

# POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Gestionale  
Classe L8 - Ingegneria dell'Informazione



## **Sviluppo di un software per la generazione automatica di una maratona cinematografica**

**Relatore**  
Prof. Fulvio Corno

**Candidato**  
Grandi Emanuele  
269431

A.A. 2021/2022



## Sommario

1	Proposta di progetto.....	1
1.1	Studente proponente .....	1
1.2	Titolo della proposta.....	1
1.3	Descrizione del problema proposto .....	1
1.4	Descrizione della rilevanza gestionale del problema.....	1
1.5	Descrizione dei data-set per la valutazione.....	2
1.6	Descrizione preliminare degli algoritmi coinvolti.....	3
1.7	Descrizione preliminare delle funzionalità previste per l'applicazione software.....	4
2	Descrizione del problema affrontato .....	4
3	Descrizione del data-set utilizzato per l'analisi .....	5
3.1	Introduzione .....	5
3.2	Tabella Data_tot .....	6
74	Descrizione ad alto livello delle strutture dati e degli algoritmi utilizzati .....	7
4.1	Descrizione delle strutture dati: i packages e le classi.....	7
4.1.1	it.polito.tdp.PF .....	7
4.1.2	it.polito.tdp.PF.db .....	8
4.1.3	it.polito.tdp.PF.model .....	8
4.2	Descrizione degli algoritmi utilizzati.....	9
4.2.1	Sezione 1: Ricerca .....	9
4.2.2	Sezione 2: Ricorsione .....	11
5	Interfaccia e video dimostrativo.....	13
5.1	Sezione Ricerca .....	13
5.2	Sezione Maratona.....	14
6	Risultati sperimentali e esempi .....	15
6.1	Esempio 1 .....	15
6.2	Esempio 2 .....	16
6.4	Esempio 1 .....	17
6.5	Esempio 2 .....	18
6.6	Esempio Playlist.....	19
7	Valutazioni sui risultati ottenuti.....	19

# **1 Proposta di progetto**

## **1.1 Studente proponente**

s269431 Grandi Emanuele

## **1.2 Titolo della proposta**

Sviluppo di un Software per la generazione automatica di una maratona cinematografica

## **1.3 Descrizione del problema proposto**

L'obiettivo del software è quello di permettere all'utente di generare automaticamente una maratona cinematografica, elaborata rispettando i vincoli dettati dall'utente stesso e comunicati al software mediante l'interfaccia applicativa.

In particolare, fornisce all'utente la possibilità di determinare una durata massima (in termini di minuti per la ricerca semplice e ore per la maratona) della sequenza dei Film (vincolo di solito assente nella ricerca dei Film nelle piattaforme di streaming più utilizzate); i generi privilegiati e il periodo di uscita dei titoli.

E' possibile, inoltre, definire un vincolo sulla valutazione minima della critica, nell'elaborazione della maratona desiderata.

Il software permette, per quanto riguarda le Serie Tv, di esprimere una preferenza per una determinata durata media degli episodi.

Funzioni implementate sono quelle di ricerca semplice (senza generare una maratona) di Film e Serie Tv (forniti i vincoli da parte dell'utente) e di aggiunta dei Film e Serie a una playlist personale "Guarda più tardi", per tenere traccia dei titoli da vedere in un secondo momento.

## **1.4 Descrizione della rilevanza gestionale del problema**

Il software si focalizza, in primo luogo, su una globale ottimizzazione dei tempi.

L'utente, infatti, grazie al vincolo sulla durata massima dei Film, delle Serie e della maratona, potrà effettuare una ricerca su misura, in base al proprio programma

giornaliero, al proprio tempo libero e in generale alle proprie esigenze in termini di tempistiche.

I titoli proposti dal software si incastreranno perfettamente nelle finestre temporali stabilite dal fruitore, evitando di dover interrompere e metà la visione di un film o di perdere tempo dietro a Serie Tv dagli episodi troppo lunghi.

Oltre a fornire informazioni generali sui titoli proposti, il software segnalerà all'utente la piattaforma di streaming in cui il titolo è attualmente disponibile (tra quelle più utilizzate), semplificando ulteriormente la ricerca.

Il processo di generazione della maratona cinematografica, infine, porterà chiaramente alla generazione di molteplici possibilità per l'utente, che potrà per esempio stabilire una valutazione minima (in termini di voto della critica e del pubblico) dei titoli che la compongono.

### **1.5 Descrizione dei data-set per la valutazione**

Il software si appoggerà a un data-set da me costruito ed elaborato a partire da alcuni data-set forniti dalla piattaforma Kaggle.

In particolare, sono stati utilizzati data-set che contengono informazioni su Film e Serie Tv presenti nelle principali piattaforme di streaming (Netflix, Disney +, Amazon Prime video), combinati con altri set di dati, per poter fornire all'utente informazioni sui titoli come: una breve descrizione del titolo, anno di uscita, ratings (dal Sito IMDb), informazioni sul genere e sul tipo di contenuti, le fondamentali informazioni sulla durata e la presenza o meno su una data piattaforma.

I dataset sono aggiornati a Novembre 2021.

Il database completo è presente nel repository (imdb.sql) ed è composto da 5 tabelle: netflix\_titles, amazon\_prime\_titles, disney\_plus\_titles, imdb\_copy e l'ultima data\_tot, che rappresenta il data-set effettivo utilizzato per l'applicazione (contenente però solamente i film e non le serie tv, poiché non considerate nella creazione della maratona), frutto della combinazione degli altri 4.

**Le righe totali del database (data\_tot) sono 1260 e gli attributi sono:**

- Nome;
- Anno di Uscita;
- Valutazione Imdb;
- Durata in minuti;
- Genere (uno o più);
- Livello di Nudità;
- Livello di Violenza;
- Livello di Alcool;
- Livello di Paura;
- Breve Descrizione del Titolo;
- Piattaforma in cui è presente (N o P o D).

### **1.6 Descrizione preliminare degli algoritmi coinvolti**

La prima parte sarà costituita da algoritmi di ricerca per la visualizzazione di record e l'elaborazione dei dati estratti.

Per la seconda parte, Il software lavorerà sul database costruito con l'obiettivo di costruire una sequenza di titoli, vincolata alle preferenze dell'utente.

Per fare ciò si baserà su un algoritmo ricorsivo, che a partire da un titolo casuale che rispetti tutti i filtri, genererà una maratona che dovrà essere ottimizzata sulla base della durata massima stabilita, della valutazione minima dei titoli richiesta dall'utente e su un serie di ulteriori parametri, che si potrà far variare a piacere (livello di paura del Film, livello di violenza...).

In tal modo, ogni volta che l'utente vorrà generare una maratona, il software cercherà di variare la proposta se possibile.

Si tratta quindi di un problema dello zaino (knapsack problem).

## **1.7 Descrizione preliminare delle funzionalità previste per l'applicazione software**

L'utente si troverà davanti a una serie di vincoli da determinare nella ricerca o nella generazione della maratona ad hoc.

Nella prima tab, l'utente potrà scegliere tramite una combobox se includere Film o Serie Tv, spuntare le piattaforme desiderate tramite checkbox e dopodichè impostare una serie di filtri, quali il genere (combobox), il periodo di uscita (2 combobox) e la durata massima dei film (durata massima media degli episodi per le Serie Tv) per la ricerca dei Titoli.

Il software evidenzia il titolo con la valutazione migliore (in caso di valutazione uguale, viene scelto il primo in ordine alfabetico) e fornisce breve descrizione (in inglese) di esso.

Sempre nella prima tab, dopo aver effettuato la ricerca, l'utente potrà aggiungere uno o più titoli, tra quelli proposti, ad una playlist "Guarda più Tardi".

Nella seconda Tab, quella predisposta alla creazione di una maratona, oltre agli stessi vincoli presenti nella prima parte, sono presenti degli slider per stabilire il livello massimo di violenza, paura, nudità e alcool, la durata totale massima in ore della maratona e la valutazione minima dei titoli che ne faranno parte.

L'applicazione mostrerà la valutazione media e la durata totale della maratona proposta.

Per quanto riguarda la creazione della maratona ho preferito escludere le serie Tv, in quanto risulta poco sensato a mio parere inserire uno o più episodi di una serie all'interno di una maratona, in sequenza con dei Film.

I risultati proposti dal software mostreranno il nome del Film o della Serie tv, l'anno di uscita e la Piattaforma in cui sono disponibili (N, P, D) (Agg. Novembre 2021).

## **2 Descrizione del problema affrontato**

Il problema affrontato è principalmente di ottimizzazione.

L'idea parte dall'osservazione del fatto che sulle principali piattaforme di streaming di Film e Serie tv non è possibile effettuare in modo semplice ed immediato una ricerca

per durata massima dei Titoli, così come risulta macchinoso limitare la ricerca ad un determinato periodo temporale di uscita dei Titoli.

Il software si pone l'obiettivo di adattarsi alle esigenze degli utenti e ai loro impegni giornalieri, che limitano le finestre temporali in cui potersi godere un buon Film o cominciare una Serie Tv.

Attraverso un algoritmo ricorsivo, l'applicazione genera in automatico una maratona ad hoc, a partire dalle preferenze dell'utente, che stabilirà la durata massima totale, il periodo di uscita del Film, la valutazione minima dei Titoli che ne faranno parte e regolerà alcuni parametri come il livello di Violenza o Paura del Film.

L'applicazione ovviamente non è pensata come un elemento sostitutivo di tali piattaforme, bensì come un elemento di supporto.

L'utente sarà informato della piattaforma in cui sono presenti i titoli, velocizzando la ricerca e la successiva fruizione.

Infine, ho ritenuto non significativa l'implementazione della ricerca per Attori o Registi, sia per problemi di completezza dei dataset, sia perché tale tipo di ricerca è facilmente eseguibile sulle piattaforme di streaming.

### **3 Descrizione del data-set utilizzato per l'analisi**

#### **3.1 Introduzione**

Per creare il software sono state utilizzate parti di un database fornito dal sito Kaggle, il quale afferma che è possibile usufruirne liberamente.

I dati forniti erano tutti in formato CSV, di conseguenza è stata svolta una conversione in formato SQL per permetterne l'importazione attraverso il software HeidiSQL.

A tal proposito è stato creato nell'ambiente di HeidiSQL un nuovo database chiamato "imdb" contenente le tabelle e i parametri necessari ad accogliere il contenuto attraverso il comando import CSV.

Al fine di non affaticare l'applicazione con eccessivi calcoli si è preferito snellire il database eliminando dati errati e/o ridondanti, nonché oggetti inutili ai fini del software.

Il database "imdb" è composto da 5 tabelle, come scritto in precedenza.

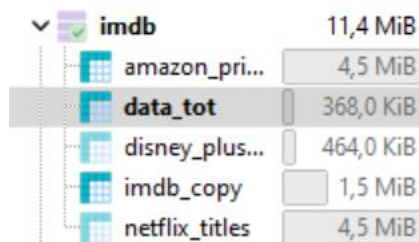
Le tabelle imdb\_copy, netflix\_titles, amazon\_prime\_titles, \_disney\_plus\_titles sono quelle originarie dal sito Kaggle (utilizzate per la ricerca semplice di titoli), tuttavia al fine di non affaticare l'applicazione con eccessivi calcoli si è preferito snellire il database



eliminando dati errati e/o ridondanti, nonché oggetti inutili ai fini del software (registi, attori, Paese, id Titoli, anno aggiunta nelle piattaforme) combinando i dati utili nella tabella data\_tot (più semplice da utilizzare per la ricorsione).

Per questa ragione descriverò solamente tale tabella, riassunto delle informazioni utili delle altre.

- Amazon\_prime\_titles
- Data\_tot
- Disney\_plus\_titles
- Imdb\_copy
- Netflix\_titles



File	Dimensione
imdb	11,4 MiB
amazon_pri...	4,5 MiB
<b>data_tot</b>	368,0 KiB
disney_plus...	464,0 KiB
imdb_copy	1,5 MiB
netflix_titles	4,5 MiB

Figura 3.1 Data-set completo

### 3.2 Tabella Data\_tot

La tabella è composta da 1260 elementi, che rappresentano i Film presenti nel database

Di seguito sono elencate le colonne contenenti informazioni utili:

- 1) \item “Nome” rappresenta il titolo del Film in inglese.
- 2) \item “Anno” identifica univocamente insieme al Nome il Titolo.
- 3) \item “Voto” contiene la valutazione Imdb del Titolo (da 2 a 9)
- 4) \item “Durata” indica durata in minuti del Film.
- 5) \item “Genere” include il genere o i generi in cui il Titolo rientra.
- 6) \item “Nud” contiene un valore da 0 a 3 che indica la presenza di Nudità nel Film.  
(0 nessuna, 1 blanda, 2 moderata, 3 significativo)
- 7) \item “Viol” contiene un valore da 0 a 3 che indica la presenza di Violenza nel Film.  
(0 nessuna, 1 blanda, 2 moderata, 3 significativa)
- 8) \item “Alc” contiene un valore da 0 a 3 che indica la presenza di Alcool nel Film. (0 nessuna, 1 blanda, 2 moderata, 3 significativa)
- 9) \item “Fri” contiene un valore da 0 a 3 che indica il livello di Paura che solitamente il Titolo suscita alla visione . (0 nessuno, 1 blando, 2 moderato, 3 significativo)

- 10) item “Descr” contiene una breve descrizione in inglese del Film.
- 11) item “Piattaf” indica la piattaforma in cui è presente il Titolo (N netflix, P prime video, D disney +).

#	Nome	Tipo di dati
1	Nome	VARCHAR
2	Anno	INT
3	Voto	INT
4	Durata	INT
5	Genere	VARCHAR
6	Nud	INT
7	Viol	INT
8	Alc	INT
9	Fri	INT
10	Descr	VARCHAR
11	Piattaf	VARCHAR

Figura 3.2 Tabella data\_tot

## 4 Descrizione ad alto livello delle strutture dati e degli algoritmi utilizzati

L'applicazione è stata sviluppata in linguaggio java seguendo il pattern MVC (Model-View-Controller) ed il pattern DAO (Data-Access-Object) tramite i quali sono stati separati l'accesso al database, i processi algoritmici per la manipolazione dei dati e l'interfaccia utente.

### 4.1 Descrizione delle strutture dati: i packages e le classi

I packages sono utili a tenere separate le varie fasi, abbiamo infatti tre packages: uno per il model, uno per il controller e uno per il database.

#### 4.1.1 it.polito.tdp.PF

Il package si occupa dell'interfaccia utente e della procedura di avvio dell'applicazione tramite tre classi:

**EntryPoint:** necessario per l'avvio dell'applicazione stessa

**Main:** gestisce avvio interfaccia nel quale verranno inseriti i dati necessari.

**FXMLController**: riceve dati in Input e collega l'interfaccia con la logica applicativa tramite l'oggetto model. I occupa quindi di interpretare e validare i dati inseriti.

In questa classe avranno luogo gli algoritmi che andranno a comporre l'UI che mostrerà anche gli output.

#### 4.1.2 it.polito.tdp.PF.db

Questo package ci permette di effettuare la connessione alla base dati e di interrogare la suddetta attraverso query scritte in linguaggio SQL.

E' composto dalla classe **DBConnect**, **IMDBDao** :

- **DBConnect** In questa classe avviene la connessione con il database tramite pooling che permette di risparmiare tempo di apertura e chiusura di una connessione grazie alla classe HikariDataSource.
- **IMDBDao** Questa classe riceve i parametri dal Model (che nella maggior parte dei casi vengono passati dal Controller) e interroga il database tramite query SQL. Crea delle strutture dati idonee all'organizzazione dei dati raccolti e li restituisce al Model.

#### 4.1.3 it.polito.tdp.PF.model

Questo package contiene la logica applicativa e svolge la maggior parte delle operazioni e dei calcoli.

È composto da 2 classi:

- **Model** È la classe portante del progetto, qui sono contenuti tutti i dati e le strutture necessari al corretto svolgimento dei processi. Il model collega il database e l'interfaccia utente e al suo interno viene creato il grafo semplice che rappresenta i film appartenenti allo stesso genere e che rispettano tutti i filtri stabiliti dall'utente (periodo temporale, valutazione minima, livelli di violenza...).
- **Titolo** I cui attributi sono le 11 colonne della tabella data\_tot.

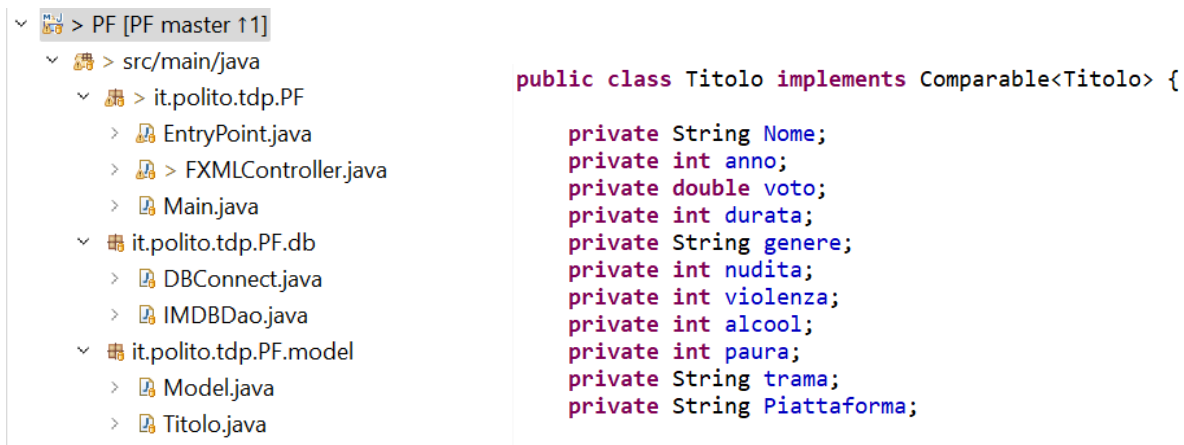


Figura 4.1 Packages e relative classi

Classe Titolo

## 4.2 Descrizione degli algoritmi utilizzati

### 4.2.1 Sezione 1: Ricerca

Nella prima tab, l'utente potrà scegliere tramite una combobox se includere Film o Serie Tv, spuntare le piattaforme desiderate tramite checkbox e dopodichè impostare una serie di filtri, quali il genere (combobox), il periodo di uscita (2 combobox) e la durata massima dei film (durata massima media degli episodi per le Serie Tv) per la ricerca dei Titoli.

Sempre nella prima tab, dopo aver effettuato la ricerca, l'utente potrà aggiungere uno o più titoli, tra quelli proposti, ad una playlist "Guarda più Tardi".

L'utente in questa prima parte può scegliere se effettuare la ricerca per genere e per periodo o solo per uno dei due filtri, ciò ha reso necessario creare più metodi nel IMDBDao.

## Nel DAO

```
public List<Titolo> listTitoliNetflixGenAnno(String tipo, String genere, int anno, int durata, int anno2){
    String sql = "SELECT i.NOME, i.ANNO, i.VOTO, i.DURATA, i.GENERE, i.NUD, i.VIOL, i.ALC, i.FRI, n.description, n.Piattaforma "
        + "FROM imdb_copy i, netflix_titles n "
        + "WHERE i.NOME=n.title && i.TIPO=? && i.GENERE LIKE CONCAT('%',?, '%') && i.ANNO=? && i.ANNO<=? && i.DURATA<=?";
    List<Titolo> result = new ArrayList<Titolo>();
    Connection conn = DBConnect.getConnection();

    try {
        PreparedStatement st = conn.prepareStatement(sql);
        st.setString(1, tipo);
        st.setString(2, genere);
        st.setInt(3, anno);
        st.setInt(4, anno2);
        st.setInt(5, durata);
        ResultSet res = st.executeQuery();
        while (res.next()) {
            Titolo s = new Titolo(res.getString("i.NOME"), res.getInt("i.ANNO"), res.getDouble("i.VOTO"),
                res.getInt("i.DURATA"), res.getString("i.GENERE"), res.getInt("i.NUD"), res.getInt("i.VIOL"), res.getInt("i.ALC"),
                res.getInt("i.FRI"), res.getString("n.description"), res.getString("n.Piattaforma"));
            result.add(s);
        }
        conn.close();
        return result;
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}

public List<Titolo> listTitoliNetflixGen(String tipo, String genere, int durata){
    String sql = "SELECT i.NOME, i.ANNO, i.VOTO, i.DURATA, i.GENERE, i.NUD, i.VIOL, i.ALC, i.FRI, n.description, n.Piattaforma "
        + "FROM imdb_copy i, netflix_titles n "
        + "WHERE i.NOME=n.title && i.TIPO=? && i.GENERE LIKE CONCAT('%',?, '%') && i.DURATA<=?";
    List<Titolo> result = new ArrayList<Titolo>();
    Connection conn = DBConnect.getConnection();

    try {
        PreparedStatement st = conn.prepareStatement(sql);
        st.setString(1, tipo);
        st.setString(2, genere);

        st.setInt(3, durata);
        ResultSet res = st.executeQuery();
        while (res.next()) {
            Titolo s = new Titolo(res.getString("i.NOME"), res.getInt("i.ANNO"), res.getDouble("i.VOTO"),
                res.getInt("i.DURATA"), res.getString("i.GENERE"), res.getInt("i.NUD"), res.getInt("i.VIOL"),
                res.getInt("i.ALC"), res.getInt("i.FRI"), res.getString("n.description"), res.getString("n.Piattaforma"));
            result.add(s);
        }
        conn.close();
        return result;
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}

public List<Titolo> listTitoliNetflixAnno(String tipo, int anno, int durata, int anno2){
    String sql = "SELECT i.NOME, i.ANNO, i.VOTO, i.DURATA, i.GENERE, i.NUD, i.VIOL, i.ALC, i.FRI, n.description, n.Piattaforma "
        + "FROM imdb_copy i, netflix_titles n "
        + "WHERE i.NOME=n.title && i.TIPO=? && i.ANNO=? && i.ANNO<=? && i.DURATA<=?";
    List<Titolo> result = new ArrayList<Titolo>();
    Connection conn = DBConnect.getConnection();

    try {
        PreparedStatement st = conn.prepareStatement(sql);
        st.setString(1, tipo);

        st.setInt(2, anno);
        st.setInt(3, anno2);
        st.setInt(4, durata);
        ResultSet res = st.executeQuery();
        while (res.next()) {
            Titolo s = new Titolo(res.getString("i.NOME"), res.getInt("i.ANNO"),
                res.getDouble("i.VOTO"), res.getInt("i.DURATA"), res.getString("i.GENERE"),
                res.getInt("i.NUD"), res.getInt("i.VIOL"), res.getInt("i.ALC"), res.getInt("i.FRI"),
                res.getString("n.description"), res.getString("n.Piattaforma"));
            result.add(s);
        }
        conn.close();
        return result;
    } catch (SQLException e) {
        e.printStackTrace();
        return null;
    }
}
```

3 metodi ripetuti anche per le altre 2 tabelle: Amazon\_Prime\_titles e Disney\_plus\_titles.

### 4.2.2 Sezione 2: Ricorsione

La ricorsione si basa sulla creazione di un grafo semplice i cui vertici sono tutti quei film che durano meno della durata massima della maratona impostata dall'utente e che rispettano tutti i filtri impostati (vertici ottenuti tramite metodo nel DAO, che lavora sul data-set finale data-tot). I vertici del grafo saranno poi tutti collegati fra loro e verranno utilizzati per la creazione della maratona ottimale a partire da uno di essi scelto in modo casuale (Math.random), nel metodo Crea Cammino (), il quale si appoggia sul metodo ricorsivo cerca().

## NEL DAO

```
public List<Titolo>getVertici(String genere,double nud,double viol,double alc,double fri,double vot,int anno1,int anno2){

String sql="SELECT d.Nome,d.Anno,d.Voto,d.Durata,d.Genere,d.Nud,d.Viol,d.Alc,d.Fri,d.Descr,d.Piattaf "
+ "FROM data_tot d "
+ "WHERE d.Nud<=? && d.Viol<=?&& d.Alc<=? && d.Fri<=? && d.Genere LIKE CONCAT('%',?,'%') && d.Voto>=? && d.Anno>="

List<Titolo> result = new ArrayList<Titolo>();
Connection conn = DBConnect.getConnection();

try {
    PreparedStatement st = conn.prepareStatement(sql);
    st.setDouble(1, nud);
    st.setDouble(2, viol);

    st.setDouble(3, alc);
    st.setDouble(4, fri);
    st.setString(5, genere);
    st.setDouble(6, vot);
    st.setInt(7, anno1);
    st.setInt(8, anno2);
    ResultSet res = st.executeQuery();
    while (res.next()) {

        Titolo s = new Titolo(res.getString("d.Nome"),res.getInt("d.Anno"),res.getDouble("d.Voto")
        ,res.getInt("d.Durata"),res.getString("d.Genere"),res.getInt("d.Nud"),res.getInt("d.Viol"),
        res.getInt("d.Alc"),res.getInt("d.Fri"),res.getString("d.Descr"),res.getString("d.Piattaf"));
        result.add(s);
    }
    conn.close();
    return result;
} catch (SQLException e) {
    e.printStackTrace();
    return null;
}
```

## NEL MODEL

```
public void CreaGrafo(String genere,double nud,double viol,double alc,double fri,List<String>piattaforme,double vot,int duratatot,int anno1,int anno2) {
    this.grafo=new SimpleGraph<>(DefaultEdge.class);
    List<Titolo>titolicompl=new ArrayList<Titolo>();

    titolicompl=dao.getVertici(genere, nud, viol, alc, fri,vot,anno1,anno2);
    titolipiatt=new ArrayList<Titolo>();

    for(Titolo t:titolicompl) {
        if(piattaforme.contains(t.getPiattaforma())&& t.getDurata()<=duratatot) {
            titolipiatt.add(t);
        }
    }

    Graphs.addALLVertices(grafo, titolipiatt);

    for(Titolo t1:titolipiatt) {
        for(Titolo t2: titolipiatt) {
            if(!t1.equals(t2)) {
                Graphs.addEdgeWithVertices(grafo, t1, t2);
            }
        }
    }
}
```

## Metodo CreaCammino ()

```
public List<Titolo> CreaCammino(int duratatot){  
    finale =new ArrayList<Titolo>() ;  
    List<Titolo>parziale=new ArrayList<>();  
  
    if (titolipiatt.size()==0) {  
        return null;  
    }  
    int indice=(int)(Math.random()*titolipiatt.size());  
    Titolo Part=titolipiatt.get(indice);  
  
    connesse=new LinkedList<Titolo>(Graphs.neighborListOf(this.grafo,Part));  
    parziale.add(Part);  
    cerca(parziale,duratatot,connesse);  
    return finale;  
  
}
```

## Metodo cerca ()

```
private void cerca(List<Titolo> parziale, int duratatot, List<Titolo>connesse) {  
    int tottitoli=parziale.size();  
    if(tottitoli>finale.size()|| finale==null) {  
        finale=new ArrayList<>(parziale);  
    }  
    for(Titolo t:connesse) {  
        if(!parziale.contains(t)) {  
            parziale.add(t);  
            if(aggiuntavalida(parziale,duratatot,t)) {  
                cerca(parziale,duratatot,connesse);  
            }  
            parziale.remove(t);  
        }  
    }  
}
```

## 5 Interfaccia e video dimostrativo

### 5.1 Sezione Ricerca

Ricerca

Maratona

TIPO

PIATTAFORMA

☐ Netflix

☐ Prime Video

☐ Disney +

RICERCA

Ricerca Film e Serie Tv

GENERE

DA

A

DURATA MAX FILM/DURATA MAX MEDIA EPISODI (MINUTI)

RESETTA

Valutazione Migliore

Descrizione

Playlist "Guarda più Tardi"

TITOLO

AGGIUNGI

RESETTA PLAYLIST



## 5.2 Sezione Maratona

Ricerca

Maratona

### Maratona Cinematografica

NUDITA'

0 1 2 3

PAURA

0 1 2 3

ALCOOL

0 1 2 3

VIOLENZA

0 1 2 3

GENERE

DURATA MAX (ORE)

VOTO MIN

PIATTAFORMA

☐ Netflix

☐ Prime Video

☐ Disney +

DA

A

GENERA

RESET

Valutazione Media

Durata Totale

Qui potrete trovare un video dimostrativo: <https://www.youtube.com/watch?v=wbMAJAl2gM>

## 6 Risultati sperimentali e esempi

### 6.1 Esempio 1

#### Ricerca

Input:

**Ricerca Film e Serie Tv**

TIPO: Film

PIATTAFORMA: ☒ Netflix, ☒ Prime Video, ☒ Disney +

GENERE: Comedy

DA: 1950 A: 1980

DURATA MAX FILM/DURATA MAX MEDIA EPISODI (MINUTI): 120

RICERCA RESETTA

Output:

Monty Python and the Holy Grail (1975) (N)  
Popeye (1980) (N)  
The Jungle Book (1967) (N)  
Harold and Maude (1971) (P)  
Annie Hall (1977) (P)  
Young Frankenstein (1974) (P)  
The Graduate (1967) (P)  
The Rocky Horror Picture Show (1975) (P)  
Robin Hood (1973) (D)

**Valutazione Migliore: Monty Python and the Holy Grail (8.2)**

**Descrizione**

The Monty Python comedy clan skewers King Arthur and his Knights of the Round Table as they quest far and wide for the Holy Grail.

#### Valutazione

**Pur scegliendo tutte e 3 le piattaforme, una durata massima di 120 minuti (2 ore) e un genere molto comune(Comedy), il numero di Film consigliati è relativamente limitato.**

**Questo è dovuto al periodo scelto (1950-1980), che limita molto la scelta all'interno del data-set.**

## 6.2 Esempio 2

### Ricerca

Input:

TIPO

Film

PIATTAFORMA

☒ Netflix

☒ Prime Video

☒ Disney +

RICERCA

Ricerca Film e Serie Tv

GENERE

Comedy

DA

1990

A

2021

DURATA MAX FILM/DURATA MAX MEDIA EPISODI (MINUTI)

120

RESETTA

Output:

My Little Pony: A New Generation (2021) (N)

The Starling (2021) (N)

Grown Ups (2010) (N)

Afterlife of the Party (2021) (N)

A Cinderella Story (2004) (N)

Letters to Juliet (2010) (N)

Mars Attacks! (1996) (N)

The Interview (2014) (N)

The Nutty Professor (1996) (N)

He's All That (2021) (N)

The Kissing Booth 3 (2021) (N)

Valutazione Migliore: Bo Burnham: Inside (8.7)

Descrizione

A new comedy special shot and performed by Bo Burnham, alone, over the course of the past year.

### Valutazione

Modificando solamente il periodo rispetto all'esempio precedente, la lista di titoli proposta aumenta notevolmente (1990-2021), poiché la maggior parte dei titoli nel database appartiene a tale periodo.

## 6.3 Esempio 3

### Ricerca

Input:

TIPO

Series

PIATTAFORMA

☐ Netflix

☐ Prime Video

☒ Disney +

RICERCA

Ricerca Film e Serie Tv

GENERE

History

DA

1990

A

2021

DURATA MAX FILM/DURATA MAX MEDIA EPISODI (MINUTI)

120

RESETTA

## Output:

Genius (2019) (D)	<b>Valutazione Migliore: Genius (8.3)</b>
	<b>Descrizione</b> A teen genius juggles the roles of college student and junior high bad boy.

## Valutazione:

Includendo solamente le serie tv e la piattaforma Disney +, nonostante il periodo specificato sia quello a cui corrispondono più film, il software è in grado di suggerire solo un titolo, in base ai filtri stabiliti.

Tutto ciò perché nel database i titoli presenti su Disney + sono relativamente limitati rispetto alle altre 2 piattaforme.

Inoltre, nel data-set utilizzato le serie tv costituiscono circa il 30% del totale dei Titoli.

## 6.4 Esempio 1

### Ricorsione

## Input:

**Maratona Cinematografica**

<b>NUDITA'</b> 0 1 2 3	<b>GENERE</b> Comedy	<b>PIATTAFORMA</b> <input checked="" type="checkbox"/> Netflix
<b>PAURA</b> 0 1 2 3	<b>DURATA MAX (ORE)</b> 7	<input checked="" type="checkbox"/> Prime Video
<b>ALCOOL</b> 0 1 2 3	<b>VOTO MIN</b> 3	<input checked="" type="checkbox"/> Disney +
<b>VIOLENZA</b> 0 1 2 3		<b>DA</b> 1970
<b>GENERA</b> <b>RESET</b>		<b>A</b> 2000

## Output:

The Emperor's New Groove (2000) (D) The Golden Child (1986) (N) Beethoven (1992) (N) The Monster Squad (1987) (P)	<b>Valutazione Media:</b> <b>6.5</b> <b>Durata Tot: 5 ore e 38 minuti</b>
--	---

## Valutazione

Applicando tali filtri e impostando una durata massima non elevata (7 ore) la creazione della maratona è immediata.

### 6.5 Esempio 2

#### Ricorsione

Input:

The screenshot shows a web form titled "Maratona Cinematografica". It features four sliders on the left for "NUDITA'", "PAURA", "ALCOOL", and "VIOLENZA", all set to 3. In the center, there is a "GENERE" dropdown set to "Comedy", a "DURATA MAX (ORE)" input field with "10", and a "VOTO MIN" input field with "8". On the right, under "PIATTAFORMA", there are three checked checkboxes for "Netflix", "Prime Video", and "Disney +". Below these are two decade dropdown menus: "DA" set to "1990" and "A" set to "2021". At the bottom left are "GENERA" and "RESET" buttons.

Output:

The screenshot shows the output of the marathon creation. On the left, a list of movies is displayed: "Captain Fantastic (2016) (N)", "Mimi (2021) (N)", "Flipped (2010) (N)", "Bo Burnham: Inside (2021) (N)", and "Frances Ha (2012) (N)". On the right, the "Valutazione Media:" is shown as "8.2" and the "Durata Tot:" is "8 ore e 33 minuti".

Valutazione:

Aumentando la durata totale a 10, lasciando i parametri massimi degli slider a 3, scegliendo il periodo da cui provengono la maggior parte dei Titoli (1990-2021), selezionando tutte le piattaforme, pur scegliendo una valutazione minima molto alta (8), il software impiega circa 4 secondi per creare la maratona (procedura non più immediata).

## 6.6 Esempio Playlist

La cmbTitoli () per l'aggiunta dei titoli alla playlist viene riempita con i titoli suggeriti dal software nel punto precedente.

Non appena il titolo viene aggiunto alla playlist viene immediatamente rimosso dalla combobox.

The screenshot shows a web interface for movie recommendations. On the left, a box lists suggested movies: Jaws: The Revenge (1987) (N), Prom Night (1980) (N), Candyman (1992) (N), Child's Play (1988) (N), Event Horizon (1997) (N), Final Destination (2000) (N), and Friday the 13th (1980) (N). To the right, a box displays the 'Valutazione Migliore: Candyman (6.7)' and a 'Descrizione' of the movie: 'Grad student Helen Lyle unintentionally summons the Candyman, a hook-handed creature made flesh by other people's belief in him.' Below these, a section titled 'Playlist "Guarda più Tardi"' contains a dropdown menu for 'TITOLO' with 'Friday the 13th (1...' selected. There are two buttons: 'AGGIUNGI' (blue) and 'RESETTA PLAYLIST' (red). To the right of the buttons is a box containing the titles 'Candyman (1992) (N)' and 'Jaws: The Revenge (1987) (N)'.

## 7 Valutazioni sui risultati ottenuti

La pulizia dei data-set e la scelta degli attributi effettivamente presenti in tutti i record ha portato ad una riduzione della dimensione del database finale.

L'avere a disposizione un database relativamente limitato di Titoli ha rappresentato uno svantaggio per quanto riguarda la prima parte, votata alla ricerca semplice, ma d'altra parte ha rappresentato un requisito fondamentale per permettere all'algoritmo ricorsivo di fornire risultati in tempi accettabili nella seconda parte.

Mentre nella prima parte ho potuto lasciare una certa libertà nella compilazione dei vincoli (l'utente può selezionare il genere e il periodo temporale o solo uno dei 2; il periodo scelto deve essere di almeno 20 anni, per avere più risultati), nella seconda, per esigenze dettate dalla scelta della ricorsione (che può dilatare i tempi di creazione della maratona), l'utente è obbligato a compilare tutti i campi ed è consigliato farlo cercando di limitare il campo di ricerca dei titoli.

Nella creazione della maratona, infatti, il periodo temporale scelto non deve essere maggiore di 35 anni e la durata massima deve essere minimo di 3 ore (per cercare di non avere solo un Film consigliato) ed è preferibile non superi le 11 ore (ovviamente in relazione alla ristrettezza degli altri vincoli).

Inoltre, il database finale (ripulito) è popolato per la maggior parte da titoli presenti su Netflix, penalizzando la ricerca per le altre 2 piattaforme.

Si nota inoltre uno sbilanciamento per quanto riguarda il periodo di uscita dei titoli, a favore dei titoli più recenti (più numerosi).

Tenendo presente le precedenti considerazioni riguardo i filtri e il database di partenza, il software può rappresentare uno strumento molto utile e divertente da utilizzare.



This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

To view a copy of this license, visit

<http://creativecommons.org/licenses/by-nc-sa/4.0/>