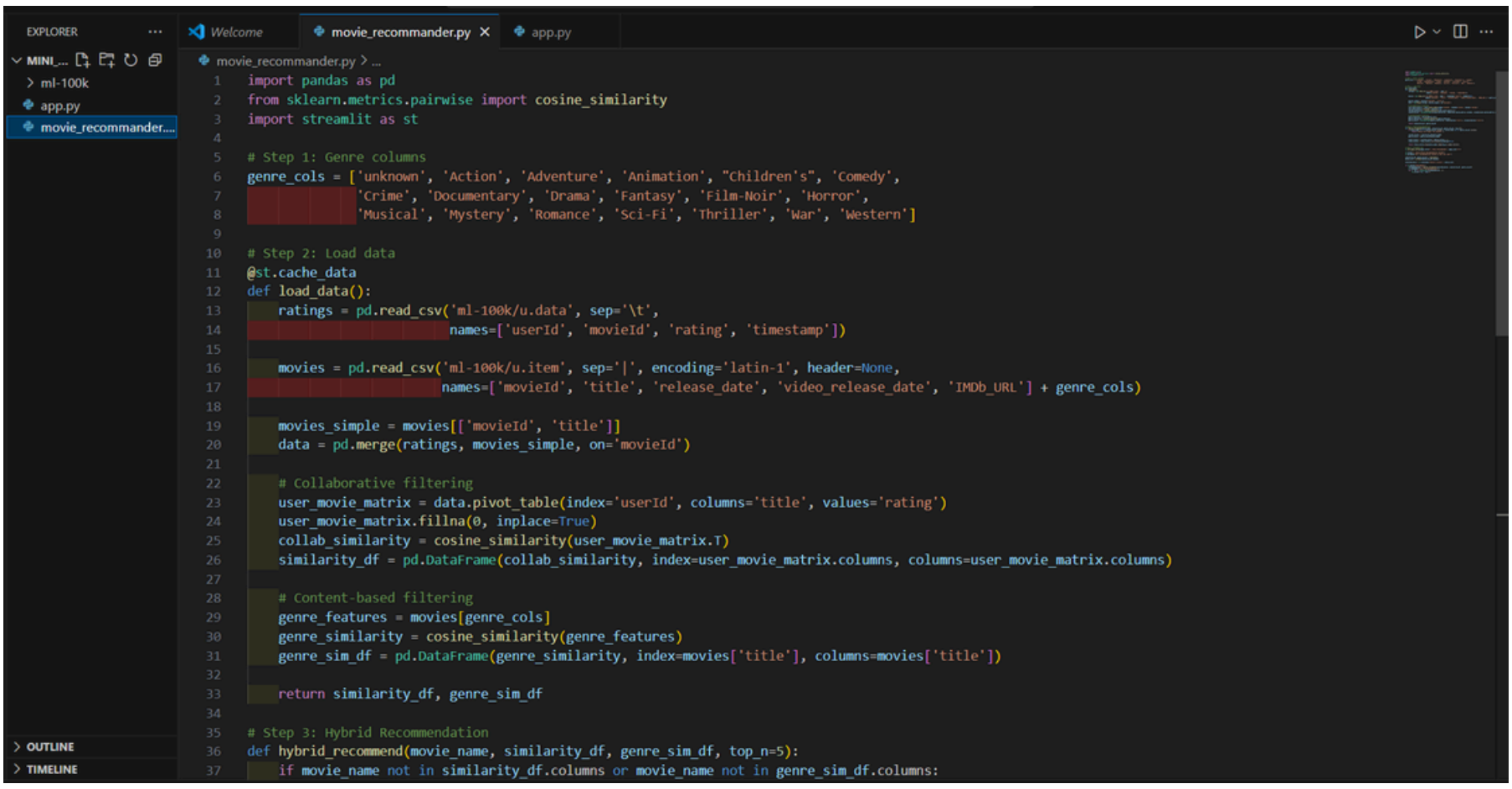
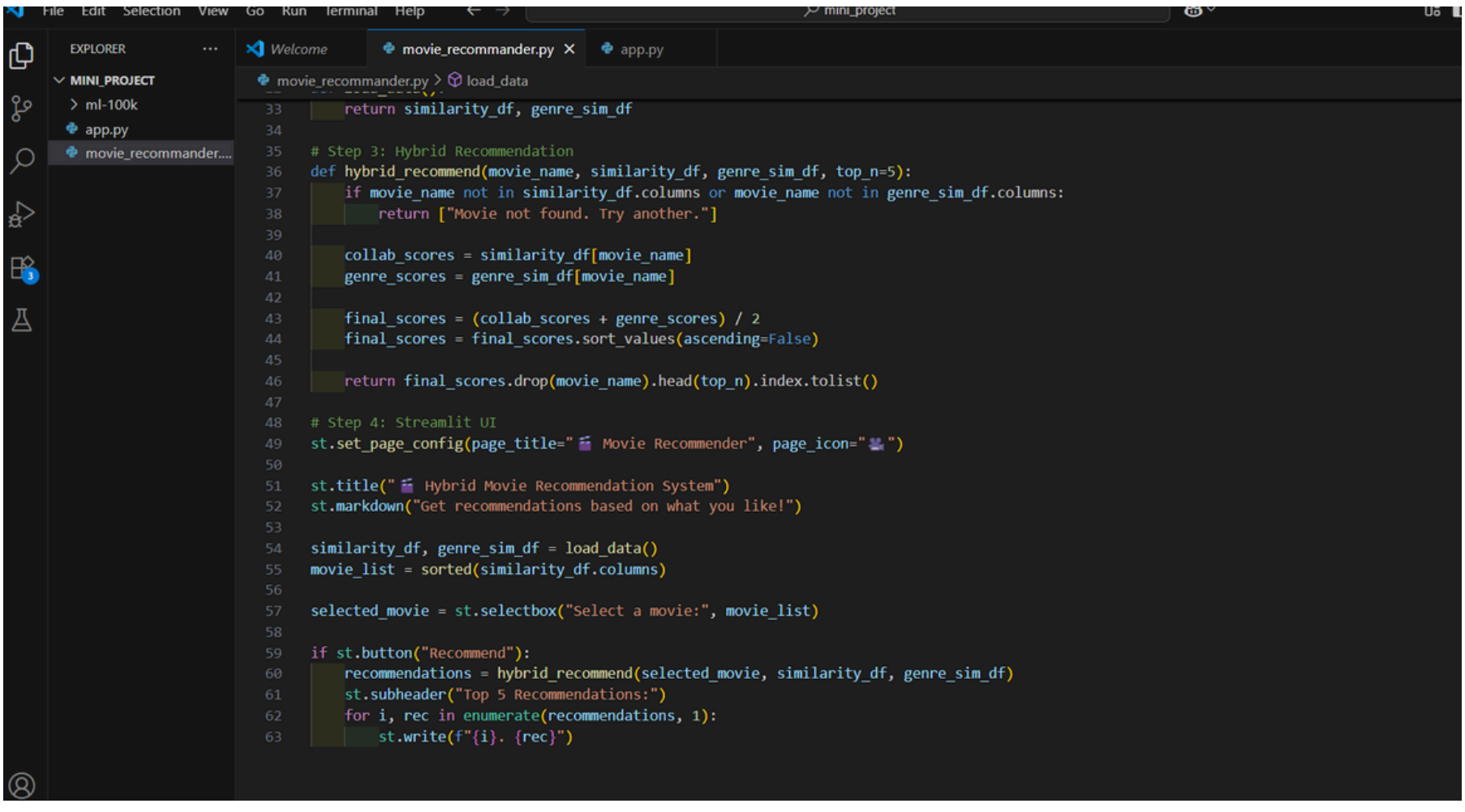
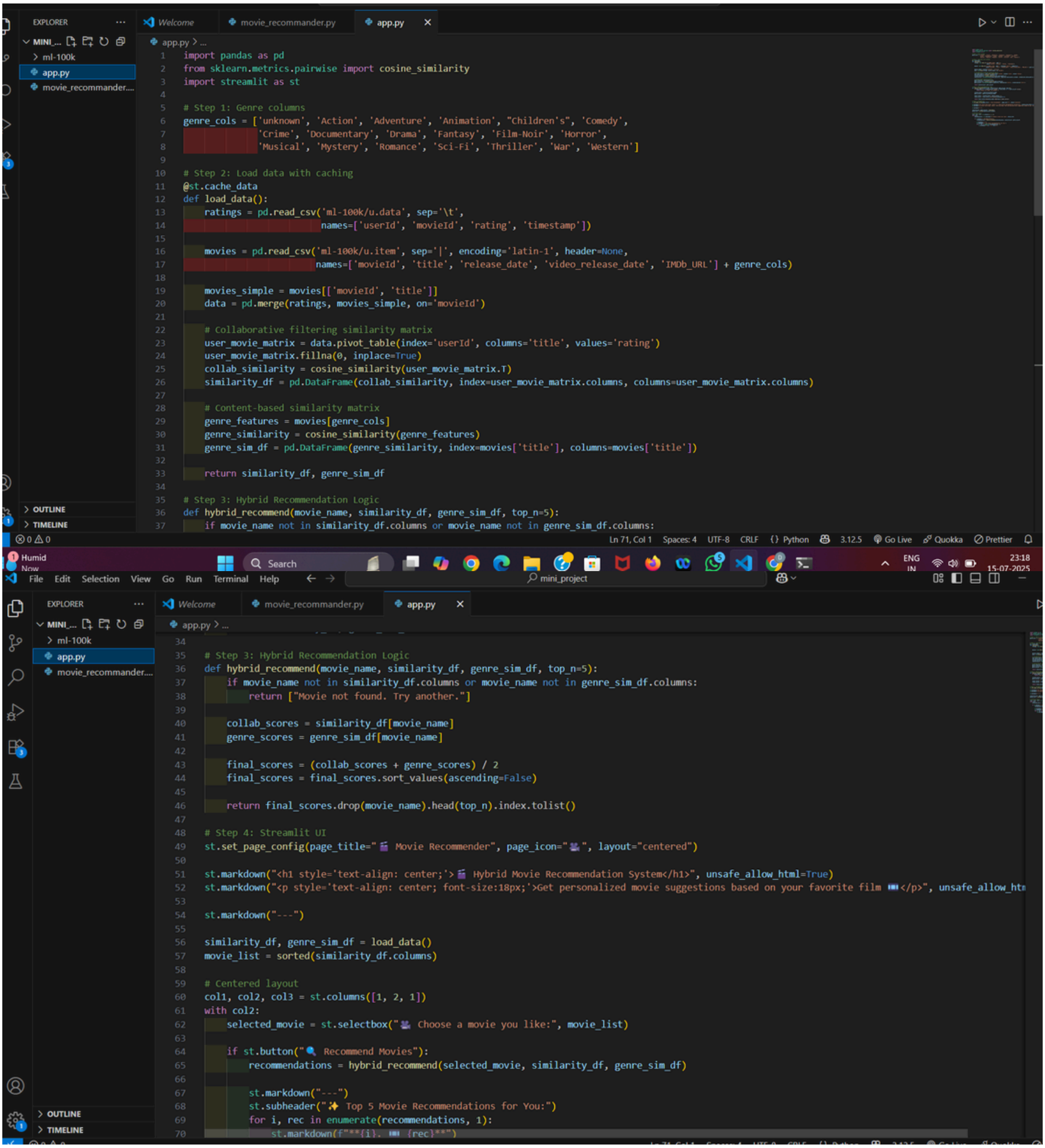
Movie Recommendation System








```python
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
import streamlit as st

# Step 1: Genre columns
genre_cols = ['unknown', 'Action', 'Adventure', 'Animation', "Children's", 'Comedy',
              'Crime', 'Documentary', 'Drama', 'Fantasy', 'Film-Noir', 'Horror',
              'Musical', 'Mystery', 'Romance', 'Sci-Fi', 'Thriller', 'War', 'Western']

# Step 2: Load data with caching
@st.cache_data
def load_data():
    ratings = pd.read_csv('ml-100k/u.data', sep='\t',
                          names=['userId', 'movieId', 'rating', 'timestamp'])

    movies = pd.read_csv('ml-100k/u.item', sep='|', encoding='latin-1', header=None,
                         names=['movieId', 'title', 'release_date', 'video_release_date', 'IMDb_URL'] + genre_cols)

    movies_simple = movies[['movieId', 'title']]
    data = pd.merge(ratings, movies_simple, on='movieId')

    # Collaborative filtering similarity matrix
    user_movie_matrix = data.pivot_table(index='userId', columns='title', values='rating')
    user_movie_matrix.fillna(0, inplace=True)
    collab_similarity = cosine_similarity(user_movie_matrix.T)
    similarity_df = pd.DataFrame(collab_similarity, index=user_movie_matrix.columns, columns=user_movie_matrix.columns)

    # Content-based similarity matrix
    genre_features = movies[genre_cols]
    genre_similarity = cosine_similarity(genre_features)
    genre_sim_df = pd.DataFrame(genre_similarity, index=movies['title'], columns=movies['title'])

    return similarity_df, genre_sim_df

# Step 3: Hybrid Recommendation Logic
def hybrid_recommend(movie_name, similarity_df, genre_sim_df, top_n=5):
    if movie_name not in similarity_df.columns or movie_name not in genre_sim_df.columns:
        return ["Movie not found. Try another."]

    collab_scores = similarity_df[movie_name]
    genre_scores = genre_sim_df[movie_name]

    final_scores = (collab_scores + genre_scores) / 2
    final_scores = final_scores.sort_values(ascending=False)

    return final_scores.drop(movie_name).head(top_n).index.tolist()

# Step 4: Streamlit UI
st.set_page_config(page_title="🎬 Movie Recommender", page_icon="🎥", layout="centered")

st.markdown("<h1 style='text-align: center;'>🎬 Hybrid Movie Recommendation System</h1>", unsafe_allow_html=True)
st.markdown("<p style='text-align: center; font-size:18px;'>Get personalized movie suggestions based on your favorite film 🎞️</p>", unsafe_allow_htm

st.markdown("---")

similarity_df, genre_sim_df = load_data()
movie_list = sorted(similarity_df.columns)

# Centered layout
col1, col2, col3 = st.columns([1, 2, 1])
with col2:
    selected_movie = st.selectbox("🎥 Choose a movie you like:", movie_list)

    if st.button("🔍 Recommend Movies"):
        recommendations = hybrid_recommend(selected_movie, similarity_df, genre_sim_df)

        st.markdown("---")
        st.subheader("✨ Top 5 Movie Recommendations for You:")
        for i, rec in enumerate(recommendations, 1):
            st.markdown(f"**{i}. 🎞️ {rec}**")
```


Hybrid Movie Recommendation System
Get personalized movie suggestions based on your favorite film
Choose a movie you like:
101 Dalmatians (1996)
Recommend Movies
Top 5 Movie Recommendations for You:
1. First Kid (1996)
2. Santa Clause, The (1994)
3. Home Alone (1990)
4. Matilda (1996)