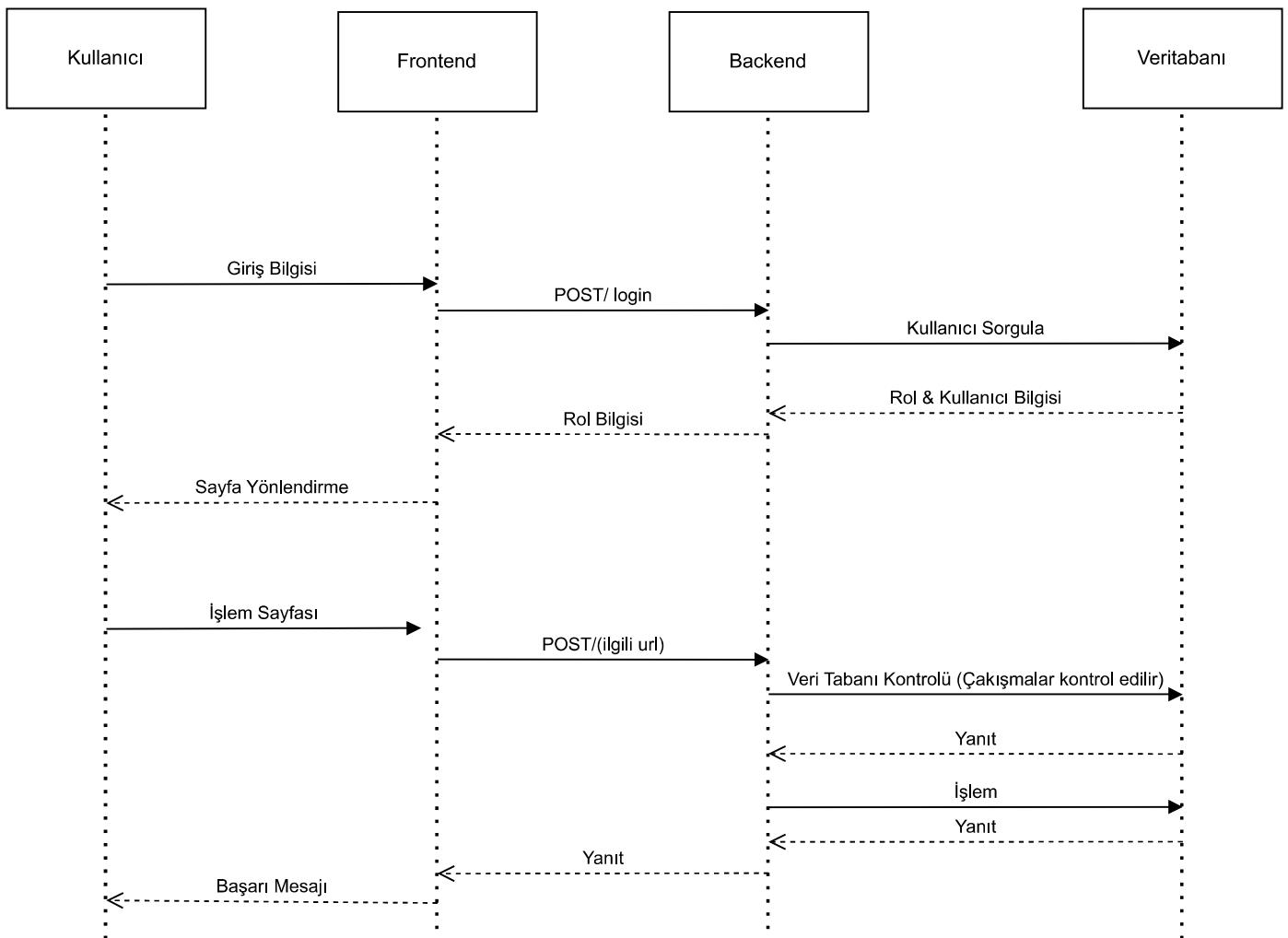
Aktivite Diagramı



Class Diagram



Veri Akışı Diyagramı



14. Kaynakça

1. <https://app.diagrams.net/>
2. <https://developer.mozilla.org/en-US/docs/Web/HTML>
3. <https://uxdesign.cc/>
4. [https://medium.com/@kubraasahin98/er-diagram%C4%B1-entity-relationship:](https://medium.com/@kubraasahin98/er-diagram%C4%B1-entity-relationship)
5. <https://docs.djangoproject.com/en/5.2/topics/forms/>
6. <https://docs.djangoproject.com/en/5.2/topics/db/models/>
7. https://en.wikipedia.org/wiki/Unified_Modeling_Language
8. <https://getbootstrap.com/docs/5.0/getting-started/introduction/>