

**SVEUČILIŠTE U ZAGREBU  
FAKULTET ORGANIZACIJE I INFORMATIKE  
VARAŽDIN**

**Matteo Rajkov**

# **VIDEOTEKA**

## **PROJEKT**

**Zagreb, Veljača 2021**

**SVEUČILIŠTE U ZAGREBU**  
**FAKULTET ORGANIZACIJE I INFORMATIKE**  
**V A R A Ž D I N**

**Matteo Rajkov**

**Matični broj: 45597/17-Izv**

**Studij: Primjena informacijske tehnologije u poslovanju**

**VIDEOTEKA**  
**PROJEKT**

**Mentor:**

Izv. prof. dr. sc. Markus Schatten

**Zagreb, Veljača 2021**

## **Izjava o izvornosti**

Izjavljujem da je moj projekt izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

*Autor*  
*Matteo Rajkov*

## SADRŽAJ

1. Sažetak .....	1
2. Opis Aplikacije .....	2
3. Opis modela baze podataka.....	3
4. Opis koda baze podataka .....	4
4.1 Tablice 'Genre', 'Director', 'Actor', 'Customer' .....	4
4.2 Tablice 'Customer_To_Movie', 'Actor_To_Movie', 'Movie' .....	5
4.3 Unos podataka u tablice 'Actor', 'Director', 'Genre', 'Movie', 'Customer' .....	6
4.4 Ispis podataka iz baze podataka .....	7
5. Opis aplikacijske funkcionalnosti .....	11
5.1 Glavni izbornik digitalne videoteke.....	11
5.2 [1] Ispis svih dostupnih filmova u videoteci .....	12
5.3 [2] Ispis svih redatelja koji su režisirali filmove u bazi podataka videoteke.....	14
5.4 [3] Ispis svih glumaca koji su glumili u filmovima u bazi podataka videoteke .....	14
5.5 [4] Posudba filma .....	15
5.6 [5] Ispis liste svih posuđenih filmova .....	17
5.7 [6] Ispis detalja filma .....	17
5.8 [9] Izlaz iz aplikacije.....	18
6. Opis programskog koda – Java.....	19
6.1 Klasa 'DatabaseManagerImpl' .....	21
6.2 Sučelje 'DatabaseManager' .....	23
6.3 Klasa 'InputHandler' .....	24
6.4 Klasa 'Printer' .....	26
6.5 Klasa 'Main' .....	28
7. Zaključak.....	29
8. Literatura.....	30
9. Popis slika.....	31

# 1. Sažetak

U ovome projektu izrađena je aplikacija koja služi kao digitalna videoteka. Uz nju je izrađena i prikladna baza podataka koja sadrži podatke za korištenje primitivnog oblika digitalne videoteke. Ovaj projekt izrađen je u jeziku SQL za upravljanje bazama podataka te u Java programskom jeziku.

## 2. Opis Aplikacije

Aplikacija izrađena u ovome projektu služi kao digitalna videoteka u koju korisnici mogu doći i posuditi razne filmove te ima svrhu obične videoteke kojih je sve rjeđe i rjeđe.

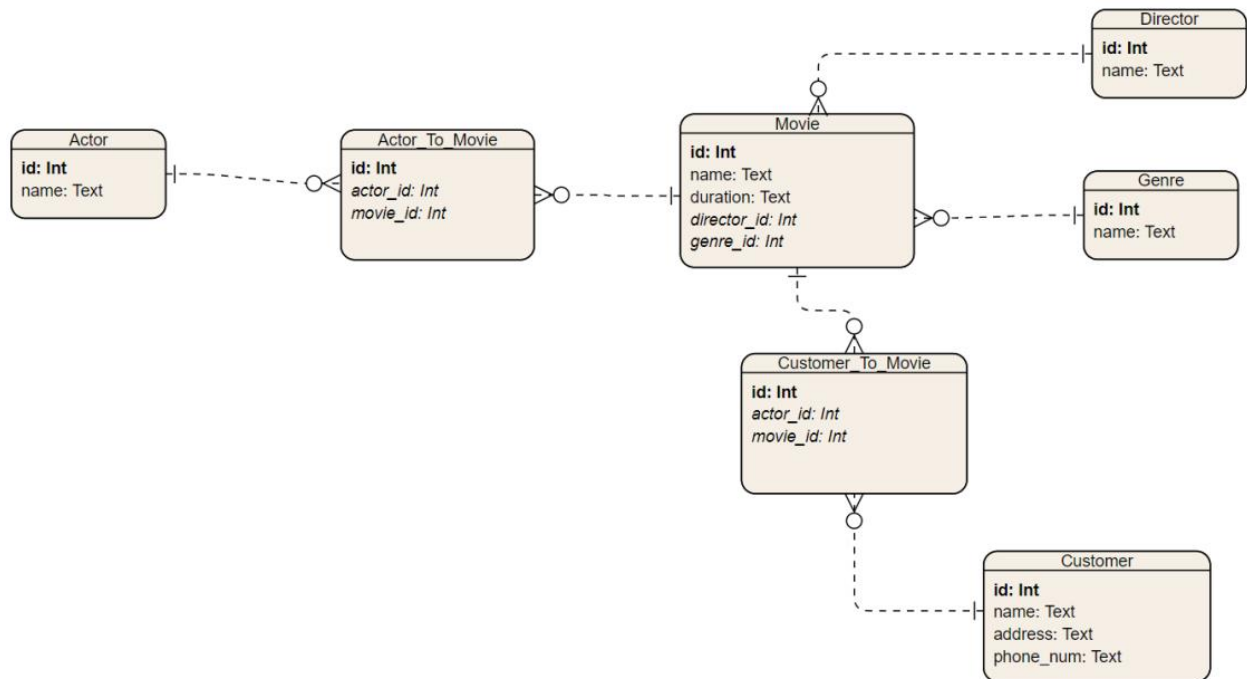
Za ovu aplikaciju koristila se baza podataka u kojoj su kreirane tablice koje sadrže podatke kakve bismo pronašli u običnoj videoteci, kao što su: glumci, redatelji, žanrovi, korisnici, i naravno, filmovi.

Aplikacija ima svrhu dohvaćanja svih podataka koje posjeduje baza podataka te iste videoteke, te pruža mogućnosti dohvaćanja informacija na način na koji bi se kupcu olakšalo doći do željenog filma.

Aplikacija ima sljedeće mogućnosti te je napisana u konzolnom obliku:

- Ispis trenutno dostupnih filmova u videoteci
- Ispis liste redatelja svih filmova koji su režisirali filmove u videoteci
- Ispis liste glumaca koji glume u filmovima u videoteci
- Posudba filmova iz videoteke
- Ispis liste svih filmova koji su posuđeni
- Ispis detalja filma

### 3. Opis modela baze podataka



Slika 1: ER Dijagram

Na slici je prikazan ER dijagram baze podataka koji prikazuje odnose između tablica i atributa u tablicama baze podataka.

## 4. Opis koda baze podataka

Za kreiranje ove aplikacije kreirano je 7 tablica, 4 od kojih služe za pohranu osnovnih podataka te sadrže samo primarne ključeve, dvije koje sadrže strane ključeve te služe kao poveznica između tablica i 1 koja služi kao poveznica između više tablica.

### 4.1 Tablice 'Genre', 'Director', 'Actor', 'Customer'

Ova tablica kreirana je kao osnovna tablica za pohranu podataka o žanrovima filmova.

Zbog restrikcija u SQL-u za atribut '*id*' nije bilo moguće koristiti tip podataka '*SERIAL*' koja služi za automatsko generiranje unikatnih brojeva tipa podataka 'Integer', jer kao što će biti prikazano u daljnjim isječcima koda (4.b), za određivanje stranog ključa korišten je tip podataka '*INT*', no SQL ne dozvoljava da je strani ključ različit tip podataka od vrijednosti '*id*'.

Kako bismo zaobišli taj problem, korištena je ključna riječ '*AUTO\_INCREMENT*' koja služi kao zamjena za tip podataka '*SERIAL*', te ima sličnu funkcionalnost kao i '*SERIAL*' tip podataka, gdje se vrijednosti automatski uvećavaju za jedan ako nisu definirane.

```
CREATE TABLE Genre (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    name TEXT NOT NULL  
);  
  
CREATE TABLE Director (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    name TEXT NOT NULL  
);  
  
CREATE TABLE Actor (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    name TEXT NOT NULL  
);  
  
CREATE TABLE Customer (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,
```



```
name TEXT NOT NULL,  
address TEXT,  
phone_num TEXT  
);
```

Za svaku gore navedenu tablicu morali smo deklarirati atribut 'name' kako bismo mogli unositi vrijednosti u gore navedene attribute tablica.

## 4.2 Tablice 'Customer\_To\_Movie', 'Actor\_To\_Movie', 'Movie'

U gore navedenim tablicama za kreaciju tablica '*Customer\_To\_Movie*' i '*Actor\_To\_Movie*', morali smo se poslužiti

```
CREATE TABLE Actor_To_Movie (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    actor_id INT NOT NULL,  
    movie_id INT NOT NULL,  
    FOREIGN KEY (actor_id) REFERENCES Actor(id),  
    FOREIGN KEY (movie_id) REFERENCES Movie(id)  
);
```

```
CREATE TABLE Customer_To_Movie (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    customer_id INT NOT NULL,  
    movie_id INT NOT NULL,  
    FOREIGN KEY (customer_id) REFERENCES Customer(id),  
    FOREIGN KEY (movie_id) REFERENCES Movie(id)  
);
```

```
CREATE TABLE Movie (  
    id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,  
    name TEXT NOT NULL,  
    duration TEXT NOT NULL,
```

```

director_id INT NOT NULL,
genre_id INT NOT NULL,
FOREIGN KEY (director_id) REFERENCES Director(id),
FOREIGN KEY (genre_id) REFERENCES Genre(id)
);

```

Tablica *'Actor\_To\_Movie'* bila je potrebna kako bi se ostvarila veza između tablice *'Movie'* i tablice *'Actor'* koja je kardinalnosti *'više na više'*, također isto pravilo vrijedi i za tablicu *'Customer\_To\_Movie'*. Tablica *'Actor\_To\_Movie'* sadrži jedan primarni ključ *'id'* te dva vanjska ključa *'actor\_id'* i *'movie\_id'* koji odgovaraju primarnim ključevima tablice *'Actor'* i *'Movie'* respektivno.

Tablica *'Customer\_To\_Movie'* sadrži jedan primarni ključ *'id'* te dva vanjska ključa *'customer\_id'* i *'movie\_id'* koji odgovaraju primarnim ključevima tablice *'Customer'* i *'Movie'* respektivno.

Tablica *'Movie'* je najkompleksnija od svih jer sadrži kombinaciju informacija i vanjskih ključeva.,

**id** – primarni ključ tipa *'Integer'* koji ne smije biti *'NULL'* i čija se vrijednost automatski povećava

**name** – tekstualni tip podataka koji sadrži ime filma

**duration** – tekstualni tip podataka koji sadrži vrijeme projekcije filma

**director\_id** – cjelobrojni tip podataka koji sadrži informaciju o tome koji režiser je režisirao film

**genre\_id** – cjelobrojni tip podataka koji sadrži informaciju o tome kojeg je žanra film

Nadalje, dodana su ograničenja nad atributima *'director\_id'* i *'genre\_id'* koja limitiraju unos tih atributa na vrijednosti primarnih ključeva tablica na koje se referenciraju što su u ovom slučaju *'Director'* i *'Genre'*.

### 4.3 Unos podataka u tablice *'Actor'*, *'Director'*, *'Genre'*, *'Movie'*, *'Customer'*

```

INSERT INTO Actor (name) VALUES ('Keanu Reeves');
INSERT INTO Actor (name) VALUES ('Daniel Radcliffe');
INSERT INTO Actor (name) VALUES ('Jack Black');
INSERT INTO Actor (name) VALUES ('Johnny Depp');
INSERT INTO Actor (name) VALUES ('Elijah Wood');

```

U gore navedenom primjeru koda opisan će biti samo unos u tablicu 'Actor' zbog redundancije vrlo sličnog koda.

'INSERT INTO' je naredba koja nam omogućava unos podataka u tablicu baze podataka.

'Actor' je ime tablice u koju želimo unijeti sljedeće podatke.

U zagradama su navedena imena atributa u koje želimo popuniti vrijednosti navedene dalje u naredbi.

'VALUES' je ključna riječ koja dijeli prvi dio 'INSERT INTO' naredbe od pravih vrijednosti koje unosimo u tablicu.

U zagradama poslije ključne riječi 'VALUES' definirani su podaci koje želimo unijeti u prethodno navedena imena atributa, što su gornjem slučaju 'Keanu Reeves'.

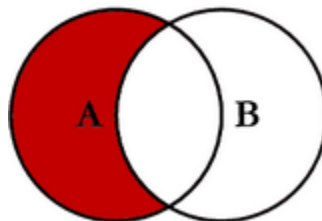
## 4.4 Ispis podataka iz baze podataka

U programskom jeziku Java se nalaze svi upiti koji se koriste u bazi podataka.

Da ponovimo funkcionalnosti naše aplikacije:

-Ispis trenutno dostupnih filmova u videoteci

```
SELECT movie.id, movie.name, movie.duration
FROM movie
LEFT JOIN customer_to_movie
ON movie.id=customer_to_movie.movie_id
WHERE customer_to_movie.movie_id IS NULL
```



Slika 2: 'Left Join'

Ovaj upit dohvaća sve trenutno dostupne filmove u videoteci (one koji nisu već posuđeni). Ovaj upit je složeniji jer se izvršava nad dvije tablice 'Customer\_To\_Movie' i 'Movie'. Upit dohvaća samo ono što se nalazi u tablici 'A' bez presjeka između tablice 'A' i 'B', tablica 'A' u ovom slučaju je 'Movie', a tablica 'B' je 'Customer\_To\_Movie'.

-Ispis liste redatelja svih filmova koji su režisirali filmove u videoteci

```
SELECT * FROM director
```

Ovaj upit dohvaća sve redatelje koji su zapisani u tablici 'Director'. Konkretno, dohvaćaju se svi atributi koji su prisutni u tablici 'Director' pomoću ključne oznake '\*'.

-Ispis liste glumaca koji glume u filmovima u videoteci

```
SELECT * FROM actor
```

Ovaj upit dohvaća sve glumce koji su zapisani u tablici 'Actor'. Konkretno, dohvaćaju se svi atributi prisutni u tablici 'Actor' pomoću ključne oznake '\*'.

-Posudba filmova iz videoteke

Korisniku je prvo prikazana lista korisnika koji su upisani u bazi podataka, nakon korisnikove identifikacije, korisniku je prikazana lista filmova te je ostavljeno njemu na izbor koji film želi posuditi. Korisnik to određuje tako što unosi identifikator filma u konzolu.

```
SELECT * FROM customer
```

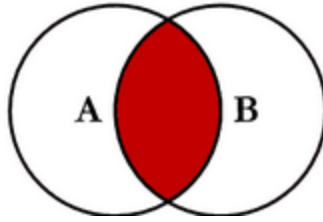
Naredba gore navedena izvlači sve vrijednosti koje su unesene u tablicu 'Customer'.

```
INSERT INTO customer_to_movie (customer_id, movie_id)
VALUES (<customer_id>, <movie_id>);
```

Posudba filmova se radi u bazi podataka tako da unesemo novi redak u tablicu 'Customer\_To\_Movie' koja, ako se prisjetimo je napravljena kako bi se ostvarila veza kardinalnosti 'više na više' između tablica 'Customer' i 'Movie'. Naredba iznad nije sintaktički ispravna već su označene u izlomljenim zagradama vrijednosti koje korisnik odabrao prilikom posuđivanja filma.

-Ispis liste svih filmova koji su posuđeni

```
SELECT movie.id, movie.name, movie.duration
       FROM movie, customer_to_movie
       WHERE movie.id=customer_to_movie.movie_id
```



Slika 3: 'Inner Join'

Ovaj upit je također kompleksniji upit koji se odvija nad dvije tablice 'Movie' i 'Customer\_To\_Movie' i radi gotovo obrnutu stvar koju čini ispis trenutno dostupnih filmova u videoteci. U ovom slučaju tražili smo presjek između dvije tablice i to smo ostvarili pomoću 'WHERE' klauzule koja je zahtjevala uvjet da je 'movie.id' bio jednak 'customer\_to\_movie.movie\_id'.

### -Ispis detalja filma

```
SELECT movie.id, movie.name, movie.duration FROM movie
```

Kako bismo korisniku što bolje iskustvo prilikom korištenja aplikacije moramo prvo ispisati sve filmove koje posjedujemo u videoteci, to se radi pomoću iznad navedenog upita koji ispisuje 'id', 'name' i 'duration' filma iz tablice 'Movie'.

```
SELECT DISTINCT m.id, m.name, m.duration, d.name as director_name,
g.name as genre_name, (SELECT GROUP_CONCAT(a2.name SEPARATOR ',')
FROM Movie m2, Actor_to_movie am2, Actor a2
WHERE m2.id = am2.movie_id
AND am2.actor_id = a2.id
AND m2.id = m.id
GROUP BY m2.id) actors
FROM Movie m, Director d, Genre g, Actor_to_movie am, Actor a
WHERE m.director_id = d.id
AND m.genre_id = g.id
AND m.id = <movieId>
AND am.actor_id = a.id
```

Ovaj upit je i ujedno daleko najkompleksniji jer koristi podupit kako bi se dohvatili svi glumci u filmu ispisani u istome retku odvojeni zarezom. 'DISTINCT' je ključna riječ koja grupira identične retke u ispisu u jedan redak, odnosno ne duplicira dva retka jednakog zapisa. U ovom upitu ispisujemo 'id' filma, ime

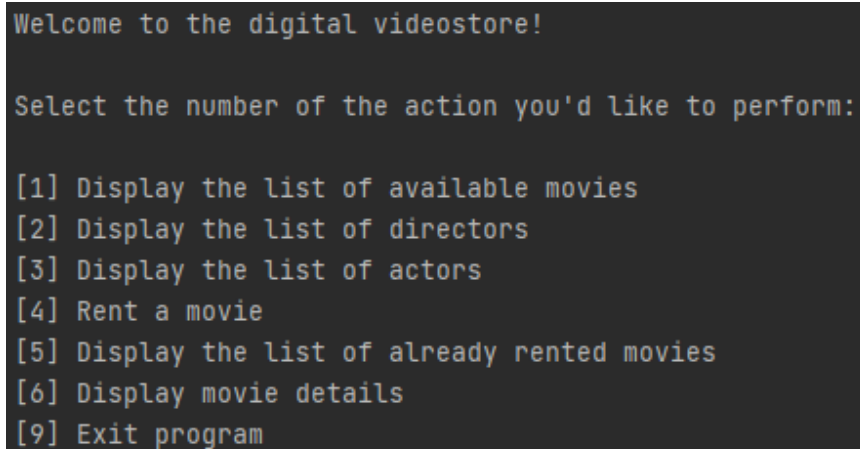
filma, duljinu projekcije filma, ime redatelja, ime žanra, te sve glumce koji glume u filmu.

'*GROUP\_CONCAT*' je funkcija koja spaja podatke iz više redaka u jedno polje. To je '*GROUP BY*' funkcija koja vraća tekstualni zapis ako ta ista grupa sadrži najmanje jednu vrijednost koja nije '*NULL*', u suprotnome vraća '*NULL*' vrijednost. Dodatno omogućava izbor odvojnika između višestrukih vrijednosti koje su spojene u jedan redak. U našem slučaju odvojnik je zarez ','. Također, bitno je napomenuti da je ova vrsta podupita korelirani podupit što znači da on koristi informacije iz vanjskog upita. U ovom primjeru to je '*AND m2.id = m.id*', s time osiguravamo da cijeli podupit izvršava se samo za film koji se po '*id*'-u odgovara filmu iz vanjskog upita, da to nije napravljeno rezultat bi bio potpuno pogrešan. Naredba u izlomljenim zagradama je vrijednost koju korisnik odabrao prilikom odabira filma.

## 5. Opis aplikacijske funkcionalnosti

U ovom dijelu teksta prikazat ćemo korisničko sučelje, objasniti osnovne funkcionalnosti aplikacije te slikovno prikazati gore navedeno korisničko sučelje.

### 5.1 Glavni izbornik digitalne videoteke



```
Welcome to the digital videostore!

Select the number of the action you'd like to perform:

[1] Display the list of available movies
[2] Display the list of directors
[3] Display the list of actors
[4] Rent a movie
[5] Display the list of already rented movies
[6] Display movie details
[9] Exit program
```

Slika 4: Glavno Sučelje

Pri pokretanju aplikacije, u konzoli nam se prikazuje izbornik koji je opredijeljen brojevima 1,2,3,4,5,6,9, te izabirom određenog broja pokreće se dio aplikacije koji izvršava zadatak ovisno o broju koji korisnik unese.

Prilikom upisa bilo kojeg drugog znaka osim brojeva 1,2,3,4,5,6,9, aplikacija izbacuje grešku te nas traži za ponovni unos.

```
Select the number of the action you'd like to perform:

[1] Display the list of available movies
[2] Display the list of directors
[3] Display the list of actors
[4] Rent a movie
[5] Display the list of already rented movies
[6] Display movie details
[9] Exit program

5
That is not an integer number, try again
8
That command was unrecognized, try again
```

Slika 5: Greška

Nakon izvršavanja svake naredbe, aplikacija se vraća na glavni izbornik te korisniku nudi ponovni unos identifikatora naredbe koju korisnik želi izvršiti (Slika 1.).

## 5.2 [1] Ispih svih dostupnih filmova u videoteci

Prilikom unosa broja '1' u konzolu, ispisuje nam se lista svih filmova koji trenutno nisu posuđeni u videoteci, te filmovi koji jesu posuđeni neće biti izlistani prilikom unosa broja '1', to se može vidjeti tako što 'id' prvog filma započinje brojem '2' umjesto '1', što implicira da film sa identifikatorom '1' je već posuđen.



1

Currently available movies are:

-----		
id	Movie name	Duration
-----		
2	Harry Potter and the Philosopher's Stone	2h 39min
4	Lord of the Rings	3h 48min
5	The Shawshank Redemption	2h 22min
6	Saving Private Ryan	2h 49min
7	Django Unchained	2h 45min
8	Once upon a time in...Hollywood	2h 41min
9	Iron Man	2h 6min
-----		

Select the number of the action you'd like to perform:

- [1] Display the list of available movies
- [2] Display the list of directors
- [3] Display the list of actors
- [4] Rent a movie
- [5] Display the list of already rented movies
- [6] Display movie details
- [9] Exit program

Slika 6: Lista filmova

### 5.3 [2] Ispih svih redatelja koji su režisirali filmove u bazi podataka videoteke

Prilikom unosa broja '2' u konzolu, ispisuje nam se lista svih redatelja koji su režisirali filmove koje imamo u bazi podataka.

```
Following is the list of all directors which have directed movies in our database
-----
| id | Director name          |
-----
| 1  | Peter Jackson           |
| 2  | Quentin Tarantino       |
| 3  | Ridley Scott            |
| 4  | Steven Spielberg        |
| 5  | David Yates             |
| 6  | Norman Bates            |
| 7  | Chad Stahelski          |
| 8  | Justin Lin              |
| 9  | Frank Darabont          |
| 10 | Jake Kasdan             |
| 11 | Jon Favreau             |
-----
```

Slika 7: Lista redatelja

### 5.4 [3] Ispih svih glumaca koji su glumili u filmovima u bazi podataka videoteke

Prilikom unosa broja '3' u konzolu, ispisuje nam se lista svih redatelja koji su režisirali filmove koje imamo u bazi podataka.

```
Following is the list of all actors which have starred in movies in our database
-----
| id | Actor name              |
-----
| 1  | Keanu Reeves            |
| 2  | Daniel Radcliffe        |
| 3  | Jack Black              |
| 4  | Johnny Depp             |
| 5  | Elijah Wood             |
| 6  | Morgan Freeman          |
| 7  | Robert Downey, jr.     |
| 8  | Tom Hanks               |
| 9  | Samuel L. Jackson      |
| 10 | Leonardo DiCaprio       |
| 11 | Will Smith              |
| 12 | Brad Pitt               |
-----
```

Slika 8: Lista glumaca

## 5.5 [4] Posudba filma

Prilikom unosa broja '4' u konzolu, ispisuje nam se lista svih korisnika koji se nalaze u bazi podataka te se odvija novi upit koji traži od korisnika da upiše identifikator kojim odabire korisnika kojeg želi koristiti pri posudbi filma. Nakon odabira korisnika, od korisnika se traži da unese identifikator filma koji želi posuditi.

```
4
```

Which customer are you

id	User name	Address	Phone number
1	Luke	10 Hollywood Lane	555-435-828
2	John	15 Bulevard Avenue	555-943-562
3	Mike	8 Times Square	555-341-456

Slika 9: Lista korisnika

U gore navedenom primjeru, odabrali smo korisnika sa identifikatorom '1'.

```
1
```

Currently available movies are:

id	Movie name	Duration
2	Harry Potter and the Philosopher's Stone	2h 39min
4	Lord of the Rings	3h 48min
5	The Shawshank Redemption	2h 22min
6	Saving Private Ryan	2h 49min
7	Django Unchained	2h 45min
8	Once upon a time in...Hollywood	2h 41min
9	Iron Man	2h 6min

Please select movie id you wish to rent out

|

Slika 10: Lista filmova trenutno u videoteci

Film koji je odabran nalazi se pod identifikatorom '2', tj. *'Harry Potter and the Philosopher's Stone'*.

Ako smo unjeli točan identifikator, u konzoli ćemo vidjeti poruku da je film uspješno posuđen.

```
Currently available movies are:
-----
| id | Movie name                | Duration |
-----
| 2 | Harry Potter and the Philosopher's Stone | 2h 39min |
| 4 | Lord of the Rings          | 3h 48min |
| 5 | The Shawshank Redemption  | 2h 22min |
| 6 | Saving Private Ryan        | 2h 49min |
| 7 | Django Unchained           | 2h 45min |
| 8 | Once upon a time in...Hollywood | 2h 41min |
| 9 | Iron Man                   | 2h 6min  |
-----

Please select movie id you wish to rent out
2
Movie has been successfully rented
```

Slika 11: Poruka uspjeha

Kako bismo provjerili uspješnost ove naredbe, opet ćemo pokrenuti naredbu pod identifikatorom '1', te možemo vidjeti da na popisu dostupnih filmova nema filma sa identifikatorom '2' tj. *'Harry Potter and the Philosopher's Stone'*.

```
Select the number of the action you'd like to perform:

[1] Display the list of available movies
[2] Display the list of directors
[3] Display the list of actors
[4] Rent a movie
[5] Display the list of already rented movies
[6] Display movie details
[9] Exit program
1

Currently available movies are:
-----
| id | Movie name                | Duration |
-----
| 4 | Lord of the Rings          | 3h 48min |
| 5 | The Shawshank Redemption  | 2h 22min |
| 6 | Saving Private Ryan        | 2h 49min |
| 7 | Django Unchained           | 2h 45min |
| 8 | Once upon a time in...Hollywood | 2h 41min |
| 9 | Iron Man                   | 2h 6min  |
-----
```

Slika 12: Provjera uspješnosti naredbe

## 5.6 [5] Ispis liste svih posuđenih filmova

Prilikom unosa broja '5' u konzolu, ispisuje nam se lista svih filmova koji su nedostupni za posudbu te se klasificiraju kao posuđeni.

```
5

Currently unavailable movies are:
-----
| id | Movie name                                | Duration |
-----
| 1  | John Wick                                | 1h 41min |
| 2  | Harry Potter and the Philosopher's Stone | 2h 39min |
| 3  | Jumanji                                 | 1h 59min |
-----
```

Slika 13: Lista svih posuđenih filmova

## 5.7 [6] Ispis detalja filma

Prilikom unosa broja '6' u konzolu, ispisuje nam se lista svih filmova te se traži od korisnika ponovni upit identifikatora filma kako bi se dohvatili detalji filma.

```
Movies stored in the database are:
-----
| id | Movie name                                | Duration |
-----
| 1  | John Wick                                | 1h 41min |
| 2  | Harry Potter and the Philosopher's Stone | 2h 39min |
| 3  | Jumanji                                 | 1h 59min |
| 4  | Lord of the Rings                        | 3h 48min |
| 5  | The Shawshank Redemption                 | 2h 22min |
| 6  | Saving Private Ryan                      | 2h 49min |
| 7  | Django Unchained                         | 2h 45min |
| 8  | Once upon a time in...Hollywood          | 2h 41min |
| 9  | Iron Man                                 | 2h 6min  |
-----
8
```

Slika 14: Lista svih filmova u bazi podataka

Prilikom unosa broja '8', tj. *'Once upon a time in...Hollywood'* ispisuju nam se svi detalji o navedenom filmu, te se poziva najkompleksniji upit za dohvaćanje podataka.

```
8
```

Printing out movie details

id	Movie name	Duration	Director name	Genre	Actors
8	Once upon a time in...Hollywood	2h 41min	Quentin Tarantino	Drama	Leonardo DiCaprio,Brad Pitt

Slika 15: Detalji o filmu

## 5.8 [9] Izlaz iz aplikacije

Prilikom unosa broja '9' u konzolu, program se zaustavlja, te na ekranu dobivamo poruku sukladnu tome.

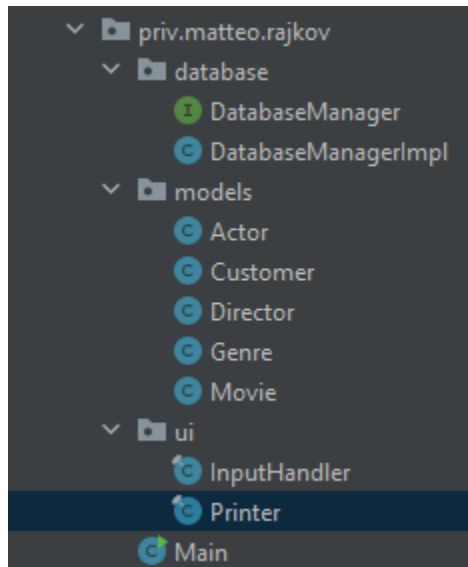
```
Select the number of the action you'd like to perform:

[1] Display the list of available movies
[2] Display the list of directors
[3] Display the list of actors
[4] Rent a movie
[5] Display the list of already rented movies
[6] Display movie details
[9] Exit program
9
Exiting Digital Videostore, thank you for visiting our store.
```

Slika 16: Izlazna poruka

## 6. Opis programskog koda – Java

Odluka za programskim jezikom Java je bio vlastiti interes za navedenim jezikom, te želja za razvijanje vlastitog znanja u objektno orijentiranom programskom jeziku velike rasprostranjenosti u današnjem svijetu.



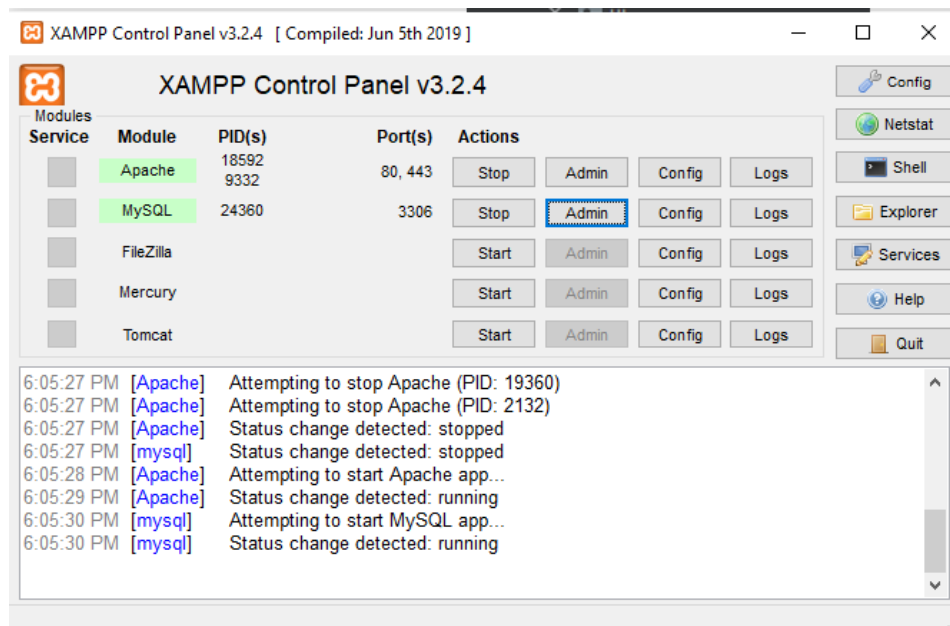
Slika 17: Stablo projekta

Na slici iznad prikazano je stablo projekta te organizacija paketa i klasa u projektu *'Digitalne Videoteke'*.

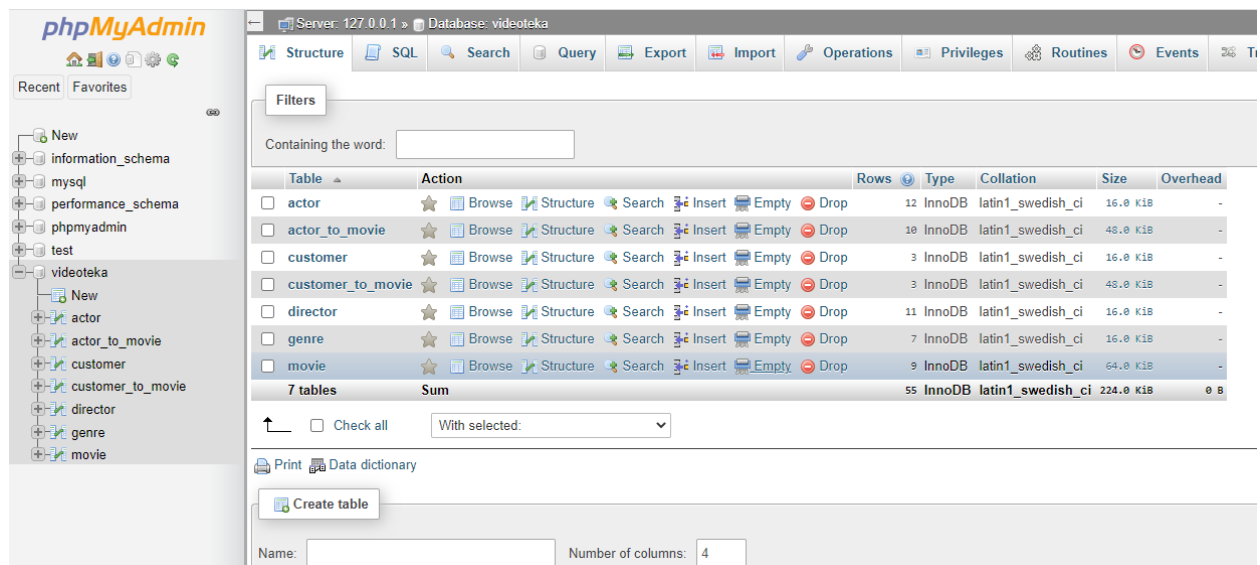
U paketu *'database'* sadržana je klasa *'DatabaseManagerImpl'* te interface *'Database Manager'*.

Klasa *'DatabaseManagerImpl'* sadrži implementaciju pomoću koje se komunicira sa bazom podataka preko *'JavaDatabaseConnectivity-jdbc'*. *'JavaDatabaseConnectivity'* je aplikacijsko sučelje za programski jezik Javu koje definira kako klijent može pristupiti bazi podataka. Dio je *'Java Standard Edition'* platforme od Oracle korporacije. Kako bi projekt funkcionirao i kako bi veza sa bazom podataka funkcionirala bilo je potrebno preuzeti *'mysql connector/J v8.0.23'* koji služi kao biblioteka klasa i programskog koda koji omogućava spajanje sa bazom podataka.

Za ostvarivanje lokalne baze podataka korišten je alat *'xampp'* te je njegov prikaz administracijske konzole *'phpMyAdmin'*.



Slika 18: 'xampp'



Slika 19: 'phpMyAdmin'



## 6.1 Klasa 'DatabaseManagerImpl'

```
public class DatabaseManagerImpl implements DatabaseManager {
    private Connection connection = null;

    @Override
    public void connect() {
        try {
            if (connection == null || connection.isClosed()) {
                connection =
DriverManager.getConnection("jdbc:mysql://localhost/videoteka",
"root", "");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    @Override
    public void releaseResources() {
        if (connection != null) {
            try {
                connection.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

Dio klase '*DatabaseManagerImpl*' prikazan gore služi za povezivanje na bazu podataka te za oslobađanje resursa i zatvaranje veze sa bazom podataka u trenutku kada se program više ne izvodi. Ova klasa implementira metode za dohvaćanje i unos iz baze podataka. Primjer jednog takvog dohvaćanja prikazan je u dolje navedenom programskom kodu.

```

@Override
public List<Director> getAllDirectors() {
    List<Director> resultList = new ArrayList<>();
    try {
        Statement statement = connection.createStatement();
        String query = "SELECT * FROM director";
        ResultSet resultSet = statement.executeQuery(query);
        while (resultSet.next()) {
            int id = resultSet.getInt("id");
            String name = resultSet.getString("name");
            resultList.add(new Director(id, name));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return resultList;
}

```

'@Override' je anotacija koja nam služi za indicaciju da je ova metoda preuzeta iz sučelja *'DatabaseManager'*.

Kao što je definirano u deklaraciji metoda, metoda vraća listu redatelja iz tablice u bazi podataka *'Director'* kako bismo postavili upit u bazu podataka, moramo se poslužiti prethodno navedenom konekcijom s bazom podataka da bismo kreirali *'statement'* objekt i onda pomoću njega izvršili upit nad bazom podataka. *'ResultSet'* sadrži pokazivač na memorijsku lokaciju gdje se nalazi idući redak sa informacijama. Zato je potrebno dohvaćati podatke unutar *'while'* petlje dok metoda *'next'* ne vrati *'false'* što indicira da smo došli do kraja seta podataka koje su dohvaćeni pomoću upita. S obzirom da metoda *'createStatement'* može baciti iznimku *'SQLException'* naš blok koda treba biti izvršen unutar *'try'* i *'catch'*.

## 6.2 Sučelje 'DatabaseManager'

```
public interface DatabaseManager {  
    void connect();  
    void releaseResources();  
    List<Movie> getAllMovies();  
    List<Movie> getAvailableMovies();  
    List<Movie> getUnavailableMovies();  
    List<Director> getAllDirectors();  
    List<Actor> getAllActors();  
    List<Customer> getAllCustomers();  
    boolean rentMovie(int customerId, int movieId);  
    Movie getMovieDetails(int id);  
}
```

Napravljeno je sa svrhom kako bi se sakrili implementacijski detalji klase '*DatabaseManagerImpl*' te služi kao apstrakcija.

### 6.3 Klasa 'InputHandler'

```
public final class InputHandler {

    private static final DatabaseManager databaseManager = new
    DatabaseManagerImpl();

    public static void handleUserMainInput(int commandNumber) {

        if (commandNumber != 9) databaseManager.connect();

        else databaseManager.releaseResources();

        switch (commandNumber) {

            case 1 -> {

                List<Movie> moviesList =
                databaseManager.getAvailableMovies();

                Printer.printMoviesList("Currently available movies
                are: ", moviesList);

            }

            case 2 -> {

                List<Director> directorsList =
                databaseManager.getAllDirectors();

                Printer.printDirectorsList("Following is the list of
                all directors which have directed movies in our database",
                directorsList);

            }

            case 3 -> {

                List<Actor> actorsList =
                databaseManager.getAllActors();

                Printer.printActorsList("Following is the list of all
                actors which have starred in movies in our database", actorsList);

            }

            case 4 -> handleMovieRenting();

            case 5 -> {
```

```

        List<Movie> moviesList =
databaseManager.getUnavailableMovies();

        Printer.printMoviesList("Currently unavailable movies
are:", moviesList);

    }

    case 6 -> {

        List<Movie> allMovies =
databaseManager.getAllMovies();

        Printer.printMoviesList("Movies stored in the database
are: ", allMovies);

        int movieId =
Printer.askForMovieIdInputUntilValid(allMovies);

        Movie movie =
databaseManager.getMovieDetails(movieId);

        Printer.printMovieDetails("Printing out movie
details", movie);

    }

    case 9 -> {

        Printer.printExitingText();

        System.exit(0);

    }

}

}...

```

Klasa gore navedena služi isključivo za obradu unosa podataka od strane korisnika. Kao što je ispisano u programskom kodu, obrađuje slučajeve u kojima korisnik upisuje brojeve 1,2,3,4,5,6,9, gdje se izvršavaju naredbe koje korisnik vidi na konzoli, tj. sučelju. U početnim linijama metode *'If'* provjerava se je li naredba koju je korisnik unio različita od broja '9' koji izlazi van iz programa te se veza sa bazom podataka obustavlja. U klasi *'Main'* detaljnije je opisano kako jedini brojevi koji se *'smiju'* unjeti su brojevi od 1-6 te broj 9. U daljnjim linijama koda koristi se *'switch'* selekcija koja ovisno o unosu korisnika obrađuje određeni set naredbi.

## 6.4 Klasa 'Printer'

```
public final class Printer {  
    ...  
    private static final Scanner scanner = new Scanner(System.in);  
    public static String askForUserInput(String question) {  
        System.out.println(question);  
        return scanner.nextLine();  
    }  
    public static String askForUserInput() {  
        return scanner.nextLine();  
    }  
    public static int askForIntegerInputUntilValid(List<Integer>  
validInputs) {  
        int parsedCommand;  
        do {  
            String scannedLine = scanner.nextLine();  
            try {  
                parsedCommand = Integer.parseInt(scannedLine);  
                for (int validInput : validInputs) if (parsedCommand  
== validInput) return parsedCommand;  
                System.out.println("That command was unrecognized, try  
again");  
            } catch (NumberFormatException e) {  
                System.out.println("That is not an integer number, try  
again");  
            }  
        } while (true);  
    }  
}
```

Klasa *'Printer'* nam služi za ispisivanje te generiranje sučelja koje korisnik vidi u konzoli tokom unosa određenog broja i izvršavanja naredbe. Također ova klasa ima pomoćne metode za traženje unosa naredbe od strane korisnika. Implementaciju takvog ponašanja možemo vidjeti u metodi *'askForUserInput'*. *'askForIntegerInputUntilValid'* metoda prima listu brojeva kao argument i radi iznova akciju pitanja

korisnika da upiše jedan od brojeva koji su poslani kao argument metode. Radnja ove metode će se ponavljati sve dok korisnik ne unese jedan od brojeva koji su poslani kao argument metode, u protivnom metoda će ispisati jednu od pogrešaka korisniku, ovisno o tome što je korisnik unio. Pogreške koje se mogu dogoditi prilikom unosa su da korisnik ne unese broj nego da unese slovo ili znak koji je neprepoznatljiv programu prilikom čega će korisnik dobiti adekvatnu poruku ili alternativno ako korisnik unese broj koji nije u listi validnih brojeva korisnik će primiti adekvatnu poruku i bit će pitao za ponovni unos.

```
public static void printMainMenu() {  
    System.out.println("\nSelect the number of the action you'd  
like to perform:\n");  
    System.out.println("[1] Display the list of available  
movies");  
    System.out.println("[2] Display the list of directors");  
    System.out.println("[3] Display the list of actors");  
    System.out.println("[4] Rent a movie");  
    System.out.println("[5] Display the list of already rented  
movies");  
    System.out.println("[6] Display movie details");  
    System.out.println("[9] Exit program");  
}
```

U gore navedenoj funkciji prikazan je ispis glavnog izbornika aplikacije.

## 6.5 Klasa '*Main*'

```
public class Main {

    public static void main(String[] args) {
        Printer.printHelloText();
        do {
            Printer.printMainMenu();
            int numberCommand;
            try {
                numberCommand =
Printer.askForIntegerInputUntilValid(Arrays.asList(1, 2, 3, 4, 5, 6,
9));

                InputHandler.handleUserMainInput(numberCommand);
            } catch (NumberFormatException e) {
                Printer.printErrorText();
            }
        } while (true);
    }
}
```

U klasi '*Main*' se nalazi metoda '*main*' koja ujedno označava početak programa, prilikom kojeg se ispisuje tekst dobrodošlice korisniku i glavni izbornik. Neposredno iza toga pita se korisnika za njegov početni izbor radnje glavnog izbornika. Kao što je prikazano iznad, unos može biti od 1-6 i broj 9. Ako je unos validan poziva se klasa '*InputHandler*' u kojoj smo prethodno vidjeli da se nalazi selekcija '*switch*'

koja obrađuje pojedinu od navedenih naredbi korisnika. Program se izvršava u beskonačnoj petlji i završava isključivo kada je unesen broj '9'.



## 7. Zaključak

Aplikacija Digitalna Videoteka je minimalistični prikaz jedne stvarne videoteke. Podaci i atributi u bazi podataka su bili prikazani na jednostavan način zbog potencijalno velikog opsega koji bi aplikacija mogla poprimiti ako bi se išlo u stvarnu realizaciju i rješavanje problema. Aplikaciji nedostaju ključne funkcionalnosti da bi mogla stvarno zaživjeti, neke od tih funkcionalnosti su prijava korisnika, registracija novih korisnika, određivanje datuma do kojeg je film posuđen, izmjena podataka o korisniku i slično. Neovisno o navedenom, smatram da je izvedba aplikacije dovoljno jednostavna za korištenje korisnika visokog tehničkog znanja ujedno i niskog tehničkog znanja, te je izvrstan primjer na kojem se mogu vježbati vještine izrade baza podataka i programskog koda. Aplikacija nema stvarno korisničko sučelje već se ono izrađeno pomoću konzole i konzolnog prikaza. Poboljšanje na aplikaciji bi se moglo napraviti u ovoj sferi, odnosno dalo bi se napraviti pravo korisničko sučelje na kakvo su današnji korisnici navikli. Za izradu aplikacije korišten je programski jezik Java, JDBC, te MySQL Connector/J biblioteka za ostvarenje i uspostavu veze između klijentskog dijela aplikacije i baze podataka.

## 8. Literatura

- [1]<https://www.w3schools.com/sql/default.asp>
- [2][https://www.w3schools.com/java/java\\_oop.asp](https://www.w3schools.com/java/java_oop.asp)
- [3]<https://www.javatpoint.com/java-string-to-int#:~:text=We%20can%20convert%20String%20to,returns%20instance%20of%20Integer%20class>
- [4]<https://dzone.com/articles/java-string-format-examples>
- [5]<https://www.techrepublic.com/article/sql-basics-query-multiple-tables/>
- [6]<https://online.visual-paradigm.com/drive/#diagramlist:proj=0&new=ERDiagram>

## 9. Popis slika

- [1] ER Dijagram
- [2] *'Left Join'*
- [3] *'Inner Join'*
- [4] Glavno sučelje
- [5] Greške
- [6] Lista filmova
- [7] Lista redatelja
- [8] Lista glumaca
- [9] Lista korisnika
- [10] Lista filmova trenutno u videoteci
- [11] Poruka uspjeha
- [12] Provjera uspješnosti naredbe
- [13] Lista svih posuđenih filmova
- [14] Lista svih filmova u bazi podataka
- [15] Detalji o filmu
- [16] Izlazna poruka
- [17] Stablo projekta
- [18] *'xampp'*
- [19] *'phpMyAdmin'*