T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Proje Raporu

Ders: Web Programlama

Dönem: 2024-2025 Güz Dönemi

Grup No: 2.Öğretim B Grubu

Konu: Kuaför/Berber İşletme Yönetim Sistemi

Öğrencinin Adı Soyadı: Burak Emre SARIKOÇ
Öğrencinin Numarası: g221210077

Dersi Veren Öğretim Görevlisi: Dr.Öğr.Üyesi Can YÜZKOLLAR

Projenin Github Adresi: https://github.com/Isholiday/kuafor

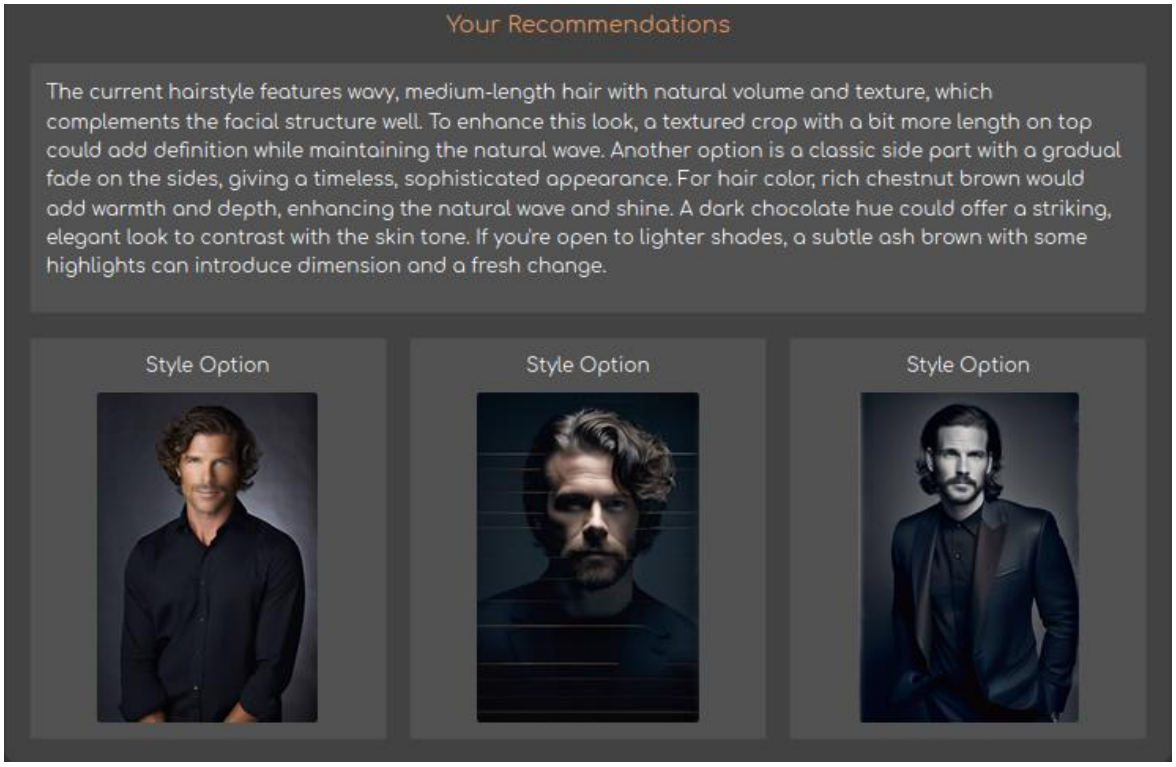
Proje Özeti

Bu projede, ASP.NET Core MVC kullanılarak bir kuaför işletme yönetim sistemi geliştirilmiştir. Sistem, kuaför/berber salonlarının işlemlerini, çalışanlarını, randevu sistemlerini yönetmelerine olanak tanır. Yapay zeka entegrasyonu ile kullanıcılar saç modeli veya renk önerisi alabilir.

Yapay Zeka Öneri Sistemi

Projenin en önemli özelliklerinden biri olan yapay zeka tabanlı saç modeli ve renk önerisi sistemi, kullanıcıların fotoğraflarını yüklemeleriyle çalışmaktadır. Bu özellik, iki temel bileşen üzerine kuruludur:

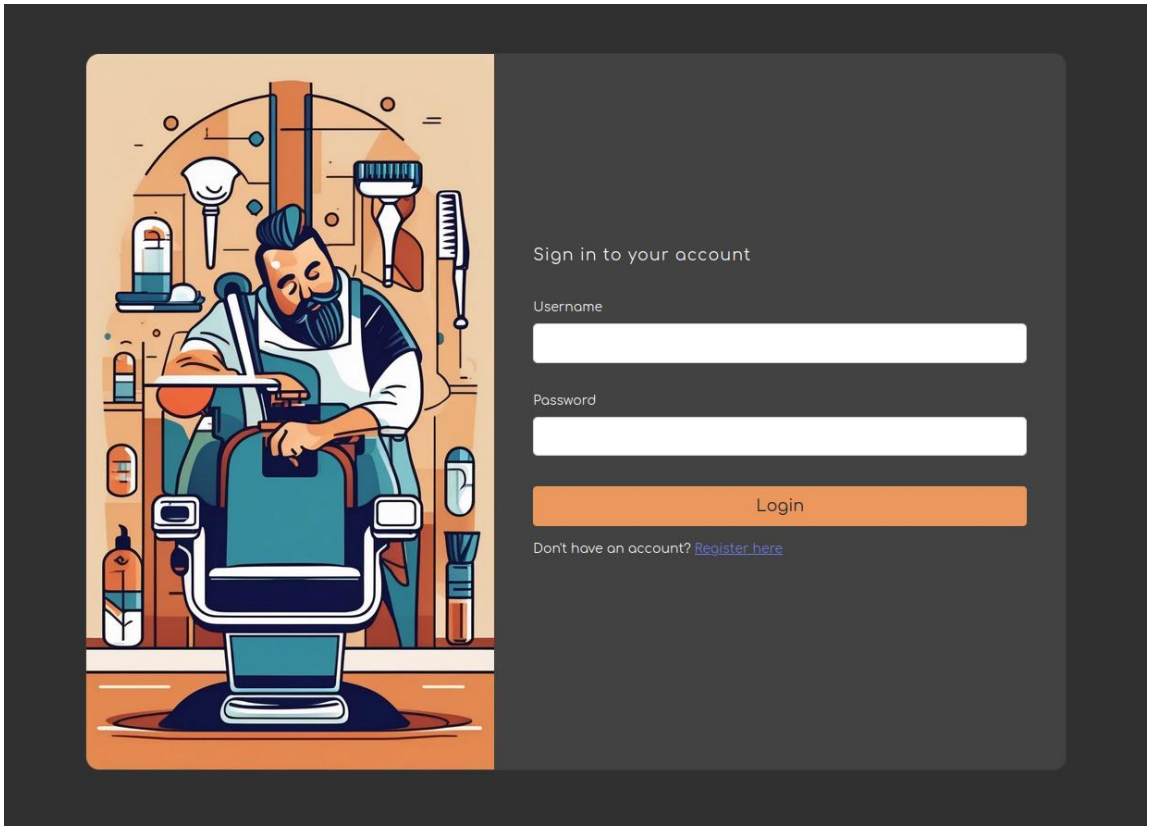
- Replicate API
  Fotoğraf analizi ve saç modeli önerileri için Replicate API kullanılmıştır. API, yüklenen fotoğrafları işleyerek kullanıcılara en uygun saç stillerini ve renklerini öneren bir yapay zeka modeliyle entegre edilmiştir. Bu sayede, kullanıcılar mevcut trendlere ve yüz şekillerine uygun kişiselleştirilmiş öneriler alabilmektedir.
- GPT-4 Modeli
  Metin tabanlı geri bildirim ve açıklamalar için OpenAI'nin GPT-4 modeli entegrasyonu yapılmıştır. Kullanıcılara, önerilen saç stillerinin avantajları ve bu stillerin yüz şekilleriyle uyumu hakkında açıklamalar sunulmaktadır. GPT-4, doğal dil işleme yeteneği sayesinde kullanıcılarla etkileşim kurarken anlaşılır ve akıcı bir dil kullanmaktadır.



Giriş ve Kayıt Sistemi

Proje, kullanıcı dostu bir giriş ve kayıt sistemi ile donatılmıştır. Bu özellikler, sistemin güvenilirliğini artırmak ve kullanıcı deneyimini iyileştirmek için özel olarak tasarlanmıştır.

- **Kayıt Olma**
  - Kullanıcılar, ad, soyad, e-posta adresi ve şifre gibi bilgilerini girerek sisteme kolayca kayıt olabilir.
  - Şifreler, veri güvenliğini sağlamak amacıyla SHA-256 şifrelemesi kullanılarak saklanmaktadır.
  - Kullanıcı kayıtları, veritabanında doğrulama ve yetkilendirme süreçlerine uygun şekilde tutulur.
- **Giriş Yapma**
  - Kayıtlı kullanıcılar, kullanıcı adı ve şifre bilgilerini kullanarak sisteme giriş yapabilir.
  - Yanlış giriş denemelerinde kullanıcılar bilgilendirilir ve güvenlik amacıyla belirli deneme sınırları uygulanır.
- **Rol Tabanlı Erişim Kontrolü**
  - Sistem, kullanıcı ve admin rolleri arasında ayrım yaparak erişim yetkilerini sınırlandırır.
  - Admin, sistem üzerinde daha geniş bir yetkiyle işlem yapabilirken, standart kullanıcılar yalnızca kendi işlemleriyle ilgili yetkilere sahiptir.
- **Hata Yönetimi**
  - Hatalı girişlerde, kullanıcılar bilgilendirilir ve önerilerle desteklenir.

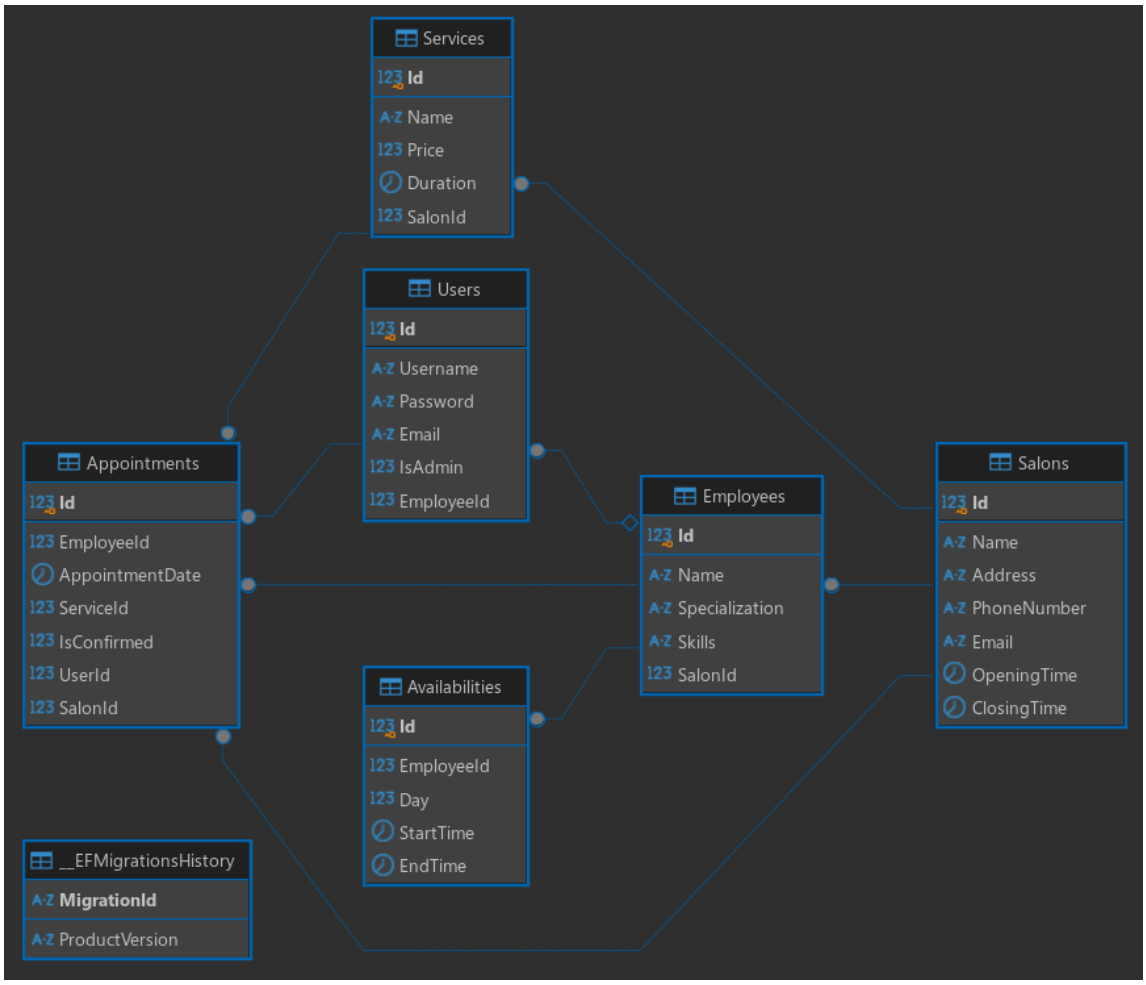


Veritabanı Yönetimi

Veritabanı yönetimi için Microsoft SQL Server kullanılmış ve işlemler Entity Framework Core (EF Core) ile gerçekleştirilmiştir. EF Core, hem ilişkisel veritabanlarıyla çalışmayı kolaylaştırmış hem de kodla veritabanı modellerinin uyumlu hale getirilmesini sağlamıştır.

- EF Core Kullanımı
  Model sınıfları ve ilişkileri, EF Core üzerinden kodla tanımlanmıştır. Veritabanı şeması, Code-First yaklaşımıyla oluşturulmuş ve düzenlenmiştir. EF Core'un sunduğu özellikler, veri erişim katmanında hem performanslı hem de güvenli bir yapı oluşturulmasına olanak tanımıştır.

- LINQ ile Veri Sorgulama
  EF Core ile veritabanı sorguları için LINQ (Language Integrated Query) kullanılmıştır. LINQ, veri sorgularının daha okunabilir ve güvenli hale getirilmesini sağlamıştır. Karmaşık ilişkili sorgular, LINQ yardımıyla kolaylıkla yazılmıştır. LINQ sorgularıyla REST API entegrasyonu sağlanarak, istemcilere dinamik veri akışı sunulmuştur.



Güvenlik ve Yetkilendirme

Projenin güvenlik gereksinimlerini karşılamak için JWT (JSON Web Token) tabanlı bir yetkilendirme sistemi uygulanmıştır. Kullanıcı ve yönetici girişlerinde güvenli bir kimlik doğrulama sağlanmıştır. JWT, kullanıcı yetkilerini ve oturum sürelerini saklamak için şifrelenmiştir. Yetkilendirme, kullanıcı rolleri arasında doğru bir şekilde ayrılmıştır (Admin ve Standart Kullanıcı).

```
public class JwtService(IConfiguration configuration) {  2 usages
    private readonly TokenValidationParameters _tokenValidationParameters = new() {
        ValidateIssuerSigningKey = true,
        IssuerSigningKey = new SymmetricSecurityKey(
            Encoding.UTF8.GetBytes(configuration["Jwt:Key"]!)
        ),
        ValidateIssuer = false,
        ValidateAudience = false,
        ValidateLifetime = true
    };

    public string GenerateToken(int userId, string username, string role) {  1 usage
        var claims = new[] {
            new Claim( type: ClaimTypes.NameIdentifier, userId.ToString()),
            new Claim( type: ClaimTypes.Name, username),
            new Claim( type: ClaimTypes.Role, role)
        };

        var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(configuration["Jwt:Key"]!));
        var credentials = new SigningCredentials(key,  algorithm: SecurityAlgorithms.HmacSha256);
        var expires :DateTime  = DateTime.Now.AddMinutes(120);

        var token = new JwtSecurityToken(
            claims: claims,
            expires: expires,
            signingCredentials: credentials
        );

        return new JwtSecurityTokenHandler().WriteToken(token);
    }

    public ClaimsPrincipal? ValidateToken(string token) {  1 usage
        try {
            return new JwtSecurityTokenHandler().ValidateToken(token, _tokenValidationParameters, out _);
        } catch {
            return null;
        }
    }
}
```