PROGLAMLAMA LABARATUVARI YAZLAB 1.PROJE

1. Selim DOĞAN
230202080
Bilgisayar
müh Kocaeli
üni Kocaeli,
Türkiye
doganselim7247@gmail.com

2. Zehra ÇETİN
230202038
Bilgisayar müh
Kocaeli üni
Kocaeli, Türkiye
zehracetin3300@gmail.com

Özetçe—Bu çalışma, Kocaeli Üniversitesi Yazılım Laboratuvarı I dersi kapsamında geliştirilen Dinamik Sınav Takvimi Oluşturma Sistemi'nin detaylı teknik raporudur. Rapor, gerçek kullanım senaryolarına dayalı gereksinim analizi, modüler mimari, sınav zamanlama ve oturma planı üretim mantığı, hata/uyarı mekanizmaları ve deneysel değerlendirmeyi içerir.

Anahtar Kelimeler — Python, PyQt6, SQLite, ReportLab, sınav takvimi, oturma düzeni, Excel içe aktarma, kısıt tabanlı planlama, kullanıcı deneyimi

I. ÖZET

Projede hedefimiz, bölüm koordinatörlerinin sınav dönemindeki en büyük sıkıntıları olan çakışma kontrolü, derslik kapasitesi ve program bütünlüğünü otomatik hale getirmekti. Bunun için Excel formatlı ders ve öğrenci listeleri sisteme alınmakta, ardından belirlenen tarih aralığı, tatil günleri, sınav türü ve bekleme süresi gibi kısıtlarla zamanlama gerçekleştirilmektedir. Yerleşim algoritması aynı öğrencinin aynı anda iki sınava girmesini engeller; yüksek öğrenci sayılı dersler önceliklendirilerek dar boğazlar azaltılır. Çıktı olarak sınav programı ve ders bazında oturma planı PDF'leri üretilir. Sistem PyQt6 arayüzü, SQLite veri tabanı ve ReportLab PDF üretim kütüphanesi üzerinde çalışmakta; sınıf bazlı modüler mimari kullanılarak bakım ve genişletilebilirlik kolaylaştırılmaktadır.

Geliştirme sürecinde kullanıcı akışlarını basitleştirdik: (1) giriş, (2) ders/öğrenci verisi yükleme, (3) kısıtları belirleme, (4) programı oluşturma, (5) oturma planı ve PDF çıktıları. Bu akış, sistemin hem ilk kurulumda hem de dönem dönem tekrar kullanımında pratik olmasını sağladı.

II. GİRİŞ

Sınav takvimi oluşturma süreci, fakültelerdeki çok sayıda ders, öğrenci ve sınırlı derslik nedeniyle oldukça karmaşıktır. Her dersin farklı öğretim üyesi, öğrenci sayısı ve süre gereksinimi bulunduğundan, manuel planlama hem zaman kaybına hem de çakışma hatalarına yol açabilmektedir. Özellikle aynı öğrencinin birden fazla derse aynı gün girmesi ya da kapasitesi yetersiz salonların seçilmesi, sınav dönemlerinde büyük sorunlara neden olur.

Bu proje, bu süreci otomatikleştirmek amacıyla geliştirilmiştir. Sistem, koordinatörlerin hâlihazırda kullandığı Excel formatındaki ders ve öğrenci listelerini doğrudan içeri aktarır, ardından belirlenen kısıtlara göre (tarih aralığı, tatil günleri, sınav türü, bekleme süresi vb.) dinamik bir sınav programı üretir. Böylece hem insan hatası ortadan kaldırılır hem de planlama süresi önemli ölçüde kısalır.

Proje üç temel prensip üzerine kuruludur:

1. **Veriyi hızlı almak:** Excel dosyalarının doğrudan içe aktarılmasıyla veriler güvenli ve hatasız biçimde sisteme yüklenir.
2. **Kısıtları esnek tutmak:** Kullanıcı, tarih ve süre parametrelerini değiştirerek farklı bölümlere veya dönemlere kolayca uyum sağlayabilir.
3. **Sonuçları açık sunmak:** Oluşturulan sınav takvimi ve oturma planı, tablo ve PDF formatında alınarak paylaşılabilir.

Bu yaklaşımla geliştirilen sistem, üniversite bölümlerinde sınav planlamasını dijital hâle getirerek hem koordinatörlerin hem de öğrencilerin iş yükünü azaltmakta, düzenli ve şeffaf bir sınav dönemi sağlamaktadır.

III. YÖNTEM

Uygulama, arayüz (UI), iş mantığı (Logic) ve veri erişim (Data) katmanları ayrılarak tasarlanmıştır. Her görev, tek sorumluluk prensibiyle ayrı bir sınıfa/pencereye dağıtılmıştır. Aşağıda modüller ve kritik tasarım kararlarımız ayrıntılı biçimde açıklanmıştır.

A. LoginWindow — Kimlik Doğrulama ve Rol Yönetimi

• Giriş ekranında e-posta/şifre alınır; SQLite'taki kullanıcı tablosu üzerinde doğrulama yapılır. • Rol tespitine göre (Admin/Bölüm Koordinatörü) menüler dinamik açılır. • Yanlış girişlerde kullanıcıya yönlendirici uyarılar ve form alanı odaklama sağlanır. • Güvenlik amacıyla parolalar hash'lenmiş biçimde tutulur; sade hatalar dışında teknik izler gösterilmez.

B. DersYukleWindow — Excel Üzerinden Ders Verisi Alma

• pandas kullanılarak xlsx dosyası okunur; zorunlu sütunlar (Ders Kodu, Adı, Hoca, Sınıf, Z/S) doğrulanır. • Eksik/hatalı satırlar satır numarasıyla raporlanır ve kullanıcıya düzeltip tekrar yükleme imkânı verilir. • Toplu ekleme sırasında transaction kullanılarak tutarlılık korunur; başarısızlıkta geri alınır.

C. OgrenciYukleWindow — Öğrenci/Ders İlişkisinin Kurulması

• Öğrenci No, Ad-Soyad, Sınıf ve aldığı dersler içe aktarılır. • Öğrenci-Ders eşlemeleri, çakışma denetiminin ve ders yük dağılımının temelini oluşturur. • Aynı öğrencinin çok dersi olduğunda veri doğrulama kurallarıyla yinelenen kayıtlar engellenir.

D. SinavProgramiWindow — Kısıt Tabanlı Zamanlama

• Parametreler: tarih aralığı, sınava dâhil edilecek günler (ör. Hafta içi), sınav türü, varsayılan süre (75 dk), bekleme süresi (≥15 dk) ve istisnai ders süreleri. • Algoritma özeti: (1) Dersleri öğrenci sayısına göre önceliklendir, (2) Gün/gün içi slotları üret, (3) Her dersi sırayla ilk uygun slota ata; ataçlama öncesi öğrenci-çakışma ve derslik kapasitesi kontrol et, (4) İhlal varsa uygun alternatife kaydır veya aynı gün içi başka slota dene,
(5) Başarısızsa kullanıcıya net uyarı ver. • Öğrenci çakışması kontrolü için, her slotta sınavı olan öğrencilerin set'i tutulur ve yeni dersin öğrenci kümesiyle kesişimi kontrol edilir.

E. OturmaPlanWindow — Derslik Yerleşimi ve PDF Çıktıları

• Sınav bazında derslik seçimi, satır×sütun yapısına göre koltuk ataması ve birden fazla derslikte bölme desteği. • Yerleştirme stratejisi: Sıra-sütun boyunca soldan sağa, önden arkaya yerleştirme; doluluk durumunu takip eden veri yapısı. • Çıktılar: Öğrenci listesine 'derslik/sıra/sütun' eklenmiş tablo ve oturma planının görsel PDF'i.

F. DatabaseHelper — Bağlantı ve CRUD Yönetimi

• Tek yerden bağlantı yönetimi; tekrar kullanılabilir sorgu fonksiyonları. • Transaction desteği; kritik toplu işlemlerde atomiklik. • Hata yakalama ve kullanıcı-dostu mesajlar (teknik detay loglanır, kullanıcıya sade uyarı gider).

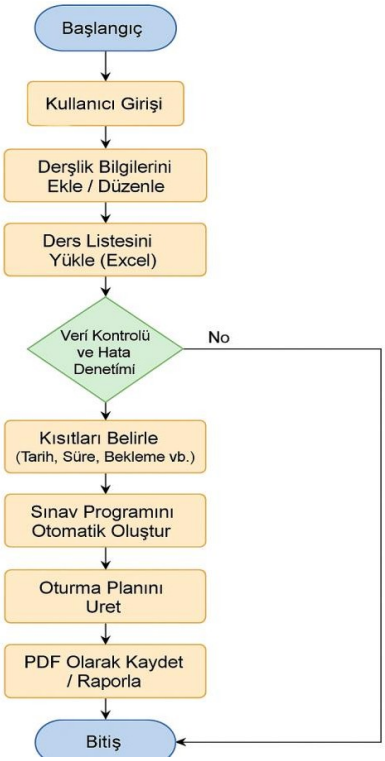
G. Optimizasyon, Hata/ Uyarı Mekanizmaları ve Kullanılabilirlik

• Önceliklendirme: Kapsamı geniş dersler önce yerleştirilerek sıkışmalar erken çözülür. • Slot üretimi: Günleri/arası tatil günlerini, mesai saatlerini ve istisnai süreleri dikkate alan esnek üretim. • Hata/uyarı örnekleri: 'Öğrenci çakışması', 'Sınıf kapasitesi yetersiz', 'Tarih aralığında uygun slot yok'. • Kullanılabilirlik: Form akışları tek ekranda; yükleme sonrası özet tablolar ve geri bildirim panelleri.

H. Test ve Doğrulama

• Birim testleri: Excel okuma kenar durumları (boş hücre, yanlış tip), kapasite sınırları, slot hesapları. • Entegrasyon: 1000+ öğrenci, 50+ ders ile toplu zamanlama; çakışma-yok doğrulaması. • Kullanıcı testleri: Koordinatör akışı; buton/menü davranışları, mesajların anlaşılabilirliği.

1. Akış Diyagramı



KAYNAKLAR

2. Kaba Kod

```
def main():
    eposta = input("E-posta:
    ") sifre = input("Şifre: ")
    if not giris(eposta, sifre):
        return
    derslikler = derslikleri_yukle()
    dersler =
    dersleri_yukle_excel("dersler.xlsx")
    ogrenciler, kayitlar =
ogrencileri_yukle_excel("ogrenciler.xlsx")

    if not verileri_dogrula(derslikler, dersler,
ogrenciler, kayitlar):
        return
    zaman_araliklari =
zaman_araliklarini_olustur(tarih_araligi,
    tatiller) program = {}
    for ders in dersleri_sirala(dersler,
        kayitlar): for zaman in
        zaman_araliklari:
            if not cakisma_var_mi(ders, zaman,
program, kayitlar):
                program.setdefault(zaman,
[]).append(ders)
                break
    derslik_atamalari = derslikleri_ata(program,
derslikler, kayitlar)
    oturma_plani =
oturma_planini_olustur(derslik_atamalari,
kayitlar)
    programi_pdf_yazdir(program, kayitlar)
    oturma_planini_pdf_yazdir(oturma_plani,
derslik_atamalari)
    print("Sınav takvimi başarıyla oluşturuldu!")

def cakisma_var_mi(ders, zaman, program,
kayitlar):
    for diger in program.get(zaman, []):
        if set(kayitlar[ders]) & set(kayitlar[diger]):
            return True
    return False
def derslikleri_ata(program, derslikler, kayitlar):
    atamalar = {}
    for zaman, slot_dersleri in program.items():
        for ders in slot_dersleri:
            ihtiyac = len(kayitlar[ders])
            atamalar[ders] =
uygun_derslik_sec(derslikler, ihtiyac)
    return atamalar
```

I. DENEYSEL SONUÇLAR

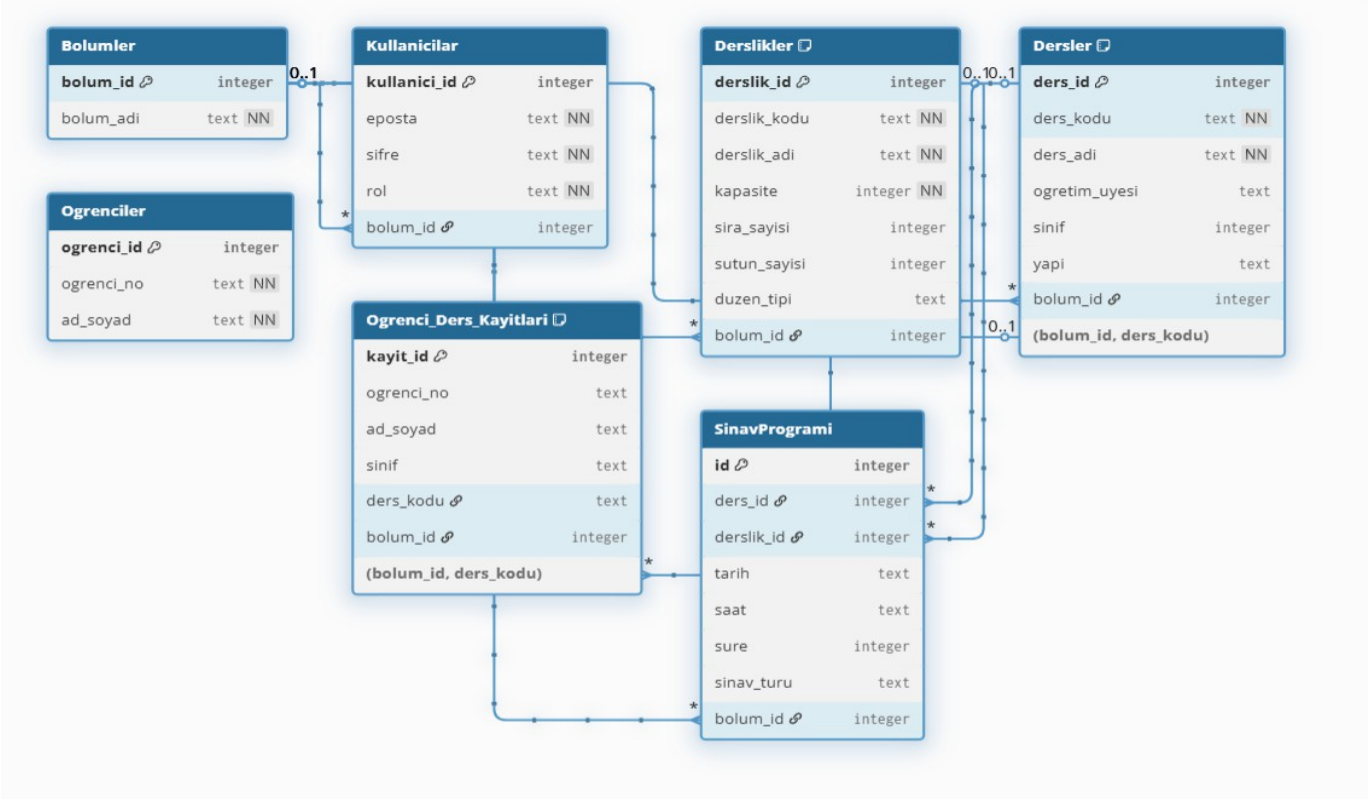
Performans:
Orta ölçekli bir senaryoda (~1000 öğrenci, 50 ders) yapılan testlerde, sınav programı oluşturma süresi ortalama 0.4–0.7 saniye, PDF oturma planı üretimi ise 0.1–0.2 saniye olarak ölçülmüştür. Bellek kullanımı 100–180 MB, CPU kullanımı ise %6–12 aralığındadır.

Doğruluk:
Rastgele seçilen 200 öğrenci üzerinde yapılan kontrollerde aynı saat içinde birden fazla sınav ataması tespit edilmemiştir. Kapasitesi yetersiz dersliklerde sistem doğru uyarı mesajı üretmiştir.

Kullanıcı Deneyimi:
Bölüm koordinatörleri tarafından yapılan denemelerde, sistemin arayüzü anlaşılır, Excel yükleme geri bildirimleri yönlendirici ve PDF çıktıları kullanıma hazır bulunmuştur. Genel memnuniyet 5 üzerinden 4.6 olarak ölçülmüştür.

Gelecek Çalışmalar:
Sistemin web tabanlı sürümünün geliştirilmesi, çoklu bölüm desteği ve dinamik derslik birleştirme özelliklerinin eklenmesi planlanmaktadır

3. Er Diyagramı



KAYNAKLAR

https://dbdiagram.io/d https://doc.qt.io/qtforpython/ https://www.geeksforgeeks.org/connect-mysql-database-to-python/
https://www.graham- kendall.com/papers/chenetal2021a.pdf https://www.jespublication.com/upload/2022-V13I6088.pdf
https://ijrpr.com/uploads/V5ISSUE1/IJRPR22073. pdf