

# REPLIKASI DATASET CSV MENGGUNAKAN PYTHON DAN JAVA

made by :

**Nindi Laura Salsabila (202410370110300)**

**Nabiila Izzati Zatadini (202410370110325)**

**Karina Putri Ani (202410370110354)**

# DATA Description

Data set ini **berisi informasi tentang film-film terpopuler yang diambil dari The Movie Database (TMDB) API.** Data tersebut mencakup atribut utama film seperti ID film, judul, tanggal rilis, popularitas, jumlah suara, dan rata-rata suara.

- Total Halaman yang Diambil: 500
- Total Film yang Termasuk: 10.000+

**data source :**

<https://www.kaggle.com/datasets/muhammadatifkhan181/top-rated-movies-in-tmdb-csv>



**10.000+**  
**data film**  
pada data ini

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# TANTANGAN UTAMA

## Implementasi Data

- Format data mentah "kotor".
- Banyak Judul Film mengandung tanda koma (,)  
(Contoh: "Schindler's List, The").
- Pemisahan kolom standar (Split by Comma)  
menyebabkan kerusakan struktur data.

	,id,title,vote_average,vote_count,release_date,popularity
0,278,	The Shawshank Redemption,8.711,29158,1994-09-23,25.3318
1,238,	The Godfather,8.685,22026,1972-03-14,26.3705
2,240,	The Godfather Part II,8.571,13315,1974-12-20,13.4802
3,424,	Schindler's List,8.566,16836,1993-12-15,10.6071
4,389,	12 Angry Men,8.548,9503,1957-04-10,9.0143
5,129,	Spirited Away,8.534,17608,2001-07-20,15.8954
6,155,	The Dark Knight,8.524,34658,2008-07-16,33.2459
7,19404,	Dilwale Dulhania Le Jayenge,8.5,4518,1995-10-20,6.8739
8,497,	The Green Mile,8.502,18519,1999-12-10,14.272

# Pendekatan 1

## Python – Prototyping

**Tujuan :** Validasi data dan eksplorasi visualisasi cepat.

**Library :**

- **pandas** : Untuk manipulasi data tabular (DataFrame).
- **streamlit** : Untuk antarmuka web dashboard.
- **plotly** : Untuk grafik interaktif.

---

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)



# Code Snippet

## (Python Load Data)

Fungsi `pd.read_csv` pada Python dapat dikategorikan sebagai fungsi tingkat tinggi (high-level function). Fungsi ini mampu secara otomatis mengenali pemisah kolom dalam file CSV serta membedakannya dari tanda koma yang merupakan bagian dari data, seperti pada judul film. Meskipun sintaks yang digunakan relatif sederhana, proses pengolahan data yang dilakukan di balik layar bersifat kompleks dan melibatkan mekanisme parsing serta validasi data secara menyeluru

```
@st.cache_data
def load_data():
    current_dir = os.path.dirname(os.path.abspath(__file__))
    file_path = os.path.join(current_dir, '..', 'data', 'raw', 'Top_Rated_Movies.csv')

    if os.path.exists(file_path):
        df = pd.read_csv(file_path)
        # Bersihkan tanggal agar bisa jadi grafik waktu
        if 'release_date' in df.columns:
            df['release_date'] = pd.to_datetime(df['release_date'], errors='coerce')
            df['year'] = df['release_date'].dt.year
        return df
    return None
```

# Code Snippet

## Strategi Data Cleaning

Fokus: Bagaimana mengatasi data tanggal yang rusak/error.

```
if 'release_date' in df.columns:  
    df['release_date'] = pd.to_datetime(df['release_date'], errors='coerce')  
    df['year'] = df['release_date'].dt.year  
    return df  
💡 return None
```

Parameter `errors='coerce'` digunakan dalam proses konversi kolom `release_date` ke dalam format `datetime`. Penerapan parameter ini bertujuan untuk mengantisipasi adanya data tanggal yang tidak valid atau tidak sesuai dengan format standar pada file CSV. Apabila sistem menemukan data yang bermasalah, nilai tersebut secara otomatis dikonversi menjadi `NaT` (Not a Time). Dengan mekanisme ini, proses pembacaan dan pengolahan data tetap dapat berjalan tanpa terhenti oleh kesalahan (fault tolerance), sehingga meningkatkan stabilitas dan keandalan sistem.

# Code Snippet

## Strategi Data Cleaning



A screenshot of a movie dashboard application. The interface is dark-themed with white and light gray text. At the top right, there are buttons for "Deploy" and three vertical dots. On the left side, there's a sidebar titled "Movie Dashboard" with a popcorn icon. It contains a section for selecting analysis modules: "Home Dashboard" (selected), "Top Charts (Leaderboard)", "Tren & Statistik", and "Movie Search Engine". Below this is a note: "Aplikasi Visualisasi Data Film TMDB." In the main content area, the title "Executive Summary" is displayed in large bold letters. A subtitle below it says "Ringkasan performa database film saat ini." To the right of the title are four data cards: "Total Film" (9,980), "Total User Votes" (20.5 M+), "Avg Quality Rating" (6.7 / 10), and "Data Update" (Dec 2025). At the bottom, there's a section titled "Quick Insights" with a lightbulb icon, containing the text "Gunakan menu di samping untuk melihat grafik tren dan pencarian detail."

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# Code Snippet

## Feature Engineering (Ekstraksi Data)

Bagaimana mendapatkan 'Tahun' dari 'Tanggal Lengkap'

```
df['release_date'] = pd.to_datetime(df['release_date'])
df['year'] = df['release_date'].dt.year
```

Dalam rangka mendukung analisis tren berbasis waktu, dilakukan proses ekstraksi fitur (feature engineering) pada data tanggal. Melalui penggunaan aksesori `.dt.year`, komponen tahun dipisahkan dari format tanggal lengkap (`release_date`).

Proses ini mengubah granularitas data dari tingkat harian menjadi tahunan, sehingga data siap digunakan untuk analisis time-series dan visualisasi tren berdasarkan tahun.

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# Code Snippet

## Feature Engineering (Ekstraksi Data)

The screenshot shows a dark-themed web application interface for movie analysis. On the left, a sidebar titled "Movie Dashboard" lists four modules: "Home Dashboard" (selected), "Top Charts (Leaderboard)", "Tren & Statistik", and "Movie Search Engine". A blue callout box at the bottom of the sidebar says "Aplikasi Visualisasi Data Film TMDB.". The main content area is titled "Hall of Fame" with a trophy icon. It displays a list of the "100 Film dengan Rating Tertinggi" (Top 100 Rated Movies). A slider indicates a "Minimal Jumlah Vote" of 1000. Below the slider is a table showing the top five movies:

title	vote_average	vote_count	release_date
The Shawshank Redemption	8.711	29158	1994-09-23 00:00:00
The Godfather	8.685	22026	1972-03-14 00:00:00
The Godfather Part II	8.571	13315	1974-12-20 00:00:00
Schindler's List	8.566	16836	1993-12-15 00:00:00
12 Angry Men	8.548	9503	1957-04-10 00:00:00

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# Code Snippet

## Data Aggregation (Logika Grafik)

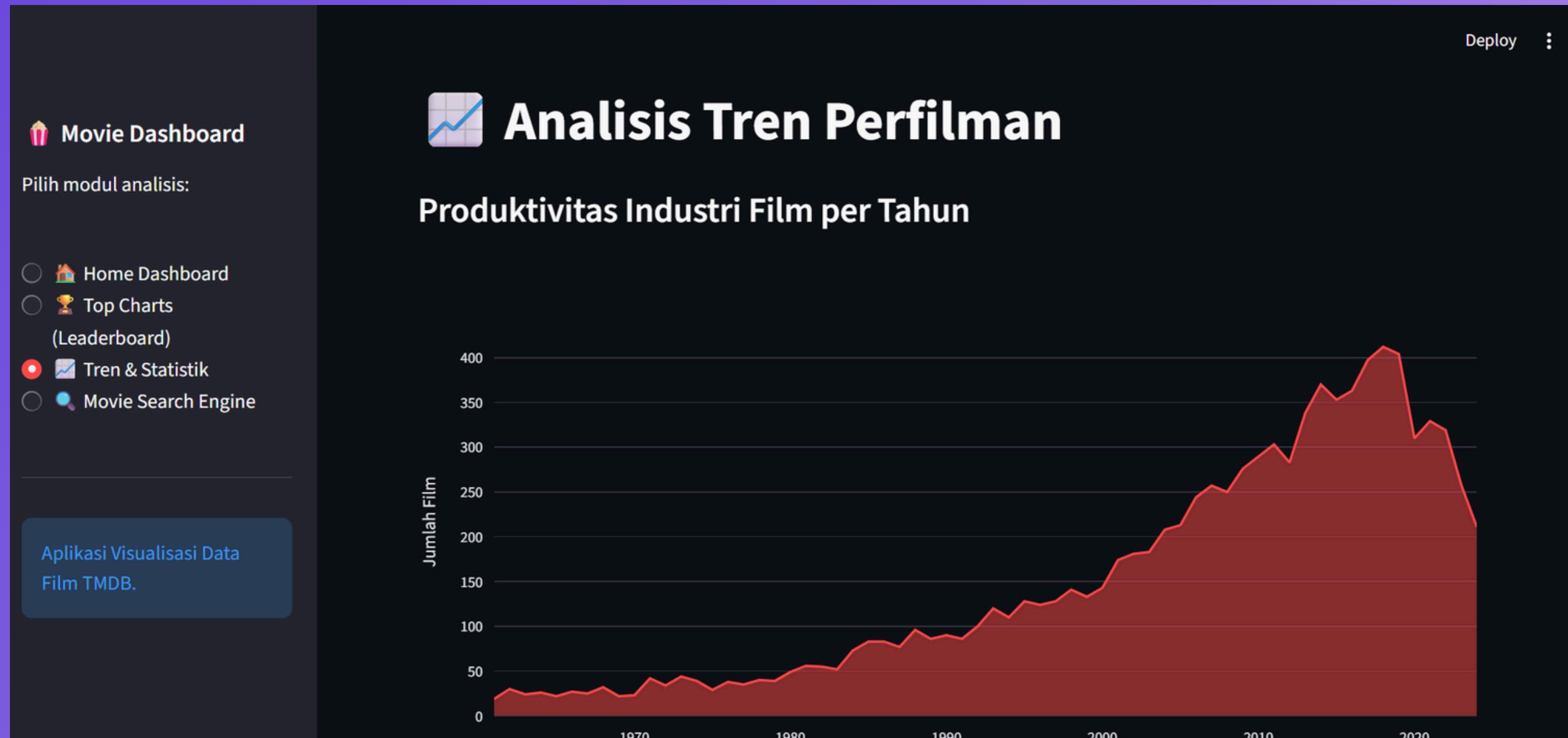
Fokus: Bagaimana menghitung jumlah film per tahun untuk grafik merah.

```
if 'year' in df.columns:  
    film_per_tahun = df.groupby('year').size().reset_index(name='Jumlah Film')  
    film_per_tahun = film_per_tahun[(film_per_tahun['year'] > 1960) & (film_per_tahun['year'] <= 2024)]  
  
    fig_line = px.area(film_per_tahun, x='year', y='Jumlah Film', color_discrete_sequence=['#FF4B4B'])  
    fig_line.update_layout(plot_bgcolor="rgba(0,0,0,0)")  
    st.plotly_chart(fig_line, use_container_width=True)
```

Untuk membangun visualisasi jumlah film per tahun, diterapkan proses agregasi data menggunakan fungsi groupby. Seluruh data film dikelompokkan berdasarkan kesamaan tahun rilis, kemudian fungsi .size() digunakan untuk menghitung total jumlah film pada setiap tahun. Proses ini menghasilkan dataset terstruktur yang terdiri atas variabel Tahun dan Jumlah Film, yang selanjutnya digunakan sebagai dasar pembuatan Area Chart guna merepresentasikan tren produksi film dari waktu ke waktu.

# Code Snippet

Data Aggregation (Logika Grafik)



LINK REPOSITORY

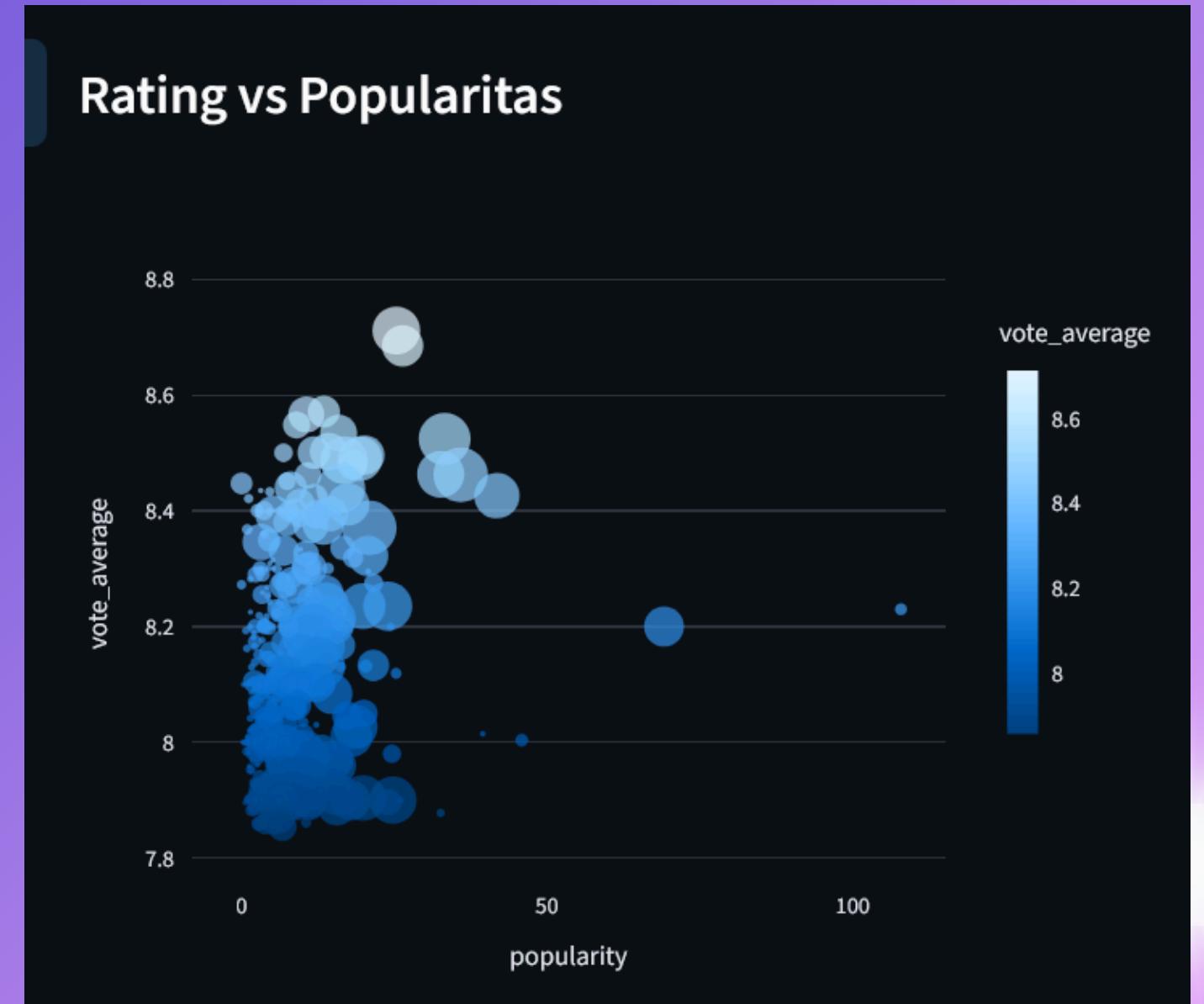
[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# Code Snippet

## Python Scatter

Visualisasi Scatter Plot Multivariat untuk menguji korelasi antara Popularitas (X-axis) dan Kualitas Rating (Y-axis). Grafik ini memetakan empat dimensi data sekaligus: posisi sumbu (variabel dependen/independen), skala warna (intensitas rating), dan ukuran titik (volume vote count). Analisis ini bertujuan untuk mendeteksi pola anomali, seperti film dengan popularitas tinggi namun memiliki rating rendah, atau sebaliknya

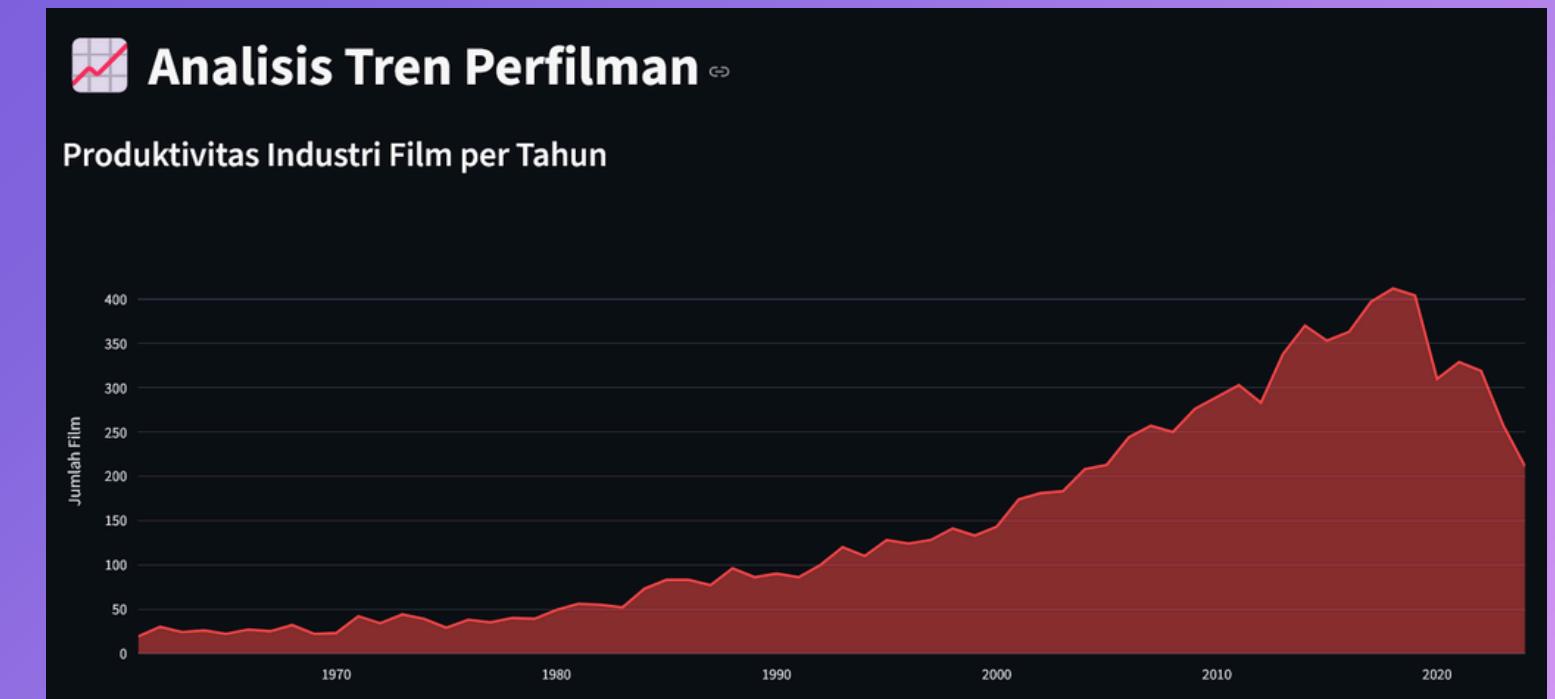
```
with col_b:  
    st.subheader("Rating vs Popularitas")  
    col_pop = 'popularity' if 'popularity' in df.columns else df.columns[-1]  
    col_rating = 'vote_average' if 'vote_average' in df.columns else df.columns[2]  
  
    fig_scat = px.scatter(df.head(500), x=col_pop, y=col_rating, color=col_rating, size='vote_count')  
    st.plotly_chart(fig_scat, use_container_width=True)
```



# Code Snippet

## Analisis Temporal (Tren Produktivitas Industri)

Analisis Time-Series menggunakan Area Chart untuk memetakan tren produktivitas industri perfilman dari waktu ke waktu. Algoritma memanfaatkan fungsi groupby pada Pandas untuk mengelompokkan frekuensi rilis film berdasarkan tahun. Visualisasi menggunakan library Plotly Express (px.area) memungkinkan eksplorasi data yang interaktif, memperlihatkan lonjakan volume produksi film secara signifikan pada era modern.



```
st.subheader("Produktivitas Industri Film per Tahun")
if 'year' in df.columns:
    film_per_tahun = df.groupby('year').size().reset_index(name='Jumlah Film')
    film_per_tahun = film_per_tahun[(film_per_tahun['year'] > 1960) & (film_per_tahun['year'] <= 2024)]

    fig_line = px.area(film_per_tahun, x='year', y='Jumlah Film', color_discrete_sequence=[ '#FF4B4B' ])
    fig_line.update_layout(plot_bgcolor="rgba(0,0,0,0)")
    st.plotly_chart(fig_line, use_container_width=True)
```

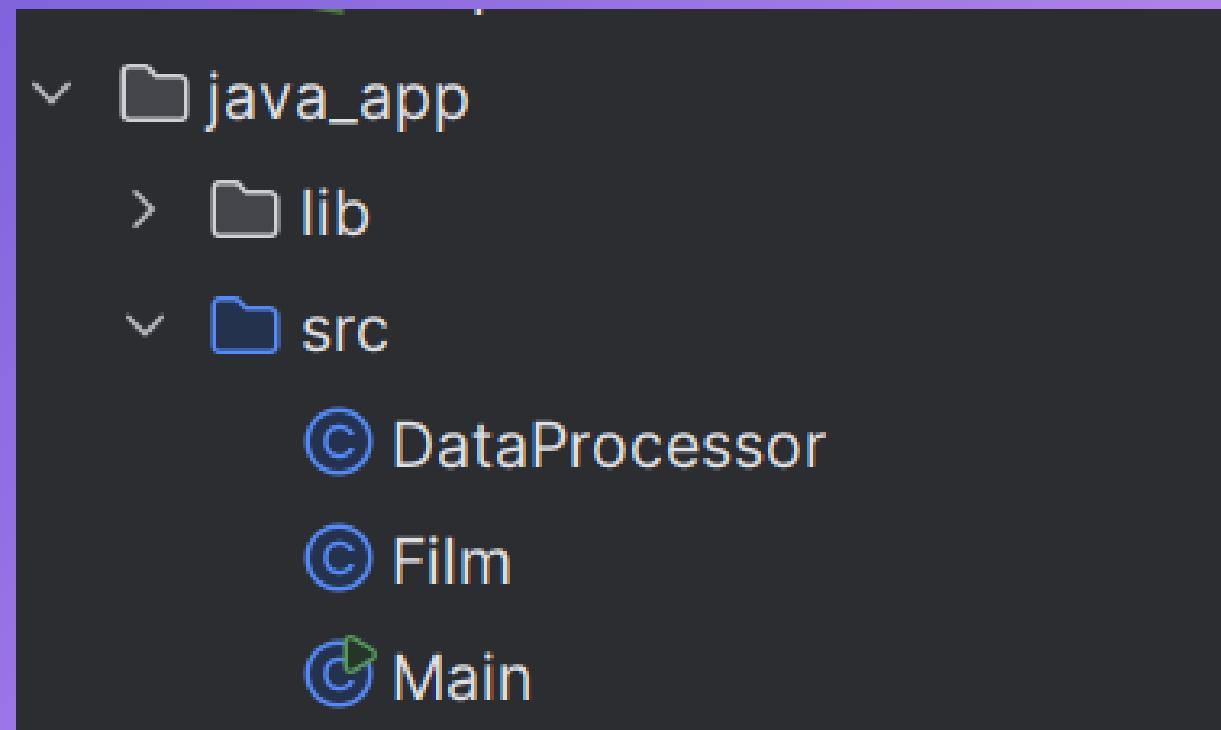
LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# Pendekatan 2 - Java

Membangun aplikasi Desktop Native yang KUAT menggunakan JavaFX.

1. Model (Film.java): Representasi objek data (OOP).
2. Logic (DataProcessor.java): Pemrosesan raw data manual.
3. View (Main.java): GUI Modern dengan JavaFX (Tabs & Charts).



Tanpa Library Pembantu: Tidak menggunakan library CSV eksternal, murni algoritma Java.

# Solusi Algoritma Java (The "Advanced" Part)

**Masalah:** Java String.split(",") akan memotong judul film yang memiliki koma.

**Solusi:** Menggunakan \*Regex Lookahead\*.

**Fungsi Parsing Manual** (DataProcessor.java):\*

Berbeda dengan metode pemisahan string standar (split biasa), sistem ini mengimplementasikan algoritma parsing berbasis Regular Expression (Regex) dengan mekanisme Lookahead Assertion.

Algoritma ini dirancang untuk melakukan validasi kontekstual terhadap tanda koma. Sistem secara cerdas membedakan antara koma yang berfungsi sebagai delimiter (pemisah antar-kolom) dan koma leksikal yang merupakan bagian dari data teks

```
try (BufferedReader br = new BufferedReader(new FileReader(csvFile))) {  
    br.readLine(); // Lewati header (baris pertama: ,id,title,...)  
    while ((line = br.readLine()) != null) {  
        // Split berdasarkan koma, tapi abaikan koma di dalam tanda kutip ("')  
        // PERHATIAN: Line dimulai dari 0,278,The Shawshank...  
        String[] data = line.split(regex: "(?=(?:[^\\"]*\"[^\\"]*\")*[^\"]*$)");
```

# Code Snippet

## (Implementasi model data OOP)

### Class Film.java

- Menggunakan JavaFX Property agar data sinkron dengan Tabel GUI secara real-time.

```
public class Film {  
    // Properti Data Murni (sesuai kolom di CSV)  
    2 usages  
    private StringProperty title;  
    3 usages  
    private DoubleProperty voteAverage;  
    2 usages  
    private DoubleProperty popularity;  
    1 usage  
    private StringProperty releaseDate;  
    1 usage  
    private DoubleProperty voteCount; // Menggunakan DoubleProperty untuk kemudahan parsing awal
```

Terdapat perbedaan fundamental dalam arsitektur data kedua bahasa. Python menerapkan Dynamic Typing, dimana tipe data bersifat fleksibel dan tidak memerlukan definisi struktur yang ketat.

Sebaliknya, Java menerapkan Strong Static Typing dan Encapsulation. Data wajib dibungkus dalam objek (Class Film) untuk menjamin keamanan dan integritas struktur.

LINK REPOSITORY

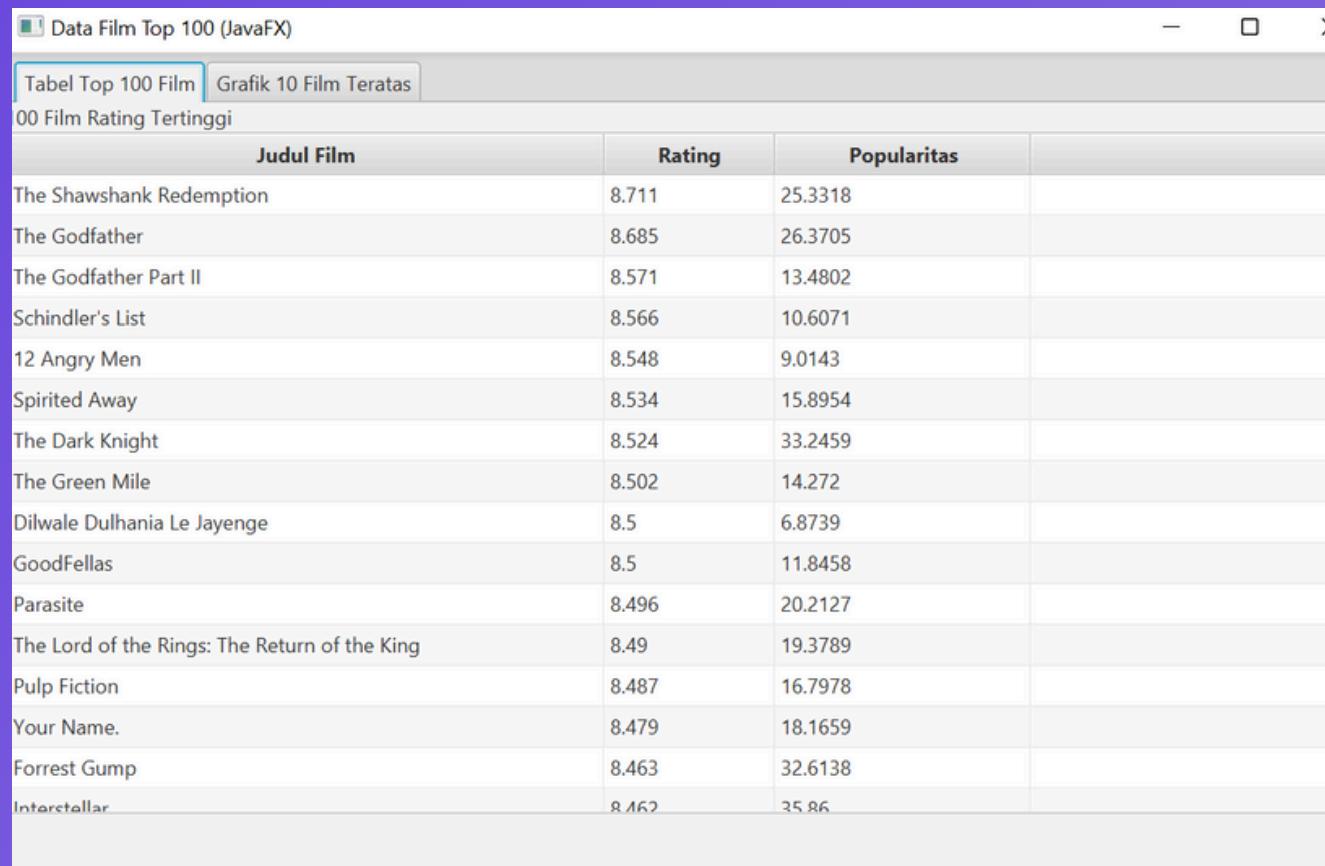
[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# Hasil Visualisasi (JavaFX GUI)

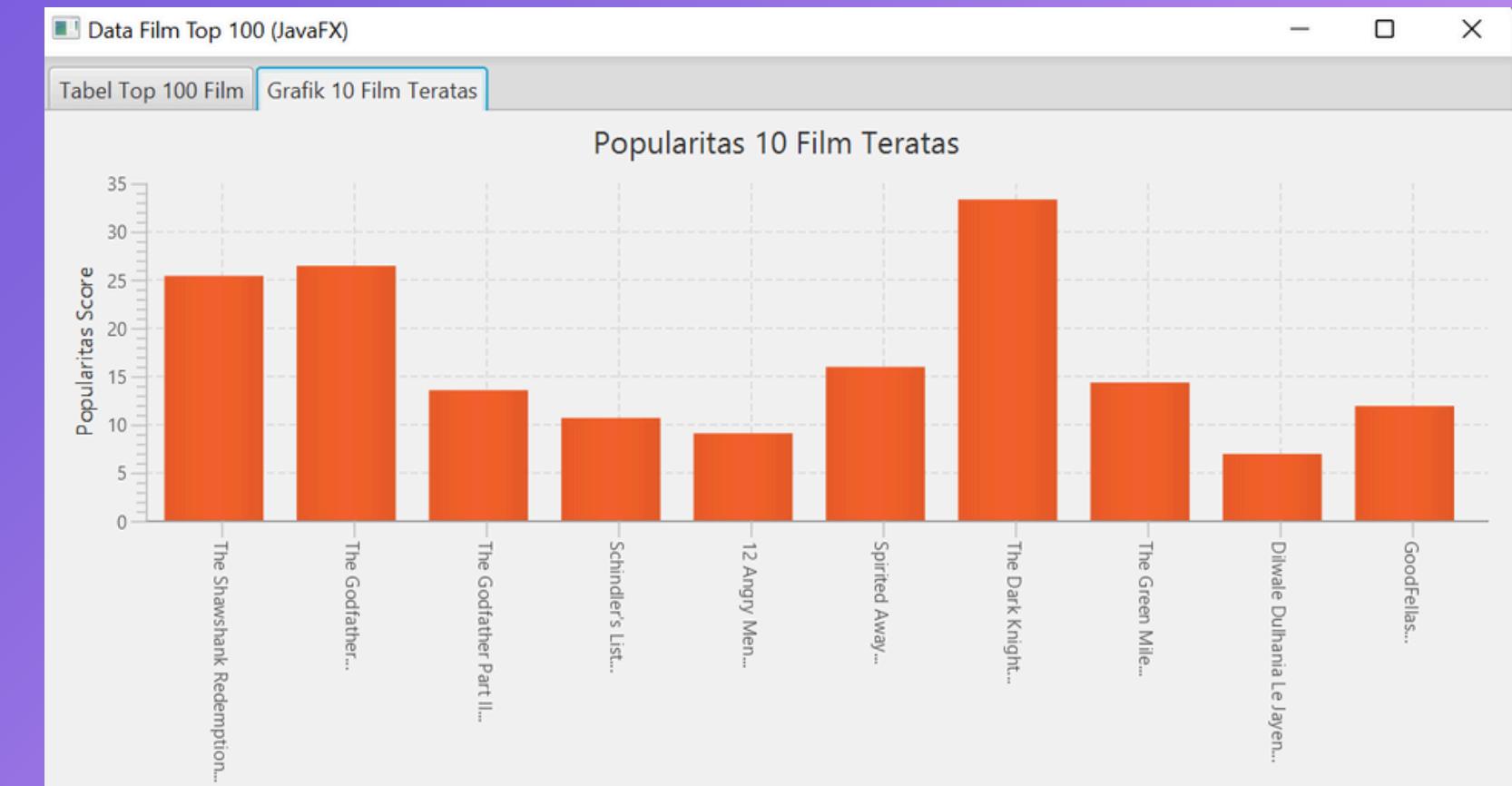
[Home](#)[About Us](#)[Contact](#)

## Fitur Aplikasi:

- **Tab Layout:** Memisahkan tampilan Tabel dan Grafik.
- **TableView:** Menampilkan 100 Data Teratas dengan kolom (Judul, Rating, Popularitas).
- **BarChart:** Visualisasi perbandingan popularitas 10 film teratas.



Judul Film	Rating	Popularitas
The Shawshank Redemption	8.711	25.3318
The Godfather	8.685	26.3705
The Godfather Part II	8.571	13.4802
Schindler's List	8.566	10.6071
12 Angry Men	8.548	9.0143
Spirited Away	8.534	15.8954
The Dark Knight	8.524	33.2459
The Green Mile	8.502	14.272
Dilwale Dulhania Le Jayenge	8.5	6.8739
GoodFellas	8.5	11.8458
Parasite	8.496	20.2127
The Lord of the Rings: The Return of the King	8.49	19.3789
Pulp Fiction	8.487	16.7978
Your Name.	8.479	18.1659
Forrest Gump	8.463	32.6138
Interstellar	8.462	35.86



Terlepas dari performa tinggi Java dalam pengolahan Big Data, strategi pembatasan data (Top 100 entries) diterapkan pada antarmuka pengguna guna mencegah beban kognitif berlebih dan menjaga responsivitas aplikasi secara real-time

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# Komparasi Akhir (Python vs Java)

[Home](#)[About Us](#)[Contact](#)

Aspek	Python (Streamlit)	Java (JavaFX)
Parsing data	Otomatis (Pandas Library)	Manual menggunakan Regex Logic
Gaya Coding	Otomatis (Pandas Library)	OOP (terstruktur & ketat)
Tipe data	Dinamis (dapat berubah)	Statis (harus didefinisikan)
Hasil Akhir	Web Dashboard	Aplikasi Desktop (.exe)
Kesimpulan	Cocok untuk Analisis Data Scripting (Cepat & Pendek)	Cocok untuk Software Engineering

Python gampang buat analisa, tapi Java melatih logika pemrograman yang sebenarnya (karena harus ngurusin memori, tipe data, dan parsing manual). Punya kelompokmu lebih "Engineering".

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)

# THANK YOU For Attention

LINK REPOSITORY

[https://github.com/izzadin24/Advanced\\_Programming\\_MoviesApp](https://github.com/izzadin24/Advanced_Programming_MoviesApp)