Project Report:

Simple Movie Recommendation System

Project Name: Content-Based Movie Recommender (Genre Similarity)

1. Introduction and Project Goal

The primary goal of this project was to develop a simplified, single-file movie recommendation system that demonstrates the core principles of Content-Based Filtering (CBF), a foundational technology in recommendation engines.

This system is designed to provide movie suggestions to a user based on the genre profile of a movie they have selected. By treating genres as features, the system identifies other movies in the dataset that are most similar to the user's input, thus providing relevant recommendations.

While the system is labeled "Hybrid" in its conceptual framework, this implementation focuses exclusively on the Content-Based (genre similarity) component for simplicity and demonstration purposes.

2. Methodology: Content-Based Filtering

Content-Based Filtering relies on the characteristics (or "content") of items to find similar items. In this project, the characteristics are the movie genres.

The methodology involves three main steps:

2.1 Feature Representation (TF-IDF)

1. **Text Preprocessing:** The list of genres for each movie (e.g., "Action|Crime|Drama") is first converted into a single string of space-separated words ("Action Crime Drama").
2. **TF-IDF Vectorization:** The **Term Frequency-Inverse Document Frequency (TF-IDF)** technique is used to transform the genre strings into a numerical vector space.
    - **Term Frequency (TF):** Measures how frequently a genre term appears in a specific movie's genre list.
    - **Inverse Document Frequency (IDF):** Measures how important a genre term is across the whole dataset (i.e., common genres like 'Drama' get a lower weight than rare genres).
    - The output is a large sparse matrix where each row represents a movie, and each column represents a weighted genre feature.

3.. Implementation Details (Python)

The solution is implemented as a single Python script utilizing the following key libraries:

Library	Purpose
pandas	Data handling and structure (DataFrame).
sklearn.feature_extraction.text.TfidfVectorizer	Numerical feature engineering from text data (genres).
sklearn.metrics.pairwise.linear_kernel	Efficient calculation of Cosine Similarity.

Key Functions Breakdown:

1. **create_and_load_data():**
   - Initializes the movie dictionary.
   - Creates the Pandas DataFrame.
   - Cleans the Genres string by replacing the | delimiter with a space, making it ready for the TF-IDF tokenizer.
2. **get_recommendations(title, movies_df, cosine_sim, num_recommendations=5):**
   - Finds the index of the input title within the DataFrame.
   - Extracts the similarity scores for that movie from the cosine_sim matrix.
   - Sorts the scores in descending order.
   - Selects the top N movies (excluding the movie itself).
   - Maps the indices back to movie titles and returns the list of recommendations.

4. System Execution and Results

The main() function orchestrates the process:

1. It loads the data and builds the model once.
2. It prompts the user to enter a movie title from the available list.
3. It calls get_recommendations() using the user's input.
4. It prints a structured output, displaying the top recommendations and their respective genres, allowing the user to understand *why* the recommendation was made (i.e., genre overlap).

Example Output: If the user enters **"Heat"** (Genres: Action Crime Drama), the system compares its genre vector to all others. It should recommend **"The Dark Knight"** (Action Crime Drama) because the two movies share a 100% genre match, resulting in a high cosine similarity score.