İZMİR KÂTİP ÇELEBİ ÜNİVERSİTESİ
2010
FEN BİLİMLERİ ENSTİTÜSÜ

Film ve Dizi Arşiv Yönetim Sistemi

Yazılım Mühendisliği Ana Bilim Dalı

Dönem Projesi

Gencay Ceyhan

Proje Danışmanı: Dr. Öğr. Üyesi Serpil Yılmaz

Ağustos 2024

Film ve Dizi Arşivi Yönetim Sistemi

Öz

Günümüz dijital çağında, medya içeriklerinin sayısının artmasıyla birlikte, bu içeriklerin düzenlenmesi ve erişimi büyük bir önem arz etmektedir. Mevcut çözümler genellikle sınırlı özelleştirme seçenekleri ve kullanıcı dostu ara yüz eksiklikleri gibi sorunlarla karşılaşmaktadır. Bu projede geliştirilen film ve dizi arşivi uygulaması, kullanıcının kişisel koleksiyonlarını düzenlemesini sağlayacak kapsamlı bir platformun arka planını sunmaktadır. Uygulama, kullanıcıların film ve dizi bilgilerini kolayca kayıt edebilecekleri bir backend uygulamadır. Veri tabanı yönetimi ve medya veri entegrasyonu gibi önemli teknik özellikler de bu projede yer almaktadır.

Anahtar Sözcükler: Mühendislik, yazılım, film ve dizi arşivi, veri tabanı yönetimi, yazılım geliştirici

Movie and TV Series Archive Management Program

Abstract

In today's digital age, with the increasing number of media content, the organization and access to these contents are of great importance. Existing solutions often face problems such as limited customization options and lack of user-friendly interfaces. The film and series archive application developed in this project provides the background of a comprehensive platform that will allow the user to organize their personal collections. The application is a backend application where users can easily record film and series information. Important technical features such as database management and media data integration are also included in this project.

Keywords: Engineering, software, movie and TV series archives, database management, backend

Teşekkür

Proje çalışmasına katkılarından dolayı arkadaşım Rıfat Türkmen'e teşekkürlerimi borç bilirim.

İçindekiler

Şekiller Listesi

Kısaltmalar Listesi

API	Application Programming Interface
BIOS	Basic Input-Output System
CD	Compact Disc
CI/CD	Continuous Integration/Continuous Deployment
CLI	Command Line Interface
http	Hyper Text Transfer Protocol
JPA	Java Persistence API
JSON	JavaScript Object Notation
LTS	Long Term Support
ORM	Object Relation Mapping
ORDBMS	Nesne-İlişkisel Veri Tabanı Yönetim Sistemi
RAM	Random Access Memory
SQL	Structured Query Language
UI	User Interface
VTSY	Veri Tabanı Yönetim Sistemi
YAML	Ain’t Markup Language
XML	Extensible Markup Language

Bölüm 1

Giriş

Günümüz dijital çağında, medya içeriklerinin hızla artması ve çeşitlenmesi, bireylerin ve kurumların film ve dizi arşivlerini etkin bir şekilde yönetme gereksinimini doğurmuştur. Film ve Dizi Arşivi Yönetim Sistemi, bu ihtiyaca yanıt olarak geliştirilmiş bir yazılım çözümüdür. Bu sistem, kullanıcıların sahip oldukları film ve dizileri düzenlemelerine, arşivlemelerine ve kolayca erişim sağlamalarına olanak tanıyacak programın backend kısmını oluşturmaktadır. Ayrıca, genişleyen medya koleksiyonlarının organize edilmesini sağlar.

Medya arşivlerinin yönetimi, sadece bireysel kullanıcılar için değil, aynı zamanda film stüdyoları, televizyon kanalları, eğitim kurumları ve kütüphaneler gibi kurumlar için de büyük önem taşır. Film ve dizi koleksiyonlarının etkin bir şekilde yönetilmesi, hem zamandan tasarruf sağlar hem de bilgiye hızlı erişimi mümkün kılar. Bu bağlamda geliştirilecek olan Film ve Dizi Arşivi Yönetim Sistemi, kullanıcıların medya içeriklerini kategorize etmelerine, meta verileri düzenlemelerine ve arama fonksiyonları sayesinde istedikleri içeriğe hızlıca ulaşmalarına yardımcı olacaktır.

Bu sistemin temel özelliklerinden biri, kullanıcı dostu bir arkayüze sahip olmasıdır. Günümüz dijital çağında, medya içeriklerinin hızla artması, bu tür verilerin etkin bir şekilde saklanması, yönetilmesi ve erişilmesi gerekliliğini doğurmuştur.

Proje kapsamında geliştirilecek olan sistemin bir diğer önemli özelliği ise güvenlik ve yedekleme fonksiyonlarıdır. Kullanıcıların medya arşivlerinin güvenliğini sağlamak ve veri kaybını önlemek amacıyla, sistem düzenli aralıklarla yedekleme yapma yeteneğine sahip olacaktır. Ayrıca, kullanıcı hesapları ve verileri güvenli bir şekilde saklanacak, yetkisiz erişimlere karşı korunacaktır.

Sonuç olarak, Film ve Dizi Arşivi Yönetim Sistemi, medya içeriklerinin düzenli ve etkin bir şekilde yönetilmesini sağlamak için geliştirilmiş kapsamlı bir yazılımdır. Bu proje, bireysel kullanıcıların yanı sıra kurumsal kullanıcıların da medya arşivlerini

daha etkin bir şekilde yönetmelerine olanak sağlayacak backend uygulamayı barındırmaktadır.

Bölüm 2

Genel Bilgiler

2.1. Yazılım

Yazılım, elektronik aygıtların belirli bir işi yapmasını sağlayan programların tümüne verilen isimdir. Bir başka deyişle var olan bir problemi çözmek amacıyla bilgisayar dili kullanılarak oluşturulmuş anlamlı anlatımlar bütünüdür. Kelime işlemci programları, bilgisayarınız açıldığında CD sürücünüzü, sabit sürücüleri, RAM'i tanıyan BIOS; işletim sistemi, web tarayıcınız, virüslerin kendileri, antivirüs programları hep birer yazılımdır.

2.1.1. Bir Yazılımda Olması Gereken Temel Özellikler

Doğruluk: Yazılımın belirtilmiş ihtiyaçlarını karşılamasıdır.

Güvenilirlik: Gerekli işlevi ne hassaslıkla yerine getireceği beklentisidir.

Verimlilik: İşlevin gerçekleştirilmesi için kullanılması gereken bilgisayar kaynakları ve kod miktarıdır.

Güvenlik(Bütünlük): Yazılım ve bilgilerine, istenmeyen insanlarca ulaşımın ne derece engellenebildiğidir.

Kullanılabilirlik: Programın öğrenilmesi, çalıştırılması, girdi hazırlama ve çıktı yorumlama işlemlerinin kolaylık derecesidir.

Hata bulma kolaylığı: Hatanın yerini bulma ve düzeltme kolaylığıdır.

Esneklik: Yazılımda değişiklik yapma kolaylığıdır.

Taşınabilirlik: Programın farklı donanımlarda ve yazılım sistemi ortamlarında kullanılmasıdır.

Tekrar kullanılabilirlik: Yazılım tamamının ya da bir bölümünün farklı bir uygulamada kullanılabilmesidir.

Birlikte çalışabilirlik: Bir yazılım sisteminin diğerleri ile bağlantı sağlaması kolaylığıdır.

Bilgisayar yazılımları genel olarak 2 ana grupta incelenebilir.

- Sistem yazılımları (system software): Bilgisayarı yöneten, denetleyen, kontrol eden yazılımlardır. Örnek: Linux, Pardus, Windows vb.
- Uygulama yazılımları (application software): Belli bir alana ve uygulamaya ilişkin olarak kullanıcılar için geliştirilen yazılımlardır. Örnek: Web hazırlama araçları, programlama araçları, vb.

Bütün sistem programları içinde en temel yazılım işletim sistemidir ki, bilgisayarın bütün donanım ve yazılım kaynaklarını kontrol ettiği gibi kullanıcılara ait uygulama yazılımlarının da çalıştırılmalarını ve denetlenmelerini sağlar.

2.2. Programlama Araçları

2.2.1. Yazılım Çeşitleri

İster genel ister özel amaçlı olsun tüm uygulama ve sistem yazılımları programlama dilleriyle yazılır. İster genel ister özel amaçlı olsun tüm uygulama ve sistem yazılımları programlama dilleriyle yazılır. Bir programlama dili, insanların bilgisayara çeşitli işlemler yaptırmasına imkân veren her türlü sembol, karakter ve kurallar grubudur. Programlama dilleri insanlarla bilgisayarlar arasında tercümanlık görevi yapar. Programlama dilleri, bilgisayara neyi, ne zaman, nasıl yapacağını belirten deyim ve komutlar içerir.

Bu programlama dilleri şunlardan oluşur.

- **Genel komutlar:** Programlama dilinin anlayacağı komutlardır.
- **Gelişmiş komutlar:** Genel komutları kullanarak oluşturulmuş komutlardır.
- **API komutları:** İşletim sisteminin sunduğu özellikleri kullanan komutlardır.

- **Derleyici komutları:** Komut içinde çalışmayıp derleme esnasında alınan bilgilere göre derleme yapılmasını sağlar.
- **Aktif nesneler:** Buton, Menü, Gösterge çubuğu ve Tabpanel‟ gibi bileşenlerin genel adıdır.

2.2.2. Programlama Dilleri Çeşitleri

Bir programlama dili ya insan ya da makine anlayışına yakındır. İnsan anlayışına daha yakın programlara dillerine yüksek seviyeli programlama dilleri, makineye yakın olanlara ise düşük seviyeli programla dilleri denir.

- Yüksek seviye programlama ile yazılan projelerin kaynak kodları kısa, derlenmiş halleri ise uzun olur. Çalışma hızları ise yavaştır.
- Alçak seviye programlama dilleri ile yazılan projelerin kaynak kodları uzun, derlenmiş halleri ise kısa olur. Çalışma hızları ise en yüksek seviyededir. Programlama dillerini seviyelerine göre 5 ana gruba ayırabiliriz:

➢ Çok yüksek seviyeli diller ya da görsel diller (Access, Foxpro, Paradox, Xbase, Visual Basic, Oracle Forms)
➢ Yüksek seviyeli diller (Bunlara algoritmik diller de denir; Fortran, Foxpro, Paradox, Xbase, Visual Basic, Oracle Forms)
➢ Orta seviyeli diller C, C++, C#. Orta seviyeli diller daha az kayıpla makine diline çevrilebildiğinden daha hızlı çalışır
➢ Alçak seviyeli programlama dilleri (Sembolik makine dili, Assembler)
➢ Makine dili (En aşağı seviyeli programlama dilidir (Saf makine dili ve 0’lardan oluşuyor.

Film ve dizi arşiv yönetim programımda Spring Boot framework’ünü kullandım. Spring Boot üzerinde çalıştığı dil olan Java, yüksek seviyeli programlama dilleri kategorisine girer. Java, Spring Boot gibi büyük ve karmaşık uygulamaların geliştirilmesini kolaylaştıran zengin kütüphanelere ve araçlara sahip, soyutlama düzeyi yüksek bir dildir. Bu nedenle Spring Boot, Java'nın bir ürünü olarak, yüksek seviyeli programlama dilleri kategorisi altında değerlendirilebilir.

2.2.3. Veri Tabanı Sistemleri

2.2.3.1. Veri Nedir?

Veri (Data) ve Bilgi (Information) terimleri genellikle birbiriyle karıştırılır ve birbirinin yerine kullanılır. Veri, olaylar veya yerler, insanlar veya diğer nesnelerle ilgili gerçekler olarak tanımlanabilir. Bu nedenle, veri kavramı, farklı alanlarda farklı anlamlar taşıyabilir. Bilgisayar ortamında, veriyi, bir manyetik disk (örn. sabit disk), yarı iletken bellek birimi (örn. RAM) veya bir veri tabanında işlemeye hazır durumda bulunan kayıtlar olarak kabul ederiz. Veri, ilk bakışta anlamsız ve kullanışsız gibi görünen, ancak işlenmeye veya analiz edilmeye hazır birçok kaydı ifade eder.

Bilgi ise, işlenmiş ve kullanıcıya yararlı ve kullanışlı hale getirilmiş verilerdir. Örneğin, bir öğrenci veri tabanını düşünürsek; bu veri tabanından "Bilgisayar Teknolojisi ve Programlama" bölümü öğrencileri arasında, "Veri Tabanı Yönetim Sistemleri" dersinden bütünlemeye kalan öğrencilerin listesini elde edersek, bu bilgi, dersi veren öğretim üyesi için çok değerli ve anlamlı olacaktır.

2.2.3.2. Veri Tabanı Nedir?

Veri tabanı, birbirleriyle ilişkili verilerin tutulduğu, kullanım amacına uygun şekilde organize edilmiş veri topluluklarının hem mantıksal hem de fiziksel tanımlarının bulunduğu bilgi depolarıdır. Veri tabanları, gerçek dünyadaki nesneleri ve bu nesneler arasındaki ilişkileri modelleyen sistemlerdir.

Veri tabanı; banka, üniversite, okul, seyahat şirketi, hastane ve devlet dairesi gibi kuruluşların işleyebilmesi için gerekli olan operasyonel verilerin bir arada tutulduğu sistemdir. Örneğin, ticari bir şirket için müşteri, satış, ürün ve ödeme bilgileri; bir okul için öğrenci bilgileri, ders kayıtları ve öğretmen bilgileri; bir hastane için ise hasta bilgileri, doktor bilgileri ve yatak durumu gibi çeşitli veriler veri tabanlarında tutulur.

Belirli bir konu hakkında toplanmış veriler, bir veri tabanı programı altında organize edilir. Bu verilerden, ihtiyaç duyulan bilgiler istenildiğinde görüntülenebilir,

yazdırılabilir veya bu bilgilerden yeni bilgiler türetilip farklı amaçlar için kullanılabilir.

2.2.3.3. Neden Veri Tabanı Kullanılır?

Bilgisayar ortamında verilerin depolanması, saklanması ve erişilmesi konusunda yıllar içinde çeşitli yöntemler geliştirilmiştir. Geleneksel Yaklaşım, verilerin ayrı ayrı dosyalarda gruplanmasına dayanır. Veri tabanı programları öncesinde, programlama dillerinde sıralı ve rastgele dosyalama sistemleri kullanılırdı. Ancak verilerin sayısının artması, aynı anda erişim ihtiyacı ve bu verilerin düzenlenmesi gibi gereksinimler arttıkça geleneksel yaklaşım yetersiz kalmıştır.

Veri tabanı yaklaşımının avantajları şunlardır:

- Verilerin tekrarlanmasının önlenmesi, merkezi denetim ve tutarlılık sağlanması,
- Veri paylaşımının kolaylaştırılması,
- Fiziksel yapı ve erişim yöntemi karmaşıklıklarının kullanıcıdan gizlenmesi,
- Kullanıcılara sadece ilgilendikleri verilerin sunulması,
- Uygulama yazılımı geliştirme sürecinin kolaylaştırılması,
- Veri bütünlüğü ve güvenlik mekanizmalarının sağlanması,
- Yedekleme, yeniden başlatma ve onarma gibi işletim sorunlarına çözüm getirilmesi.

2.2.3.4. Veri Tabanı Yönetim Sistemi Nedir?

Veri tabanı Yönetim Sistemi (VTYS), bir veri tabanını oluşturmak, düzenlemek, geliştirmek ve bakımını yapmak gibi karmaşık işlemleri gerçekleştiren, birden fazla programdan oluşan bir yazılım sistemidir. VTYS, kullanıcı ile veri tabanı arasında bir arabirim sağlar ve veri tabanına erişimi mümkün kılar. Bu sistem, fiziksel bellek ve veri tiplerini kullanıcı adına yönetir ve kullanıcılarına standart bir SQL ara yüzü sunar, böylece veri giriş-çıkışı için uygun ara yüzlerin geliştirilmesine olanak tanır. VTYS'de kullanıcılar, roller ve gruplar vardır ve bunlar verileri düzenleyip, erişim haklarını

belirlerler. Her kullanıcının, veri tabanı yöneticisi tarafından tanımlanmış belirli yetkileri vardır; bu yetkiler artırılabilir, kısıtlanabilir veya kaldırılabilir.

2.2.3.5. Veri Tabanı Yönetim Sisteminin Sağladığı Yararlar

- Veriler tek bir merkezde tutulur, aynı veri farklı bilgisayarlarda tekrar tekrar saklanmaz.
- Eklenen, düzenlenen veya silinen bir bilgi merkezde güncellenmezse, veri tutarsızlığı oluşabilir. Aynı verinin birden çok kopyasının bulunması bakım zorluklarını da beraberinde getirir.
- Veri tabanı Yönetim Sistemi kullanılmayan sistemlerde veriye erişim sıralı yapılır ve birden fazla kullanıcı aynı anda aynı veriye erişemez. VTYS'de ise veri bütünlüğü korunarak saniyede yüzlerce, binlerce erişim yapılabilir.
- Bir tabloda yapılan işlem, ilgili diğer tablolarda da güncellemeye neden olur. Örneğin, bir öğrencinin kaydı silindiğinde, öğrencinin not, harç ve diğer bilgileri de otomatik olarak silinir.
- Verilerin kasıtlı veya yanlış kullanımını önlemek için güvenlik önlemleri bulunur. Örneğin, veri tabanına giriş için kullanıcı adı ve şifre istenmesi, kişilerin yalnızca kendilerine verilen haklarla işlem yapabilmesi gibi.
- Programcı, verilerin yapısı, organizasyonu ve yönetimi ile ilgilenmeden, veri tabanının bunları yönetmesi sağlanır. Veri bağımsızlığı, VTYS'nin en temel amaçlarından biridir.

Film ve dizi arşivi yönetim programında Oracle database programını kullandım.

Bölüm 3

Proje ile İlgili Uygulamalar

3.1. Spring Boot

Spring Boot, Spring Framework'ün daha hızlı ve basit bir şekilde kullanılmasını sağlayan bir eklenti ya da araçtır. Spring Framework, Java platformu üzerinde büyük ve karmaşık uygulamaları geliştirmek için kullanılan popüler bir framework'tür. Ancak, Spring Framework'ün konfigürasyonu karmaşık ve zaman alıcı olabilir. Bu sorunu çözmek için Spring Boot geliştirilmiştir.

Spring Boot, minimal konfigürasyon ile bir Spring uygulaması başlatmanıza olanak tanır ve "standalone" çalışabilen, gömülü bir sunucuya sahip bir uygulama oluşturmayı kolaylaştırır. Spring Boot, tipik olarak "konfigürasyon üzerinden anlaşma" (convention over configuration) yaklaşımını benimser, bu da geliştiricilerin gereksiz yapılandırma yapmadan projelerine hızlı bir şekilde başlayabilmelerini sağlar.

Spring Boot'un temel özellikleri arasında şunlar bulunur:

- Otomatik Yapılandırma: Spring Boot, uygulamanın ihtiyaçlarına göre otomatik olarak yapılandırmalar yapar.
- Gömülü Sunucular: Spring Boot, Tomcat, Jetty veya Undertow gibi sunucuları uygulamanın bir parçası olarak gömülü şekilde çalıştırabilir.
- Üretime Hazır Özellikler: Spring Boot, sağlık kontrolleri, uygulama konfigürasyonu gibi özelliklerle üretim ortamında kullanılabilirliği artırır.
- Spring CLI: Spring Boot projelerini komut satırı ara yüzü üzerinden hızlıca başlatmayı sağlayan bir araçtır. (https://www.baeldung.com/spring-boot)

3.2. Java SE 11

Java 11, Oracle tarafından Eylül 2018'de piyasaya sürülen ve uzun vadeli destek (LTS) sunan bir sürümdür. Java 11, hem dil seviyesinde hem de API'lerde çeşitli iyileştirmeler ve yeni özellikler getirmiştir. Bu sürüm, Java 8'den sonra en yaygın kullanılan ve kurumsal uygulamalar için tercih edilen bir sürüm haline gelmiştir.

Java 11'in getirdiği bazı önemli yenilikler şunlardır:

- **Yeni API'ler:** `HttpClient`, Java 11 ile standart hale gelmiştir ve HTTP/2 desteği sunar. `Files.readString()` ve `Files.writeString()` gibi yeni yardımcı metotlar eklenmiştir.
- **Geri Dönüşümler:** Java 11, gereksiz özelliklerin kaldırılması ile de dikkat çeker. Örneğin, `java.xml.ws`, `java.xml.bind` gibi modüller artık kaldırılmıştır.
- **Lambda ile Var Kullanımı:** Lambda ifadelerinde `var` anahtar kelimesinin kullanımı desteklenmiştir.
- **Yüksek Performans ve Verimlilik:** Java 11, daha düşük hafıza tüketimi ve daha hızlı başlatma süreleri ile performansı artırmıştır.

Java 11 aynı zamanda daha sıkı güvenlik özellikleri, çeşitli hata düzeltmeleri ve geliştirmeler ile de gelmiştir, bu da onu üretim ortamlarında tercih edilen bir sürüm yapmaktadır (Schildt, 2019).

3.3. PostgreSQL

PostgreSQL, açık kaynak kodlu, güçlü ve nesne-ilişkisel bir veritabanı yönetim sistemidir (ORDBMS). İlk olarak 1986 yılında University of California, Berkeley'de başlatılan POSTGRES projesinin bir devamı olarak geliştirilen PostgreSQL, veri bütünlüğü, genişletilebilirlik ve SQL uyumluluğu ile tanınır. Ayrıca, karmaşık veri tiplerini destekler ve gelişmiş sorgu optimizasyonu sunar (Douglas, 2005).

PostgreSQL'in öne çıkan bazı özellikleri şunlardır:

- **Genişletilebilirlik:** PostgreSQL, kullanıcı tanımlı veri türleri, fonksiyonlar, operatörler ve indeksler gibi birçok genişletme seçeneği sunar.
- **ACID Uyumluluğu:** PostgreSQL, güvenilir veri işlemleri sağlamak için ACID (Atomicity, Consistency, Isolation, Durability) özelliklerine tam uyumludur.
- **SQL Uyumluluğu:** PostgreSQL, ANSI SQL standartlarını büyük ölçüde destekler ve SQL:2011 standardına uyumludur.
- **JSON ve XML Desteği:** Yapılandırılmamış verileri saklamak ve işlemek için JSON ve XML gibi modern veri formatlarını destekler.
- **Replikasyon ve Yük Dengeleme:** PostgreSQL, veritabanı replikasyonu ve yük dengeleme için çeşitli mekanizmalar sunar.

PostgreSQL'in bu güçlü özellikleri, onu hem küçük uygulamalarda hem de büyük ölçekli kurumsal projelerde tercih edilen bir veritabanı yönetim sistemi yapar (Momjian, 2001).

3.4. Swagger Doc UI

Swagger, RESTful web hizmetlerini tanımlamak, tasarlamak, belgelemek ve tüketmek için yaygın olarak kullanılan bir araçtır. Swagger, özellikle API'lerin açık, anlaşılır ve otomatik olarak oluşturulmuş belgelerle sunulmasını sağlar. Swagger UI, bu belgeleri interaktif bir şekilde sunan bir kullanıcı ara yüzüdür, böylece geliştiriciler API'leri doğrudan web tarayıcısından test edebilir ve anlayabilir.

Swagger'ın bazı temel özellikleri şunlardır:

- **API Belgeleme:** Swagger, API uç noktalarını, parametrelerini, döndürülen veri yapılarını ve hata durumlarını açıkça tanımlayan bir belge oluşturur.
- **Otomatik Üretim:** Swagger, API tanımına dayalı olarak JSON veya YAML formatında belge oluşturur. Bu belgeler, Swagger UI ile interaktif bir şekilde sunulabilir.

- **Dilin Bağımsızlığı:** Swagger, dil bağımsızdır ve herhangi bir programlama dili veya platformla entegre edilebilir.

- **Swagger UI:** Bu araç, API'leri test etmeyi kolaylaştırır, çünkü geliştiriciler doğrudan belge üzerinden API istekleri yapabilir ve yanıtları anında görebilir.

Swagger ve Swagger UI, API geliştirme süreçlerinde geliştirici deneyimini iyileştirir ve API'nin doğru ve eksiksiz bir şekilde belgelenmesini sağlar (Ponelat, 2022).

3.5. Spring Data JPA

Spring Data JPA, Java Persistence API (JPA) için Spring Framework tarafından sağlanan bir modüldür. JPA, Java uygulamalarında veritabanı işlemlerini gerçekleştirmek için kullanılan bir standarttır. Spring Data JPA, JPA'nın gücünü basit ve güçlü bir şekilde kullanabilmenizi sağlar. Veritabanı erişim katmanını hızlıca geliştirmek için kullanılan bu modül, veri erişim süreçlerini büyük ölçüde kolaylaştırır.

Spring Data JPA'nın öne çıkan bazı özellikleri şunlardır:

- **Otomatik CRUD İşlemleri:** Spring Data JPA, standart CRUD (Create, Read, Update, Delete) işlemleri için hazır metodlar sunar, böylece veri erişim kodlarını manuel olarak yazmaya gerek kalmaz.

- **JPA Repositories:** Repository arayüzleri tanımlayarak, veritabanı erişim işlemlerini kolayca gerçekleştirmek mümkündür. Spring Data JPA, bu arayüzler üzerinden otomatik sorgu üretimi sağlar.

- **Sorgu Türetme:** Repository metot isimlerinden otomatik olarak sorgular türetebilirsiniz. Örneğin, `findByLastName(String lastName)` metodu, otomatik olarak bir sorgu oluşturur.

- **Paging ve Sorting Desteği:** Büyük veri setlerinde sayfalama ve sıralama işlemlerini kolayca yapmanızı sağlar.

- **Entity Mapping:** JPA'nın sunduğu entity mapping özellikleri ile veri modelleri ve veritabanı tabloları arasında kolayca eşleme yapabilirsiniz.

Spring Data JPA, özellikle veri tabanıyla ilgili rutin işlemleri hızlandırarak, geliştiricilerin daha karmaşık iş mantığına odaklanmalarını sağlar (Kainulainen, 2014)

3.6. Dockerfile

Docker, uygulamaları ve bağımlılıklarını bir araya getirip taşınabilir konteynerler içinde çalıştırmanıza olanak tanıyan açık kaynaklı bir platformdur. Docker, geliştiricilerin uygulamalarını farklı ortamlarda tutarlı bir şekilde çalıştırmalarını sağlar. Dockerfile ise, Docker konteynerlerinin nasıl oluşturulacağını tanımlayan bir betik dosyasıdır. Dockerfile, bir Docker imajını oluşturmak için talimatlar içerir; bu talimatlar, işletim sistemi katmanı, bağımlılıklar, ortam değişkenleri ve uygulamanın nasıl başlatılacağını belirler.

Docker ve Dockerfile'ın temel özellikleri şunlardır:

- **Kapsayıcılı (Containerized) Uygulamalar:** Docker, uygulamaları izole edilmiş kapsayıcılar içinde çalıştırarak taşınabilirlik ve güvenlik sağlar.
- **Dockerfile:** Docker imajlarını oluşturmak için kullanılan Dockerfile, basit komutlar ile imaj yapılandırmasını tanımlar. Bu komutlar arasında `FROM`, `RUN`, `COPY`, `ENTRYPOINT` gibi direktifler yer alır.
- **Versiyon Kontrolü ve Paylaşım:** Docker imajları, Docker Hub gibi platformlarda paylaşılabilir ve sürüm kontrolüne tabi tutulabilir.
- **Kapsayıcı Yönetimi:** Docker, birden fazla kapsayıcıyı yönetmek ve ölçeklendirmek için kolayca entegre edilebilecek araçlar sunar.

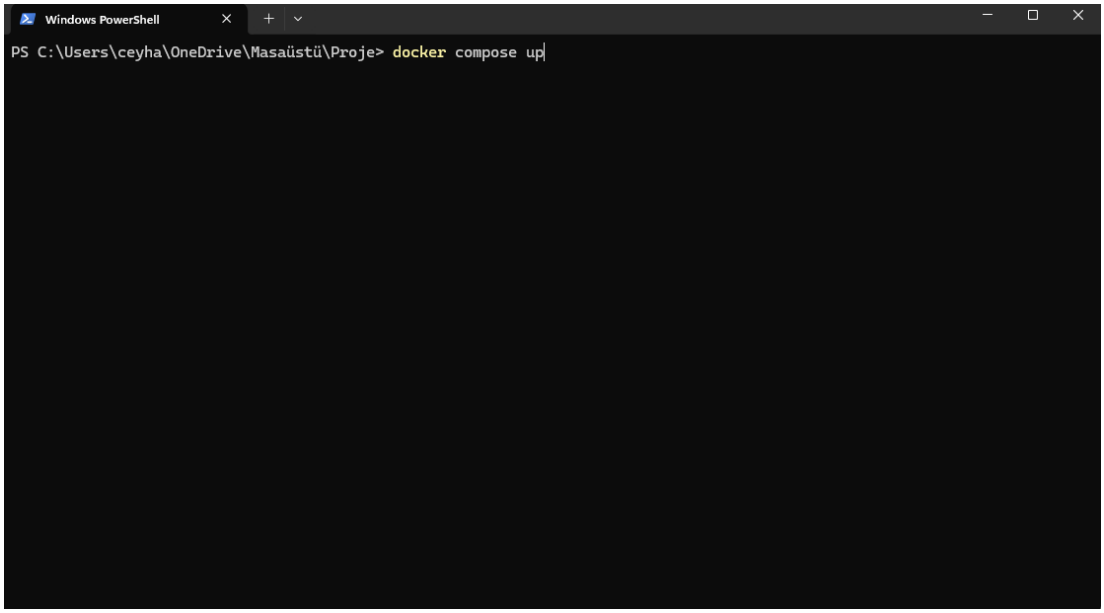
Dockerfile ile imaj oluşturmak, geliştiricilere ve operasyon ekiplerine tutarlı geliştirme, test ve dağıtım ortamları sunar. Bu da özellikle microservices mimarileri ve CI/CD (Continuous Integration/Continuous Deployment) süreçlerinde büyük avantajlar sağlar (Mouat, 2015).

Bölüm 4

Uygulama Çalıştırma Aşamaları

4.1. Projenin Çalıştırılması

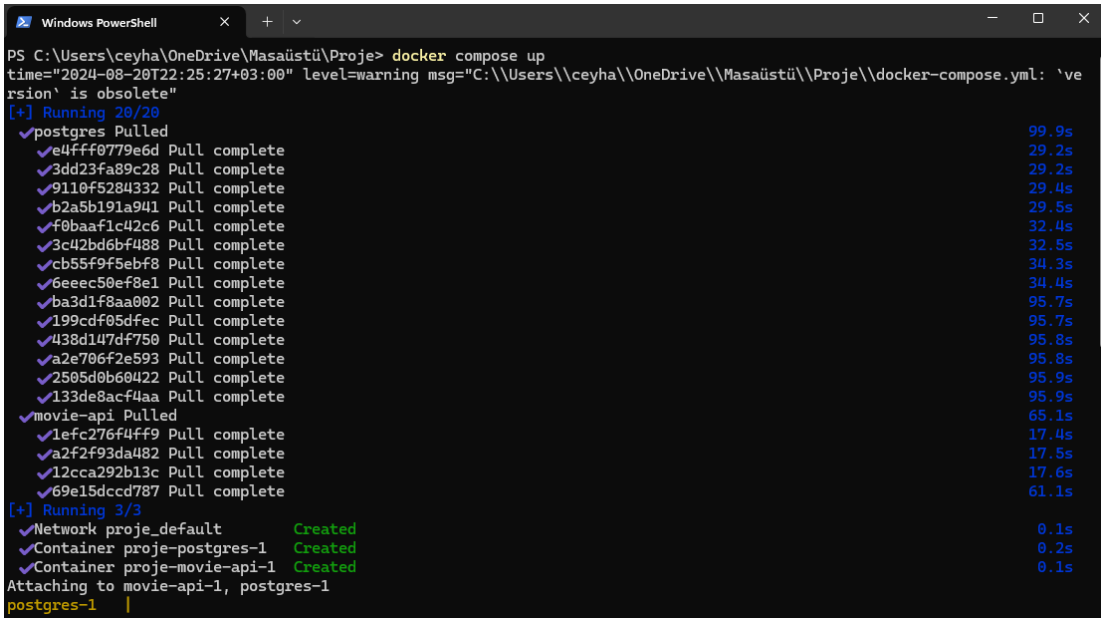
Projenin terminal üzerinde dizinine gidip docker compose up komutu ile docker imajlarını oluşturup projenin container olarak ayağa kalkmasını sağlıyoruz (Şekil 4.1).



Şekil 4.1: Docker compose up komutu

4.2. İmaj Oluşturulması

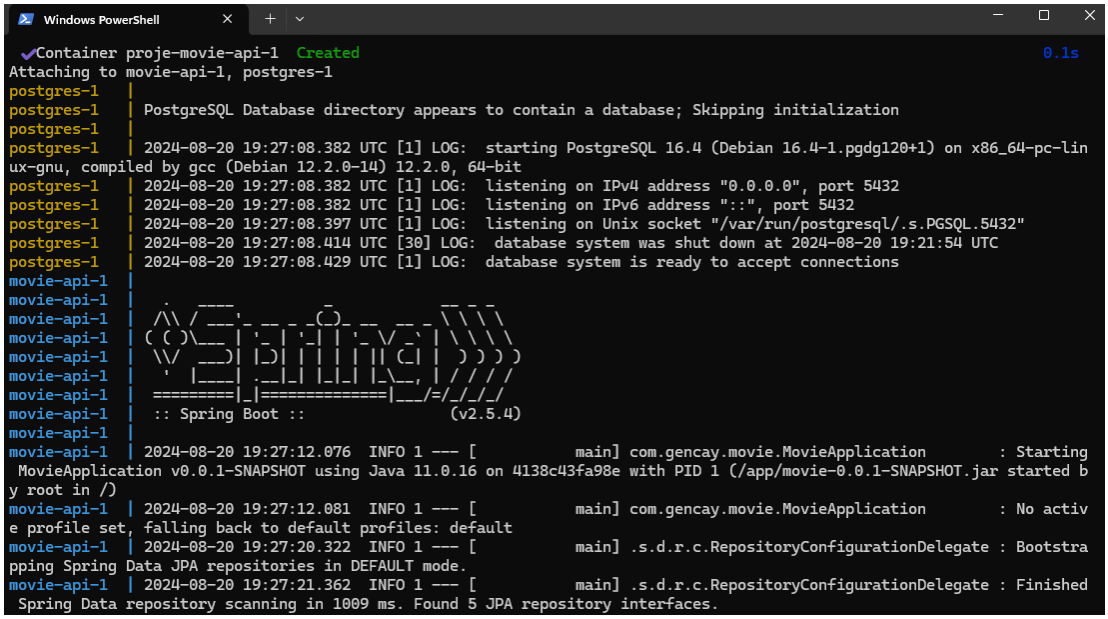
Bu adımda Şekil 4.2'de görüldüğü üzere PostgreSQL imajı ve projenin imajı oluşturuldu.



Şekil 4.2: İmaj oluşturulması

4.3. Database ve Container Çalıştırılması

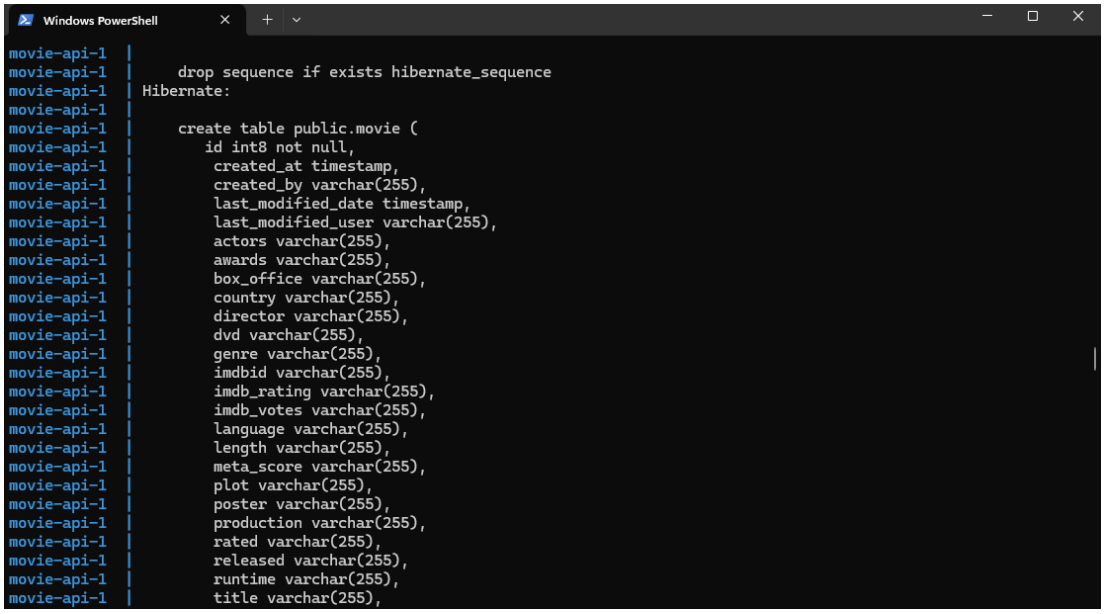
Bu adımda Şekil 4.3'de görüldüğü üzere PostgreSQL database'in ve ardından movie app container'ın çalıştığı görülmektedir.



Şekil 4.3: Database ve movie app container çalıştırılması

4.4. Spring Boot Çalıştırılması ve Database Objeleri Oluşturulması

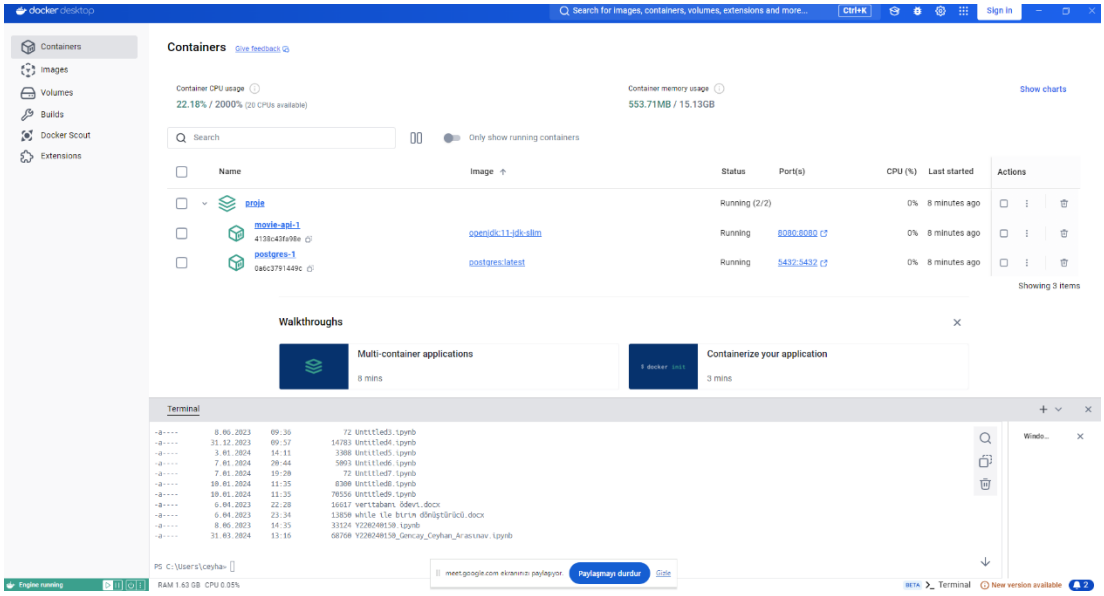
Bu adımda Spring Boot projesi ayağa kalkarken hibernate ORM (object relation mapping) tool'u database objelerini oluşturdu (Şekil 4.4)



Şekil 4.4: Hibernate ORM tool çalışması

4.5. Projenin Docker Desktop Üzerinde Çalıştırılması

Şekil 4.5'de görüldüğü üzere Docker Desktop uygulamasında projenin çalıştığı görülmektedir.



Şekil 4.5: Docker Desktop üzerinde programın çalışması

Kaynaklar

Baeldung. (2024) *Introduction to Spring Boot*. https://www.baeldung.com/spring-boot.

Kainulainen, P. (2014). *Spring Data: Modern Data Access for Enterprise Java*. Packt Publishing.

Momjian, B. (2001). *PostgreSQL: Introduction and Concepts*. Addison-Wesley Professional.

Mouat, A. (2015). *Using Docker: Developing and Deploying Software with Containers.* O'Reilly Media.

Ponelat, J. & Rosenstock, L. (2022). *Designing APIs with Swagger and OpenAPI*. O'Reilly Media.

Schildt, H. (2019). *Java: The Complete Reference, Eleventh Edition*. McGraw-Hill Education.