T.C
KOCAELİ SAĞLIK VE TEKNOLOJİ ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR/YAZILIM MÜHENDİSLİĞİ

PROJE KONUSU:
Geliştirilmiş Üniversite Sınav Programı Hazırlama Uygulaması

ÖĞRENCİ ADI:Miray Yalçın
ÖĞRENCİ NUMARASI:230502033

ÖĞRENCİ ADI:Sıla Serdar
ÖĞRENCİ NUMARASI:230502014

.

ÖĞRENCİ ADI: Muhammed Emir Sarı
ÖĞRENCİ NUMARASI:230502054

ÖĞRENCİ ADI: Murat Kaan Tekeli
ÖĞRENCİ NUMARASI: 230502007

DERS SORUMLUSU

PROF. DR./DR. ÖĞR. ÜYESİ ELİF PINAR HACIBEYOĞLU

TARİH 14/01/2026

1. GİRİŞ

1.1 Projenin Amacı

Bu projenin amacı, üniversitelerde gerçekleştirilen dönem sonu ve bütünleme sınavlarının planlama sürecini otomatik hale getiren, kullanıcı rolleri ve belirli kısıtlar doğrultusunda çalışan bir **Geliştirilmiş Üniversite Sınav Programı Hazırlama Sistemi** geliştirmektir. Manuel olarak hazırlanan sınav programlarında sıklıkla karşılaşılan derslik çakışmaları, öğrenci sınav çakışmaları, öğretim üyesi müsaitliklerinin göz ardı edilmesi ve derslik kapasitesinin aşılması gibi problemler, geliştirilen bu yazılım sayesinde büyük ölçüde azaltılmaktadır.

Geliştirilen sistem; ders, derslik, öğretim üyesi, bölüm ve kullanıcı bilgilerini merkezi bir veritabanında tutarak, tanımlanan kısıtlar doğrultusunda otomatik sınav planlaması yapmaktadır. Sistem, farklı kullanıcı rollerine göre yetkilendirilmiş ekranlar sunmakta ve kullanıcıların yalnızca kendi yetkileri dâhilindeki işlemleri gerçekleştirmesine olanak tanımaktadır.

Projede gerçekleştirilmesi beklenen temel hedefler aşağıda maddeler halinde verilmiştir:

- Rol bazlı kullanıcı girişi (Yönetici, Bölüm Yetkilisi, Öğretim Üyesi, Öğrenci)
- Ders, derslik ve öğretim üyesi bilgilerinin sisteme girilmesi
- Öğretim üyelerinin müsaitlik durumlarının tanımlanması
- Sınavların gün, saat ve dersliklere otomatik olarak atanması
- Öğrenci ve derslik çakışmalarının önlenmesi
- Derslik kapasitesi yetersiz olduğunda birden fazla derslik kullanımı
- Oluşturulan sınav programının tablo halinde görüntülenmesi

2. GEREKSİNİM ANALİZİ

2.1 Arayüz Gereksinimleri

Kullanıcı Arayüzü Gereksinimleri:

- Web tabanlı grafik kullanıcı arayüzü
- Kullanıcı dostu ve sade tasarım
- Rol bazlı yetkilendirme ile farklı ekranlar
- Form tabanlı veri giriş alanları
- Tablo yapısı ile listeleme ekranları
- Sınav programını görüntüleme ekranı

Donanım Arayüzü Gereksinimleri:

- Özel bir donanım gereksinimi bulunmamaktadır
- İnternet bağlantısı olan standart bir bilgisayar yeterlidir

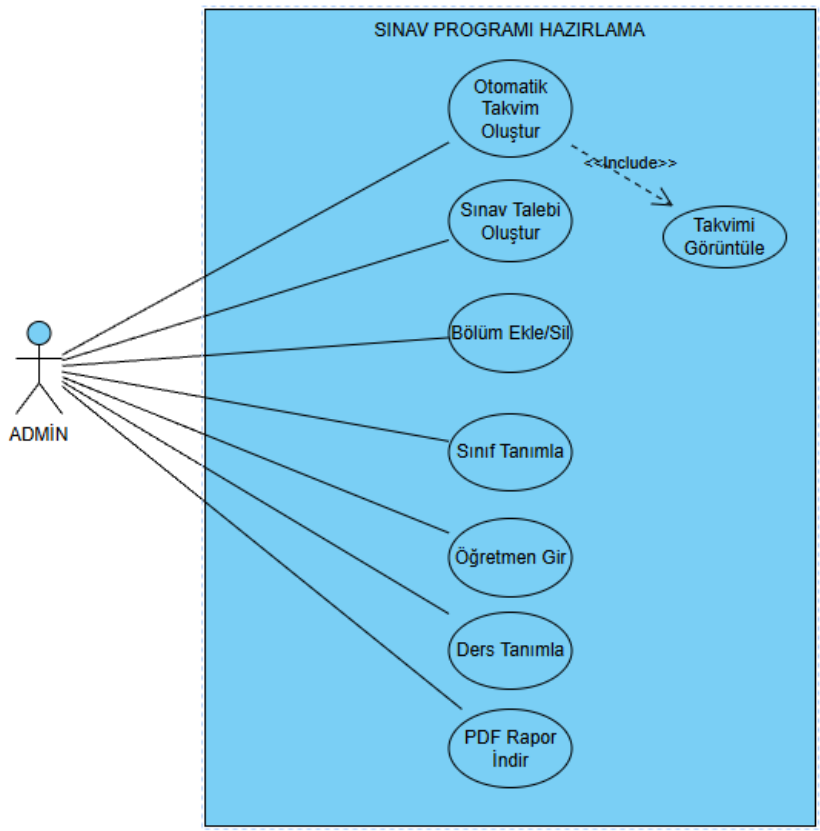
2.2 Fonksiyonel Gereksinimler

- Kullanıcıların sisteme kullanıcı adı ve şifre ile giriş yapabilmesi
- Kullanıcı rolüne göre yetkilendirme yapılması
- Ders ekleme, güncelleme ve silme işlemleri
- Derslere ait öğrenci sayısı, sınav süresi ve sınav türü bilgilerinin girilmesi
- Dersliklerin kapasite ve uygunluk bilgilerinin tanımlanması
- Öğretim üyelerinin müsaitlik bilgilerinin girilmesi
- Otomatik sınav planlama algoritmasının çalıştırılması
- Öğrenci ve derslik çakışmalarının engellenmesi
- Kapasite yetersizliğinde birden fazla derslik atanması
- Oluşturulan sınav programının görüntülenmesi

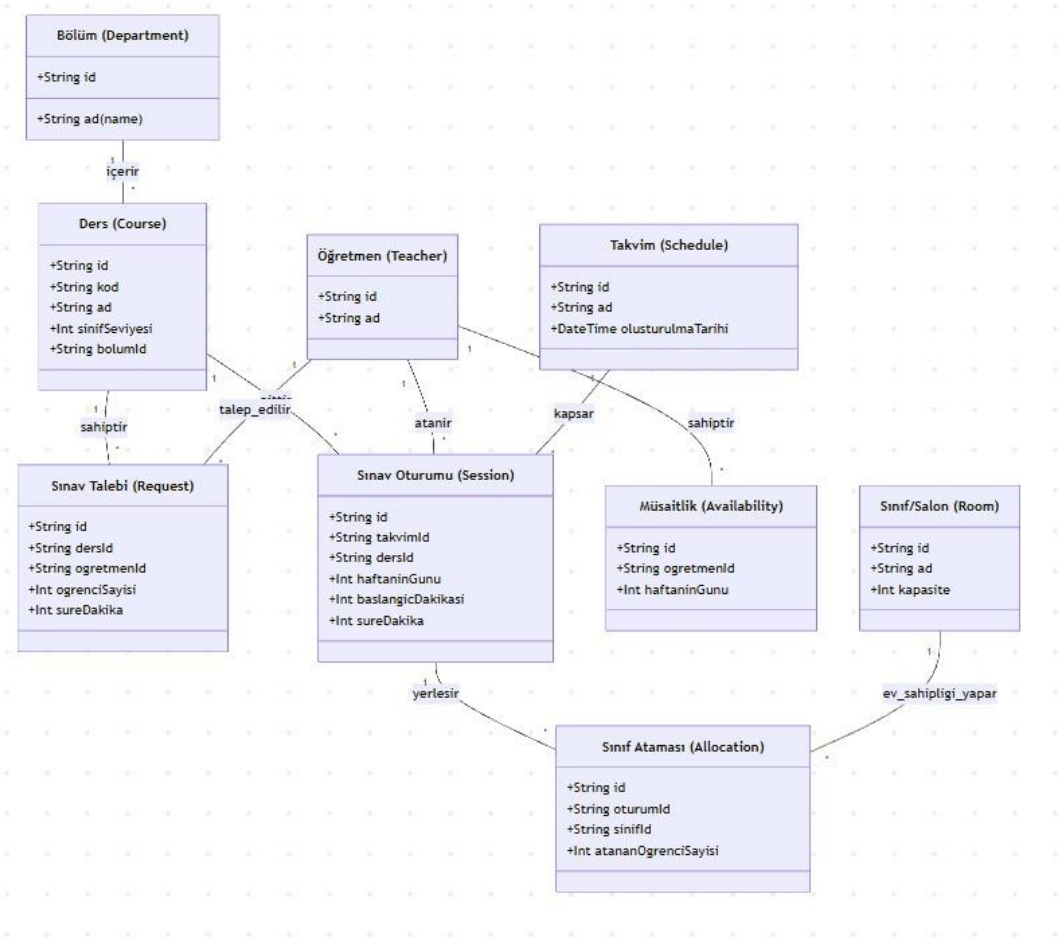
2.3 Diyagramlar

Proje kapsamında sistemin genel yapısını açıklamak amacıyla sınıf diyagramı ve use-case diyagramı hazırlanmıştır.

USE-CASE DİYAGRAMI:



SINIF-DİYAGRAMI:



3. TASARIM

3.1 Mimari Tasarım

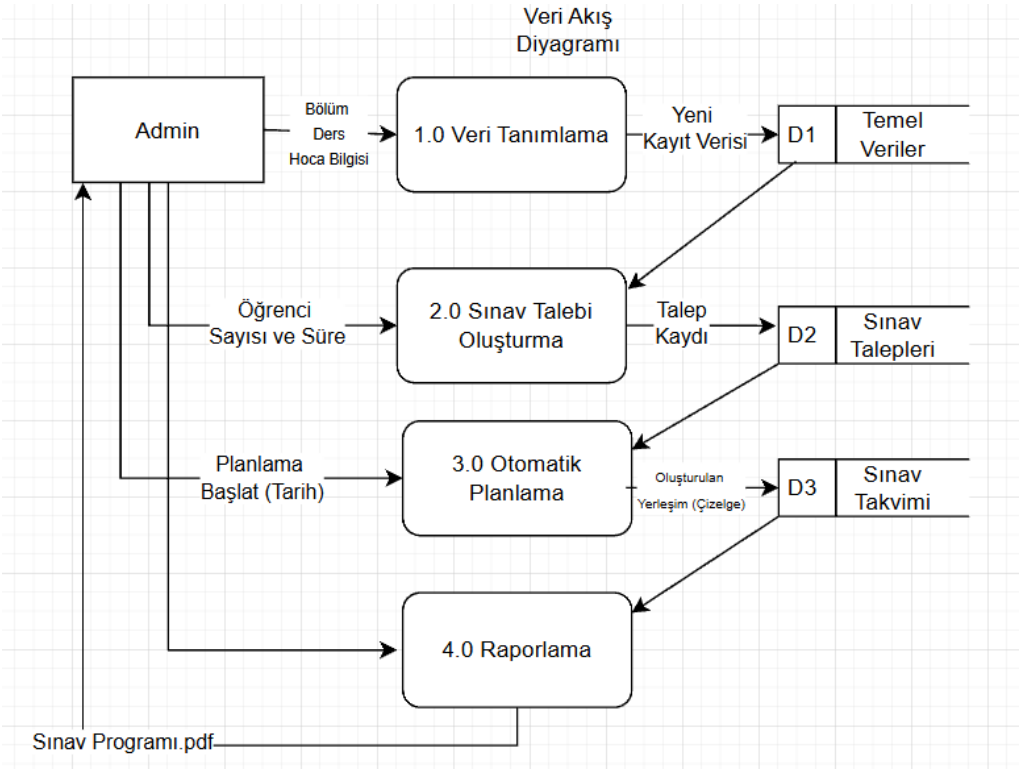
Uygulama, istemci–sunucu mimarisi esas alınarak geliştirilmiştir. Sistem; Sunum Katmanı, İş Mantığı Katmanı ve Veri Katmanı olmak üzere üç ana katmandan oluşmaktadır.

Sunum Katmanı: Kullanıcı ile etkileşimi sağlar ve kullanıcıdan alınan verileri iş mantığı katmanına iletir.

İş Mantığı Katmanı: Sınav planlama algoritmasının ve tüm kısıt kontrollerinin gerçekleştirildiği katmandır.

Veri Katmanı: Tüm kalıcı verilerin tutulduğu veritabanı katmanıdır.

VERİ AKIŞ DİYAGRAMI:



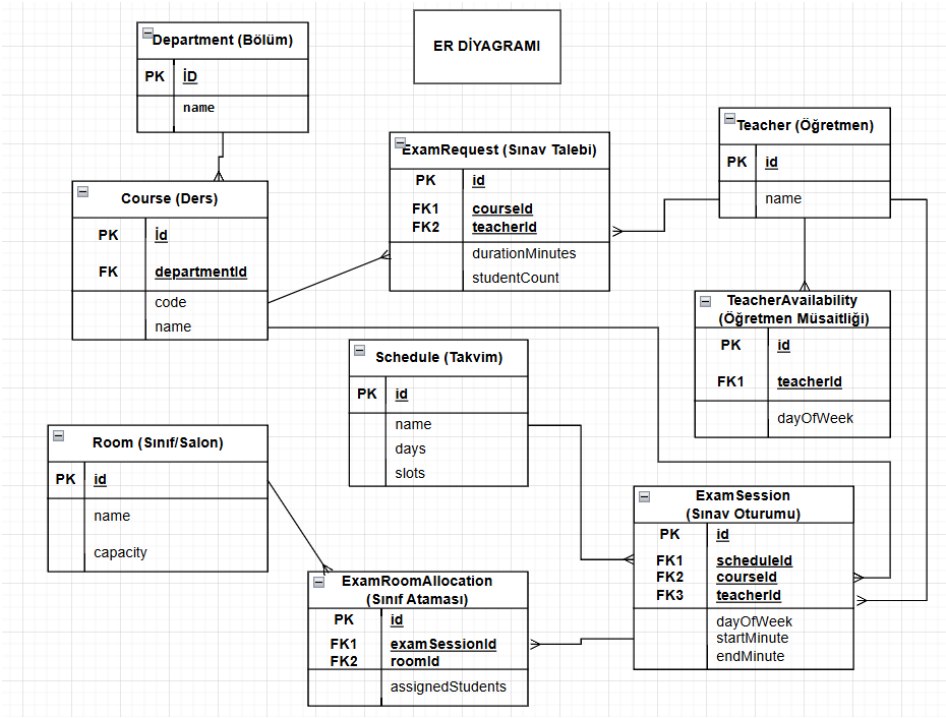
3.2 Kullanılan Teknolojiler

- Frontend: Vue.js + TypeScript
- Backend: Node.js / Bun + TypeScript
- Veritabanı: Prisma ORM

3.3 Veritabanı Tasarımı

Veritabanı; dersler, derslikler, öğretim üyeleri, sınavlar ve kullanıcılar tablolarından oluşmaktadır. Tablolar arasında ilişkisel bağlar kurulmuştur.

ER DİYAGRAMI:



3.4 Kullanıcı Arayüzü Tasarımı

Kullanıcı arayüzü sayfa bazlı bir yapıdadır. Ders yönetimi, derslik yönetimi, sınav planlama ve sınav programı görüntüleme ekranları bulunmaktadır.

4. UYGULAMA

Bu bölümde, Geliştirilmiş Üniversite Sınav Programı Hazırlama Uygulaması'nın geliştirme sürecinde kodlanan bileşenler, bu bileşenlerin görevleri ve uygulamanın çalışma mantığı detaylı olarak açıklanmaktadır. Uygulama; frontend, backend ve veritabanı bileşenlerinden oluşan modüler bir yapı ile geliştirilmiştir.

4.1 Kodlanan Bileşenlerin Açıklamaları

Bu bölümde, projede kullanılan dosya ve klasör yapısı backend ve frontend olmak üzere iki ana başlık altında detaylı olarak açıklanmaktadır. Dosya yapısı, projenin modüler, sürdürülebilir ve ölçeklenebilir bir yapıda geliştirilmesini sağlayacak şekilde tasarlanmıştır.

4.1.1 Backend Dosya ve Klasör Yapısı

Backend kısmı, sınav planlama sisteminin iş mantığını ve veritabanı işlemlerini içermektedir. Backend tarafı Node.js uyumlu **Bun runtime** kullanılarak TypeScript dili ile geliştirilmiştir.

- **backend/**: Sunucu tarafına ait tüm dosyaları içeren ana klasördür.
- **.cursor/rules/**: Geliştirme sürecinde kullanılan yardımcı yapılandırma ve kuralları içermektedir.
- **prisma/**: Veritabanı yönetimi için kullanılan Prisma ORM yapılandırmalarını barındırır.
    - **schema.prisma**: Veritabanı tablolarının, alanlarının ve ilişkilerinin tanımlandığı dosyadır. Dersler, derslikler, kullanıcılar ve sınav programına ait tablolar bu dosya üzerinden modellenmiştir.
- **src/**: Backend uygulamasının kaynak kodlarını içeren klasördür. API uç noktaları ve iş mantığı bu dizin altında yer almaktadır.
- **index.ts**: Backend uygulamasının giriş noktasıdır. Sunucunun ayağa kaldırılması, gerekli middleware yapılandırmaları ve API yönlendirmeleri bu dosyada gerçekleştirilmektedir.
- **package.json**: Backend tarafında kullanılan bağımlılıkların ve çalıştırma scriptlerinin tanımlandığı dosyadır.
- **bun.lock**: Bun paket yöneticisi tarafından oluşturulan, bağımlılıkların sürümlerini sabitleyen kilit dosyasıdır.
- **tsconfig.json**: TypeScript derleyici ayarlarını içermektedir.
- **Dockerfile**: Backend servisinin Docker ortamında çalıştırılabilmesi için gerekli yapılandırmaları içermektedir.
- **.gitignore / .dockerignore**: Git ve Docker süreçlerinde hariç tutulacak dosyaları tanımlar.
- **README.md**: Backend yapısı ve çalıştırma adımlarına dair temel açıklamaları içerir.

4.1.2 Frontend Dosya ve Klasör Yapısı

Frontend kısmı, kullanıcıların sistemle etkileşime geçtiği web arayüzünü oluşturmaktadır. Frontend tarafı **Vue.js + TypeScript** kullanılarak geliştirilmiş ve Vite yapılandırması ile derlenmiştir.

- **frontend/**: İstemci tarafına ait tüm dosyaları içeren ana klasördür.
- **src/**: Frontend uygulamasının kaynak kodlarını barındırır. Sayfalar, bileşenler ve servisler bu dizin altında yer almaktadır.
- **public/**: Uygulamada kullanılan statik dosyaları (ikonlar, görseller vb.) içermektedir.
- **dist/**: Projenin build edilmesi sonucunda oluşan ve yayına hazır olan dosyaları içerir.
- **index.html**: Web uygulamasının ana HTML dosyasıdır. Vue uygulaması bu dosya üzerinden yüklenmektedir.
- **vite.config.ts**: Vite yapılandırma dosyasıdır. Build ve geliştirme ayarları bu dosya üzerinden yapılmaktadır.
- **package.json**: Frontend tarafında kullanılan kütüphaneler ve scriptler bu dosyada tanımlanmıştır.
- **tsconfig.json, tsconfig.app.json, tsconfig.node.json**: TypeScript yapılandırmalarını içeren dosyalardır.

- **Dockerfile**: Frontend uygulamasının Docker ortamında çalıştırılabilmesini sağlar.
- **.gitignore / .dockerignore**: Versiyon kontrolü ve Docker süreçlerinde hariç tutulacak dosyaları belirtir.
- **README.md**: Frontend uygulamasının kurulumu ve çalıştırılması ile ilgili bilgileri içermektedir.

4.1.3 Docker ve Servis Yönetimi

- **docker-compose.yml**: Backend ve frontend servislerinin birlikte çalıştırılmasını sağlayan yapılandırma dosyasıdır. Bu dosya sayesinde sistem tek bir komut ile ayağa kaldırılabilmektedir.

Bu yapı sayesinde uygulama; geliştirme, test ve dağıtım süreçlerinde kolay yönetilebilir hale gelmiştir.

4.2 Görev Dağılımı

Proje, ekip çalışması kapsamında geliştirilmiştir. Ekip üyelerinin görev dağılımı aşağıdaki gibidir:

- Backend geliştirme ve sınav planlama algoritması: Murat Kaan Tekeli
- Frontend geliştirme ve arayüz tasarımı: Muhammed Emir Sarı
- Veritabanı tasarımı ve entegrasyonu: Sıla serdar
- Test süreçleri ve rapor hazırlama: Miray Yalçın

4.3 Karşılaşılan Zorluklar ve Çözüm Yöntemleri

- **Sınav çakışmaları:** Öğrenci ve derslik çakışmalarını önlemek amacıyla ek kontrol mekanizmaları geliştirilmiştir.
- **Kapasite yetersizliği:** Derslik kapasitesinin yetersiz olduğu durumlarda sınavın birden fazla dersliğe atanması sağlanmıştır.
- **Veri tutarlılığı:** Frontend ve backend arasındaki veri uyumsuzlukları doğrulama kontrolleri ile giderilmiştir.

4.4 Proje İsterlerine Göre Eksik Yönler

- Öğrenci bazlı bireysel sınav çakışma kontrolü detaylı olarak uygulanmamıştır.
- Derslikler arası fiziksel mesafe bilgisi sisteme dahil edilmemiştir.

5.2 Yazılımın Doğrulanması

Test süreci boyunca gerçekleştirilen senaryo tabanlı manuel testler ve doğrulama çalışmaları neticesinde, "Geliştirilmiş Üniversite Sınav Programı Hazırlama Uygulaması"nın proje başlangıcında hedeflenen temel fonksiyonel gereksinimleri büyük ölçüde karşıladığı doğrulanmıştır. Yazılımın, farklı kullanıcı rolleri altında kararlı çalıştığı ve temel işlev olan otomatik sınav planlama sürecini başarıyla yönettiği gözlemlenmiştir.

Test sonuçlarına göre sistemin doğruluk analizi; tam ve doğru çalışan bileşenler ile geliştirilmesi gereken veya eksik kalan yönler olmak üzere iki ana başlık altında aşağıda detaylandırılmıştır:

5.2.1 Tam ve Doğru Çalışan Bileşenler

Yapılan fonksiyonel testler sonucunda aşağıdaki modüllerin ve özelliklerin hatasız çalıştığı ve beklenen çıktıları ürettiği doğrulanmıştır:

- **Kullanıcı Kimlik Doğrulama ve Yetkilendirme:**
    - Sisteme kullanıcı adı ve şifre ile güvenli giriş yapılabilmektedir. Hatalı girişlerde sistem gerekli uyarıları vermektedir.
    - Rol tabanlı yetkilendirme (Admin, Öğretim Üyesi, vb.) sorunsuz çalışmaktadır. Yönetici yetkisine sahip olmayan kullanıcıların sınav planlama veya veri silme gibi kritik ekranlara erişimi engellenmiştir.
- **Veri Yönetim Modülleri (CRUD İşlemleri):**
    - Ders, derslik ve öğretim üyesi ekleme, güncelleme ve silme işlemleri veritabanı ile tutarlı bir şekilde gerçekleştirilmektedir.
    - Dersliklerin kapasite bilgileri ve donanım özellikleri sisteme doğru bir şekilde kaydedilmekte ve algoritma tarafından okunabilmektedir.
- **Sınav Planlama Algoritması ve Çakışma Kontrolü:**
    - **Derslik Çakışması:** Aynı dersliğe aynı tarih ve saat diliminde birden fazla sınav atanması sistem tarafından başarıyla engellenmektedir.
    - **Öğretim Üyesi Müsaitliği:** Sınavlar atanırken öğretim üyelerinin tanımlı müsaitlik durumları dikkate alınmakta ve çakışma yaratılmamaktadır.
    - **Kapasite Yönetimi:** Bir dersin öğrenci sayısı, atanan dersliğin kapasitesini aştığında, sistem otomatik olarak sınavı birden fazla dersliğe bölerek atayabilmektedir.
- **Raporlama ve Görüntüleme:**
    - Algoritma tarafından oluşturulan sınav programı, kullanıcı dostu bir tablo formatında eksiksiz olarak görüntülenebilmektedir.

5.2.2 Eksik veya Hatalı Çalışan Bileşenler

Proje isterlerine ve test sonuçlarına göre, yazılımın beklenen işlevleri tam olarak yerine getiremediği veya ileriki sürümlerde geliştirilmesi gereken kısımlar şunlardır:

- **Bireysel Öğrenci Bazlı Çakışma Kontrolü:**
    - Sistem genel dönem/bölüm bazlı çakışmaları engellese de, bireysel öğrenci bazında detaylı bir sınav çakışma kontrolü (örneğin; alttan/üstten ders alan öğrencilerin çakışmaları) henüz tam kapsamlı olarak uygulanmamıştır.
- **Fiziksel Mesafe ve Lojistik Kontrolü:**
    - Sınav planlaması yapılırken derslikler arasındaki fiziksel mesafe dikkate alınmamaktadır. Arka arkaya olan sınavlarda öğrencilerin binalar arası geçiş süresi hesaba katılmamıştır.
- **Otomatik Test Altyapısı:**

- Yazılımın doğrulanması şu an için yalnızca manuel senaryo testlerine dayanmaktadır. Kod kalitesini ve sürdürülebilirliği artıracak olan otomatik birim testleri (unit tests) ve entegrasyon testleri eksiktir.

6.GİTHUB LİNKİ:
https://github.com/MuratDev41/examplanner