TC. SAKARYA ÜNİVERSİTESİ

BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ PR. (İÖ)

WEB PROGRAMLAMA DERSİ

Adı Soyadı : SELİM ALTIN

Öğrenci No: G231210558

2. A GRUBU

FitnessCenter

Öğr.Gör.Dr.Ahmet şanslı

GitHub Bağlantısı : https://github.com/SelimAltn/FitnessCenter.git

1. Giriş ve Projenin Amacı
Bu proje, ASP.NET Core MVC mimarisi kullanılarak geliştirilen bir Spor Salonu (Fitness Center) yönetim ve randevu sistemidir. Uygulamanın temel amacı; şubelerin (salonların) yönetimi, eğitmen ve hizmet tanımları, üyelik süreçleri, randevu oluşturma ve onay mekanizması gibi operasyonların tek bir sistem üzerinden yürütülmesini sağlamaktır.
Bunlara ek olarak projede, kullanıcıdan alınan fotoğraf veya ölçü bilgilerine göre yapay zekâ destekli antrenman/beslenme önerisi üreten ve fotoğrafa dayalı "hedefe göre dönüşüm" görseli üretebilen bir modül geliştirilmiştir. Ayrıca en az bir modülde REST API üzerinden LINQ filtreleme yapılarak veri sunumu sağlanmıştır.

2. Kullanılan Teknolojiler
- **Framework:** ASP.NET Core 8.0 MVC
- **Dil:** C#
- **ORM:** Entity Framework Core (Code-First)
- **Veritabanı:** SQL Server (LocalDB)
- **Kimlik Doğrulama / Yetkilendirme:** ASP.NET Core Identity + Policy tabanlı Authorization
- **Arayüz:** Bootstrap 5, jQuery
- **Takvim:** FullCalendar.js
- **REST API Dokümantasyonu:** Swagger (Development ortamında)
- **Yapay Zekâ Servisleri:**
    - **Groq Vision API** (görsel analiz)
    - **DeepSeek API** (metin tabanlı plan üretimi)
    - **OpenAI Image API** (image-to-image dönüşüm görseli üretimi)

3. Sistem Mimarisi ve Genel Yapı
Proje, ASP.NET Core MVC yaklaşımıyla oluşturulmuştur. Temel yapı aşağıdaki bileşenlere ayrılmıştır:
- **Controllers:** İstekleri karşılayan ve iş akışını yöneten katman.
- **Models/Entities:** Veritabanı tablolarına karşılık gelen entity sınıfları.
- **Data/Context:** EF Core DbContext, migration ve seed işlemleri.
- **Services:** AI servisleri ve bazı yardımcı işlevler.
- **Views (Razor):** Arayüz sayfaları.
- **Areas:** Rol bazlı panel ayrımı:
    - **Admin** paneli
    - **BranchManager (Şube Müdürü)** paneli
    - **Trainer (Eğitmen)** paneli

Yetkilendirme tarafında Program.cs içinde policy'ler tanımlanmış, controller'lar [Authorize(Policy="...")] ile korunmuştur. Menü görünürlüğü _LoginPartial.cshtml üzerinden rol kontrolüyle düzenlenmiştir.

4. Veritabanı Tasarımı ve İlişkiler
Veritabanı, EF Core Code-First ile tasarlanmıştır. Identity altyapısı ile entegre kullanıcı yapısı kullanılmıştır.
4.1 Temel Entity'ler
- **Salon:** Şube bilgileri, çalışma saatleri ve şube müdürü ataması (ManagerUserId).
- **Uye:** Üye profili; randevular ve AI geçmişi ile ilişkilidir.

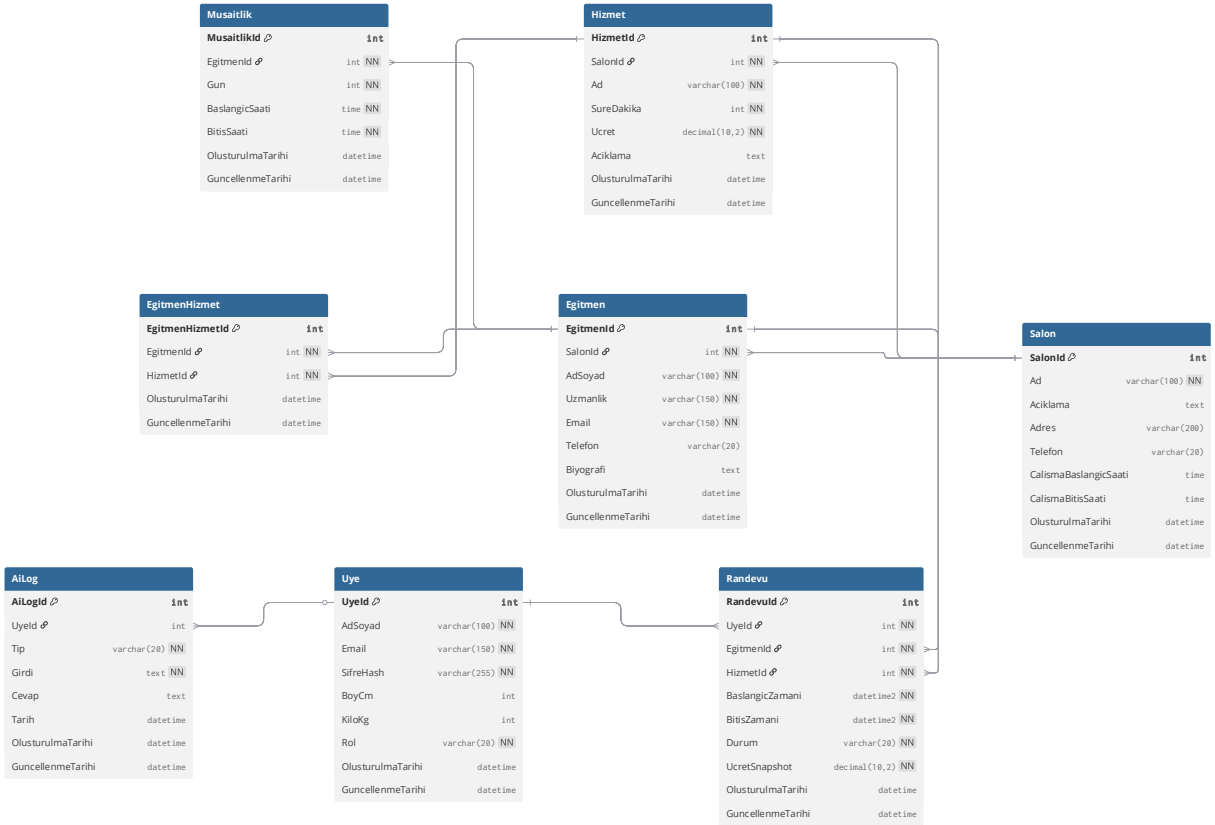
- **Uyelik:** Üye–Salon ilişkisini tutar. Bir üye birden fazla şubede üyeliğe sahip olabilir.
- **Egitmen:** Eğitmen profili; salonla (N-1), hizmetlerle (N-N) ve uzmanlıklarla (N-N) ilişkilidir.
- **Hizmet:** Hizmet adı, açıklaması ve süre (SureDakika) bilgilerini içerir.
- **Randevu:** Üye, eğitmen, salon ve hizmet ilişkilerini; başlangıç/bitiş zamanı ve durum bilgisi ile tutar.
- **Musaitlik:** Eğitmenlerin haftalık müsaitlik gün/saat aralıklarını tutar.

4.2 Köprü Tablolar

- **EgitmenHizmet:** Eğitmen–Hizmet N-N ilişkisi
- **EgitmenUzmanlik:** Eğitmen–UzmanlikAlani N-N ilişkisi

4.3 Destekleyici Modüller

- **AiLog:** AI istek/yanıt geçmişi, süre ve başarı bilgisi, giriş hash'i (InputHash) gibi alanlar.
- **SupportTicket:** Kullanıcı destek talepleri ve admin yanıtları.
- **Bildirim:** Sistem bildirimleri.
- **Mesaj:** Kullanıcılar arası mesajlaşma.



Şekil 1: Veritabanı ER Diyagramı

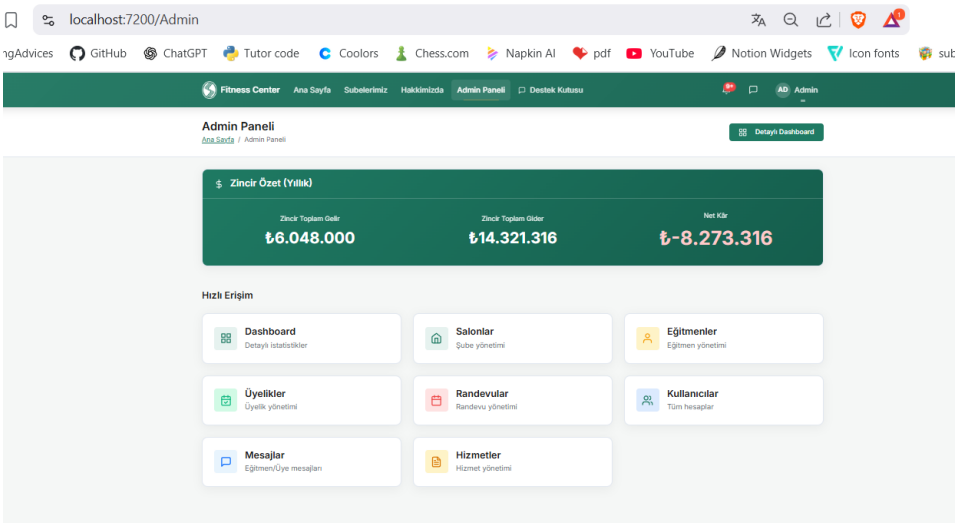
5. Kullanıcı Rolleri ve Yetkilendirme Yapısı

Sistem dört ana role sahiptir:

- **Admin:** Tüm sistem yönetimi (şube, eğitmen, üye, hizmet, randevu, destek vb.).
- **Member (Üye):** Üyelik, randevu, takvim ve AI modülü kullanım hakkı.
- **Trainer (Eğitmen):** Kendi randevularını ve mesajlarını yönetir.
- **BranchManager (Şube Müdürü):** Yalnızca kendi şubesine ait yönetim işlemleri.

Policy örnekleri: AdminOnly, MemberOnly, TrainerOnly, BranchManagerOnly, AdminOrBranchManager.

Ayrıca Admin/Trainer/BranchManager alanları Area yapısı ile ayrıştırılarak yetki sınırları netleştirilmiştir.



Şekil 2: Admin rolü için login sonrası navbar görünümü (_LoginPartial.cshtml)

6. Randevu Sistemi ve İş Kuralları

Randevu modülü, üyelerin belirli bir şubede uygun eğitmen ve hizmete göre randevu almasını sağlar.

6.1 Randevu Oluşturma Akışı

1. Üye aktif üyeliği bulunan şubeyi seçer.
2. Tarih/saat ve hizmet seçimi yapılır.
3. Uygun eğitmenler API üzerinden dinamik şekilde çekilir.
4. Form gönderildiğinde doğrulamalar geçerse randevu “Beklemede” durumuyla kaydedilir.

6.2 Eğitmen Uygunluk Kontrolleri

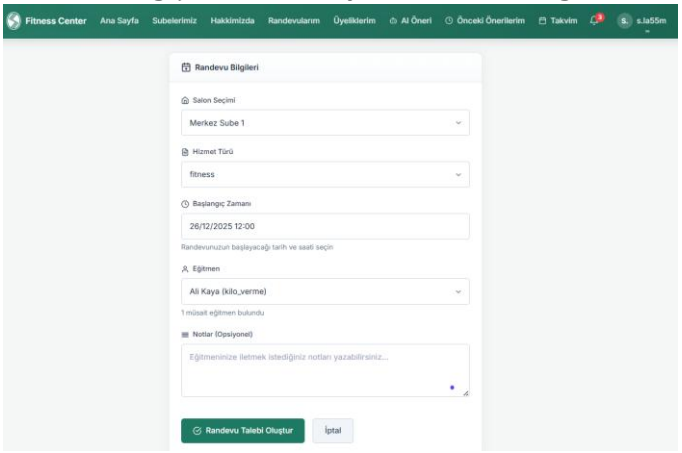
Uygun eğitmen listesi oluşturulurken şu kontroller uygulanır:

- Eğitmenin seçilen şubede çalışması
- Eğitmenin seçilen hizmeti verebilmesi
- Müsaitlik kaydının (gün/saat aralığı) uygun olması
- Aynı zaman aralığında çakışan randevusu olmaması

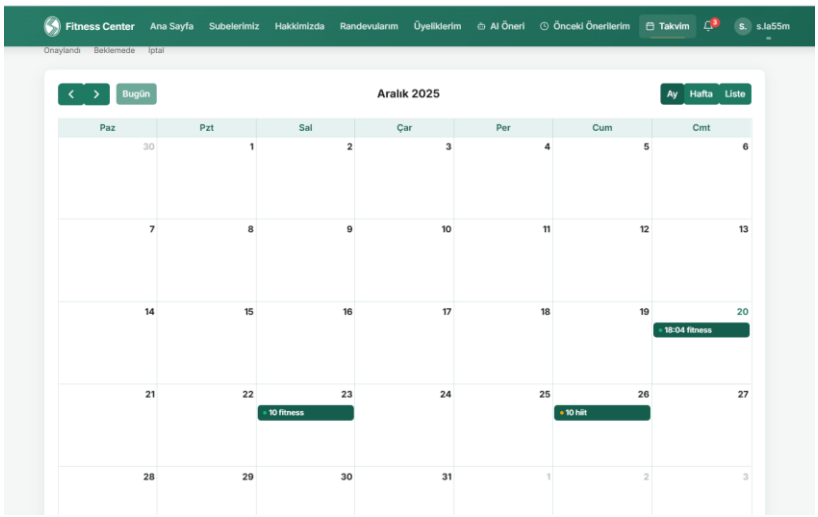
6.3 Onay Mekanizması

Randevu durumu:

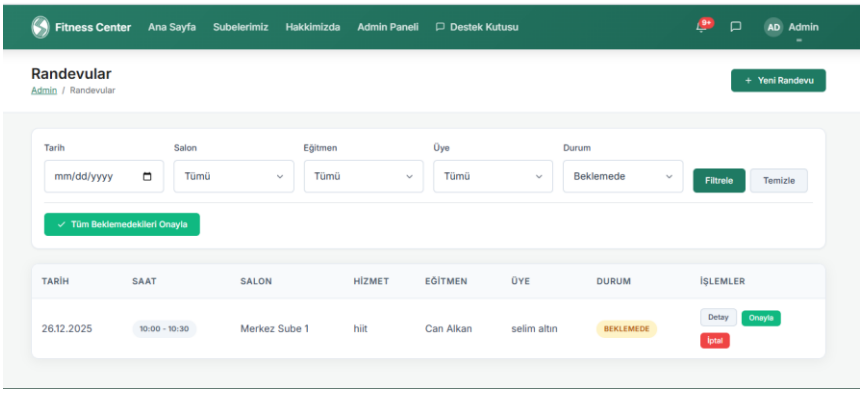
- **Beklemede → Onaylandı / İptal**
  Admin veya BranchManager randevuları tekil veya toplu şekilde onaylayabilir. Durum değişimlerinde üyelere bildirim gönderilebilir.



Şekil 3: Randevu oluşturma sayfası



Şekil 4: Takvim görünümü (FullCalendar)



Şekil 5: Admin randevu yönetimi ve durum değiştirme ekranı

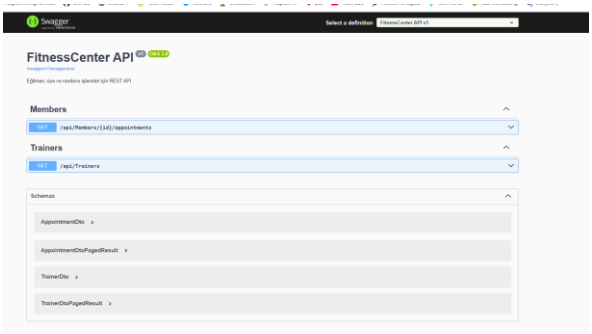
7. REST API Kullanımı ve LINQ Filtreleme

Projenin belirli bölümlerinde REST API üzerinden veri sunumu yapılmıştır. API tarafında LINQ filtreleme ve sayfalama desteği bulunmaktadır.

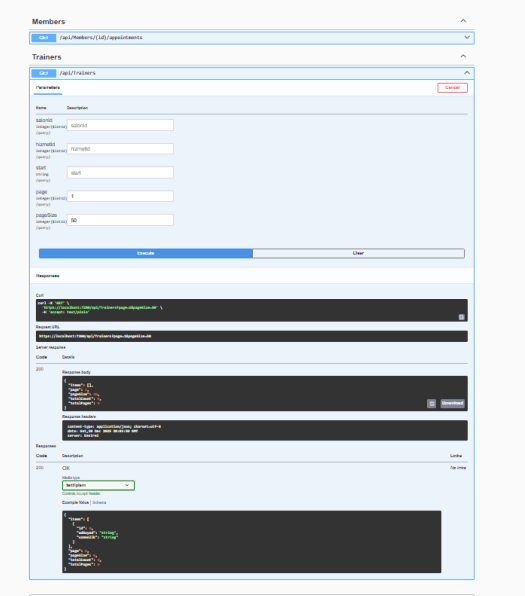
Örnek olarak:

- Uygun eğitmenleri şube/hizmet/tarih bilgisine göre getiren endpoint
- Üyenin randevularını tarih aralığı ve durum filtresiyle getiren endpoint

Swagger, Development ortamında aktif edilerek API uç noktalarının test edilmesi sağlanmıştır.



Şekil 6: Swagger UI ekranı



Şekil 7: API çağrısı örnek response ekranı

8. Yapay Zekâ Entegrasyonu

Projede yapay zekâ modülü, kullanıcıya kişiselleştirilmiş antrenman ve beslenme önerileri sunmak ve hedefe göre "dönüşüm" görseli üretmek amacıyla geliştirilmiştir. AI modülü üç farklı servis kullanır:

8.1 Groq Vision API (Görsel Analiz)

- **Dosya:** GroqVisionService.cs
- **Model:** meta-llama/llama-4-scout-17b-16e-instruct

Görevi: Kullanıcının yüklediği fotoğrafı analiz ederek fotoğrafta insan olup olmadığını kontrol eder (IsHuman). İnsan varsa kişinin fiziksel özelliklerine dair bir özet çıkarır (ör. vücut tipi, cinsiyet tahmini, boy/kilo tahmini) ve sonucu JSON formatında döndürür.

Örnek:

"Kullanıcı fotoğraf yükler → Groq Vision analiz eder → 'Bu fotoğrafta orta kilolu bir erkek var, tahmini 85kg'"

8.2 DeepSeek API (Metin Üretimi)

- **Dosya:** DeepSeekService.cs
- **Model:** deepseek-chat

Görevi: Groq Vision analiz çıktısını veya kullanıcının girdiği boy/kilo/hedef verisini kullanarak Türkçe ve detaylı şekilde:

- Haftalık antrenman planı
- Beslenme önerileri

üretir.

Örnek:

"Groq Vision: 'Şişman erkek, 85kg' + Hedef: 'Kilo Verme' → DeepSeek: 'Pazartesi 30dk koşu… günlük 1800 kalori…'"

8.3 OpenAI Image API (Görsel Dönüşüm – Image-to-Image)

- **Dosya:** OpenAIImageService.cs
- **Model:** gpt-image-1
- **Endpoint:** /images/edits (image-to-image)

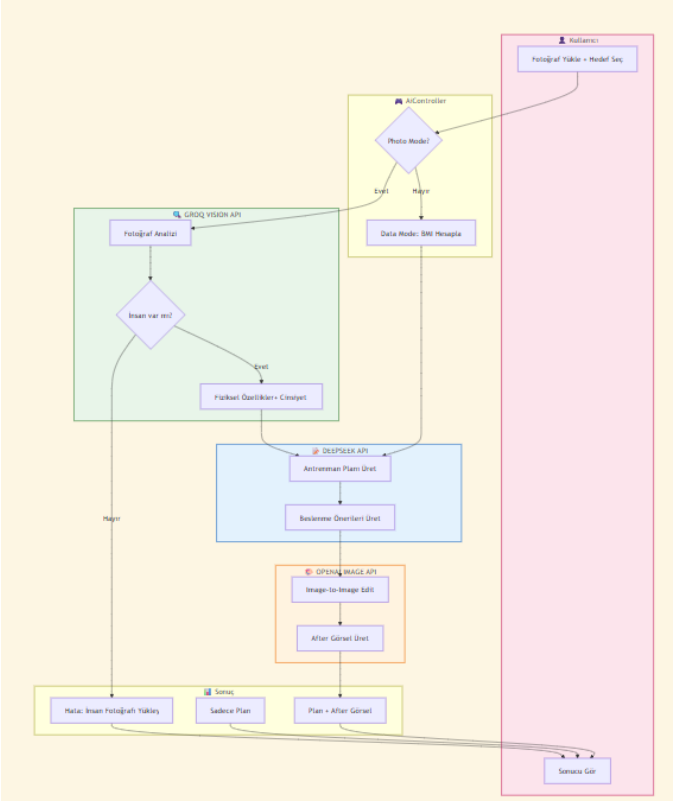
Görevi: Kullanıcının yüklediği "before" fotoğrafını referans alarak hedef doğrultusunda "after" dönüşüm görseli üretir. Aynı kişi, aynı ortam ve aynı cinsiyet korunur; yalnızca vücut görünümü hedefe göre (zayıf/kaslı/fit) dönüştürülür.

Örnek:

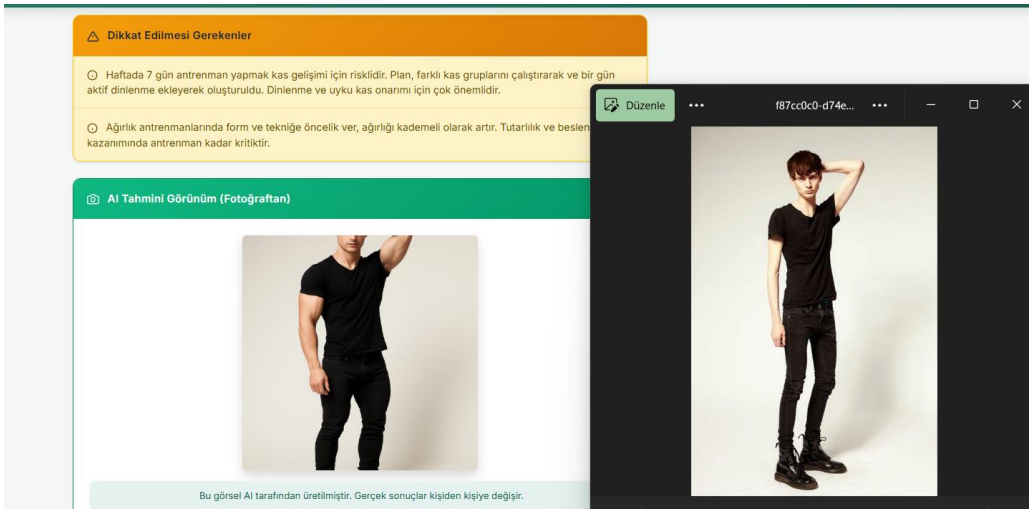
“Fotoğraf + ‘Kilo Verme’ hedefi → OpenAI Image: ‘6 ay sonraki görünüm’ görselini üretir.”

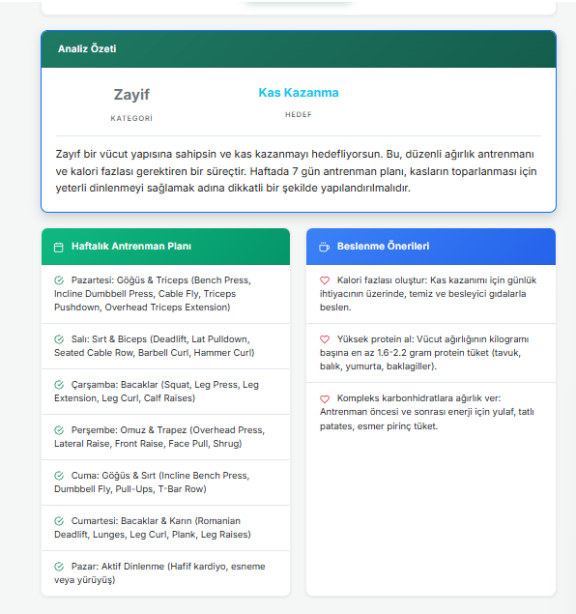
8.4 Photo Mode Tam Akış

1. Kullanıcı fotoğraf yükler + hedef seçer
2. **Groq Vision:** Fotoğraf analizi → (IsHuman, description)
3. **DeepSeek:** Analiz + hedef ile plan üretir
4. **OpenAI Image:** After görsel üretir
5. Sonuçlar kullanıcıya gösterilir



Şekil 8: AI modülü akış diyagramı



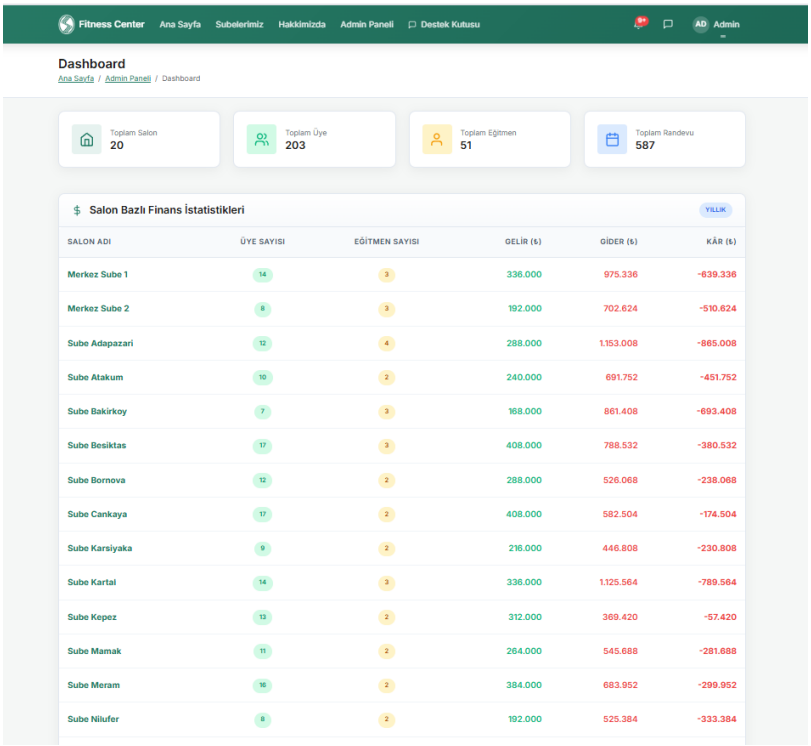


Şekil 9: AI sonuç ekranı

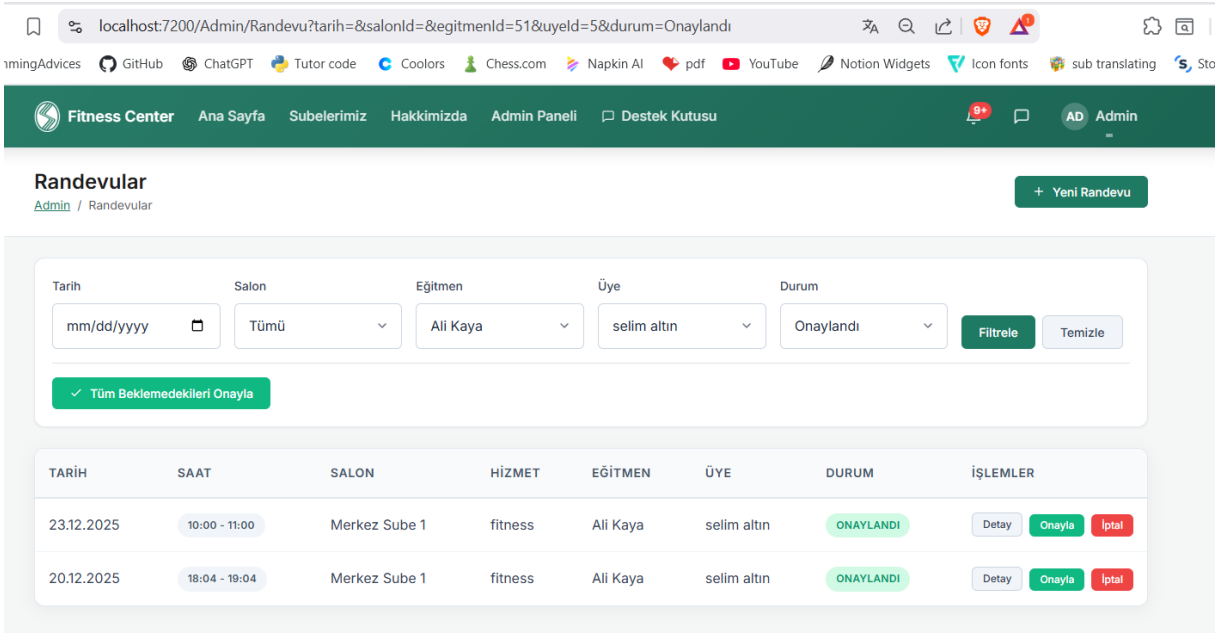
---

9. Kullanıcı Arayüzü ve Ekranlar

Arayüz Bootstrap 5 ile tasarlanmıştır. Üye tarafında randevu listeleme/oluşturma ve takvim ekranı bulunur. Admin tarafında şube, eğitmen, hizmet, uzmanlık, müsaitlik ve randevu yönetimi yapılır. BranchManager ve Trainer panelleri Area yapısı ile ayrılmıştır.



Şekil 10: Admin paneli dashboard



Şekil 11: Üye randevu listesi ve filtreleme ekranı

10. Sonuç ve Değerlendirme

Bu projede; rol tabanlı yetkilendirme, randevu motoru, REST API üzerinden LINQ filtreleme ve üç servisli (Groq Vision + DeepSeek + OpenAI Image) yapay zekâ entegrasyonu bir araya getirilerek kapsamlı bir fitness yönetim sistemi oluşturulmuştur. Sistem hem yönetimsel işlemleri (CRUD, onay, raporlama) hem de kullanıcı deneyimini artıran AI tabanlı özellikleri desteklemektedir. Geliştirilen yapı, modüler servis yaklaşımı ve Area temelli panel ayrımı sayesinde genişletilebilir bir altyapı sunmaktadır.

Ekler

- **Ek-1:** ER Diyagramı
- **Ek-2:** Örnek ekran görüntüleri (Şekil 2–10)
- **Ek-3:** Swagger / API örnekleri