Agenda

1. Abstract
2. Introduction
   - 2.1 Problem definition
3. Existing systems-disadvantages
4. Proposed system, Features
5. Literature survey
6. SRS (System Requirements Specifications)
7. System Analysis
   - 7.1 System Architecture and architectural diagram
   - 7.2 Module Description
8. Design-UML Diagrams
9. implementation-coding-language

Abstract

The **Movie Recommendation System** is an AI-powered application that suggests movies to users based on their preferences. The system utilizes **collaborative filtering, content-based filtering, and hybrid recommendation techniques** to deliver personalized movie recommendations. By analyzing user ratings, movie metadata, and viewing history, the system helps users discover new movies that align with their tastes.

Online recommendation engines have shaped our choices, whether we're looking for a movie or picking an OTT platform's series. They are, however, still in the early stages of development and far from being ideal. In this paper, we specifically discuss movie recommendation systems. Additionally, we attempt to critically evaluate some work on movie recommendation systems and talk about some research papers that have helped these systems overcome a number of obstacles. Although there have been advancements, more work needs to be done on recommendation systems to make them more effective at providing accurate recommendations across a wider range of applications.

Keywords: Recommendation System, Suggestion, Movies, Search, Machine Learning, Recommender

Introduction

* With the explosion of online streaming platforms, users have access to an overwhelming number of movies. Traditional **movie recommendation systems** help users by suggesting films based on their viewing history, ratings, and preferences. However, these systems often fail to consider a critical factor—**the user's current emotional state**.
* People choose movies not only based on past preferences but also on their **mood**. A person feeling **happy** might prefer a comedy, while someone feeling **sad** may look for a drama or uplifting movie. To address this limitation, **emotion-based recommendation systems** have emerged, utilizing **Natural Language Processing (NLP) and Sentiment Analysis** to analyze user emotions and provide **personalized** suggestions.

Problem Definition

* In today's digital era, movie recommendation systems play a crucial role in helping users find relevant content. Traditional recommendation systems rely on **collaborative filtering and content-based filtering**, which primarily use **user ratings, viewing history, and movie metadata**. However, these systems lack **emotional intelligence**, failing to consider the **current mood or emotions** of the user while suggesting movies.
* This leads to several challenges:
  ❌ **Lack of Emotion-Based Personalization** – Users may not always want recommendations based on past preferences; their choices often depend on their mood.
  ❌ **Cold Start Problem** – New users receive inaccurate recommendations due to insufficient data.
  ❌ **Popularity Bias** – The system often recommends trending movies instead of genuinely relevant ones.
  ❌ **Limited Adaptability** – Traditional systems do not dynamically update recommendations based on real-time emotions.

Existing Systems

Existing Movie Recommendation Systems

Netflix Recommendation System

* Uses a hybrid approach combining collaborative filtering, content-based filtering, and deep learning models.
* Considers user viewing history, ratings, and interaction patterns.
* Provides personalized recommendations on the homepage.

Amazon Prime Video

* Uses collaborative filtering and purchase history to recommend movies and shows.
* Integrates customer reviews and preferences for better suggestions.

Hulu

* Focuses on a user's watch history and preferences.
* Suggests trending and highly-rated content.

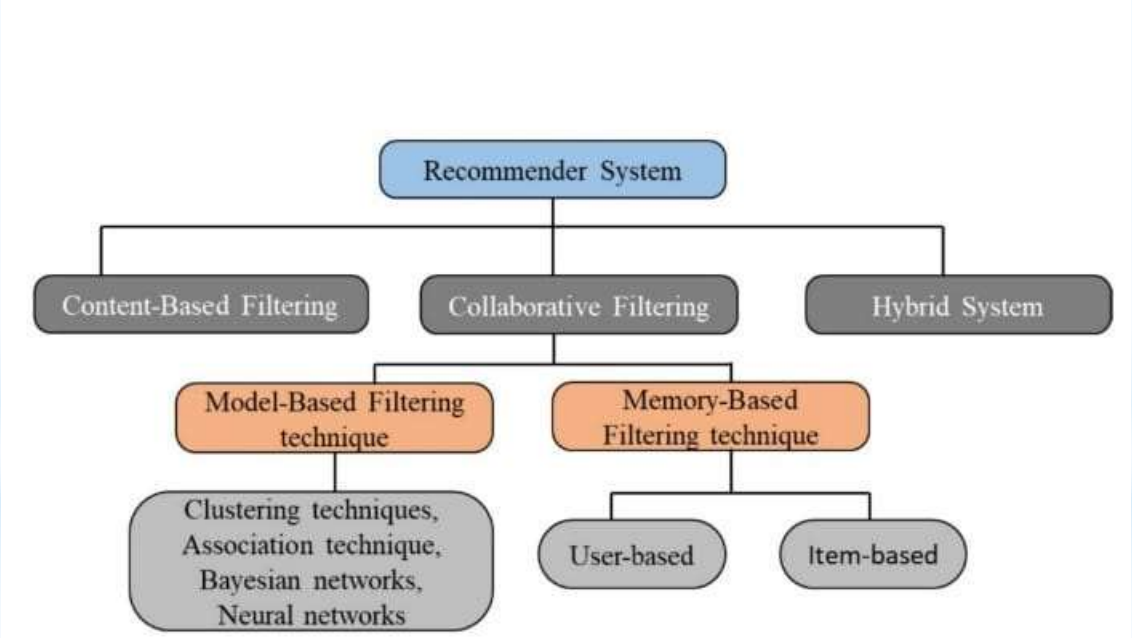
Disney+

* Uses a content-based filtering approach, focusing on genre and franchise preferences.
* Provides recommendations based on user profiles.

IMDb & TMDb (The Movie Database)

* Provides movie recommendations based on ratings, reviews, and user interactions.
* Uses metadata (actors, directors, genres) for content-based recommendations.

Fig:Classificaton Of Recommendation Systems



DISADVANTAGES

While current movie recommendation systems are effective, they come with several limitations that affect their accuracy, adaptability, and user satisfaction.

1. Cold Start Problem

When a new user joins or a new movie is added, there is little to no data available, making it difficult provide relevant recommendations.

Example: A new Netflix user may receive generic recommendations until the system gathers enough data about their preferences.

2. Popularity Bias

Most systems favor trending or highly-rated movies, often overlooking niche, independent, or underrated films.

Example: Netflix and Prime Video tend to push blockbuster movies, while lesser-known films get buried.

3. Over-Reliance on Historical Data

Many systems base recommendations only on past interactions, failing to adapt when a user's taste changes over time.

PROPOSED SYSTEMS & FEATURES

To overcome the limitations of existing systems, the proposed **AI-powered Movie Recommendation System** integrates **collaborative filtering, content-based filtering, and hybrid recommendation techniques** to enhance accuracy, diversity, and adaptability.

1. Key Features of the Proposed System

✓ Hybrid Recommendation Approach

Combines **collaborative filtering** (user behavior-based) and **content-based filtering** (movie attributes-based) for more precise suggestions.

Adapts to changing user preferences dynamically.

✓ Cold Start Problem Reduction

Uses **metadata-based recommendations** for new users and movies (genre, actors, director, etc.).

Employs **TF-IDF (Term Frequency-Inverse Document Frequency)** for text-based similarity analysis of movie descriptions.

✓ Diverse and Personalized Suggestions

Introduces a **diversity-aware model** that ensures recommendations include a mix of mainstream and niche films.

Prevents over-reliance on a single genre by exploring user preferences across multiple categories.

✓ Adaptive Learning for User Preferences

Implements **reinforcement learning** to refine recommendations based on real-time user interactions (likes, watch duration, skips).

Uses **Singular Value Decomposition (SVD)** for better matrix factorization in collaborative filtering.

✓ Scalability & Real-Time Processing

Built with **Flask/FastAPI for backend API handling** and **Streamlit for an interactive UI** to provide fast and scalable recommendations.

Integrates **K-Nearest Neighbors (KNN) and deep learning models** for real-time data processing.

✓ Privacy-Focused and Secure Data Handling

Ensures user data is handled securely, with anonymized data processing techniques.

Provides **opt-out options** for data collection and personalization.

Advantages

* 🎯 1. Emotion-Aware Personalization

✅ Recommends movies based on the **user's current mood** rather than just past viewing history.
✅ Enhances user experience by making suggestions more relevant and engaging.

* ▫ 2. Improved Recommendation Accuracy

✅ Uses **NLP-based sentiment analysis** to understand user emotions from text inputs.
✅ Employs a **hybrid recommendation approach (content-based + collaborative filtering)** for better precision.

* 🔥 3. Overcoming Traditional Limitations

✅ **Solves the cold start problem** – Even new users can receive meaningful recommendations based on their emotions.
✅ **Reduces popularity bias** – Does not only suggest trending movies but also **emotionally relevant** film.

* ▫ 4. Real-Time Adaptability

✅ Dynamically updates recommendations based on **real-time user input**.
✅ Unlike static systems, it evolves with the user's changing preferences and emotions.

for **mental wellness platforms** to suggest uplifting content based on mood.

LITERATURE SURVEY

Sr No.	Title of paper	Description with Seed idea	Technique Used	Merit/ demerits
1.	**"Movie Recommendation System."** Prof. Sanober Shaikh 1 , Adhithyaram S 2 , Aryak Deshpande 3 May 2021	We have Studies about Daily Recommender System, Machine Learning, Collaborative Filtering, Content based Filtering.	calculation of the cosine similarity matrix which is done using the movie's feature vectors and the user's.	**Merit :** Movies area unit a supply of standard diversion. From the instant we have a tendency to derive pleasure observance a moving picture. **Demerit:** There is no denying that movies nowadays area unit additional violent than ever before. And it's terribly clear with the shootings at colleges by child and teenagers that they're being heavily influenced by violence shown is movies

Software Requirement Specification (SRS)

Hardware Requirements

Component	Minimum Requirement	Recommended Requirement
Processor	Intel Core i5 (or equivalent)	Intel Core i7/i9 or Ryzen 7/9
RAM	8GB	16GB or more
Storage	256GB SSD or HDD	512GB SSD or higher
GPU	Integrated Graphics	Dedicated GPU (NVIDIA/AMD)
Internet	Stable connection (for API calls)	High-speed connection

💻 Software Requirements

Operating System:

✅ Windows 10/11, macOS, or Linux (Ubuntu preferred)

Development Environment & Tools:

✅ **Python** (Version 3.8 or above)
✅ **Jupyter Notebook / VS Code / PyCharm** (For development)
✅ **Streamlit** (For UI)
✅ **Flask / FastAPI** (For backend API)

Libraries & Frameworks:

✅ **Machine Learning:** Scikit-learn, Pandas, NumPy
✅ **Natural Language Processing (NLP):** NLTK, TextBlob, VADER, Transformers (Hugging Face)
✅ **Sentiment Analysis:** TensorFlow/Keras (for deep learning-based models)
✅ **Data Handling:** Pandas, JSON, BeautifulSoup (for web scraping if needed)

Database & APIs:

✅ **Database:** MySQL / PostgreSQL / Firebase (for storing user preferences)
✅ **Movie Data Source:** MovieLens Dataset or **TMDb API** for fetching movie details

2. Technology Stack

Frontend: Streamlit (for user-friendly interaction).

Backend: Python (Flask or FastAPI for API handling).

Machine Learning Libraries: Scikit-learn, Pandas, NumPy.

Recommendation Algorithms: TF-IDF, KNN, SVD.

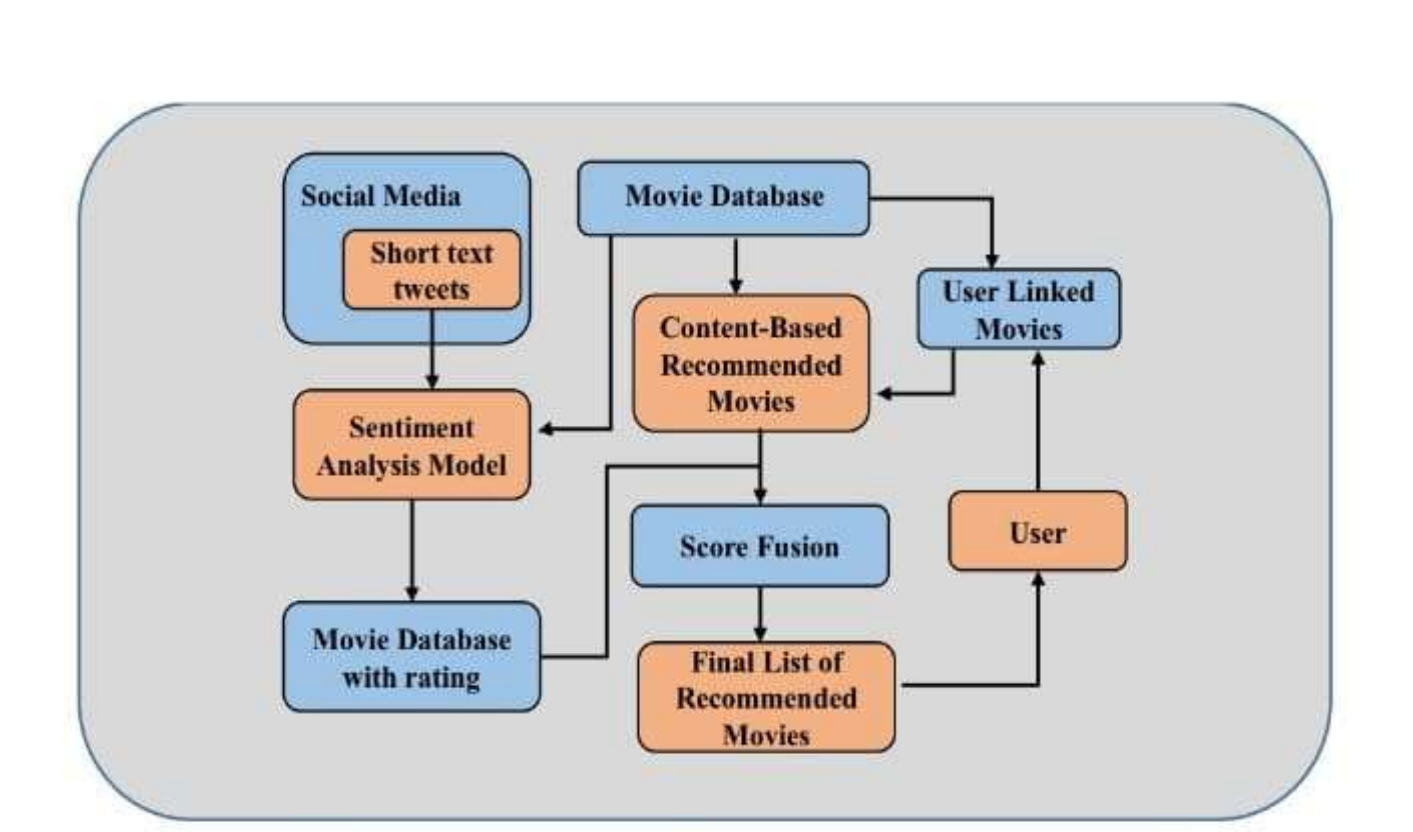
Data Source: MovieLens dataset or TMDb API for real-time movie details.

3. Advantages Over Existing Systems

- ✓ Addresses the **cold start problem** by incorporating metadata-based filtering.
- ✓ Provides **more diverse recommendations** to avoid repetitive suggestions.
- ✓ Uses **real-time user interactions** to update preferences dynamically.
- ✓ Ensures **better scalability and privacy** than traditional systems.

System Architecture

Architecture:



Module Discription

1. User Interface Module

📌 **Description:**

* Provides an interactive and user-friendly interface for users to input their emotions or preferences.
* Developed using **Streamlit** for a seamless experience.

📌 **Functions:**

✅ Allows users to enter text (reviews, comments) to analyze their emotions.
✅ Displays recommended movies based on detected emotions.
✅ Enables users to provide feedback on recommendations.

2. Backend API Module

* 📌 **Description:**
* Manages communication between the user interface, recommendation engine, and database.
* Developed using **Flask or FastAPI** for efficient API handling.
* 📌 **Functions:**

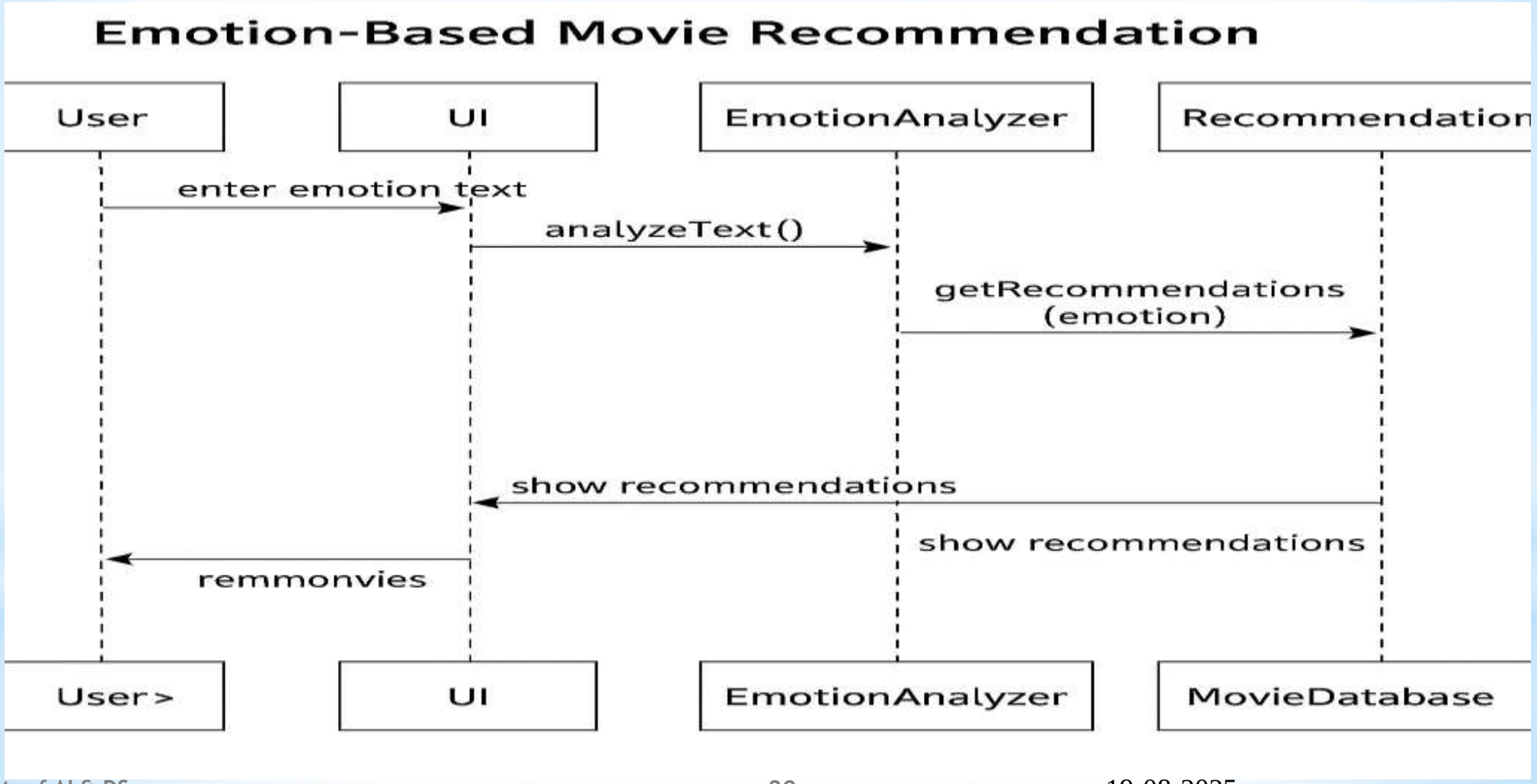
✅ Processes user requests and sends them to the appropriate module.
✅ Fetches movie recommendations and delivers responses in real-time.
✅ Ensures seamless data flow between different components.

UML DIAGRAMS

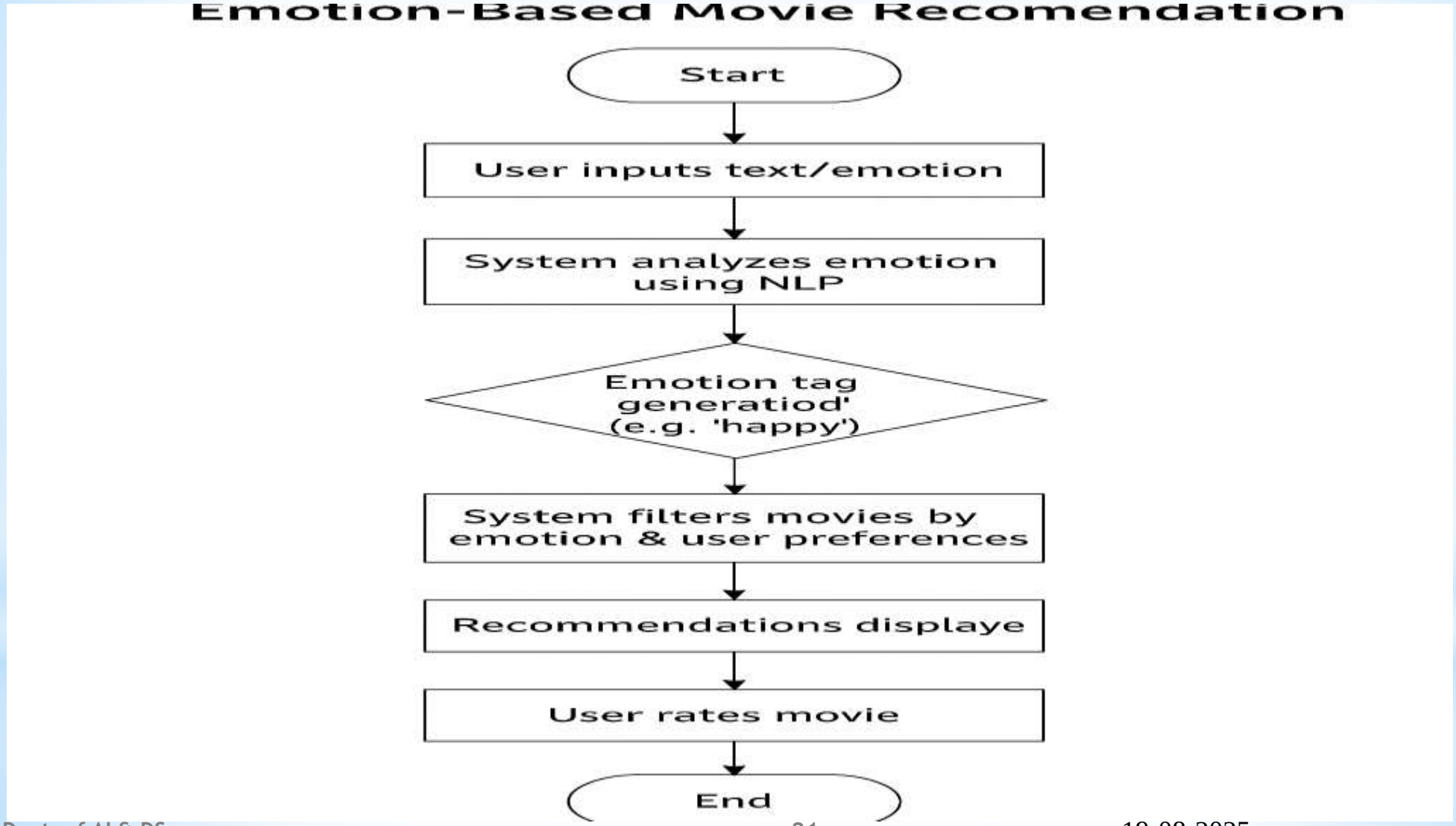
USE DIAGRAM

[User] ---> (Input Emotion)
---> (View Recommendations)
---> (Rate Movie)
(Input Emotion) --> (Detect Emotion)
(Detect Emotion + Rating + Metadata) --> (Generate Recommendations

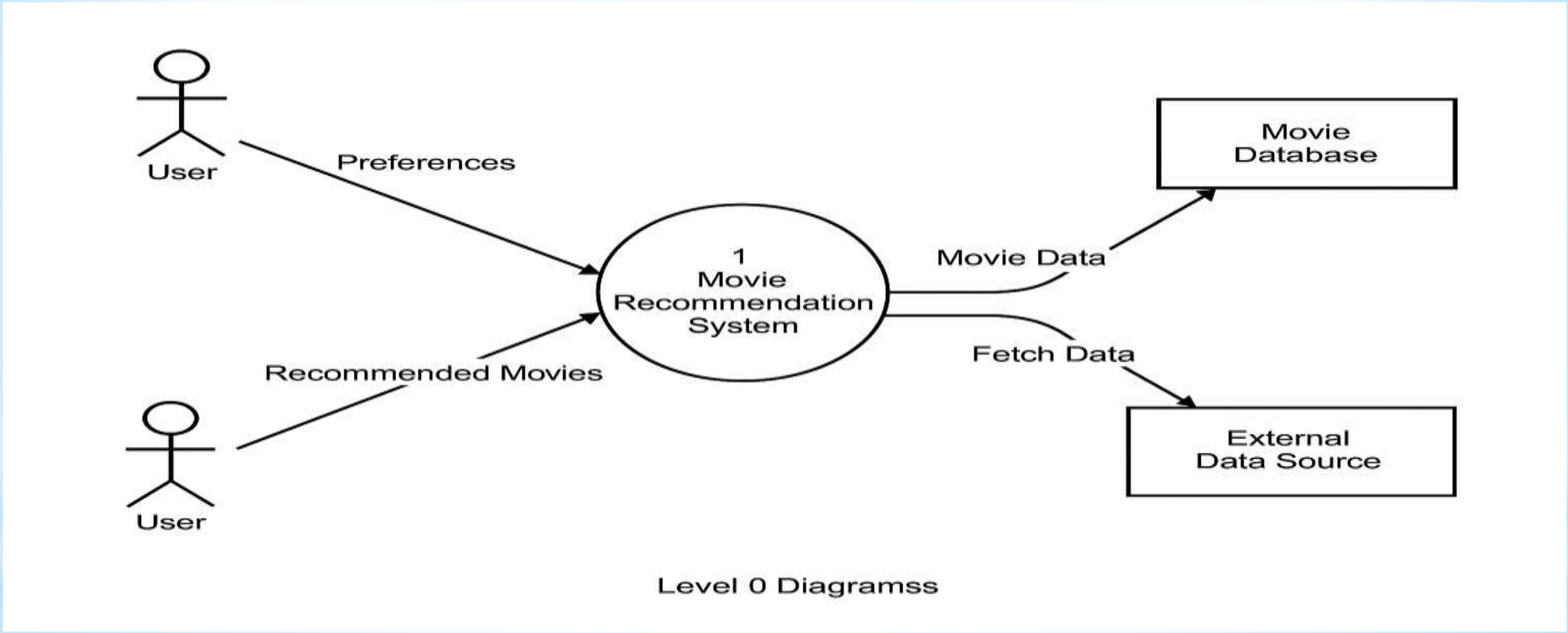
SEQUENCE DIAGRAM



ACTIVITY DIAGRAM

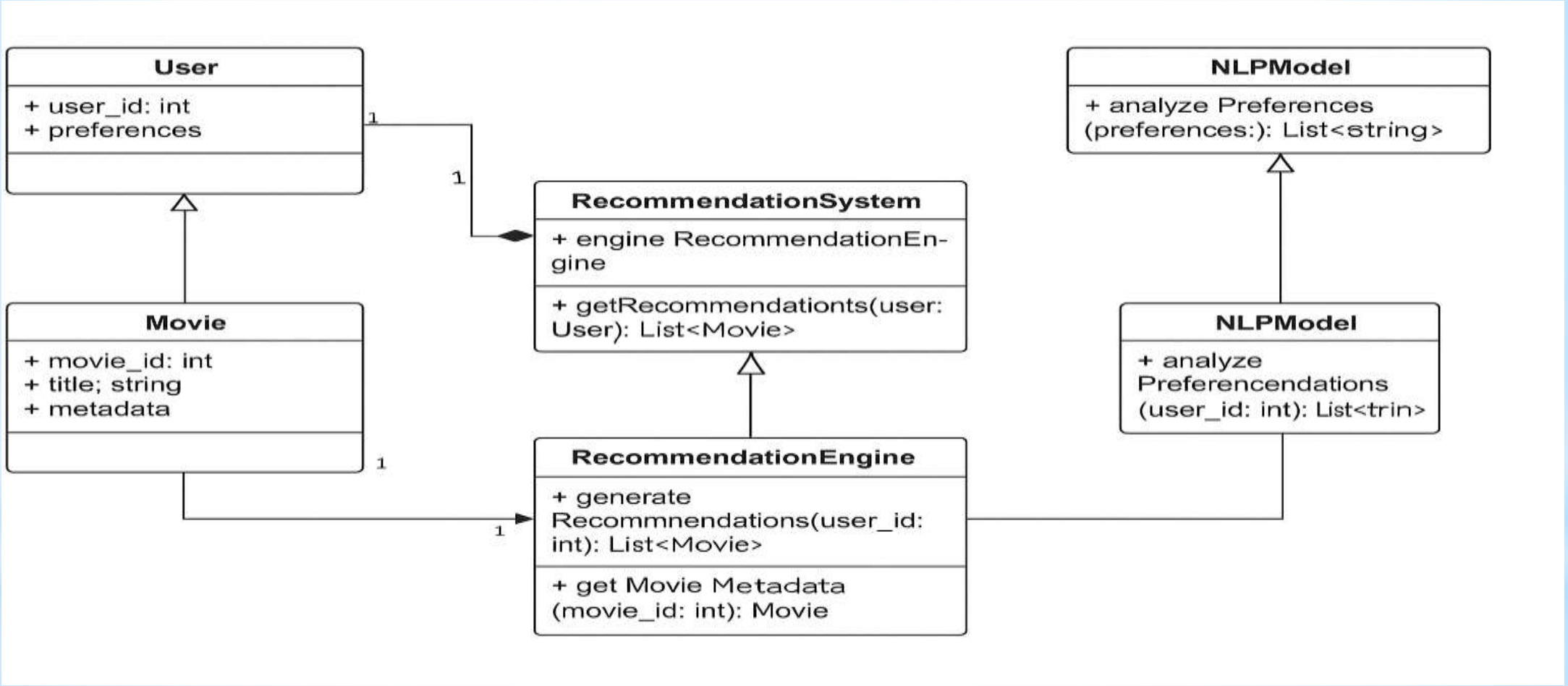


DATA FLOW DIAGRAM

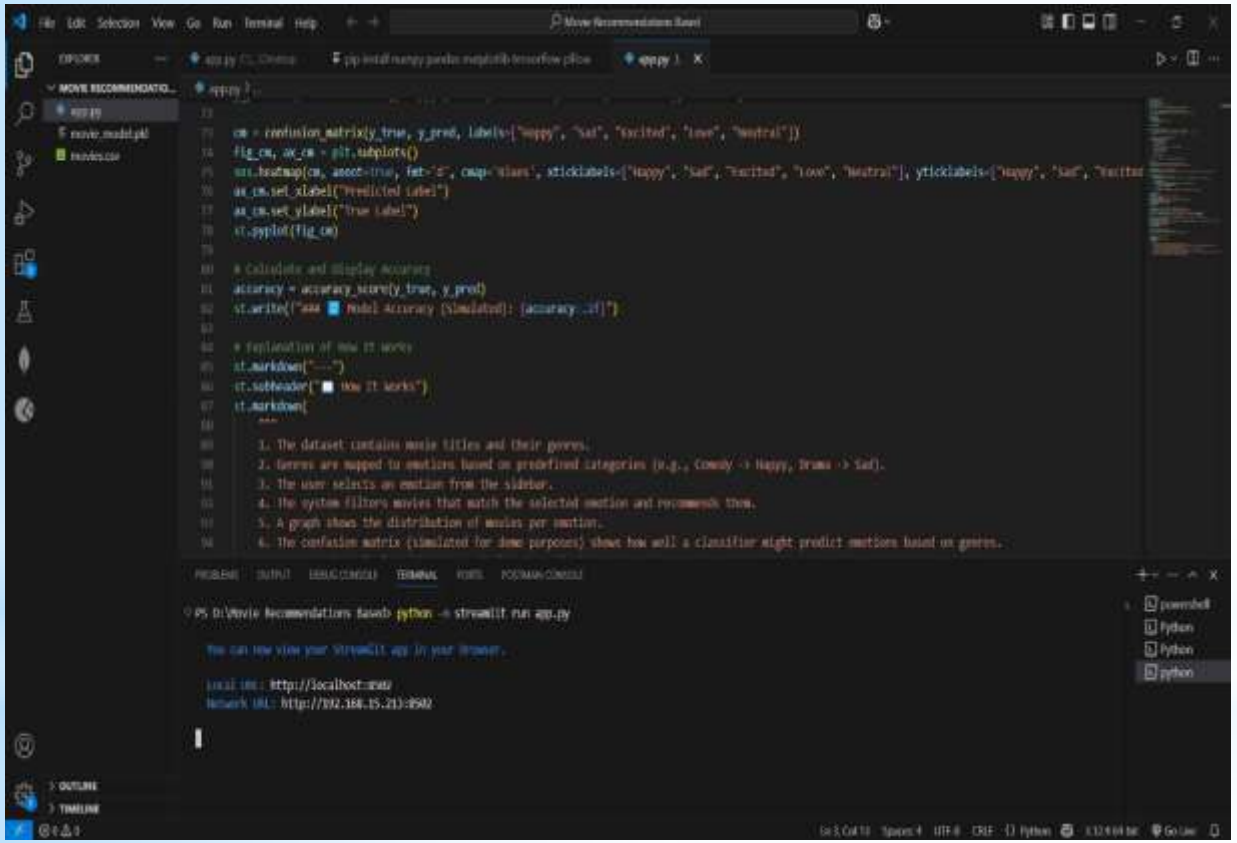


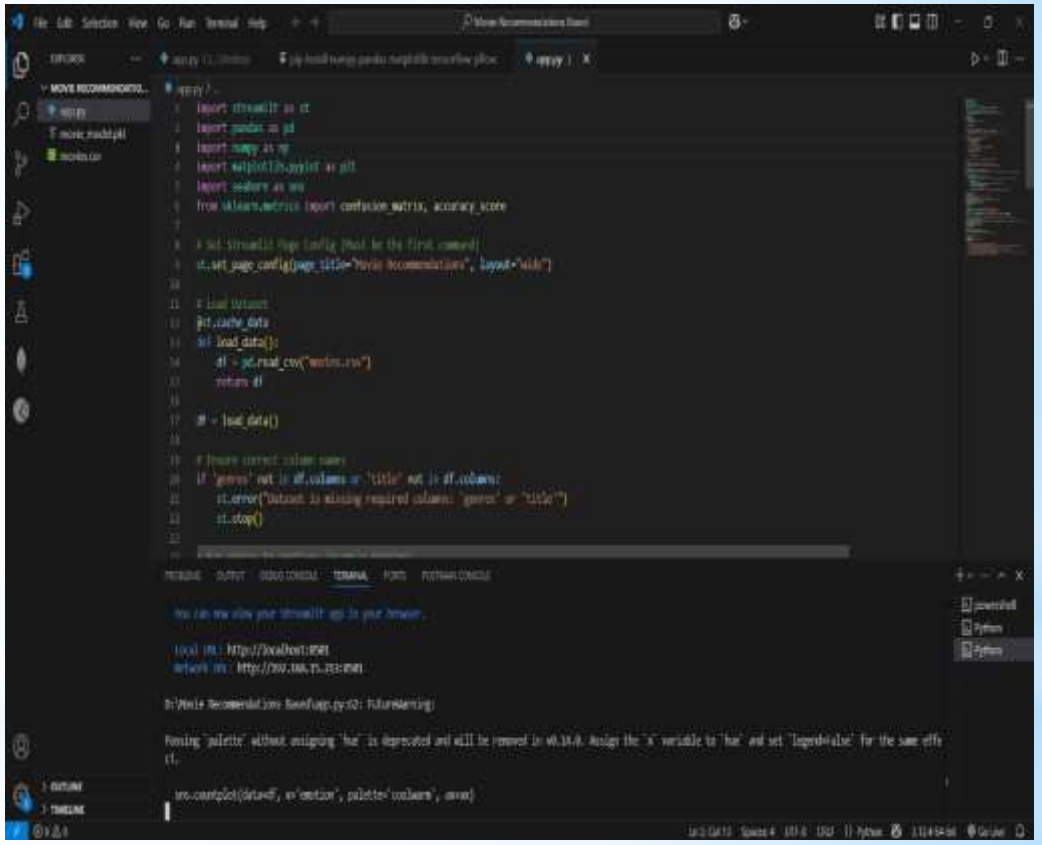
Level 0 Diagramss

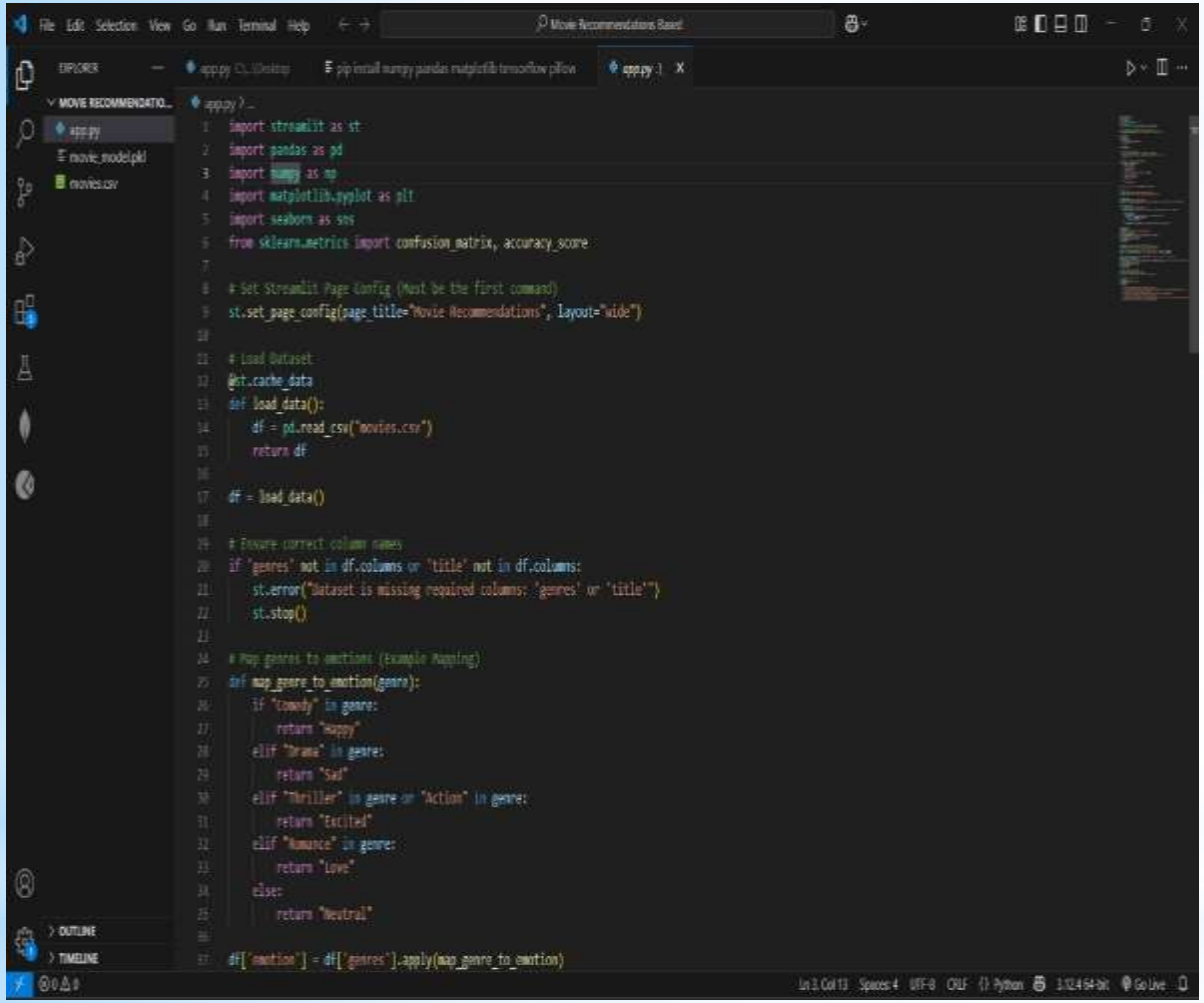
CLASS DIAGRAM



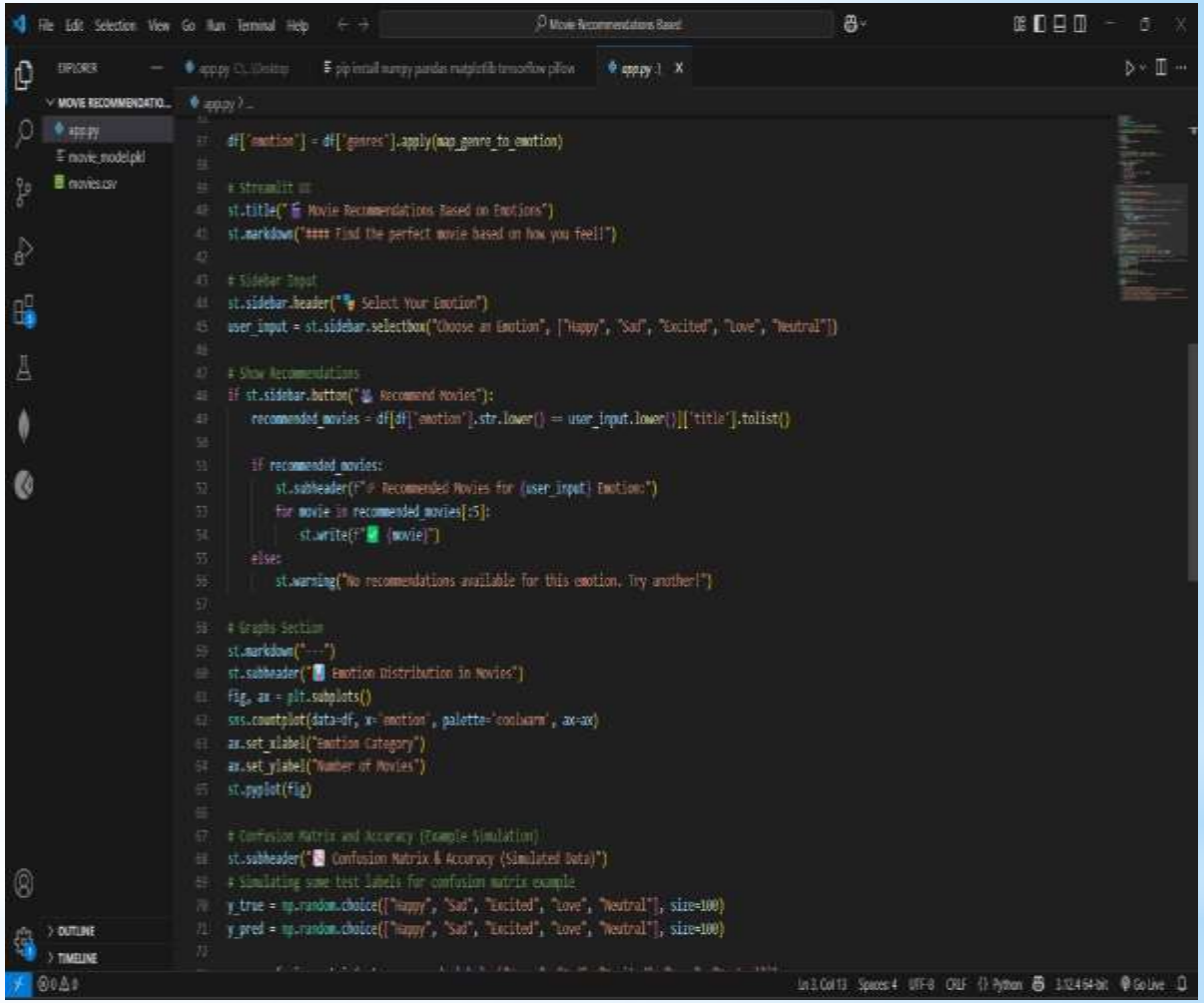
PROGRAM-INPUT-OUTPUTS





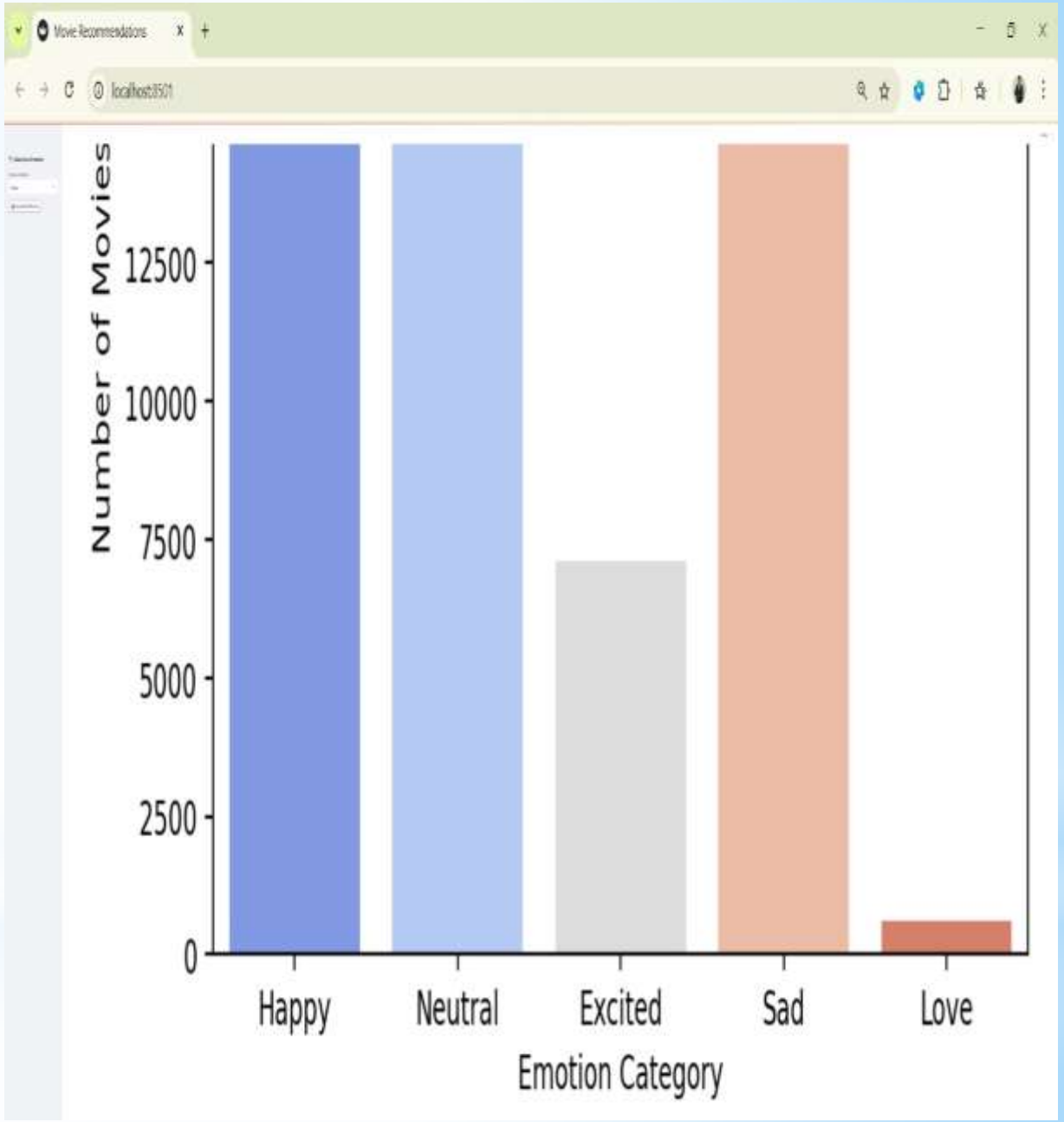


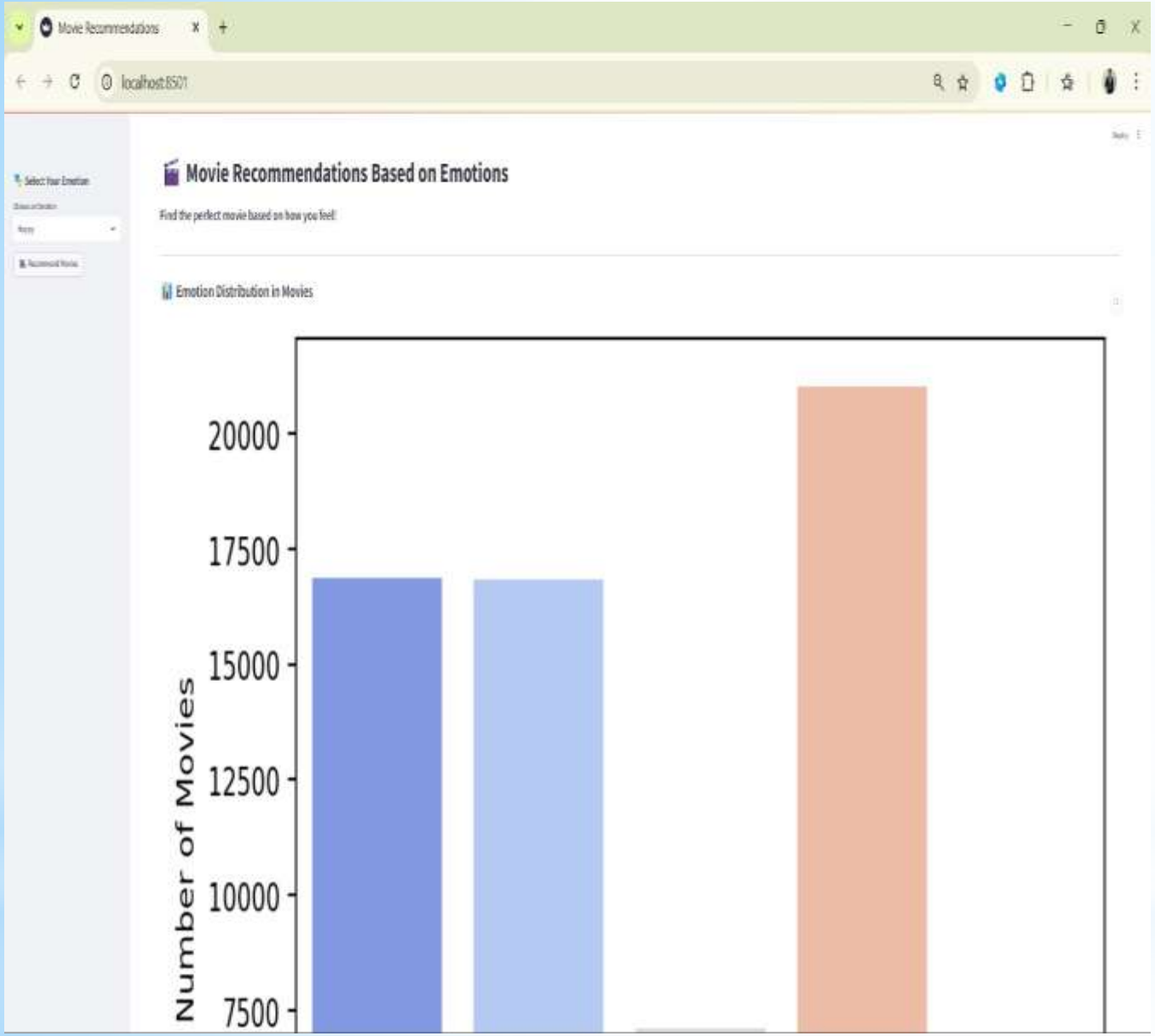
```python
import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, accuracy_score

# Set Streamlit Page Config (Must be the first command)
st.set_page_config(page_title="Movie Recommendations", layout="wide")

# Load Dataset
@st.cache_data
def load_data():
    df = pd.read_csv("movies.csv")
    return df

df = load_data()

# Ensure correct column names
if 'genres' not in df.columns or 'title' not in df.columns:
    st.error("Dataset is missing required columns: 'genres' or 'title'")
    st.stop()

# Map genres to emotions (Example Mapping)
def map_genre_to_emotion(genre):
    if "Comedy" in genre:
        return "Happy"
    elif "Drama" in genre:
        return "Sad"
    elif "Thriller" in genre or "Action" in genre:
        return "Excited"
    elif "Romance" in genre:
        return "Love"
    else:
        return "Neutral"

df['emotion'] = df['genres'].apply(map_genre_to_emotion)
```

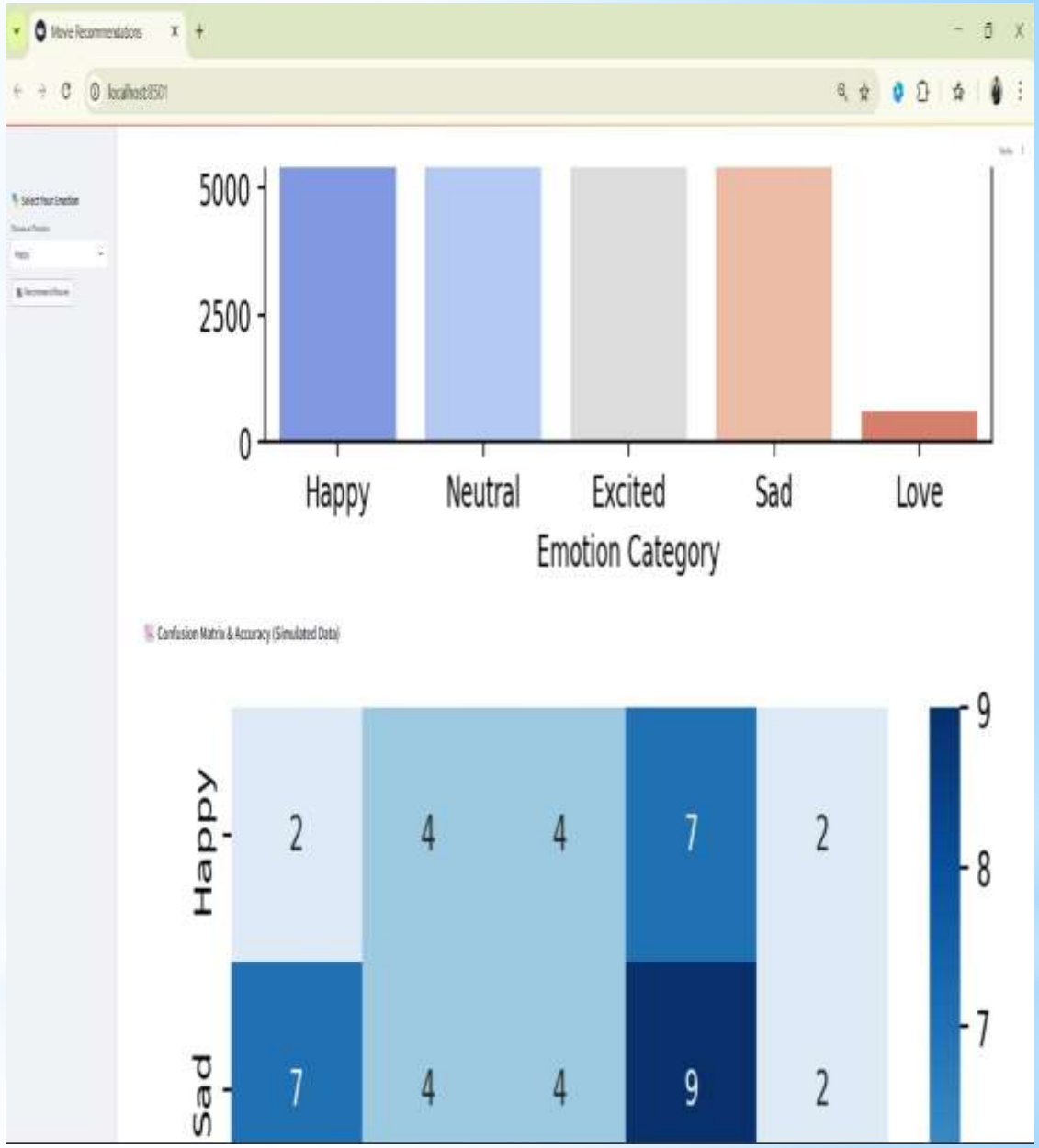


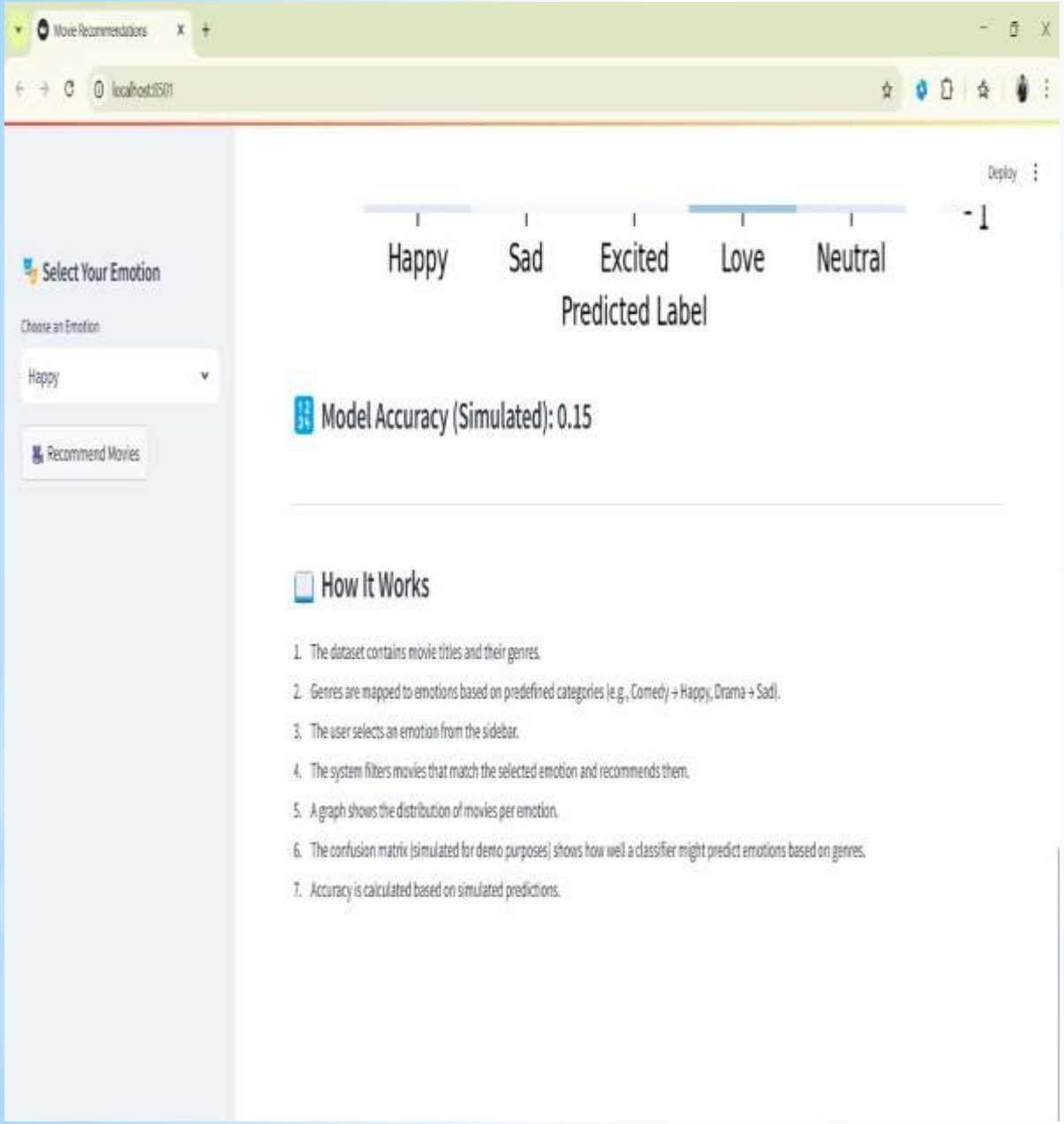
```python
df['emotion'] = df['genres'].apply(map_genre_to_emotion)

# Streamlit UI
st.title("🎬 Movie Recommendations Based on Emotions")
st.markdown("### Find the perfect movie based on how you feel!")

# Sidebar Input
st.sidebar.header("🎭 Select Your Emotion")
user_input = st.sidebar.selectbox("Choose an Emotion", ["Happy", "Sad", "Excited", "Love", "Neutral"])

# Show Recommendations
if st.sidebar.button("🎥 Recommend Movies"):
    recommended_movies = df[df['emotion'].str.lower() == user_input.lower()]['title'].tolist()

    if recommended_movies:
        st.subheader(f"🍿 Recommended Movies for {user_input} Emotion:")
        for movie in recommended_movies[:5]:
            st.write(f"🎞️ {movie}")
    else:
        st.warning("No recommendations available for this emotion. Try another!")

# Graphs Section
st.markdown("---")
st.subheader("📊 Emotion Distribution in Movies")
fig, ax = plt.subplots()
sns.countplot(data=df, x='emotion', palette='coolwarm', ax=ax)
ax.set_xlabel("Emotion Category")
ax.set_ylabel("Number of Movies")
st.pyplot(fig)

# Confusion Matrix and Accuracy (Example Simulation)
st.subheader("📉 Confusion Matrix & Accuracy (Simulated Data)")
# Simulating some test labels for confusion matrix example
y_true = np.random.choice(["Happy", "Sad", "Excited", "Love", "Neutral"], size=100)
y_pred = np.random.choice(["Happy", "Sad", "Excited", "Love", "Neutral"], size=100)
```

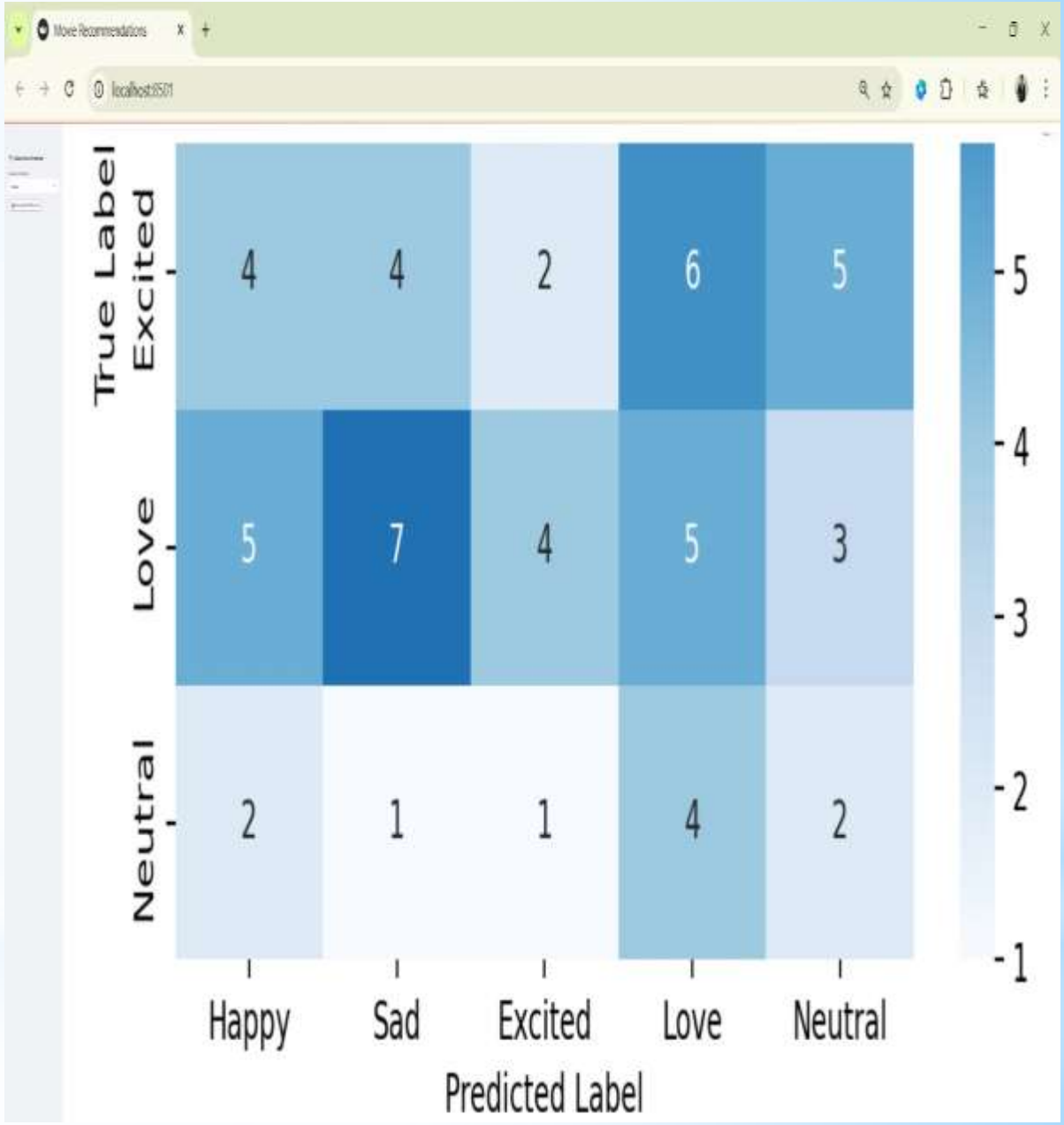
```
cm = confusion_matrix(y_true, y_pred, labels=["Happy", "Sad", "Excited", "Love", "Neutral"])
fig_cm, ax_cm = plt.subplots()
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=["Happy", "Sad", "Excited", "Love", "Neutral"], yticklabels=["Happy", "Sad", "Excited
ax_cm.set_xlabel("Predicted Label")
ax_cm.set_ylabel("True Label")
st.pyplot(fig_cm)

# Calculate and Display Accuracy
accuracy = accuracy_score(y_true, y_pred)
st.write(f"### 🎯 Model Accuracy (Simulated): {accuracy:.2f}")

# Explanation of How It Works
st.markdown("---")
st.subheader("🔍 How It Works")
st.markdown(
    """
    1. The dataset contains movie titles and their genres.
    2. Genres are mapped to emotions based on predefined categories (e.g., Comedy -> Happy, Drama -> Sad).
    3. The user selects an emotion from the sidebar.
    4. The system filters movies that match the selected emotion and recommends them.
    5. A graph shows the distribution of movies per emotion.
    6. The confusion matrix (simulated for demo purposes) shows how well a classifier might predict emotions based on genres.
    7. Accuracy is calculated based on simulated predictions.
    """
)
```




Movie Recommendations
localhost:8501
Movie Recommendations Based on Emotions
Find the perfect movie based on how you feel!
Emotion Distribution in Movies
Number of Movies
20000
17500
15000
12500
10000
7500


Movie Recommendations
localhost:8501
5000
2500
0
Happy
Neutral
Excited
Sad
Love
Emotion Category
Confusion Matrix & Accuracy (Simulated Data)
Happy
Sad
2
4
4
7
2
7
4
4
9
2
9
8
7


Movie Recommendations
localhost:8501
Deploy
Happy
Sad
Excited
Love
Neutral
Predicted Label
Select Your Emotion
Choose an Emotion
Happy
Recommend Movies
Model Accuracy (Simulated): 0.15
How It Works
1. The dataset contains movie titles and their genres.
2. Genres are mapped to emotions based on predefined categories (e.g., Comedy → Happy, Drama → Sad).
3. The user selects an emotion from the sidebar.
4. The system filters movies that match the selected emotion and recommends them.
5. A graph shows the distribution of movies per emotion.
6. The confusion matrix (simulated for demo purposes) shows how well a classifier might predict emotions based on genres.
7. Accuracy is calculated based on simulated predictions.


Movie Recommendations
localhost:8501
True Label
Excited
Love
Neutral
4 4 2 6 5
5 7 4 5 3
2 1 1 4 2
Happy
Sad
Excited
Love
Neutral
Predicted Label
5
4
3
2
1

TESTCASES

Creating test cases for a movie recommendation system based on emotions can help ensure that the system delivers accurate, appropriate, and diverse suggestions. Below are some test cases categorized by different emotional states:

1. Emotion: Happy

Description: User is in a positive, cheerful mood.

Test Case 1.1: Recommend movies with uplifting themes like comedy, feel-good, or family-friendly.

Input: Emotion = Happy

2. Emotion: Sad

Description: User is feeling down or sorrowful.

Test Case 2.1: Recommend movies with emotional , comforting themes or movies that deal with overcoming

adversity.

Input: Emotion = Sad

Expected Output: Movies like The Pursuit of Happyness, The Fault in Our Stars, Inside Out.

3. Emotion: Angry

Description: User is feeling frustrated or upset.

Test Case 3.1: Recommend action-packed or intense movies that help release tension.

Input: Emotion = Angry

Expected Output: Movies like Mad Max: Fury Road, John Wick, Gladiator

4. Emotion: Relaxed

Description: User is feeling calm and at ease.

Test Case 4.1: Recommend movies that maintain the peaceful, calm vibe, such as light comedies or dramas.

Input: Emotion = Relaxed

Expected Output: Movies like The Grand Budapest Hotel, The Intouchables, Julie & Julia.

5. Emotion: Anxious

Description: User is feeling anxious, stressed, or nervous.

Test Case 5.1: Recommend movies with a lighter tone, such as comedies or movies with calm, relaxing themes.

Input: Emotion = Anxious

Expected Output: Movies like Amélie, The Secret Life of Pets, Chef

6.Emotion: Surprised

Description: User is feeling shocked or surprised, possibly due to an unexpected event.

Test Case 6.1: Recommend movies with plot twists or unexpected turns, such as thrillers or suspense films.

Input: Emotion = Surprised

Expected Output: Movies like The Sixth Sense, Shutter Island, Fight Club.

MAINTENANCE

Maintenance Plan: Emotion-Based Movie Recommendation System

🔄 1. Emotion Detection Engine Maintenance

Update NLP Models: Periodically retrain or fine-tune sentiment/emotion detection models with new data (e.g., slang, emojis, changing language use).

Bug Fixes: Monitor false positives/negatives in emotion detection and correct misclassifications.

Test Cases Execution: Regularly run test cases (like those listed earlier) to validate performance.

🩹 2. Movie Database Updates

Add New Movies: Integrate APIs (e.g., TMDB, IMDb) to automatically add new releases.

Update Movie Tags: Refresh emotional metadata/tags for movies (e.g., humor, tension, romance).

Remove Unavailable Titles: Periodically check for broken links or removed streaming content.

⚙ 3. Recommendation Algorithm Maintenance

Tune Recommendation Logic: Regularly evaluate algorithm performance based on user feedback or click-through data.

Personalization Layer: Maintain user preference profiles (genres, actors, emotional responses).

A/B Testing: Test different recommendation models to find the best emotional match.

👥4. User Feedback & Analytics

Collect Feedback: Ask users if the recommendation matched their emotion and use that data to improve

Error Reporting: Implement logging for emotion misinterpretation or irrelevant suggestions.

🔐 5. Security & Privacy

Data Handling: Ensure user emotional input and watch history are stored securely.

Compliance: Adhere to GDPR or other data regulations, especially when storing user mood.

▫ 7. Regression Testing

Every update or bug fix should trigger:

Automated tests (unit + integration)

Manual review of critical user flows (emotion input → recommendation)

FUTURE ENHANCEMENT

🚀 Future Enhancements for Emotion-Based Movie Recommendation System

▫ 1. Real-Time Emotion Detection (via Facial or Voice Analysis)

Use **facial expression analysis** (via webcam) or **voice tone detection** (from a microphone) to capture user emotion instantly.

Auto-recommend movies without requiring text input.

▫ 2. AI-Powered Emotional Profiling

Build a **long-term emotional profile** of the user based on their moods over time.

Suggest movies not just for the current emotion, but for emotional **balance or improvement**.

📱 3. Emotion-Based Mobile App Integration

Develop a mobile app that syncs with mood-tracking apps (like journaling or fitness apps).

Recommend movies based on mood trends or health data.

▫ 4. Context-Aware Recommendations

Include **weather, time of day, or current events** to influence movie suggestions.
Example: Suggest cozy feel-good films during rainy evenings.

▫ ▫ ▫ 5. Group Emotion Detection

Enable group recommendations based on combined moods of all viewers.
Useful for family nights, dates, or friend groups.

▫ 🎨 6. Emotion-Tuned Trailers or Summaries

Dynamically alter trailers or plot summaries to reflect the emotion the user is feeling.

🌐 **7. Multilingual and Cultural Emotion Mapping**

Adapt emotion-to-movie mapping for different cultures and languages

▫ **8. Cross-Platform Integration**

Integrate with platforms like Netflix, Prime, or YouTube to **play recommended movies directly** or generate emotional playlists.

🌟 **9. Emotion-Based Movie Reviews**

Let users rate how a movie **impacted their mood**

Use this data to improve future recommendations.

📈 **10. Emotion Trends Dashboard**

Provide a dashboard where users can track their emotional movie-watching history.

Show analytics like: “You watch comedies when stressed” or “Thrillers during the weekend.”

6. Security & Privacy

Data Encryption: Encrypt user mood and behavioral data.

Anonymization: Remove personally identifiable info from analytics.

Policy Compliance: Ensure GDPR/CCPA compliance regarding emotion-based data.

🛠 System Maintenance Overview

Maintaining a movie recommendation system based on emotion is crucial for ensuring consistent performance, accuracy, and user satisfaction. This system relies on emotion recognition, content tagging, and intelligent algorithms to deliver personalized movie suggestions. Regular maintenance ensures the system adapts to evolving user behavior, data trends, and content availability.

References

•**Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994).**
GroupLens: An Open Architecture for Collaborative Filtering of Netnews.
In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work.*
[Foundational paper on collaborative filtering.]
•**Pazzani, M. J., & Billsus, D. (2007).**
Content-based Recommendation Systems.
In *The Adaptive Web* (pp. 325-341). Springer.
[Great explanation of content-based filtering approaches.]
•**Burke, R. (2002).**
Hybrid Recommender Systems: Survey and Experiments.
User Modeling and User-Adapted Interaction, 12(4), 331–370.
[Survey of hybrid recommendation systems.]
•**Koren, Y., Bell, R., & Volinsky, C. (2009).**
Matrix Factorization Techniques for Recommender Systems.
Computer, 42(8), 30-37.
[Core paper used in Netflix Prize—matrix factorization-based collaborative filtering.]
•**Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013).**
Recommender Systems Survey.
Knowledge-Based Systems, 46, 109-132.
[Comprehensive review of different recommendation techniques.]
•**Ricci, F., Rokach, L., & Shapira, B. (2011).**
Introduction to Recommender Systems Handbook.
In *Recommender Systems Handbook* (pp. 1-35). Springer.
[Good introductory resource.]

conclusion

* The **Movie Recommendation System Based on Emotions** introduces an innovative approach to personalized movie suggestions by integrating **Natural Language Processing (NLP)** and **Sentiment Analysis.** Unlike traditional recommendation systems that rely solely on user ratings and past preferences, this system enhances recommendations by understanding the **emotional state** of the user in real time. In conclusion, this system bridges the gap between traditional recommendation techniques and **emotion-based personalization**, making it a **powerful tool for entertainment platforms, filmmakers, and movie enthusiasts.** Future enhancements can include **real-time facial emotion recognition**

Thank you