PROJE OLUŞTURMA

1. Gerekli uygulama ve sürüm konrolleri yapıldı.
2. Proje “HOME” dizinine kuruldu.

VERİTABANI

1. SSMS kullanılarak .sql uzantılı dosya LocalDB veritabanına yüklendi.
2. Oluşturulan database’in ER diagramı SSMS kullanılarak oluşturuldu ve görselleştirildi.

BACKEND

1. Clean Architecture için klasör yapısı oluşturuldu.
2. ‘**dvdrental** veritabanı erişimi (**DvdRentalDBContext**) projeye DI olarak tanımlandı.
3. Interface oluşturuldu (actor) ardından service interface’I implement etti ve methodlar dolduruldu.(Repository pattern ile)
4. Contorller dosyasında Servisler kullanılarak REST komutları oluşturuldu. Async/await kullanarak, bu komutlar işlemlerin eşzamanlı ve sıra beklemeden gerçekleştirilmesini sağlar. Burada scaffold kullandımç
   1. GetAllActors [HttpGet]
   2. GetActor [HttpGet("{id}")]
   3. CreateActor [HttpPost]
   4. UpdateActor [HttpPut("{id}")]
   5. DeleteActor [HttpDelete("{id}")]
5. Dependency Injection için Program.cs dosyasına addscope ile ActorService register edildi.
6. Actor modeli için oluşturulan API’ler test edilecek, bunları yaparken Swager UI kullanılacak.
7. Swager UI projeye tanımlandı ve Actor veritabanı için REST komutları test edildi.
8. Film veritabanı için de komutlar test edildi. DI film için de AddScope kullanıldı.
9. Tüm tablolar için service ve interfaceler oluşturuldu. (Temel komutlar aynı)(değişmesi gereken şeyler var mı bunun üzerine çalışılacak.)
10. Category için de contorller tanımlandı ve swager testi yapıldı.
11. Diğer 2 modelin verileri bekletielecek actor için angular ile frontend yapılacak. Ve bağlantı sağlanacak.
12. Frontend ksımında ilk olarak MDB 5 desteği eklendi. Css formatı scss e dönüştürüldü.
13. Services ve models klasörleri oluşturuldu. Gerekli methodalr dolduruldu.
14. Actor, category ve film için form ve list yapıları oluşturuldu.
15. Form ve list’in component.ts ve html dosyaları gerekli fonksiyonlar ile dolduruldu.
16. App-component’in html I gerekli tasarıma göre değiştirildi.
17. Site üzerinden veri eklerken otomatik 0 id ataması özelliği kaldırıldı. Kullanıcının id girmesi sağlandı.
18. Hata durumları anlamlı şekilde kullancıya mesaj olarak verildi.
19. Pagination Sistemi eklendi uygulamaya
20. Arama Çubuğu eklendi.
21. Sayfa tasarımı modern bir görüntüye kavuştu.

Eklenebilecek özellikler:

* 1. Tüm özellikleri aynı iki film eklenmesini engellemek ve film mevcut uyarısı vermek.
  2. Özelliklerden sıralanabilir olanlara göre tablo sıralama özelliği dahil etmek.
  3. Filmler silindikten sonra boşalan ID’lerin listesini tutmak ve film eklendikçe listeden o ID’yi kaldırmak. (Doubly linked list kullanılabilir.)
  4. Filtreleme özellikleri eklenebilir.
  5. Arama çubuğu eklenebilir.

Sıradaki yapılabilir düzenlemeler;

FilmActor ve FilmCategory controllerları eklenecek, ardından swagger ile kontrolleri sağlnacak. Başarılı çıktı verirse client tarafına geçilecek ve categroy ve aktörlerin yanlarına yeni butonlar eklenerek içerdikleri filmler listelenecek.

Film eklenirken hangi kategoriye ait olduğu da girilecek ve filmcategory kısmına bu eşleşme eklenecek.

Aynı şey actor eklenirken oynadığı filmlerin eklenmesiyle de gerçekleşecek. Film mevcutsa güncelleme yapılacak sadece eğer film mevcut değilse onun eklemesi de yapılacak ve kategori bilgisi de istenecek.