SVEUČILIŠTE U ZAGREBU

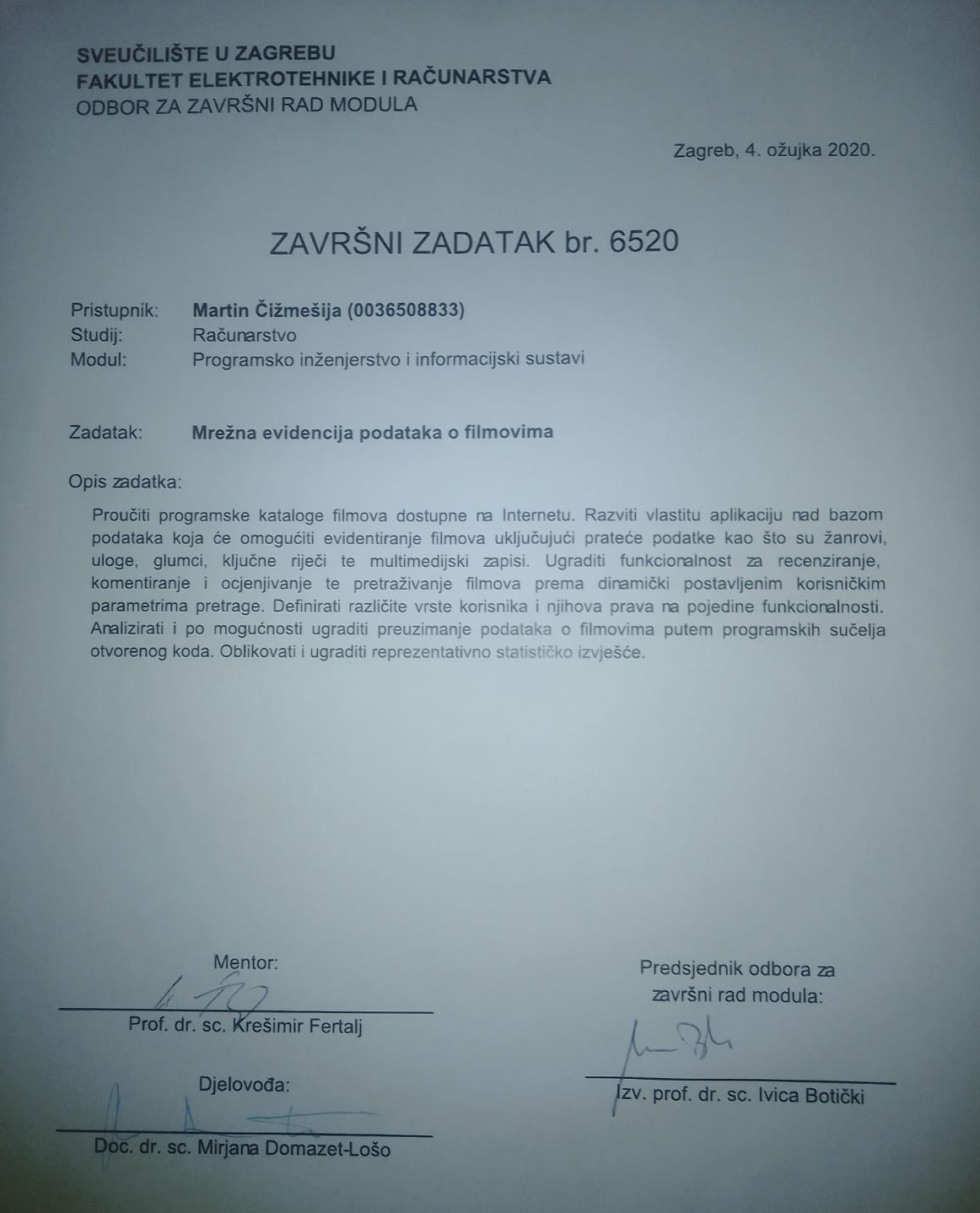
**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

ZAVRŠNI RAD br. 6520

**Mrežna evidencija podataka o filmovima**

Martin Čižmešija

Zagreb, svibanj 2020.



Sadržaj

1. **Analiza postojeće programske podrške6**
   1. Internet Movie Database6
   2. Rotten Tomatoes8
2. **Zahtjevi programske aplikacije9**
   1. Funkcionalni zahtjevi9
   2. Nefunkcionalni zahtjevi11
3. **Arhitektura rješenja11**
   1. Baza podataka11
      1. Model baze podataka12
      2. Problemi realizacije velike baze podataka18
   2. Komponente sustava22
      1. Modeli22
      2. Pogledi25
      3. Upravljači27
4. **Funkcionalnosti aplikacije31**
   1. Lista filmova32
   2. Dodavanje novog filma33
   3. Detalji filma34
   4. Uređivanje podataka o filmu35
   5. Dodavanje nepostojećih podataka u bazu36
5. **Korištene tehnologije i alati36**

## Uvod

U današnje vrijeme film je jedan od najraširenijih oblika zabave. Od komercijalnih filmova predviđenih za velika platna do filmova nezavisne produkcije te od komedija do ozbiljnih drama, u filmovima svatko može pronaći nešto što bi njemu odgovaralo i u čemu će uživati. Tako nije ni čudo da je filmska industrija danas jedna od najutjecajnijih u svijetu. Dodatno, iz godine u godinu svjedočimo novim filmskim tehnikama i postignućima potpomognutih što tehnologijom, a što neiscrpnoj inspiraciji pisaca i redatelja za stvaranje dobrog filma. S velikim brojem filmova javila se i potreba za bazama podataka u kojima će korisnici moći naći detalje o svakom filmu koji ih zanima te možda i po putu otkriti neki skriveni biser koji im je promaknuo.

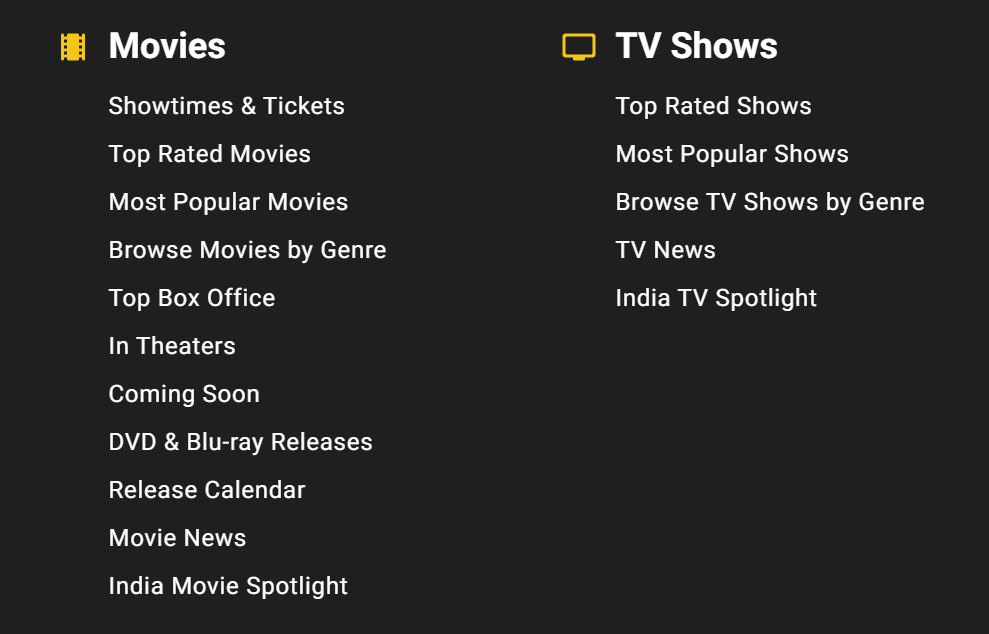
U ovom završnom radu obrađena je osnovna problematika stvaranja jedne takve baze podataka te je napravljena web aplikacija s osnovnom funkcionalnostima stranice s filmskim podatcima koja uključuje podatke o filmovima, ali posljedično i podatke o osobama iz filmske industrije. Takva aplikacija će sadrži i mogućnost ocjenjivanja i recenziranja filma od strane korisnika, kako bi se dobila prosječna ocjena te time i kvaliteta pojedinog filma.

## Analiza postojeće programske podrške

Danas postoje mnoge baze podataka filmova na mreži (eng. *online*), što javno dostupnih i besplatnih baza podataka poput IMDb-a (Internet Movie Database), a što u sklopu pružatelja internetskih *streaming* servisa poput Netflixa. Streaming servisi u svojim bazama podataka imaju isključivo filmove dostupne za gledanje na njihovoj platformi, dok otvorene baze podataka koje nisu ujedno i streaming servisi imaju podatke o svim filmovima, neovisno o platformi preko koje se mogu gledati. Funkcionalnosti obiju vrsta baza podataka su vrlo slične i većim dijelom se preklapaju, tako da ću analizirati dvije otvorene, besplatne filmske baze podataka.

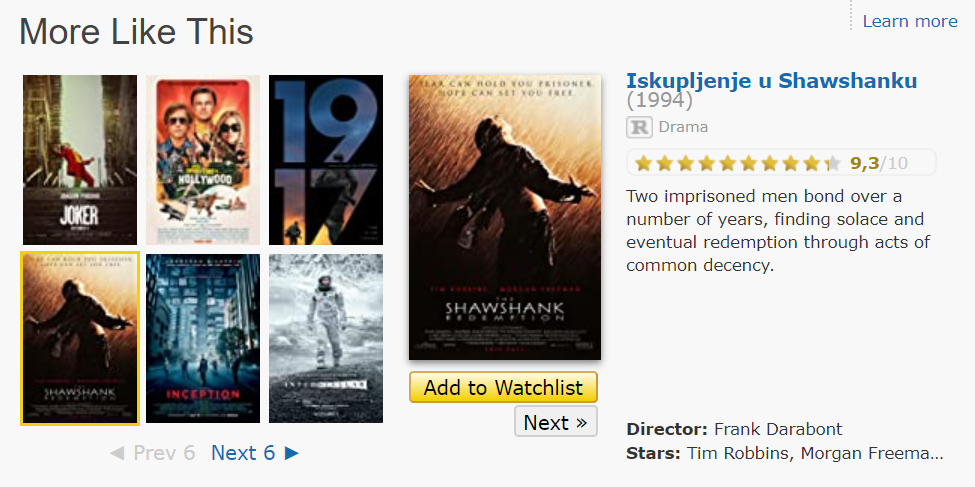
## Internet Movie Database

IMDb je online baza podataka koja sadrži informacije vezane uz filmove, serije, glumce te filmsku industriju općenito. U siječnju 2020. godine baza je sadržavala oko 6.5 milijuna naslova (uključujući pojedine epizode serija), 10.4 milijuna različitih osoba (što likova iz serija i filmova što stvarnih ljudi iz filmske industrije), te preko 83 milijuna registriranih korisnika [[1]](#IMDB). Stranica uključuje funkcionalnosti pretraživanja osoba, filmova i serija kao i ocjenjivanje pojedinog filma (ili serije) te rang liste najbolje ocijenjenog sadržaja. Dodatno, na stranici se može pretraživati po kategorijama, kao što je prikazano na slici (Slika1).



SlikaPogreška! U dokumentu nema teksta navedenog stila..1.1: Opcije pretraživanja filmova i serija

Većina funkcionalnosti na stranici je dostupna čak i neregistriranim korisnicima, pa tako svatko može preko stranice pretraživati sadržaj i vidjeti detalje o filmovima poput prosječne ocjene filma (od 1 do 10), opis, žanr, redatelja, glumce, nominacije i nagrade koje je film osvojio te ostalo. Također, kao što je prikazano na slici ([Slika 1.2](#jedan_dva)) za svaki film postoji odjeljak sa filmovima koji su na neki način slični pretraženom filmu. Uz svaki „sličan“ film stoji i njegova ocjena, kratak opis te redatelj i imena glumaca u glavnim ulogama.

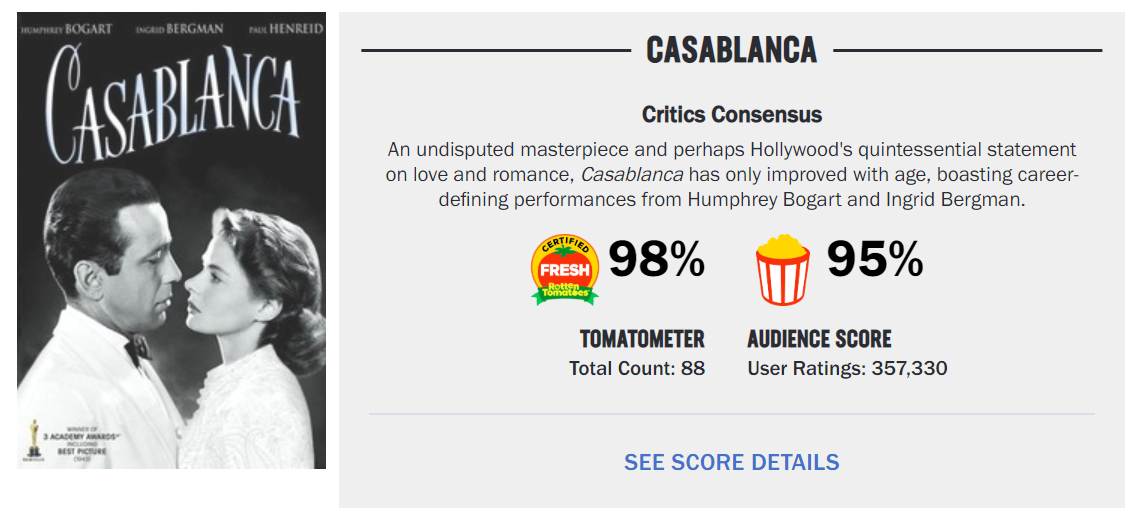


Slika 1.2: Odjeljak sa predloženim sličnim filmovima

Registrirani korisnici imaju dodatne opcije ostavljanja komentara za filmove i serije te njihovo ocjenjivanje. Završna ocjena pojedinog sadržaja koju vide i registrirani i neregistrirani korisnici se računa kao aritmetička sredina ocjena svih korisnika koji su dali ocjenu tom filmu. Dodatno, svaki registrirani korisnik može staviti pojedini film na listu gledanja (*watchlist*) te ga na taj način označiti kao film kojega će kasnije pogledati. Na kraju, svakom registriranom korisniku su prikazani i personalizirani filmovi koje nije gledao (ili ocijenio), a temeljem ocjena već ocijenjenih filmova bi mu se mogli svidjeti.

## Rotten Tomatoes

Druga vrlo poznata online baza podataka filmova Rotten Tomatoes kao i IMDb ima funkcionalnosti pregledavanja filmova i serija i ocjenjivanja te ostavljanja komentara od strane registriranih korisnika, ali uz određene razlike. Kao što se može vidjeti na slici ([Slika 1.3](#jedan_tri)) pri opisu filma Rotten Tomatoes ima dvije kategorije ocjenjivanja. Prva kategorija nazvana „Tomatometer“ se odnosi na prosječnu ocjenu akreditiranih filmskih kritičara, dok se druga ocjena „Audience score“ odnosi na prosječnu ocjenu publike.



Slika 1.3: Ocjene kritičara i publike u dvije zasebne kategorije

Dok i filmski kritičari i obični registrirani korisnici mogu ocjenjivati i ostavljati komentare o filmovima na stranici, njihove ocjene i komentari se stavljaju u odvojene kategorije [[2]](#Rotten_tomatoes). Tako osim i odvojenih ocjena filmova postoje i odvojeni komentari filmskih kritičara i običnih korisnika. Neregistrirani korisnici osim filmova i serija mogu pretraživati i registrirane korisnike i filmske kritičare, te na stranicama njihovih profila mogu naći popis filmova sa komentarima i ocjenama te osobe za pojedini film. IMDb ima sličnu opciju pretraživanja korisnika, ali odvajanjem filmskih kritičara od svakodnevnih korisnika Rotten Tomatoes je stvorio skupinu korisnika koji su potvrđeni kao pouzdani, tako da i je i sama opcija pretraživanja i pregledavanja tuđih profila dobila na važnosti.

* 1. **Sažetak analize s obzirom na domenu vlastitog rada**

U ovome radu će se implementirati osnovne funkcionalnosti koje sadrže oba analizirana primjera: Prikaz popisa filmova kao i prikaz detalja pojedinog filma uključujući žanrove, uloge, glumce, ključne riječi te multimedijske zapise u obliku filmu pripadajućih slika. Dodatno, implementirati će se i osnovne funkcionalnosti komentiranja i ocjenjivanja filmova, kao i njihovo pretraživanje po zadanim parametrima poput žanra ili ključnih riječi. Metoda ocjenjivanja filmova u ovom radu implementirati će se po uzoru na Rotten Tomatoes - svaki film će odvojeno imati navedenu prosječnu ocjenu kritičara i publike. Aplikacija je zamišljena i sa funkcionalnošću za predlaganje sličnih filmova po uzoru na onu iz IMDb-a.

1. **Zahtjevi na vlastitu aplikaciju**

U aplikaciji postoje različite kategorije korisnika od kojih svaki ima različite razine ovlasti. Tako razlikujemo neregistrirane i registrirane korisnike, administratore te filmske kritičare kao posebnu potkategoriju registriranih korisnika. Funkcionalni zahtjevi svake kategorije korisnika su navedeni u nastavku.

## Funkcionalni zahtjevi

Neregistrirani korisnici su svi posjetitelji stranice koji nemaju ili se nisu povezali sa svojim korisničkim računom. Njihovi funkcionalni zahtjevi su:

* Registracija novog korisničkog računa
* Podnošenje zahtjeva za registracijom kao filmski kritičar
* Prijava na postojeći korisnički račun
* Pregled filmova, uloga, glumaca i multimedijskih zapisa pojedinog filma
* Pretraživanje filmova po kategorijama i ključnim riječima
* Pretraživanje korisnika i filmskih kritičara kao i pregled njihovih profila i osobnih statistika.

Registrirani korisnici su korisnici koji su prijavljeni na stranicu sa svojim osobnim korisničkim računima. Njihovi funkcionalni zahtjevi su:

* Pregled filmova, uloga, glumaca i multimedijskih zapisa pojedinog filma
* Pretraživanje filmova po kategorijama i ključnim riječima
* Pretraživanje korisnika i filmskih kritičara kao i pregled njihovih profila i osobnih statistika
* Uređivanje osobnog profila i pregledavanja vlastite statistike
* Odjava iz vlastitog korisničkog računa
* Ocjenjivanje filmova
* Pregled liste filmova koje je korisnik ocijenio
* Ostavljanje i ocjenjivanje komentara na ocjenjenim filmovima
* Spremanje filmova u listu gledanja, micanje filmova iz nje i pregled te liste
* Označavanje pojedinih filmova favoritima.

Filmski kritičari su posebna vrsta registriranih korisnika koji su ovlašteni od strane administratora i čije se ocjene filmova te komentari stavljaju u posebnu kategoriju. Nasljeđuju sve funkcionalne zahtjeve registriranih korisnika.

Administratori se također tretiraju kao registrirani korisnici, ali također sa specifičnim funkcionalnim zahtjevima:

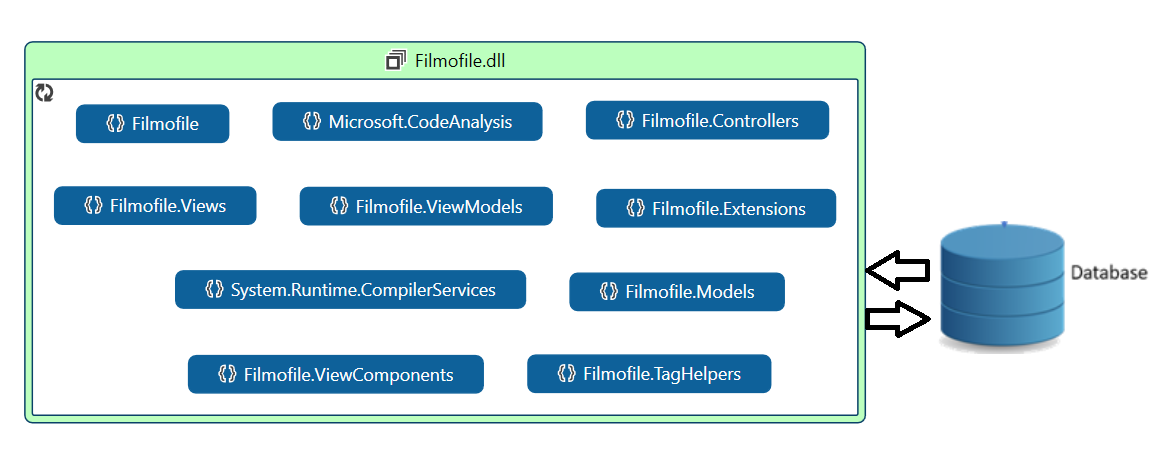
* Mogu dodavati filmove u bazu podataka
* Potvrđuju filmske kritičare
* Mogu brisati komentare drugih korisnika.

## Nefunkcionalni zahtjevi

* Sustav mora reagirati na zahtjeve korisnika te se odgovor ne bi trebao čekati dulje od 5 sekundi
* Grafičko sučelje mora biti intuitivno i lako za snalaženje kako bi prosječni korisnik mogao koristiti aplikaciju bez prethodnog upoznavanja sa njome
* Aplikacija mora raditi bez poteškoća pri istovremenom radu više korisnika
* Pogreške i neočekivano ponašanje korisnika ne smiju narušiti rad aplikacije ili integritet baze podataka
* Korisnicima ne smije biti dopušteno izvršavati radnje za koje nemaju ovlasti.

1. **Arhitektura rješenja**

Na slici ([Slika 3.1](#tri_nula)) je prikazana cjelokupna arhitektura rješenja. Baza podataka (egl. *Database*) je mjesto čuvanja podataka. U njoj se nalaze te se zapisuju i dohvaćaju svi podatci vezani za aplikaciju. Unutar rješenja aplikacije nalaze se imenski prostori (eng. *Namespace*) unutar kojih su grupirani programski razredi koji čine aplikaciju. Imenski prostor je grupiran tako da se u svakom nalaze razredi zaduženi za implementaciju određenih funkcionalnosti. Tako je na primjer imenski prostor *FIlmofile.Models* zadužen za komunikaciju s bazom podataka, dok se u imenskom prostoru *Filmofile.Views* nalazi implementacija prednje strane aplikacije. U ovome poglavlju ću analizirati arhitekturni obrazac korišten za ostvarivanje aplikacije te zatim i objasniti najbitnije komponente.



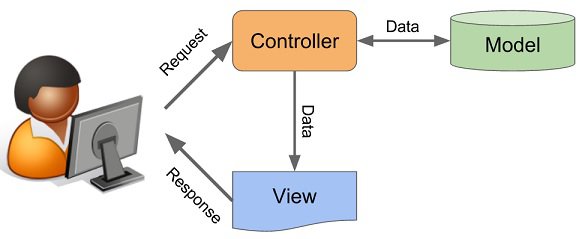
Slika 3.1: Arhitektura sustava

Baza podataka za aplikaciju ostvarena je u relacijskoj bazi *SQL Server*, dok je sama aplikacija ostvarena korištenjem arhitekturnog obrasca Model-Pogled-Upravljač (eng. *Model-View-Controller*, skraćeno MVC) [[3]](#MVC). Takav obrazac je integriran u radni okvir *ASP.NET Core* koji uključuje već pripremljene predloške za izgradnju aplikacije. Baza podataka u SQL Serveru odabrana je zbog kompatibilnosti s *.NET Core* razvojnom okolinom, dok je MVC arhitektura odabrana zbog mogućnosti odvojenog razvoja dijelova aplikacije, što je i najveća prednost takvog modela.

Arhitekturni obrazac MVC se sastoji od tri glavne komponente:

* Model (eng. *Model*) – Ovaj dio oblikovnog obrasca rukuje podatcima i logikom aplikacije. Aplikacija je povezana s bazom podataka preko modela, tako da se preko njega događaju sva dohvaćanja, ažuriranja i umetanja informacija u bazu podataka. Relacijske tablice baze podataka su preslikane u modelu koristeći najobičnije C# razrede, što je ostvareno preko radnog okvira *Entity Framework Core*.
* Pogled (eng. *View*) – U njemu je sadržana cijela prednja strana (eng. *frontend*) aplikacije. Sadrži sve informacije koje korisnik može vidjeti u svojoj interakciji sa stranicom te sve podatke koji na njoj trebaju biti prikazani. Programira se u CSS, JavaScript i HTML programskim jezicima.
* Upravljač (eng. *Controller*) – Sloj aplikacije koji obrađuje korisničke zahtjeve. Poveznica je između pogleda i modela te kontrolira interakcije među njima. Pomoću njega se s jedne strane obrađuju podatci koji će kasnije doći do baze podataka, te se s druge strane u njemu podatci pripremaju za prikaz. Također, u njemu se pozivaju pogledi koji odgovaraju korisnikovim akcijama.

Na slici ([Slika 3.2](#tri_jedan)) je prikazano međudjelovanje komponenti MVC obrasca. Korisnik aplikacije vidi pogled kao prednju stranu aplikacije. U interakciji s aplikacijom aktivira se upravljač koji na korisnikov zahtjev dohvaća željene podatke preko modela, te ih zatim prosljeđuje pogledu, a samim time i korisniku.



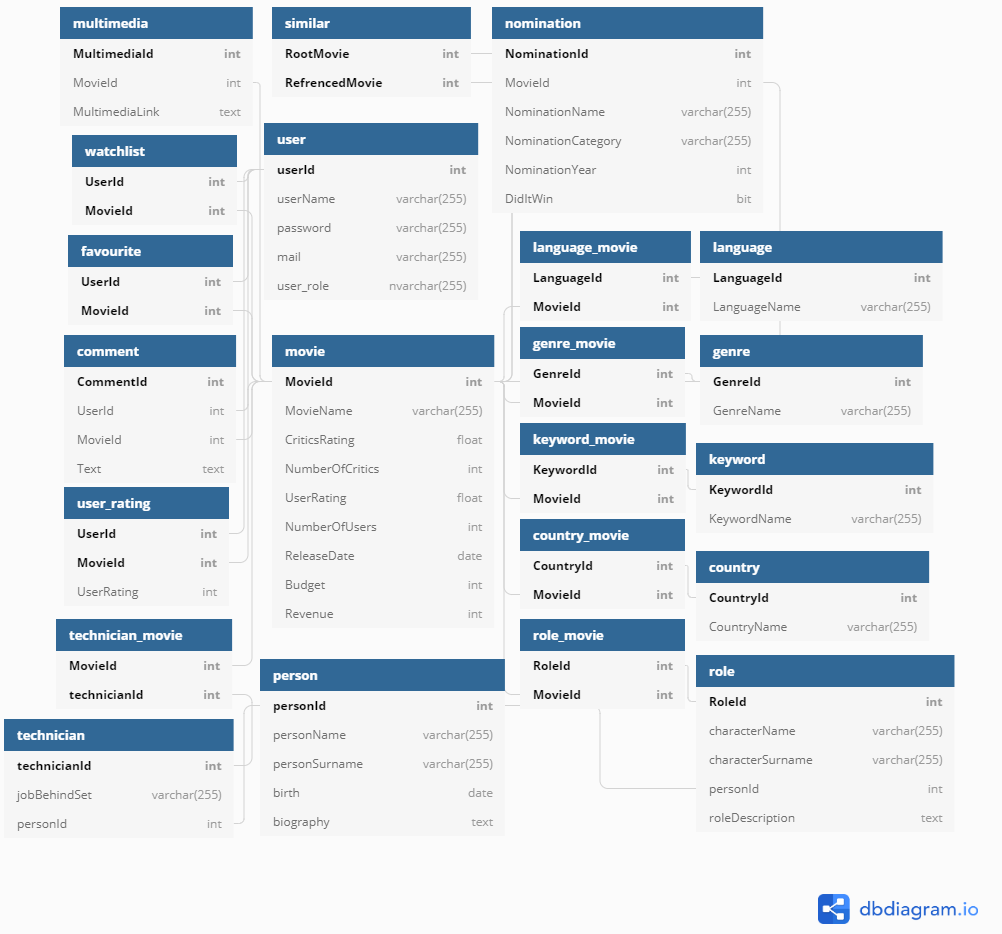
Slika 3.2: Međudjelovanje komponenti u arhitekturnom obrascu MVC [[3]](#MVC)

* 1. **Baza podataka**

Baza podataka je mjesto gdje se spremaju i iz kojega se dohvaćaju svi podatci koje aplikacija sadrži. Za svaku bazu podataka je bitno da je dobro modelirana te napravljena po pravilima normalizacije, a u slučajevima velikih baza podataka sa mnogo tablica i podataka takva pravila samo dobivaju na važnosti. Mali propust u modeliranju takve baze može dovesti do poteškoća pri manipuliranju podatcima, a u najboljem scenariju će dovesti do nepreglednosti baze te zauzimanja nepotrebnog memorijskog prostora. U ovom poglavlju ću prvo analizirati svoju bazu podataka filmova, a zatim ću se dotaknuti glavnih problema i poteškoća pri modeliranju baza velikog obujma.

* + 1. **Model Baze podataka**

Kako predviđena baza podataka ima 22 tablice ([Slika 3.3](#tri_jedan_jedan)) od kojih su neke i spojne tablice Više-Više (eng. *Many-to-Many*) veza, detaljno ću obraditi samo najbitnije tablice te njihove glavne atribute, dok ću se na ostale samo ukratko osvrnuti objašnjavajući njihovu funkciju.



Slika 3.3: Potpuni model baze podataka

**Movie:** Entitet predstavlja jedan film u aplikaciji. Sadrži atribut MovieId koji služi kao primarni ključ te atribut MovieName u kojem je zapisan naslov filma. Od ostalih atributa, UserRating i CriticsRating su prosječne ocjene dane filmu od strane kritičara i korisnika, dok ReleaseDate, Budget i Revenue sadrže podatke o datumu izlaska filma, budžetu uloženom u njegov nastanak te zaradu koju je film ostvario. Atributi NumberOfCritics i NumberOfUsers sadrže broj kritičara i korisnika koji su ocijenili film, te služe za izračun prosječne ocjene filma.

Tablica 1: Tablica s osnovnim podatcima o filmovima

|  |  |  |  |
| --- | --- | --- | --- |
| Movie | | | |
| MovieId | INT | jedinstveni identifikator | Primarni ključ, inkrement |
| MovieName | VCHAR | Naslov filma | not null |
| UserRating | FLOAT | Prosječna ocjena korisnika |  |
| CriticsRating | FLOAT | Prosječna ocjena kritičara |  |
| NumberOfCritics | INT | Broj kritičara koji su ocijenili film | Not null |
| NumberOfUsers | INT | Broj korisnika koji su ocijenili film | Not null |

**User:** Entitet predstavlja registriranog korisnika aplikacije. Primarni ključ mu je sadržan u atributu UserId. Svaki korisnik ima svoje jedinstveno korisničko ime i jedinstvenu mail adresu te lozinku s kojom se prijavljuje u svoj račun. Također, svaki korisnik ima i svoju ulogu: korisnik, kritičar ili administrator.

Tablica 2: tablica s podatcima o prijavljenom korisniku

|  |  |  |  |
| --- | --- | --- | --- |
| User | | | |
| UserId | INT | jedinstveni identifikator | primarni ključ, inkrement |
| UserName | VCHAR | jedinstveni identifikator | not null |
| Password | VCHAR | lozinka za prijavu | not null |
| Mail | VCHAR | jedinstveni identifikator | not null |
| UserRole | USER\_ROLES | uloga korisnika (enumeracija) | not null |

**Comment:** Entitet predstavlja komentar na stranici. Sadrži atribut CommentId koji mu služi kao primarni ključ. Vezan je uz entitet Movie s vezom Više-na-Jedan (eng. *Many-to-One*) pomoću atributa MovieId i User vezom Više-Jedan pomoću atributa UserId. Također, postoji atribut Text koji predstavlja sam komentar teksta.

Tablica 3: tablica s atributima za komentar

|  |  |  |  |
| --- | --- | --- | --- |
| Comment | | | |
| CommentId | INT | jedinstveni identifikator | primarni ključ, inkrement |
| UserId | INT | identifikator korisnika | strani ključ (User.UserId) |
| MovieId | INT | identifikator filma | strani ključ (Movie.MovieId) |
| Text | TEXT | Tekst komentara | not null |

**Nomination:** Entitet predstavlja nominaciju filma za određenu nagradu. Sadrži atribut NominationId koji mu služi kao primarni ključ. Vezan je uz entitet Movie s vezom Više-na-Jedan pomoću atributa MovieId. Dodatno sadrži atribute NominationName, NominationCategory i NomitaionYear koji opisuju za koju nagradu je film nominiran (Oscar, Zlatni Globus…), za koju kategoriju je nominiran (najbolji film, najbolji scenarij, najbolja glavna uloga…) i koje godine. Također, postoji atribut DidItWin tipa Boolean koji govori o tome je li film i osvojio nagradu za koju je nominiran. Na taj način se ujedno znaju i filmovi koju su pobjednici određenih nagrada.

Tablica 4: Najbitniji atributi tablice nominacija

|  |  |  |  |
| --- | --- | --- | --- |
| Nomination | | | |
| NominationId | INT | jedinstveni identifikator | primarni ključ, inkrement |
| MovieId | INT | identifikator filma | Strani ključ (Movie.MovieId) |
| DidItWin | BOOLEAN | true ako je film osvojio nagradu | Not null |

**Multimedia:** Entitet predstavlja multimedijski zapis (sliku ili video) određenog filma. Sadrži atribut MultimediaId koji mu služi kao primarni ključ. Vezan je uz entitet Movie s vezom Više-na-Jedan pomoću atributa MovieId. Dodatno sadrži atribut MultimediaLink koji je poveznica na lokalno ili mrežno spremljen multimedijski zapis.

Tablica 5: atributi tablice multimedije

|  |  |  |  |
| --- | --- | --- | --- |
| Multimedia | | | |
| MultimediaId | INT | jedinstveni identifikator | primarni ključ, inkrement |
| MovieId | INT | identifikator filma | strani ključ (Movie.MovieId) |
| MultimediaLink | TEXT | Link na multimedijski zapis | Not null |

**User\_rating:** Entitet predstavlja ocjenu koju je korisnik dao filmu. Jedinstveno je određen stranim ključevima UserId i Movieid. Pomoću njih je vezan uz entitete User i Movie vezama Više-na-Jedan. U atributu UserRating je sadržana ocjena koju je korisnik UserId pridijelio filmu s primarnim ključem MovieId. Ocjena je cijeli broj u rasponu od 1 do 5.

Tablica 6: Tablica za spajanje ocjene korisnika s filmom

|  |  |  |  |
| --- | --- | --- | --- |
| User\_rating | | | |
| UserId | INT | Identifikator korisnika | strani ključ (User.UserId) |
| MovieId | INT | Identifikator filma | strani ključ (Movie.MovieId) |
| UserRating | INT | Ocjena filma od 1 do 5 | not null |

**Role:** Entitet predstavlja glumačku ulogu osobe u određenom filmu. Sadrži atribut RoleId koji mu služi kao primarni ključ. Sadrži i atribut PersonId koji je povezan sa entitetom Person vezom Više-na-Jedan te označuje osobu koja je igrala tu ulogu.

Tablica 7: atributi glumačke uloge

|  |  |  |  |
| --- | --- | --- | --- |
| Role | | | |
| RoleId | INT | jedinstveni identifikator | Primarni ključ, inkrement |
| CharacterName | VCHAR | Ime lika | Not null |
| CharacterSurname | VCHAR | Prezime lika |  |
| RoleDescription | TEXT | Opis uloge i lika |  |
| PersonId | INT | Identifikator glumca | strani ključ (Person.PersonId) |

**Technician:** Entitet predstavlja člana filmske ekipe koja je sudjelovala u stvaranju filma, a nije bila u ulozi glumca. Primarni ključ je sadržan u atributu TechnicianId. Sadrži i atribut PersonId koji je povezan sa entitetom Person vezom jedan-na-Više (eng. One-to-Many). JobBehindSet je posao koji je osoba obavljala pri stvaranju filma (redatelj, režiser, kamerman…).

Tablica 8: atributi tehničkog posla osobe pri stvaranju filma

|  |  |  |  |
| --- | --- | --- | --- |
| Technician | | | |
| TechnicianId | INT | jedinstveni identifikator | Primarni ključ, inkrement |
| JobBehindSet | VCHAR | Uloga pri stvaranju filma |  |
| PersonId | INT | Identifikator glumca | strani ključ (Person.PersonId) |

**Person:** Entitet predstavlja osobu u filmskoj industriji. Sadrži atribut PersonId koji mu je primarni ključ. Također postoje i atributi koji sadrže ime, prezime, datum rođena i biografiju osobe.

Tablica 9: atributi osobe u filmskoj industriji

|  |  |  |  |
| --- | --- | --- | --- |
| Person | | | |
| PersonId | INT | jedinstveni identifikator | Primarni ključ, inkrement |
| PersonName | VCHAR | Ime | Not null |
| PersonSurname | VCHAR | Prezime | Not null |
| Birth | DATE | Datum rođenja | Not null |
| Biography | TEXT | Biografija |  |

**Genre, Keyword, Language i Country** tablice su šifranici u kojima su zapisani žanrovi, ključne riječi, jezici i države. Kako su ti entiteti povezani sa filmom vezom Više-Više za njihovo spajanje sa tablicom Movie koristimo spojne tablice **Genre\_movie, Keyword\_movie, Language\_movie i Country\_movie.** Zbog istog razloga se na isti način sa tablicom Movie spajaju i tablice role i Techinician (njihove spojne tablice su **Role\_movie** i **Techician\_movie**).

**Watchlist** je pomoćna tablica koja predstavlja listu filmova koje bi korisnik htio pogledati. Ima dva primarna ključa: UserId koji je povezan s tablicom User vezom Jedan-na-Jedan (eng. One-to-One) te MovieId koji je povezan s tablicom Movie vezom Jedan-na-Više.

**Favourites** je pomoćna tablica koja predstavlja listu filmova koje je korisnik označio kao favorite. Ima dva primarna ključa: userId koji je povezan s tablicom User vezom Jedan-na-Više te movieId koji je povezan s tablicom Movie vezom Jedan-na-Jedan.

**Similar** je pomoćna tablica preko koje se ostvaruje rekurzivna veza tablice Movie. Primarni ključ RootMovie je povezan sa entitetom movieId u tablici Movie vezom Više-na-Jedan te predstavlja početni film iz kojeg se referencira na njemu slične, dok je primarni ključ RefrencedMovie povezan s entitetom MovieId (također u tablici Movie) vezom jedan-na-Jedan te predstavlja referencirani film.

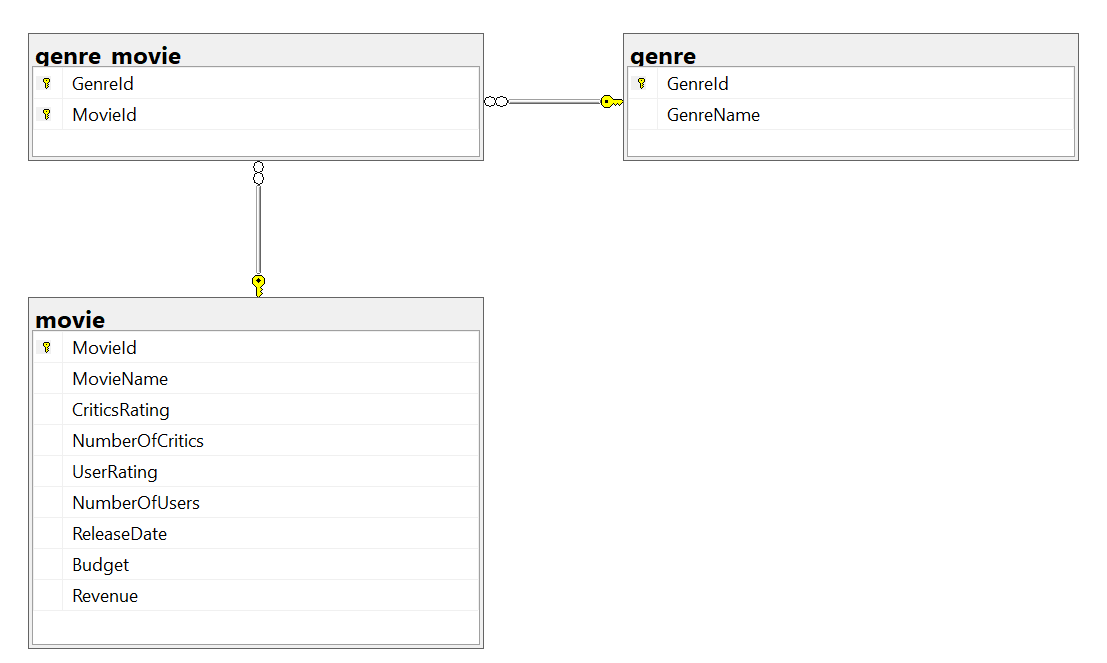
* + 1. **Problemi realizacije velike baze podataka**

Već iz primjera baze podataka filmova uočavamo sa kojim se problema možemo susreti prilikom njene realizacije. Prvi problem je nemogućnost postojanja direktnih Više-Više veza u bazi podataka. Razlog tomu su pravila normalizacije koja nalažu da se u pojedinoj tablici u bazi podataka može nalaziti samo jedna vrijednost određenog atributa. Uzmimo za primjer tablice sa žanrovima i filmovima. Jednom filmu može pripadati više žanrova (akcija, komedija, drama), a istovremeno jednom žanru pripadaju cijeli skupovi filmova. Problem koji nastaje je da bi u direktnom spoju te dvije tablice u jednoj od njih trebali unijeti više vrijednosti istoga atributa. Tako bi na primjer u tablici filmova imali atribut žanr koji bi imao vrijednosti *komedija* i *akcija* te bi svaka od tih vrijednosti bila povezana sa tablicom žanrova. Druga (još i gora i manje efikasna) metoda bi bila da u tablici žanrova postoji atribut za ID filma koji bi sadržavao sve filmove tog žanra i na taj način bio direktno povezan s tablicom filmova. Ovakve metode povezivanja baze podataka su u neskladu već i s prvom normalnom formom (1NF) baze podataka, koja nalaže da se u jednom polju atributa može nalaziti jedna vrijednost. Razlog iza postojanja te norme možemo naći i u potencijalnim problemima koji proizlaze iz gore navedenog primjera.

Prvi i najočitiji problem je nedostatak preglednosti ovakvog pristupa. Ukoliko u jednom žanru budu nazivi (tj. Identifikacijske oznake) svih filmova toga žanra, puno je teže pronaći specifičan film, pogotovo pomoću SQL upita. Naime, operacije nad tablicama koje imaju više vrijednosti atributa odvojenih zarezom (ili nekim drugim znakom) su mnogo kompleksnije. Sve vrijednosti atributa moraju se prvo odvojiti po traženom znaku, a zatim i konvertirati u prikladan format. Tek se onda može izvršiti pretraživanje po vrijednostima i dohvaćanje željenog rezultata. Osim trošenja vremena na pisanje kompliciranijih SQL upita ovaj pristup zbog istog razloga troši i više memorije, te je samim time i vrijeme izvršavanja naredbi dulje. Slična stvar se događa i kod ažuriranja jedne od vrijednosti atributa tablice. Povezana sa problemom preglednosti kod ovakvog pristupa je i povećana vjerojatnost pogreške pri manipuliranju podatcima. Ne samo da se lakše pogriješi prilikom pisanja SQL upita, nego su i pogreške poput slučajnog brisanja vrijednosti tablice mnogo teže, jer sa sobom povlače cijeli niz zapisa vrijednosti, a ne samo jednu. Naravno da je moguće i odvojiti vrijednosti tako da u svakom redu tablice postoji jedna vrijednost, ali time dolazi do ponavljanja redova tablice (npr. Prvi red bi bio *genre* s vrijednošću *comedy*, a drugi red bi opet bio *genre* s vrijednošću *action*), što bi kroz vrijeme ponovno izazvalo probleme kod dohvaćanja podataka. Dodatno, integritet podataka bi bio mnogo manji zbog manje preglednosti, ponavljana atributa i veće vjerojatnosti pogreške. Na kraju, kod modeliranje same baze podataka postavilo bi se potanje koji je maksimalan broj redova u tablici koju jedan atribut može zauzimati (npr. Koliko maksimalno žanrova može imati film).

Kako bi se Više-Više veze u bazi podataka implementirale prema standardima normalizacije koji rješavaju navedene probleme, koriste se spojne tablice (*eng*. junction tables). Sve što spojna tablica sadrži su primarni ključevi tablica koje treba spojiti, te je sa tim tablicama pomoću primarnih ključeva spojena Više-na-Jedan vezom. Time dobivamo dodatnu tablicu sa jedinstvenim uređenim parom ključeva koji služi kao spona među tablicama koje želimo povezati.

Primjer takvog spajanja se vidi na slici ([Slika 3.4](#tri_dva)) koja prikazuje tablicu žanrova koja je preko spojne tablice (*Genre\_movie*) spojena sa tablicom filmova. Na slici uočavamo kako spojna tablica ima dva atributa: GenreId koji je primarni ključ u tablici žanrova te MovieId koji je primarni ključ u tablici filmova.



Slika 3.4: Implementacija *Više-Više* veze pomoću spojne tablice

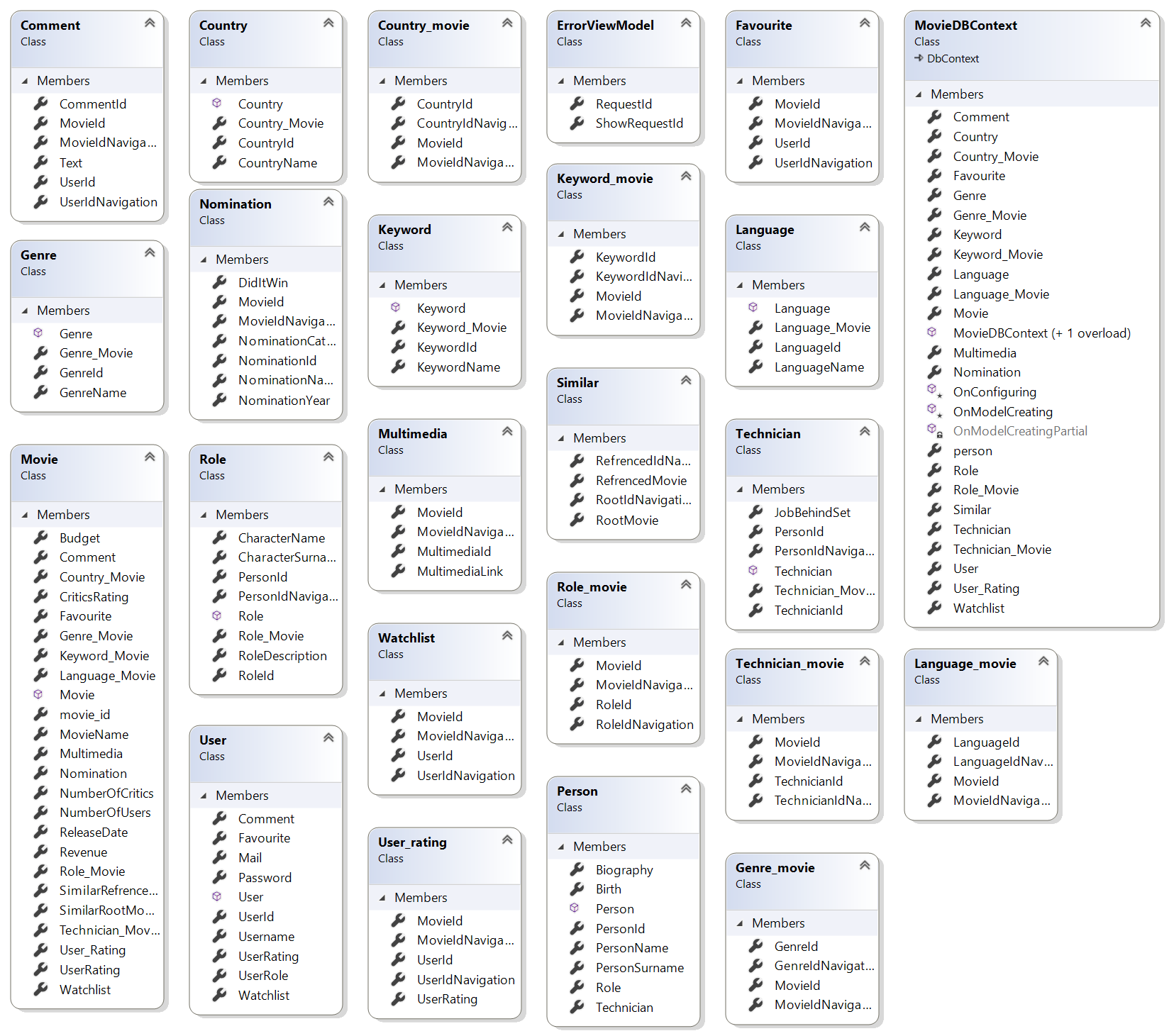
Ovakva implementacija rješava potencijalne probleme manipulacije podatcima, ali ima i izražene nedostatke. Ukoliko u bazi podataka postoji mnogo spojnih tablica (kao posljedica veze Više-Više), vrlo brzo se dolazi do situacije u kojoj se gubi mnogo vremena na implementaciju takvih tablica i njihovog spajanja s kodom i ostatkom baze podataka. Također raste i kompleksnost koda aplikacije pošto se umjesto direktnog rukovanja željenim tablicama do podataka treba doći preko jedne dodatne tablice za svaku Više-Više vezu.

* 1. **Komponente sustava**

Nakon općenite analize rada aplikacije i MVC obrasca, u ovom poglavlju ću detaljnije opisati najvažnije komponente sustava te njihove funkcionalnosti.

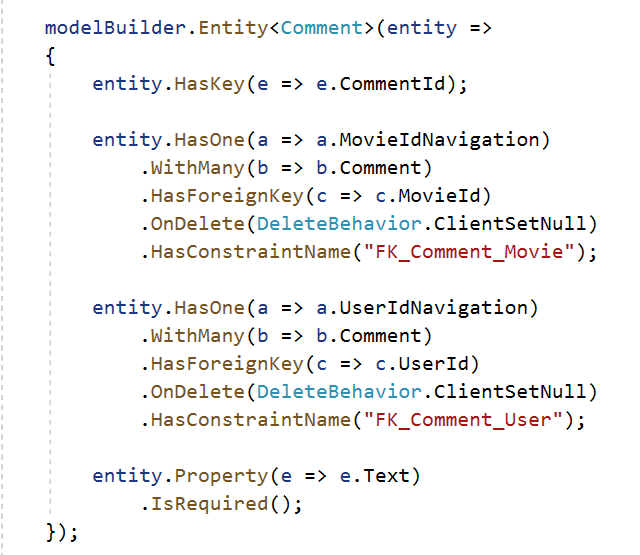
* + 1. **Modeli**

Svaka tablica u bazi podataka je opisana jednim C# razredom koji predstavlja model te tablice. Uz njih, postoji dodatna komponenta *MovieDbContext.cs* koja je ključan dio *Entity Framework Core* radnog okvira i koristi se za konkretan pristup podatcima. Ta komponenta implementira relacijske odnose među tablicama baze podataka te sadrži definicije atributa koji su sadržani u samoj bazi podataka. Na slici ([Slika 3.5](#tri_dvaa)) je prikazan cjelokupan dijagram razreda modela aplikacije.



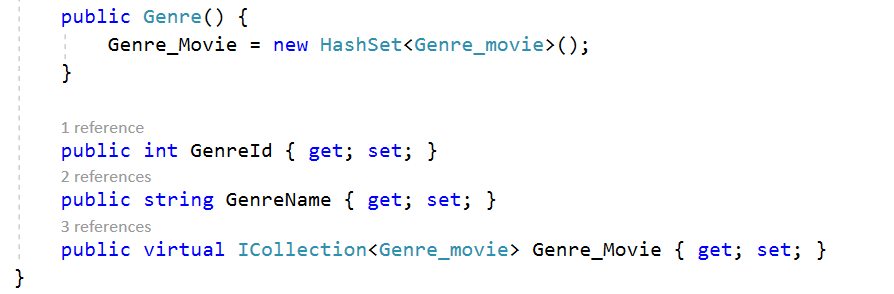
Slika 3.5: Dijagram razreda modela

Na slici ([Slika 3.6](#tri_pet)) je programski koda unutar komponente *MovieDbContext.cs* koji definira atribute tablice komentara. Pomoću parametra *modelBuilder* koji je klase *ModelBuilder* stvaramo model za navedenu tablicu. Nad varijablama iz komponente *Comment* definiramo njima pripadajuće atribute u tablici u bazi podataka te njihova ograničenja. Tako na primjer definiramo kako varijabla *Text* (koja označava tekst komentara) mora imati neku vrijednost (.*isRequired()* - ne može biti *null*), dok za varijable *MovieIdNavigation* i *UserIdNavigation* definiramo veze između komponenata.



Slika 3.6: Definiranje atributa tablice filmova pomoću *Entity Framework Corea*

Ostali modeli se zapravo koriste kao C# razredi koji se pozivaju u kontrolerima koji zatim obrađuju podatke. Sadrže atribute tablice sa pripadajućim metodama za dohvaćanje i postavljanje vrijednosti te, ukoliko je potrebno, varijable razreda s kojima su povezani u bazi podataka. Na slici ([Slika 3.7](#tri_sest)) Može se vidjeti sadržaj modela filmskog žanra *Genre*.



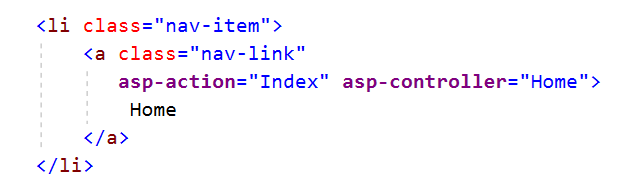
Slika 3.7: Sadržaj modela *Genre*

* + 1. **Pogledi**

Kao što je već spomenuto, pogledi čine prednju stranu aplikacije te sve ono što korisnik vidi. Zbog preglednosti programskog koda, standard je da se pogledi grupiraju u mape te da svakom od tih mapa upravlja jedan upravljač. Tako će na primjer u podmapom *Movie* upravljati upravljač *MovieController*, a u njoj će se nalaziti pogledi *Create*, *Edit*, *Details* i *Index* vezani uz filmove te pripadajući upravljač. Ova četiri navedena pogleda, uz *Delete*, su tipični pogledi MVC aplikacije s kojim je korisnik u interakciji:

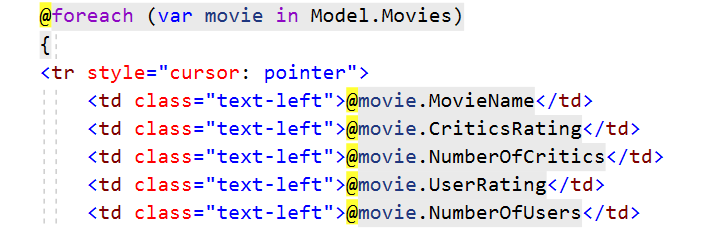
* *Create* - Pogled za stvaranje i spremanje novog entiteta
* *Edit* – Pogled za uređivanje atributa postojećeg entiteta
* *Delete –* Pogled za brisanje entiteta
* *Index* – Pogled u kojem su u navedeni svi entiteti s osnovnim atributima spremljeni u bazi podataka
* *Detail –* Pogled u kojem su navedeni svi atributi povezani s određenim entitetom.

Temelj svakog pogleda je HTML pomoću kojega se dizajnira izgled stranice, ali unutar sintakse takvog koda nalaze se i *Tag Helper* – dio koda koji opisuje što se događa sa stranicom kada korisnik dođe u interakciju s određenim elementom stranice. Pomoću njih se definira koje radnje se trebaju izvrši, koje metode upravljač treba pozvati i koji upravljači su zaduženi za određenu akciju u pogledu. Ukratko, služe kao pomoć pri stvaranju i prikazivanju HTML elemenata. Primjer sintakse HTML-a koji sadrži takav kod prikazan je na slici ([Slika 3.8](#tri_sedam)) Klikom na navigacijski link prikazan u kodu aktivira se upravljač *Home* te se pomoću njega prikazuje pogled *Index* za koji je taj upravljač zadužen.



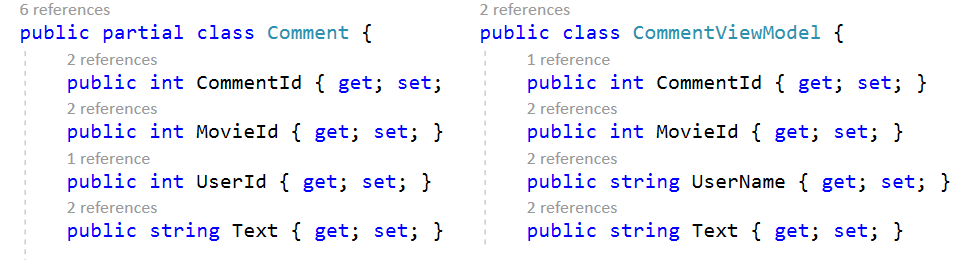
Slika 3.8: Primjer korištenja *Tag* *Helper*-a u pogledu

Dodatno, unutar svakog pogleda za generiranje dinamičkog sadržaja koristi se sintaksa *Razor* unutar kojeg se može programirati u jeziku C# što omogućuje brzo i efikasno manipuliranje podatcima. Tako pisan kod počinje sa znakom „@“ te se može koristiti u bilo kojem dijelu pogleda. Primjer takvog korištenja je prikazan na slici ([Slika 3.9](#tri_osam)) u kojoj se pomoću *Razor* sintakse dinamički stvaraju ćelije tablice sa podatcima koji se dohvaćaju iz modela korištenog u pogledu.



Slika 3.9: Primjer korištenja *Razor*-a u pogledu

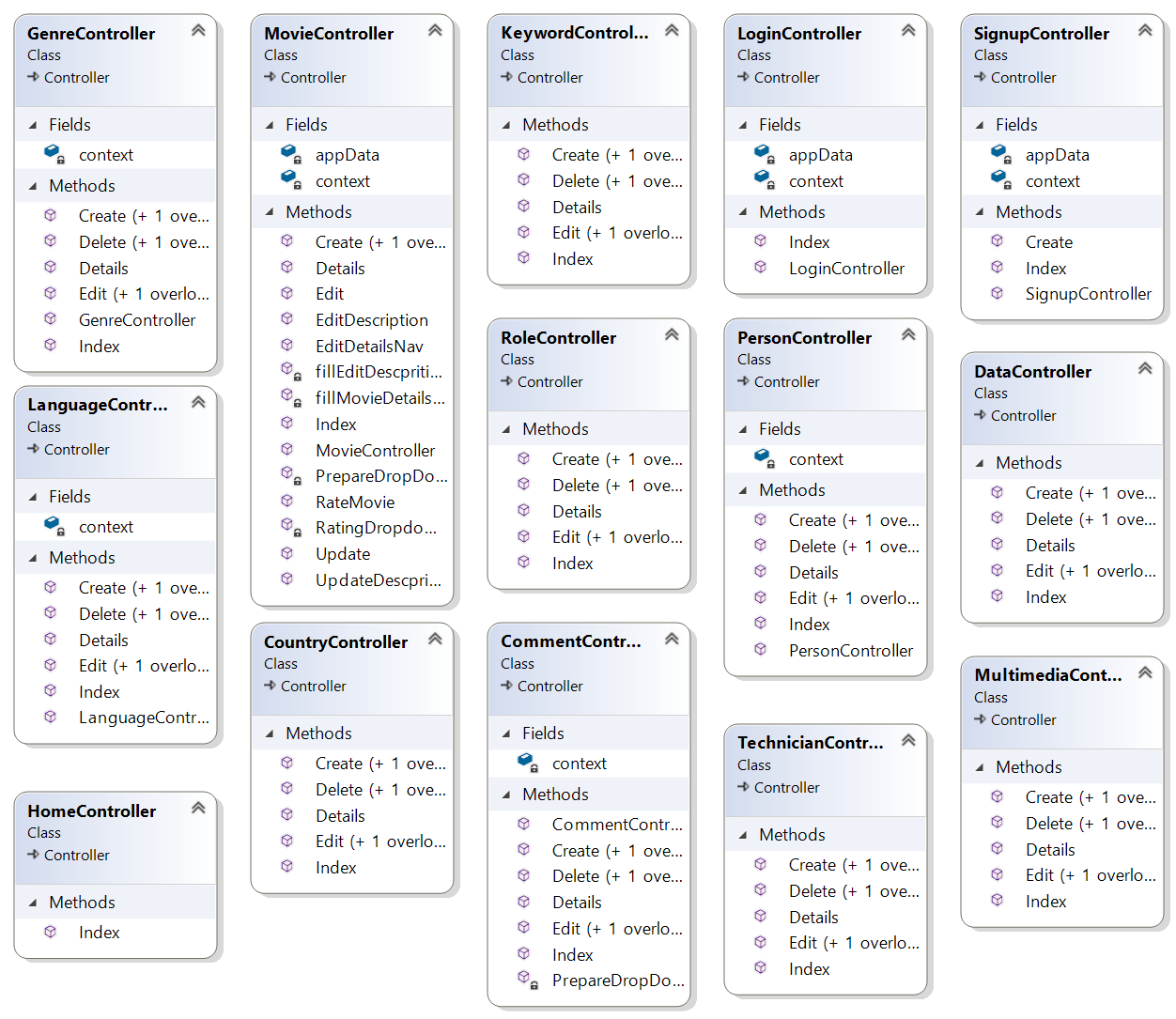
Korišteni model je naveden na početku pogleda također pomoću Razor sintakse (@model Filmofile.ViewModels.MoviesViewModel). Kao što se da primijetiti, model se nalazi u mapi *ViewModels*. Modeli u toj mapi su uglavnom različiti od onih u mapi *Models* te sadrže isključivo podatke koje se žele prikazati na stranici. Jednostavan primjer razlike između modela napravljenih u tim mapama se nalazi na slici ([Slika 3.10](#tri_devet)) koje prikazuje atribute klase *Comment* u mapi *Models* i *CommentViewModel* u mapi *ViewModels*. Razlika je što prva klasa sadrži atribut *UserId* koji se nalazi u pripadnoj tablici u bazi podataka i služi za povezivanje te tablice sa tablicom korisnika. Kako pri prikazu komentara u pogledu stranice želimo prikazati ime korisnika koji je napisao komentar umjesto njegove identifikacijske oznake, klasa modela koji se koristi u tom pogledu će imati atribut *UserName*. Njegovu vrijednost jednom definiramo u upravljaču te ju možemo jednostavno koristiti u pogledu kada nam zatreba bez kompliciranja u kodu.



Slika 3.10: Razlika u atributima razreda korištenog u modelu (lijevo) i u pogledu (desno)

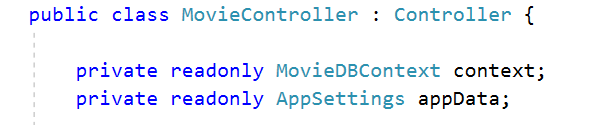
* + 1. **Upravljači**

Sve metode za operaciju nad podatcima nalaze se u upravljačima. Kao spojnica između baze podataka te završnog pogleda kojega vidi korisnik, u upravljačima se definiraju koji će podatci, i u kojem obliku, biti prikazani korisniku ili biti spremljeni u bazu podataka. Dijagram razreda upravljača aplikacije nalazi se na slici ([Slika 3.11](#tri_deset)).



Slika 3.11: Dijagram razreda upravljača

Ono što je zajedničkom svakom upravljaču jest to da manipulira podatcima iz baze podataka, tako da svaki u sebi mora imati stvorenu instancu razreda *MovieDbContext* u kojoj se nalazi opis te baze. Dodatno, inicijalizira se i varijable instance *AppSettings* koja služi za definiranje dodatnih parametara poput maksimalnog broja redova tablice po stranici (koristi se u *Index* metodi). Primjer inicijalizacije takvih varijabli vidljiv je na slici ([Slika 3.12](#tri_jedanaest)).

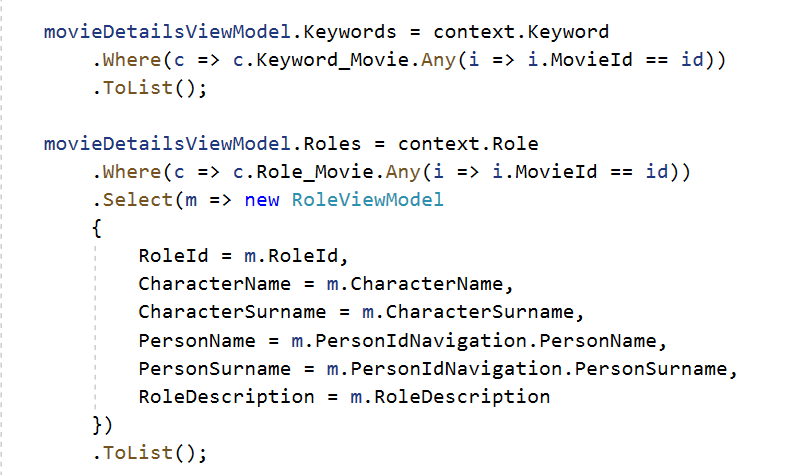


Slika 3.12: Inicijalizacija konteksta baze podataka

Svi upravljači imaju metodu *Index* pomoću koje se prikazuju osnovni podatci svih zapisa komponente iz baze podataka. Nakon pozivanja te metode prvo se iz baze podataka dohvaćaju podatci, a zatim se modeliraju u pripadajući *ViewModel* razred koji sadrži atribute kakvi će biti prikazani korisniku. Kako je već spomenuto, u metodi *Index* se koristi i instanca razreda *AppSettings.* Osim definiranja broja predmeta po stranici, pomoću nje se upravlja i sortiranjem podataka te se osigurava prikaz samo onih predmeta koji i pripadaju toj stranici (na primjer drugih 10 filmova ako smo na stranici broj 2). Na kraju se varijabla *appSettings* i zajedno sa podatcima koji će biti na stranicu sprema u varijablu dodatne klase *ViewModel*-a te se sve skupa šalje pogledu.

Upravljač *MovieController* sadrži mnogo metoda jer se unutar njega nalaze funkcije koje obavljaju operacije nad cijelim nizom tablica u bazi podataka vezanih uz filmove koje se pozivaju prilikom dodavanja, brisanja ili uređivanja popratnih sadržaja.

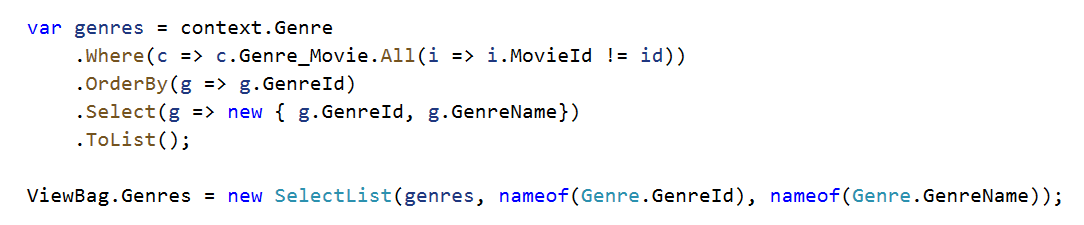
Jedna takva popratna metoda koja se koristi kako bi se dohvatili popratni podatci filmova iz različitih tablica je i *fillMovieDetailsViewModel*. Naime, kako detalji filmova uključuju i informacije o žanru filma, ključnim riječima, glumačkoj postavi i sličnim stvarima, količina podataka koja se prikazuje nije trivijalna. Konkretno u ovoj metodi iz baze podataka se dohvaćaju podatci. Ti podatci se kasnije prikazuju pomoću razreda *MovieDetailsViewModel* koji se šalje upravljaču. Za primjer dohvaćanja podataka u ovoj metodi uzeto je dohvaćanje filmskih uloga te ključnih riječi filma ([Slika 3.13](#tri_dvanaest)).



Slika 3.13: Postupak dohvaćanja i spremanja ključnih riječi i filmskih uloga za detalje o filmu

Obije tablice čiji se podatci traže su sa tablicom filmova u bazi podataka spojene vezom Više-Više, to jest pomoću spojne tablice između njih. Zbog tog razloga se prvo trebaju dohvatiti svi zapisi spojne tablice koji pripadaju filmu (MovieId == id) te se tek onda može pristupiti traženim vrijednostima. Također, na primjer ove slike možemo uočiti već objašnjenu potrebu za razredima *ViewModel*. Pri dohvaćanju uloga se preko varijable *PersonIdNavigation* oblika *Person* dolazi do imena i prezime osobe koje se nalaze u vlastitoj tablici. Vrijedi primijetiti kako za ključne riječi nije napravljen *ViewModel* pošto je tablica trivijalna i osim ključa sadrži jedino naziv koji se i treba prikazati.

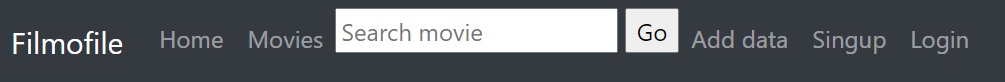
Metoda *PrepareDropDownLists* se koristi prilikom dohvaćanja podataka koji nisu pridijeljeni filmu. Naime, Prilikom nadopunjavanja detalja redundantno je prikazivati podatke koji već pripadaju filmu. Ako na primjer nekoj komediji želimo pridijeliti dodatan žanr, u metodi *PrepareDropDownLists* ćemo pripremiti sve žanrove u bazi podataka isključujući žanr komedije. Primjer takve vrste filtriranja podataka za prikaz se nalazi na slici ([Slika 3.14](#tri_trinaest)). Nakon filtriranja rezultati se u obliku liste spremaju u razred *ViewBag* kojim se može pristupiti iz pogleda.



Slika 3.14: Filtriranje podataka za prikaz

1. **Funkcionalnosti aplikacije**

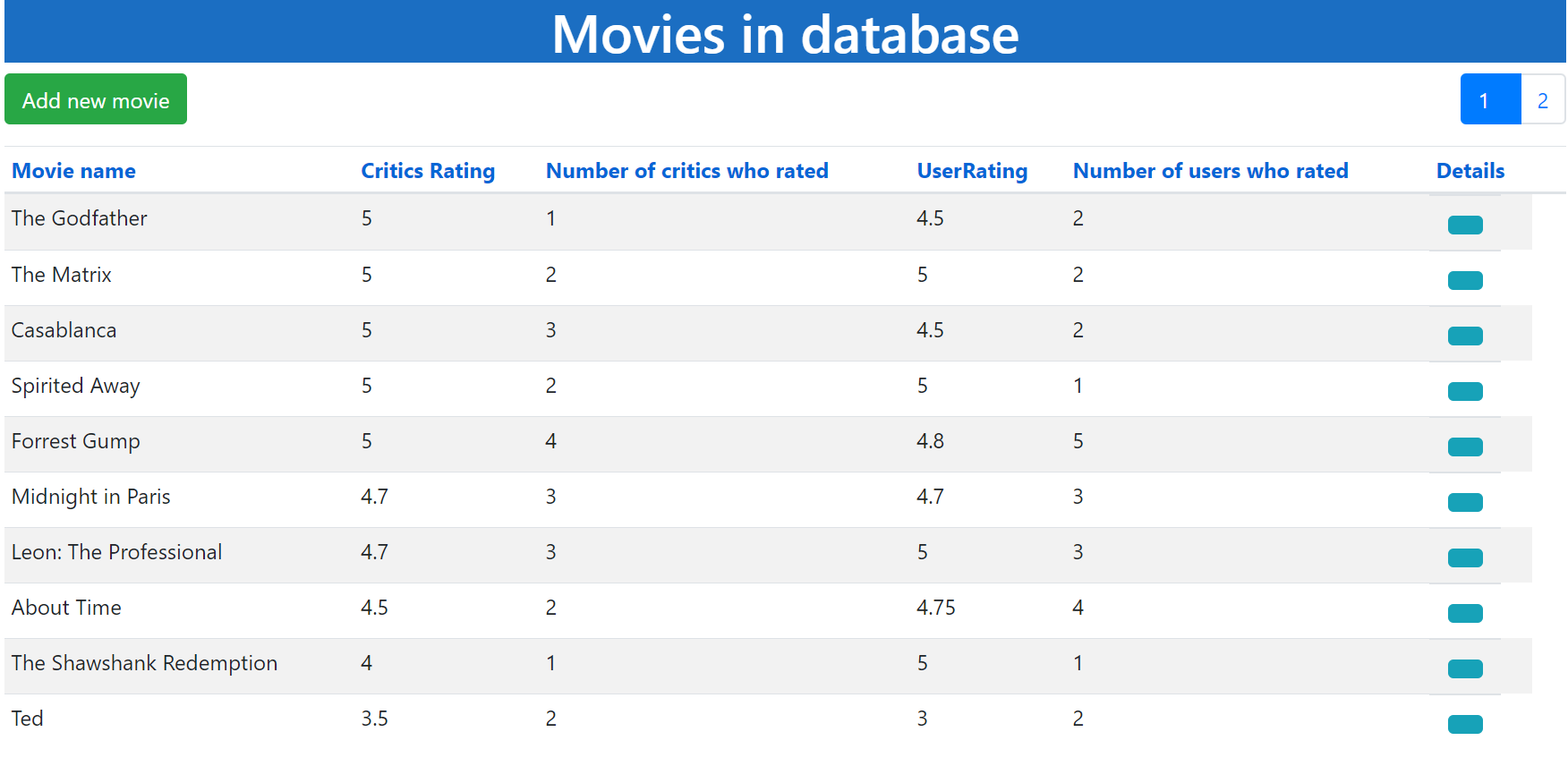
U ovom poglavlju su opisane sve implementirane funkcionalnosti aplikacije te popratno korisničko sučelje. Za početak, na vrhu aplikacije u svakoj situaciji nalazi se navigacijska traka sa navigacijskim linkovima te poljem za pretraživanje ([Slika 4.1](#cetiri_jedan)). Početna stranice se nalazi pod poveznicom *Home*. U bilo kojem trenutku korištenja aplikacije korisnik se može služiti bilo kojom od funkcionalnosti koju nudi navigacijska traka.



Slika 4.1: Navigacijska traga stranice

* 1. **Lista filmova**

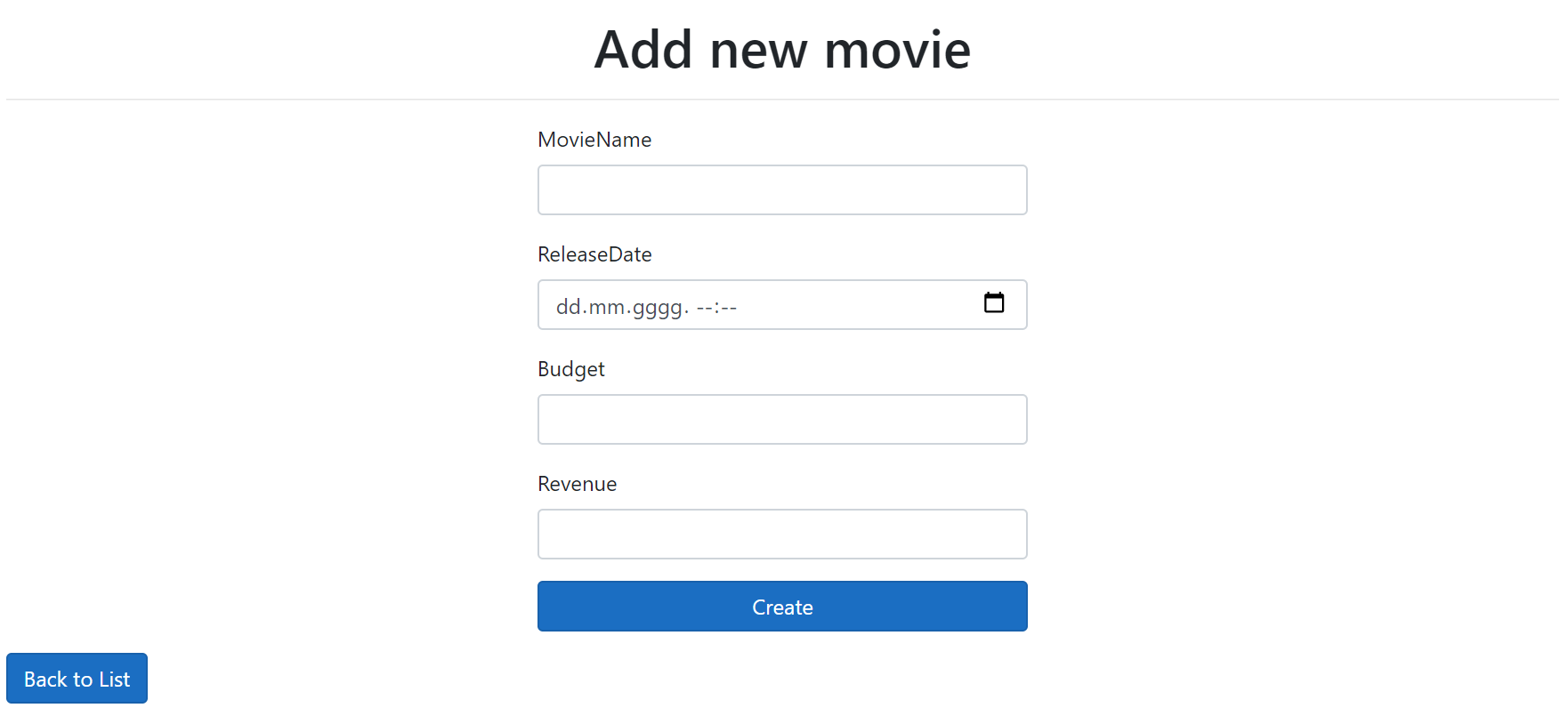
Klikom na poveznicu *Movies* navigacijske trake dolazi se do popisa filmova koji se nalaze u bazi podataka. Tu su prikazani naslovi filmova te pripadajuće prosječne ocjene i broj korisnika i filmskih kritičara koji su ocijenili film ([Slika 4.2](#cetiri_dva)). Filmovi se mogu sortirati po bilo kojoj od tih kategorija. Na jednoj stranici je navedeno 10 filmova, a do sljedeće stranice se dolazi klikom na gumb željene stranice koji se nalazi na gornjem lijevom uglu prikaza. Ukoliko se Klikne na plavi gumb u stupcu detalji, dolazi se na prozor sa prikazanim detaljima željenog filma. Dodatno, klikom na gumb *Add new movie* se dolazi na stranicu sa formularom za dodavanje novoga filma.



Slika 4.2: Lista filmova poredanih po prosječnoj ocjeni kritičara

* 1. **Dodavanje novog filma**

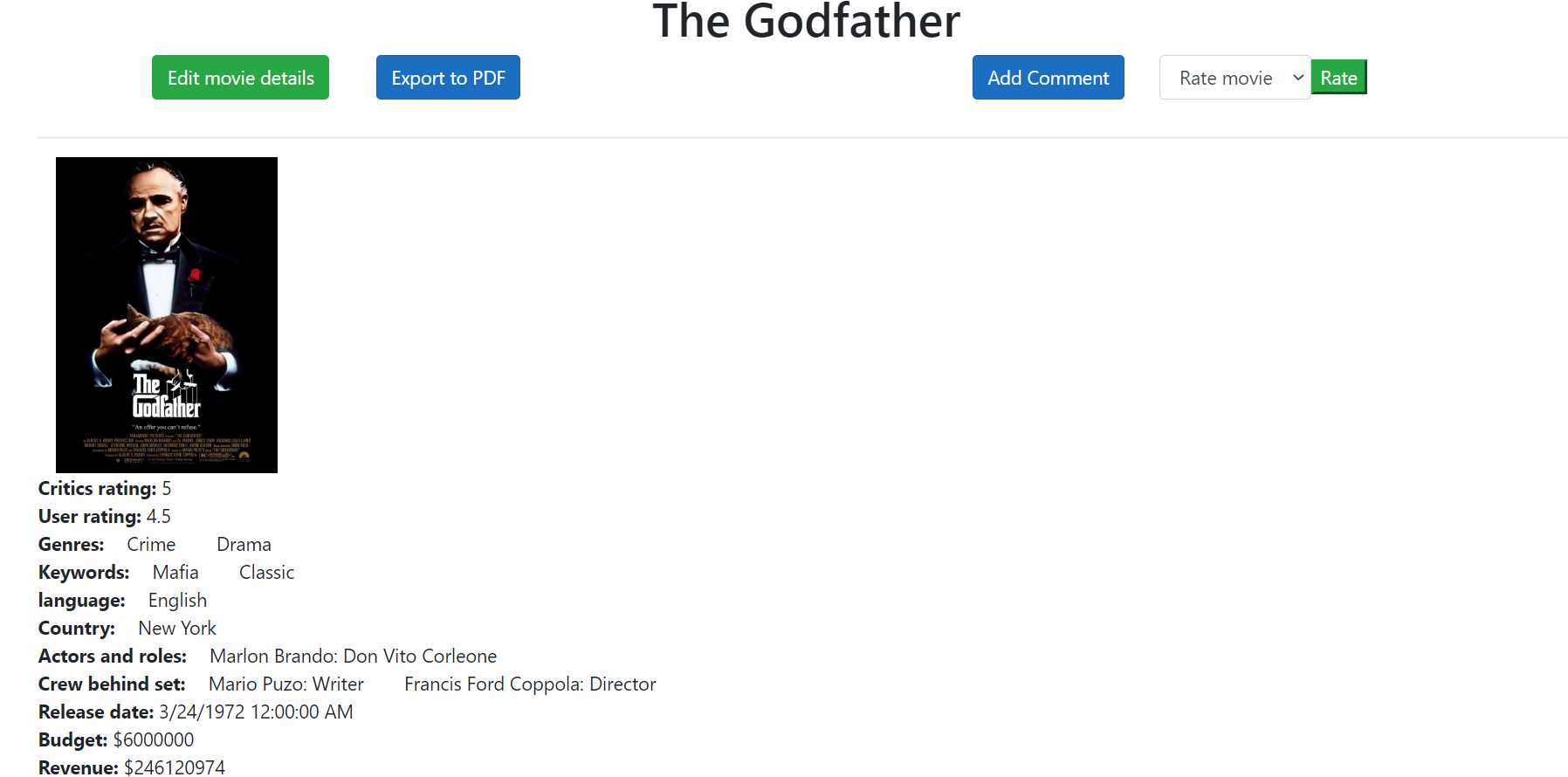
Na stranici za dodavanje novog filma ([Slika 4.3](#cetiri_tri)) nalazi se jednostavan formular čijim se popunjavanjem dodaje novi film u bazu podataka. Jedino obavezno polje je ono sa nazivom filma. Ostala polja se mogu, ali ne moraju popuniti. Klikom na gumb *Back to List* korisnik se vraća nazad na stranicu s popisom filmova. Za popunjavanje detalja o filmu poput žanrova, ključnih riječi ili uloga koriste se posebne forme. Za prekid dodavanja bez spremanja treba stisnuti gumb *Back to list* kojim se vraća na popis filmova ili na neku od poveznica kontrolne trake.



Slika 4.3: Jednostavni formular za dodavanje novog filma

* 1. **Detalji filma**

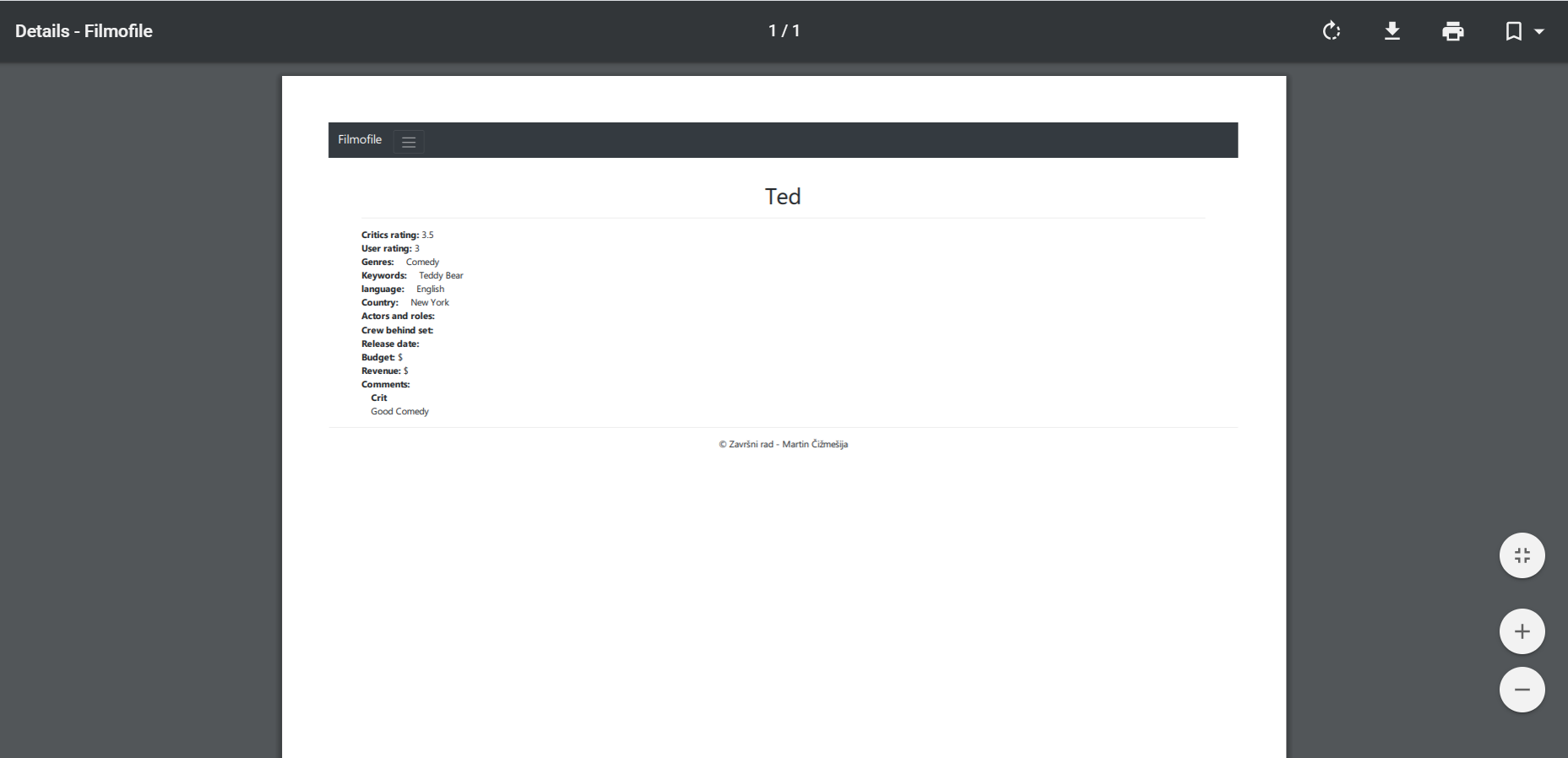
Na stranici detalja sa filmovima se nalaze svi podatci vezani uz film. Ukoliko u bazi podataka postoji poveznica na sliku odabranog filma, ona se prikazuje na stranici. Ispod nje su navedene ocjene kritičara i korisnika, a nakon toga i žanrovi i ključne riječi koje pripadaju filmu. Zatim su navedeni jezik te država u kojoj je film sniman te potom i glumačka i tehnička postava filma ([Slika 4.4](#cetiri_cetiri)). Ispod tih informacija se nalazi i budžet i podatak o tome koliko je film zaradio, a za kraj su prikazani komentari korisnika koji su taj film komentirali. Na vrhu stranice se nalazi gumb *Edit* *movie* *details* kojim se pristupa izborniku za dodavanje, brisanje i uređivanje detalja filma. Pored njega se nalazi gumb *Export to PDF* koji konvertira stranicu detalja filma u pdf oblik. Dodatno, ovdje se nalaze i funkcionalnosti komentiranja i ocjenjivanja filma. Gumb *Add Comment* vodi na posebno sučelje za komentiranje filma, dok se ocijeniti film može tako što se odabere ocjena koja mu se želi pridijeliti te potom klikne na gumb *Rate*. Na samome dnu stranice se nalazi gumbi *Back to List* kojim se vraća na popis filmova, do kojeg se može doći i klikom na poveznicu *Movies* navigacijske trake.



Slika 4.4: Dio prikaza detalja filma

* 1. **Preuzimanje filma kao PDF**

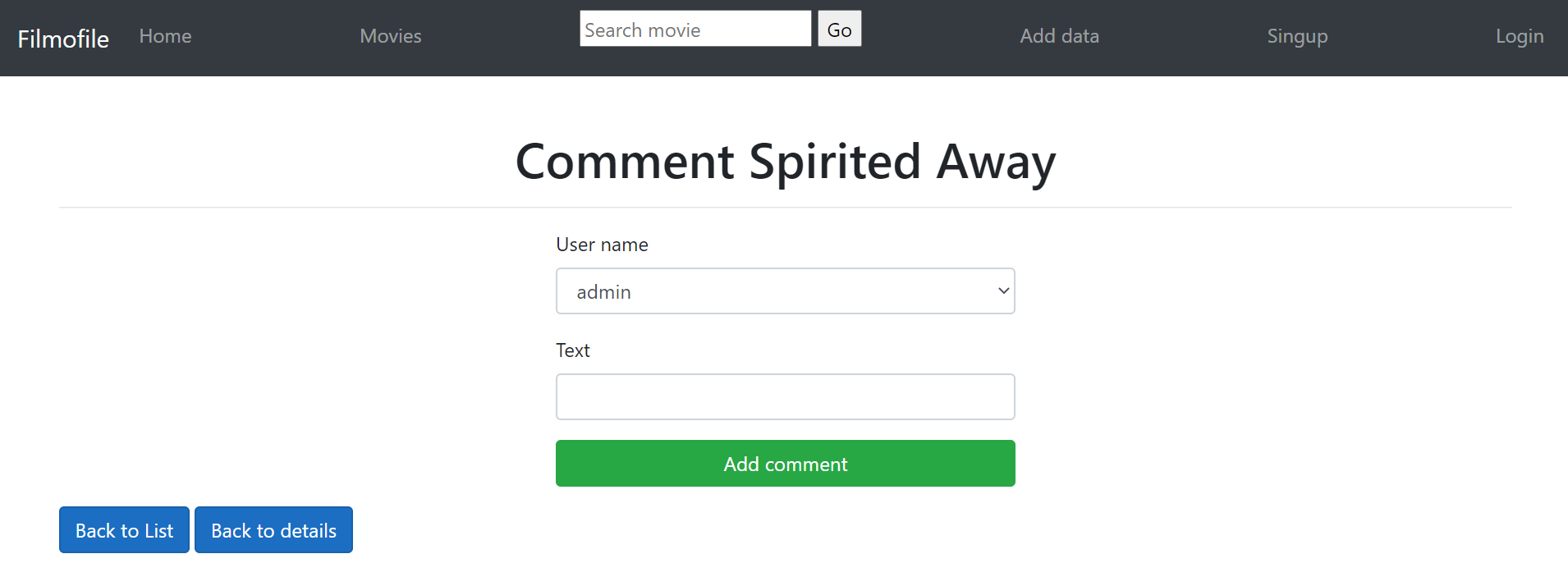
Kako je već napomenuto, klikom na gumb *Export to PDF* na stranici detalja filma generira se pdf dokument sa svim informacijama filma ([Slika 4.5](#cetiri_nula)). Takav dokument se može spremiti na računalo ili isprintati.

****

Slika 4.5: prikaz generiranog PDF obrasca filma

* 1. **Ocjenjivanje i Komentiranje filmova**

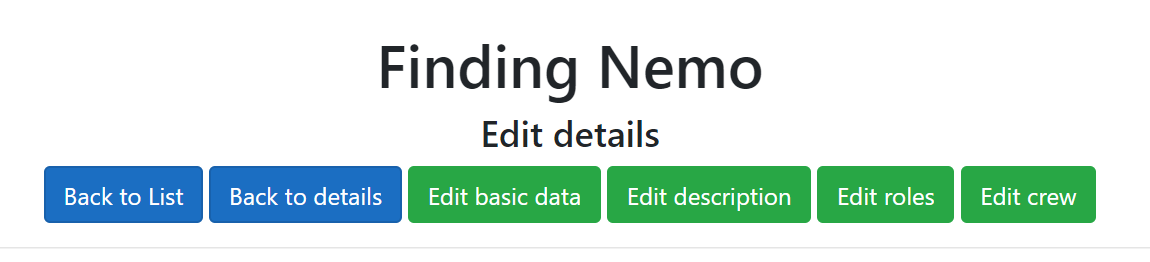
Kako je već spomenuto, ocjenjivanje filmova se vrši na stranici detalja filma. Odabirom željene ocjene filma (od 1 do 5) te pritiskom na gumb *Rate* izvršava se ocjenjivanje. Ocjena se sprema u bazu podataka, te se stranica osvježava sa ažuriranom prosječnom ocjenom filma. Kada se klikne gumb *Add Comment* otvara se novi prozor ([Slika 4.5](#cetiri_pet)) na kojem se može dodati komentar filma. Unošenjem komentara te pritiskom na gumb *Add comment* korisnik se vraća na stranicu sa detaljima komentiranog filma. Komentar se ažurira u bazu podataka te je odmah vidljiv korisniku među ostalim komentarima filma.



Slika 4.6: Primjer formulara za komentiranje filma

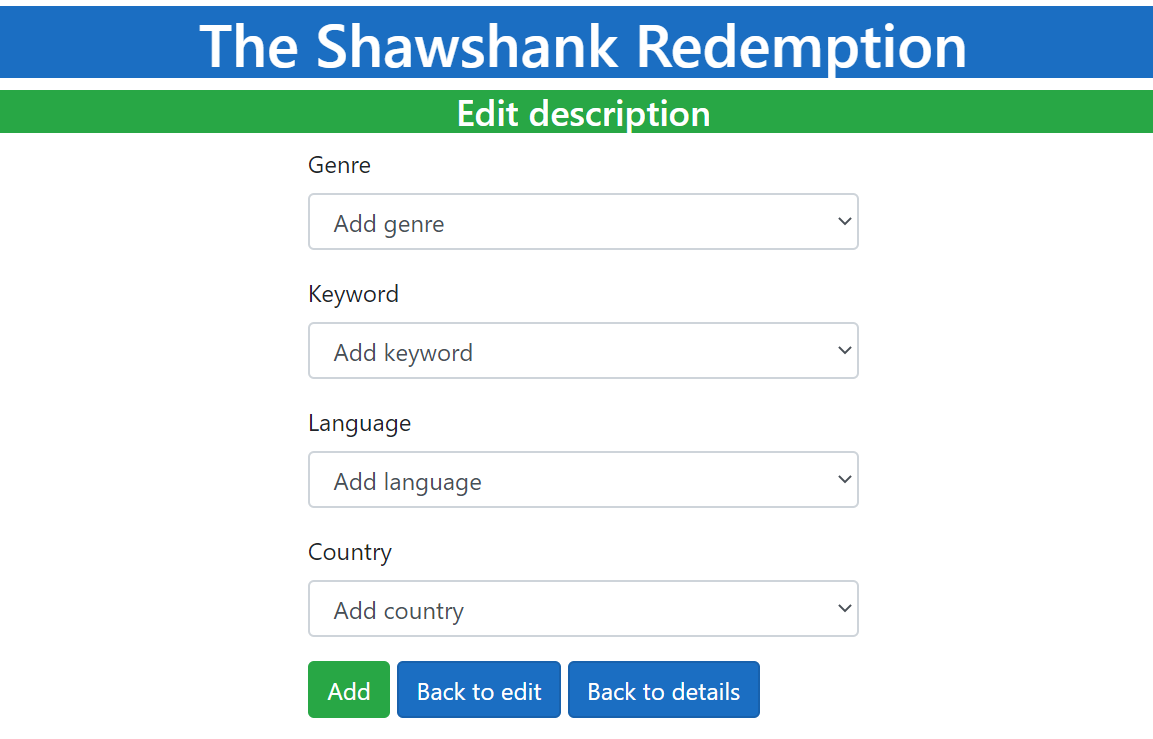
* 1. **Uređivanje podataka o filmovima**

Na stranici za uređivanje podataka o filmovima kojom se dolazi preko sučelja detalja filma nalazi se izbornik kojim se odabire unos željene vrste podatka ([Slika 4.6](#cetiri_sest)). Gumb *Back to List* vraća korisnika na tablicu s popisom filmova, dok gumb *Back to details* vraća korisnika na detalje filma kojega je odlučio urediti.



Slika 4.7: Izbornik za uređivanje podataka o filmu

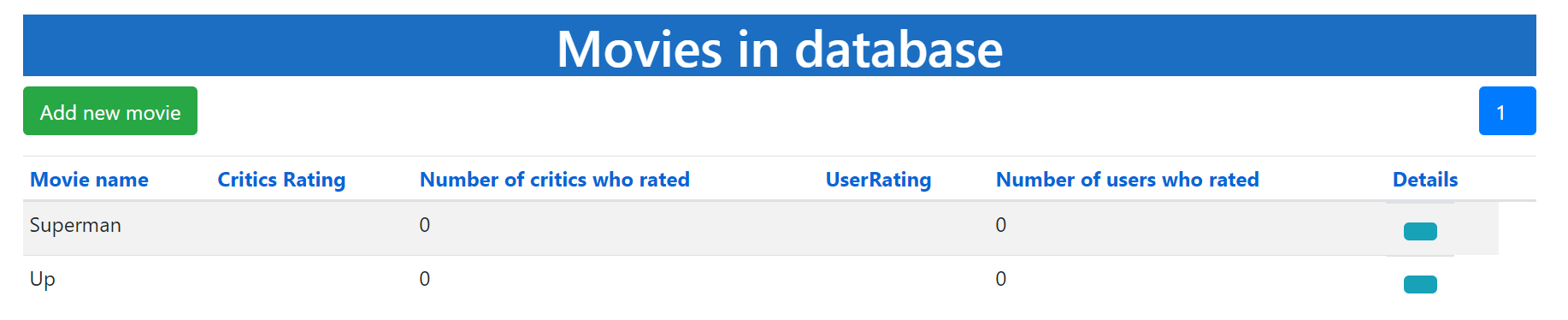
Gumb *Edit basic data* vodi do formulara sličnog onom za stvaranje filma (sa istim karakteristikama). *Edit description* vodi do formulara pomoću kojeg se filmu pridjeljuju ključne riječi, žanrovi, jezik i država u kojoj je sniman ([Slika 4.7](#cetiri_sedam)), dok gumbi *Edit roles* i *Edit crew* služe da dodavanje uloga te filmske ekipe koja je radila na filmu.



Slika 4.8: Formular za unos ključnih popratnog opisa filma

* 1. **Pretraživanje filmova**

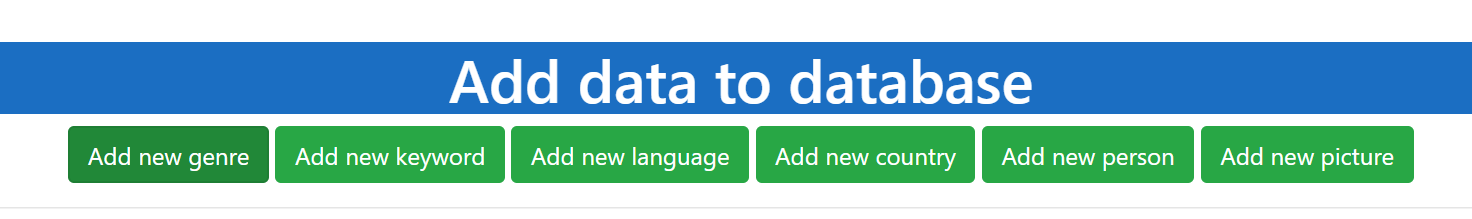
Pretraživanje filmova po naslovu omogućeno je poljem za pretraživanjem na navigacijskoj traci. Upisivanjem željenog teksta koji je naslov (ili dio naslova) filma te potom pritiskom na tipku *Go* na stranici se prikazuje lista filmova koji u naslovu sadrže uneseni tekst. Rezultati pretraživanja filmova nakon upisivanja „up“ u polje za pretraživanje su prikazani na slici ([Slika 4.8](#cetiri_osam)). Osim filma *Up* čiji je cijeli naslov u potpunosti obuhvaćen tekstom upisanom u navigacijsku traku, kao rezultat se prikazuje i film *Superman* koji u svom naslovu sadrži pretraživani izraz. Na ovaj način filmovi se mogu pretraživati prema samo djelu naslova bez potrebe da se on u potpunosti upisuje. Također, valja napomenuti kako pretraživanje nije osjetljivo na velika i mala slova.

****

Slika 4.9: Primjer rezultata pretraživanja filmova po naslovu

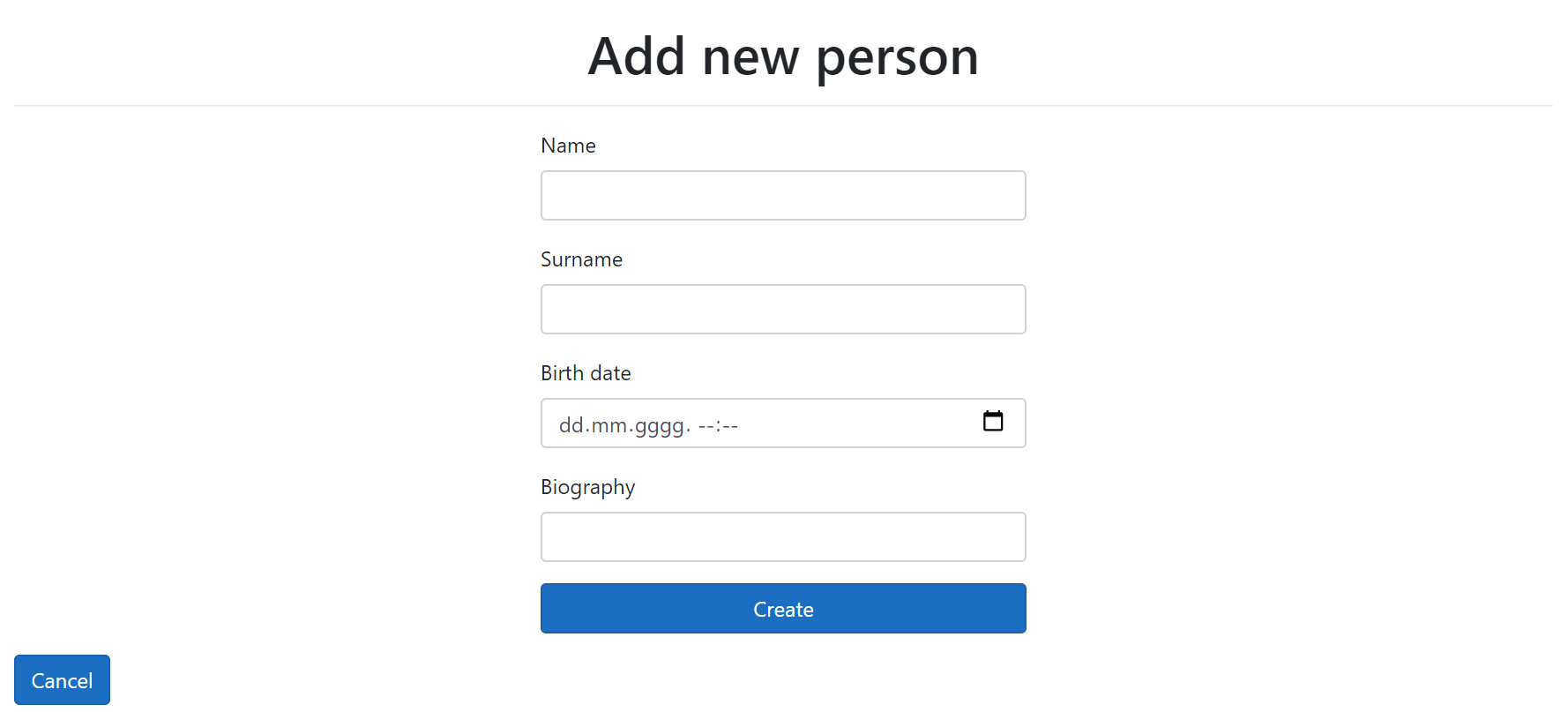
* 1. **Dodavanje podataka u bazu podataka**

Klikom na poveznicu *Add* *data* navigacijske trake dolazi se do izbornika koji služi za dodavanje još neunesenih glumaca, žanrova, jezika i ostalih popratnih filmskih sadržaja ([Slika 4.9](#cetiri_devet)). Klikom na željeni gumb dolazi se do formulara čijim se popunjavanjem vrijednost dodaje u bazu podataka.



Slika 4.10: Izbornik za dodavanje novih popratnih podataka u bazu podataka

Formular kojim se dolazi klikom na gumb *Add new person* se nalazi na slici ([Slika 4.10](#cetiri_deset)). Nakon unosa podataka o osobi klikom na gumb *Create* osoba se unosi u bazu podataka. Ukoliko neki od obaveznih podataka nije unesen (u ovom slučaju ime osobe) stranica javlja kako je taj podatak potrebno popuniti za uspješno spremanje unosa. Ukoliko korisnik želi prekinuti unošenje bez spremanja može pritisnuti gumb *Cancel*. Analogno ovom formularu su napravljeni i ostali, svaki sa poljima koji opisuju željeni podatak.



Slika 4.11: Formular za dodavanje nove osobe u bazu podataka

1. **Korištene tehnologije i alati**

Aplikacija je razvijana u razvojnom okruženju *Visual Studio Enterprise 2019 [*[*4*](#Enterprise)*]*, a pisana je u radnom okviru *ASP.NET Core 3.1* [[5](#core)]. Korištena baza podataka je *Microsoft SQL Server 2019* [[6](#server)], a za pregledavanje podataka i izradu tablica je korišten *Microsoft SQL Server Management Studio 2018* [[7](#management)]. *Entity Framework Core* [[8](#entity)] je korišten za povezivanje baze podataka s aplikacijom. Preko njega se omogućuje rad nad bazom podataka s C# razredima te minimizira pisanje koda. Za izradu relacijskog modela baze podataka korištena je online aplikacija *Draw.io* [[9](#draw)], a za izradu dijagrama razreda je korišten alat za generiranje dijagrama ugrađen u *Visual Studio Enterprise 2019*. Stražnji dio aplikacije je pisan u programskom jeziku C#, dok je prednji dio pisan u CSS-u, JavaScript-u te HTML-u pomoću u *ASP.NET Core* ugrađenih radnih okvira *Bootstrap* [[10](#bootstrap)] i *jQuerry* [[11](#jquerry)] koji uvelike olakšavaju korištenje CSS-a i JavaScript-a

**Zaključak**

Filmska industrija u današnje vrijeme raste iz dana u dan, a sa njome i potreba za pristupom podatcima vezanih uz nju. Već postoje razne aplikacije i internetske baze podataka sa raznim funkcionalnostima te sam siguran da će se u budućnosti razvijati i nove. U ovom radu je napravljena jedna takva aplikacija osnovnih funkcionalnosti te je obrađena problematika sa kojom se susreću programeri prilikom razvijanja jednog takvog sustava. Film sa sobom povlači cijeli niz drugih informacija koje su potrebne da bi zadovoljile znatiželju korisnika, a kako se širi domena podataka koju aplikacija pokriva, sa njome raste i baza podataka i posao potreban za programiranje te aplikacije.

Koliko god u ovom radu kreirana aplikacija demonstrira samo osnovne funkcionalnosti, one se lako mogu nadograđivati i dodavati. Na već napravljeno trebali bi se prvo povezati podatci za koje već postoje tablice u bazi podataka poput prikaza jezika ili države u kojoj je film sniman. Kao početni i najvažniji korak, do sada su analizirani načini razvoja jedne ovakve aplikacije te su implementirane najvažnije funkcionalnosti.

**Literatura**

[1] *IMDb* informacije, URL: <https://en.wikipedia.org/wiki/IMDb>

[2] Rotten Tomatoes informacije, URL: <https://en.wikipedia.org/wiki/Rotten_Tomatoes>

[3] Arhitekturni obrazac MVC, URL: <https://www.tech101.in/streamline-your-system-with-the-mvc-model/>

[4] Visual Studio Enterprise 2019, URL: <https://visualstudio.microsoft.com/vs/enterprise/>

[5] ASP.NET Core 3.1 download, URL: <https://dotnet.microsoft.com/download/dotnet-core/3.1>

[6] Microsoft SQL Server 2019, URL: <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>

[7] Microsoft SQL Server Management Studio 2018, URL: <https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>

[8] Entity Framework Core informacije, URL: <https://docs.microsoft.com/en-us/ef/core/>

[9] Draw.io , alat za crtanje relacijskog modela baze podataka, URL: <https://dbdiagram.io/home>

[10] Bootstrap informacije, URL: <https://getbootstrap.com/>

[11] JQuerry informacije, URL: <https://jquery.com/>

**Mrežna evidencija podataka o filmovima**

**Sažetak**

U ovome radu je opisan proces razvoja i izvedbe web aplikacije podataka o filmovima. Rad počinje sa već postojećim programskim rješenjima te njihovom implementacijom određenih funkcionalnosti. Zatim su navedeni zahtjevi aplikacije, a nakon toga se analizira baza podataka potrebna za realizaciju aplikacije te arhitektura rješenja kao i njena implementacija. Na kraju su navedene funkcionalnosti aplikacije te je zatim prikazan sažetak tehnologija korištenih u razvoju aplikacije.

**Ključne riječi:** film, filmovi, baza podataka, web aplikacija, MVC, C#, ASP.NET Core, Entity Framework Core

**Web records of movie related data**

**Summary**

This paper describes the process of developing and implementing a web application of movie related data. The work begins with already existing software solutions and their implementation of certain functionalities. Then the requirements of the application are listed, and after that the database required for the implementation of the application and the architecture of the solution, as well as its implementation are analyzed. Finally, the functionalities of the application are listed, and after it a summary of technologies used in development of the application is presented.

**Keywords:** movie, movies, film, database, web application, MVC, C#, ASP.NET Core, Entity Framework Core