IICMR-MCA

Reaching New Heights

IICMR

Inst. Code: MC-6154

ATSS's

Institute of Industrial & Computer Management & Research, Nigdi

Academic Year 2023-2024

MCA II

MASTER OF COMPUTER APPLICATION SAVITRIBAI PHULE PUNE UNIVERSITY

PROJECT REPORT

ON

"Film Finder"

By

Manas Mishra

ABSTRACT

With the exponential growth of digital content, the need for effective recommendation systems has become paramount. In the realm of entertainment, particularly movies, content-based recommendation systems have gained prominence for providing personalized suggestions based on individual user preferences. This paper presents an in-depth exploration of a Content-Based Movie Recommendation System, emphasizing its architecture, key components, and underlying algorithms.

The feature extraction process involves analyzing movie attributes such as genre, director, actors, and user-specific data like viewing history and ratings. Additionally, the utilization of advanced machine learning algorithms, such as natural language processing for sentiment analysis of user reviews, enhances the system's ability.

DECLARATION

I solemnly declare that the report of the project work entitled “**Film Finder**”, is based on my own work carried out during the Course of my study under the supervision of **Mrs. Snigdha Shukla**.

I assert that the statement made, and conclusions drawn are an outcome of the project work. I further declare that to the best of my knowledge and belief that the project report does not contain any part of work submitted by any other student of this university or any other university.

Date: yours sincerely

Place: Pune

CHAPTER 1 : INTRODUCTION

1.1 Existing System:

- Users have to physically choose movies from libraries.
- Options included reading user reviews or randomly selecting a movie.
- System suggests to us what to watch without having to put effort into deciding what to watch.

1.2 Need for System:

- **Diverse Preferences:** People have diverse tastes and preferences when it comes to movies. A recommender system helps in tailoring content recommendations to individual users, considering their viewing history, ratings, and preferences**.**
- **Vast Content Libraries:** Streaming platforms often have vast libraries of movies and TV shows. Recommender systems help users discover hidden gems and navigate through the extensive content catalog, making it easier to find content they might enjoy.
- **Enhanced User Satisfaction:** When users find content that aligns with their preferences, they are more likely to be satisfied with the platform's service. This satisfaction contributes to customer retention.
- **Differentiation:** A well-designed recommender system can set a platform apart from its competitors by offering a superior and more personalized user experience.

1.3 Scope of Work

- Accurate movie recommendations to users.
- Improves the quality of movie recommendation system, such as accuracy, quality and scalability of system through the pure approaches. This is done using content based filtering.
- Eradicate the overload of the data, recommendation system is used as information filtering tool in social networking sites. Hence, there is a huge scope of exploration in this field for improving scalability, accuracy and quality of movie recommendation systems.
- Tailor movie recommendations based on both content features and emotional sentiments.
- Explore opportunities to integrate external data sources to further refine movie recommendations.

1.4 Operating Environment

Hardware:

Processor: Intel core Processor Pentium 4
Operating System: Windows 7 and Above
RAM: 2GB and Above
Hard Disk: 40GB and Above

Software:

Frontend: Html, CSS, Javascript

Frame Work: Flask

Language: Python 3.11

API: TMDB

1.5 Detail Description of Technology Used

- **HTML**
  HTML stands for Hyper Text Mark-up Language.
  HTML is the standard markup language for creating Web pages.
  HTML describes the structure of a Web
  page.
  HTML consists of a series of elements.
  HTML elements tell the browser how to display the content.
  HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

- **CSS**
  CSS stands for Cascading Style Sheets. CSS describes how HTML elements are to be displayed on screen, paper, or in other media. CSS saves a lot of work. It can control the layout of multiple web pages all atonce. External style sheets are stored in CSS files.
- **JavaScript**
  It's the most popular programming language. It's in your browser.JavaScript also exists outside of the internet. JavaScript is Ideal for Newbies. JavaScript is Easy to Learn. You Can Create Visual Effects and Other Eye-catching. Aesthetic Features. JavaScript is Versatile.

- **Python**
  Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

- **Flask**:
  Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.

- **API**:
  API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses.

CHAPTER 2 : PROPOSED SYSTEM

2.1 Proposed System:

- Content-Based Filtering:
  It considers the characteristics of movies, such as genre, director, and actors, to recommend items that are similar to those a user has liked in the past.
- Integration with External Data Sources:
  It integrate external data sources, such as movie databases or APIs(TMDB), to enrich its catalog of movies and ensure up-to-date information.
- Sentiments Analysis Integration
  Embed sentiment analysis modules in the recommendation process to understand user emotions and reactions.
- User Interface:
  Design a user friendly interface displaying movie recommendation and sentiments on users feedback/review

2.2 Objectives of System:

- To Collect a diverse and comprehensive dataset of movie information, including details such as genre, cast, crew, plot summaries, and user reviews.
- To Extract relevant features from the movie dataset, such as genre, keywords, and metadata, to build a comprehensive representation of each movie.
- To Implement sentiment analysis on user reviews to understand the emotional tone and opinions expressed by users towards different movies.
- To Develop algorithms that analyze the content features of movies to recommend similar ones.
- To Develop an intuitive and user-friendly interface that presents movie recommendations and allows users to explore and refine their preferences.

2.3 User Requirements:

Content-Based Recommendation:

- Movie Metadata: Utilize movie features such as genre, director, actors, release year, and plot summary.

Sentiment Analysis:

- User Sentiment Analysis: Analyze user reviews, comments, or ratings to understand the sentiment associated with their preferences.

Ease of Use:

- User-Friendly Interface: Design an intuitive and user-friendly interface to make it easy for users to input preferences and explore recommended movies.

Diversity in Recommendations:

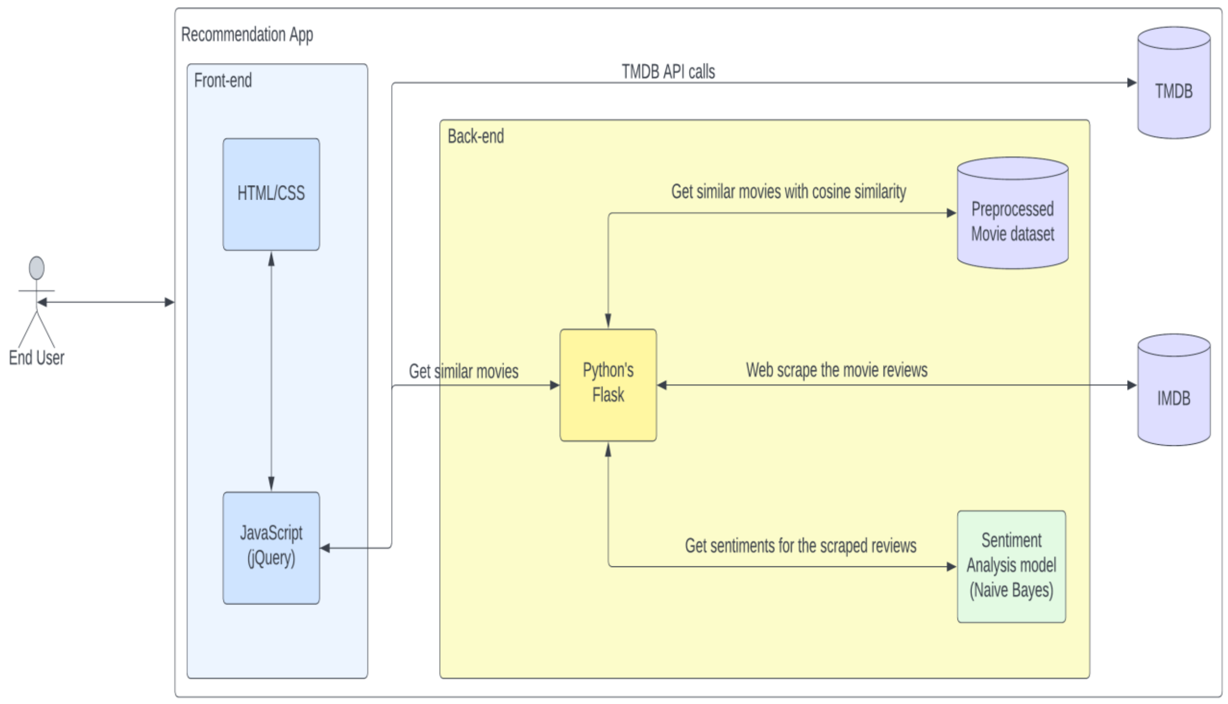
- Avoid Redundancy: Ensure that the system does not repeatedly recommend the same or similar movies.

Cross-Platform Compatibility:

- Ensure that the recommender system is accessible and functional across various platforms, such as web browsers, mobile apps, and smart TVs.

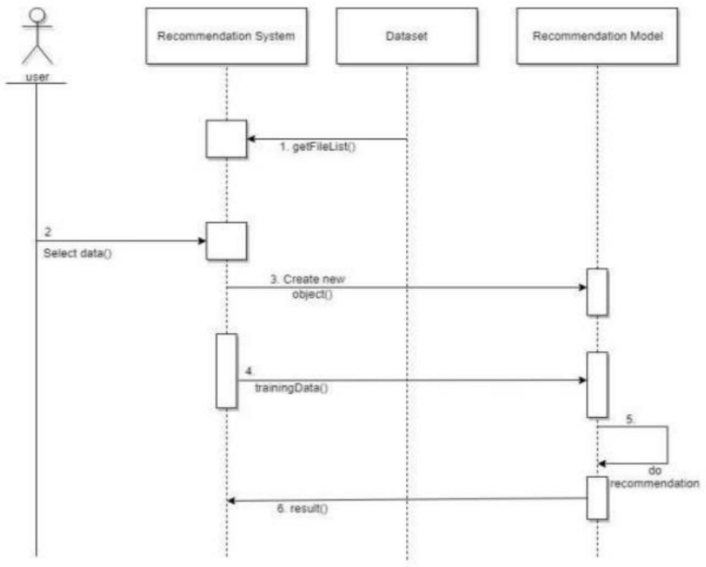
CHAPTER 3 : ANALYSIS & DESIGN

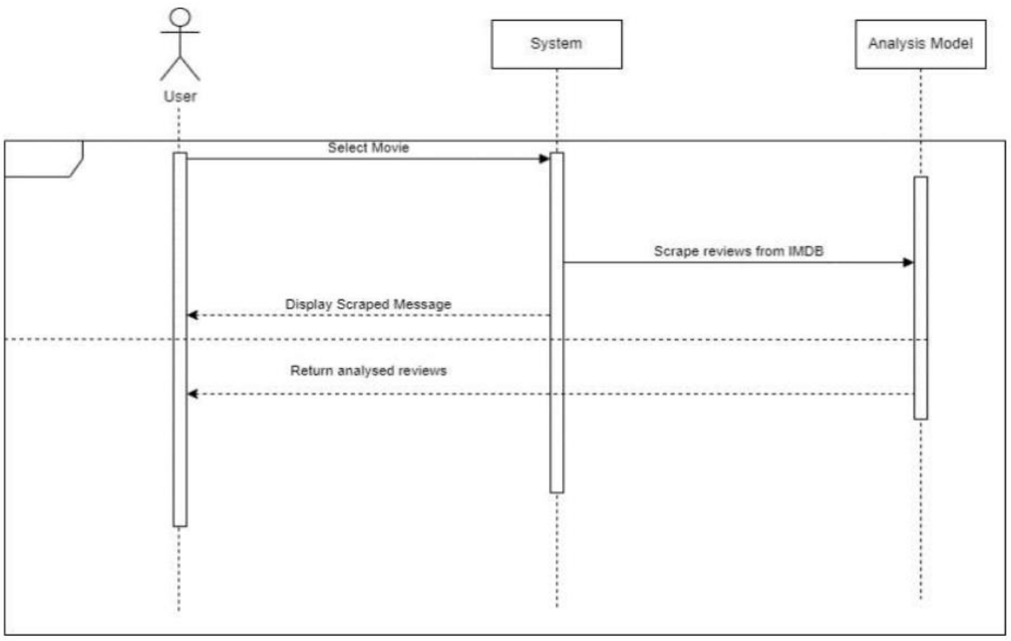
3.1 Architecture Diagram:



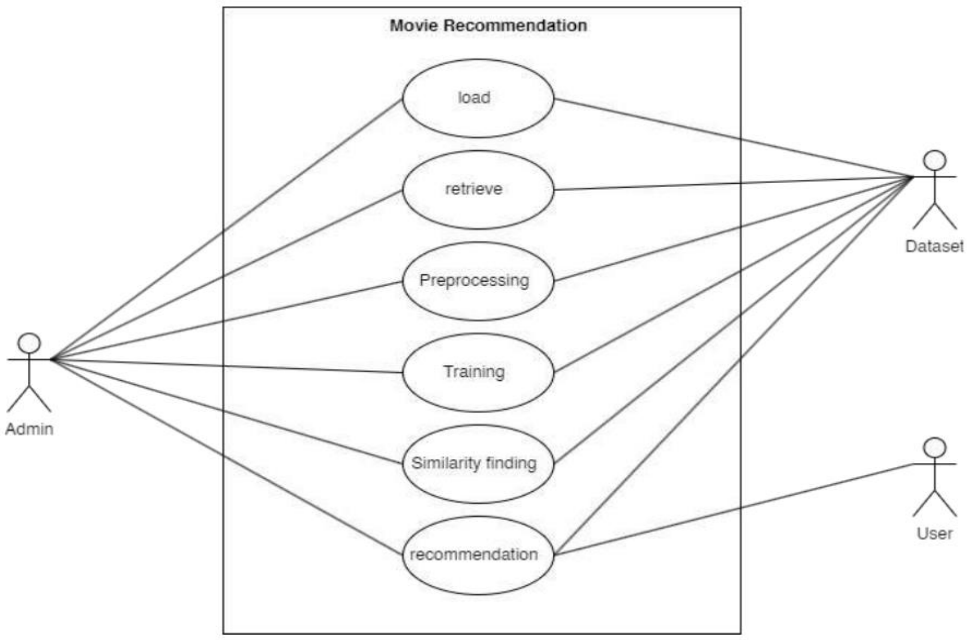
Recommendation App
Front-end
HTML/CSS
JavaScript (jQuery)
End User
TMDB API calls
TMDB
Back-end
Get similar movies with cosine similarity
Preprocessed Movie dataset
Get similar movies
Python's Flask
Web scrape the movie reviews
IMDB
Get sentiments for the scraped reviews
Sentiment Analysis model (Naive Bayes)

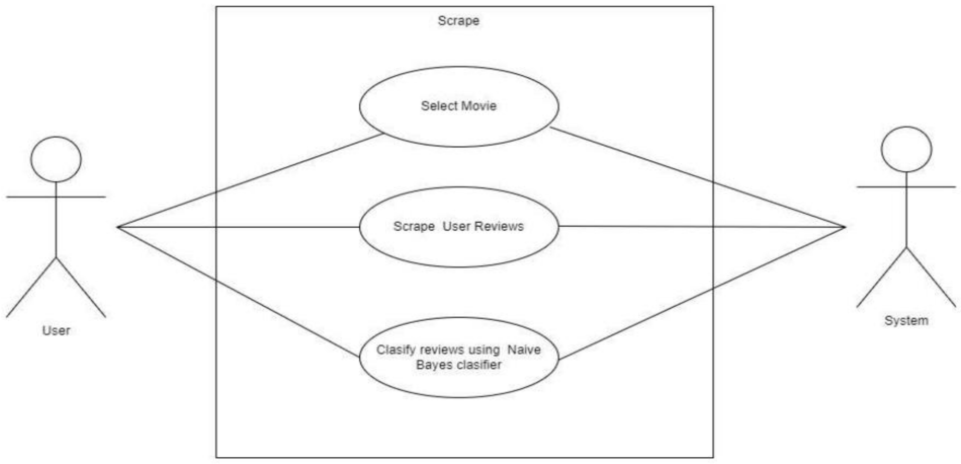
3.2 Sequence Diagram:



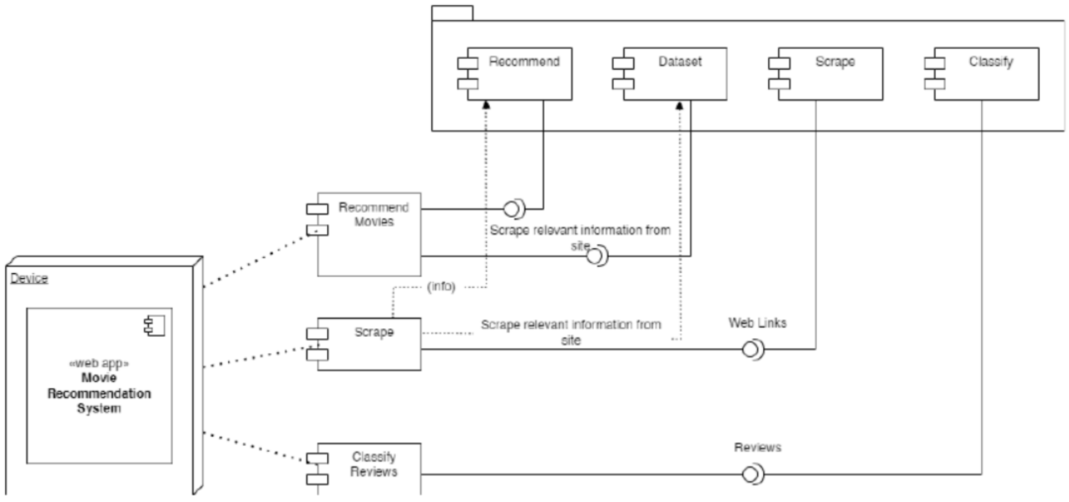


3.3 Use Case Diagrams:



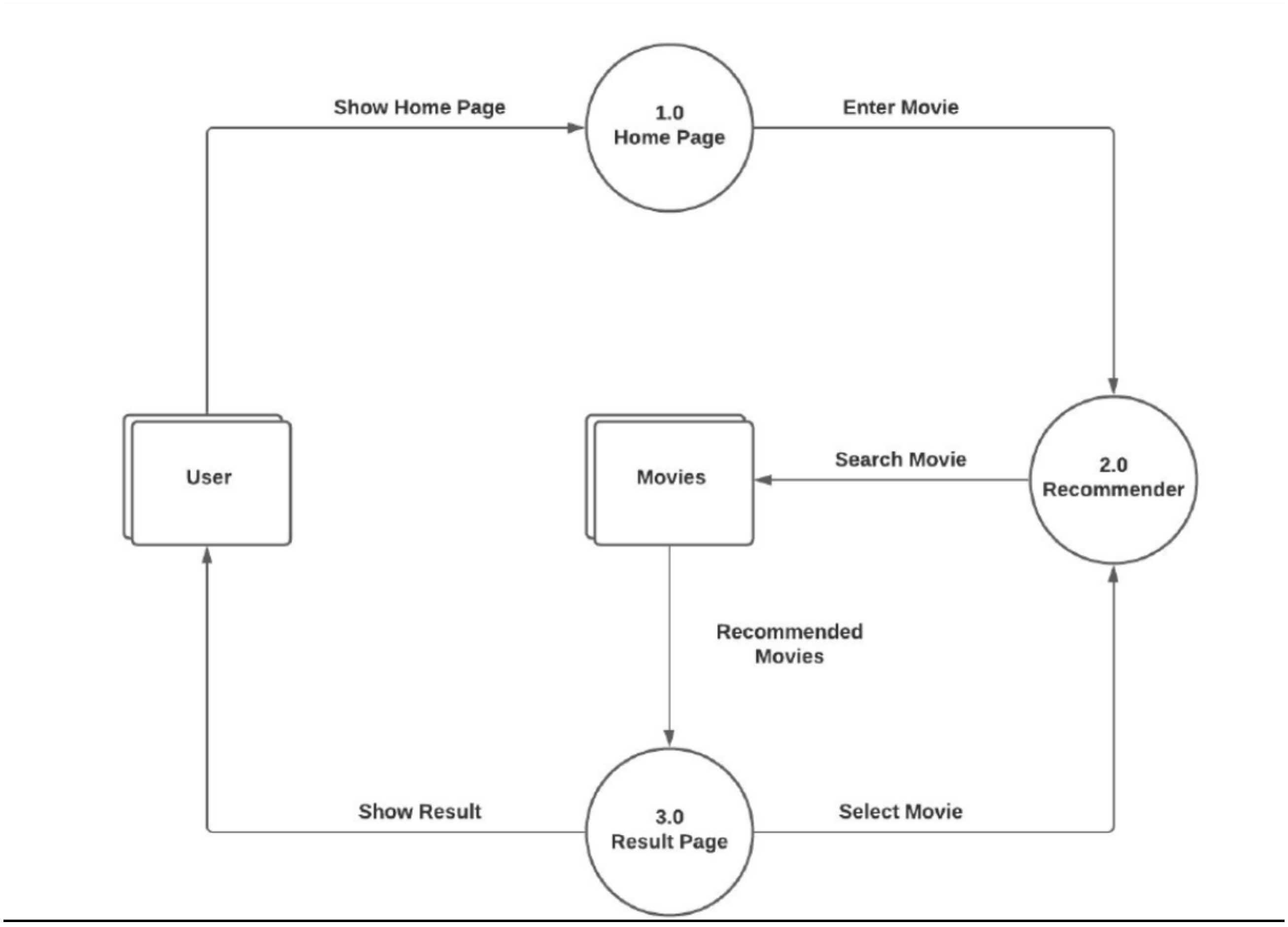


3.4 Component diagram:

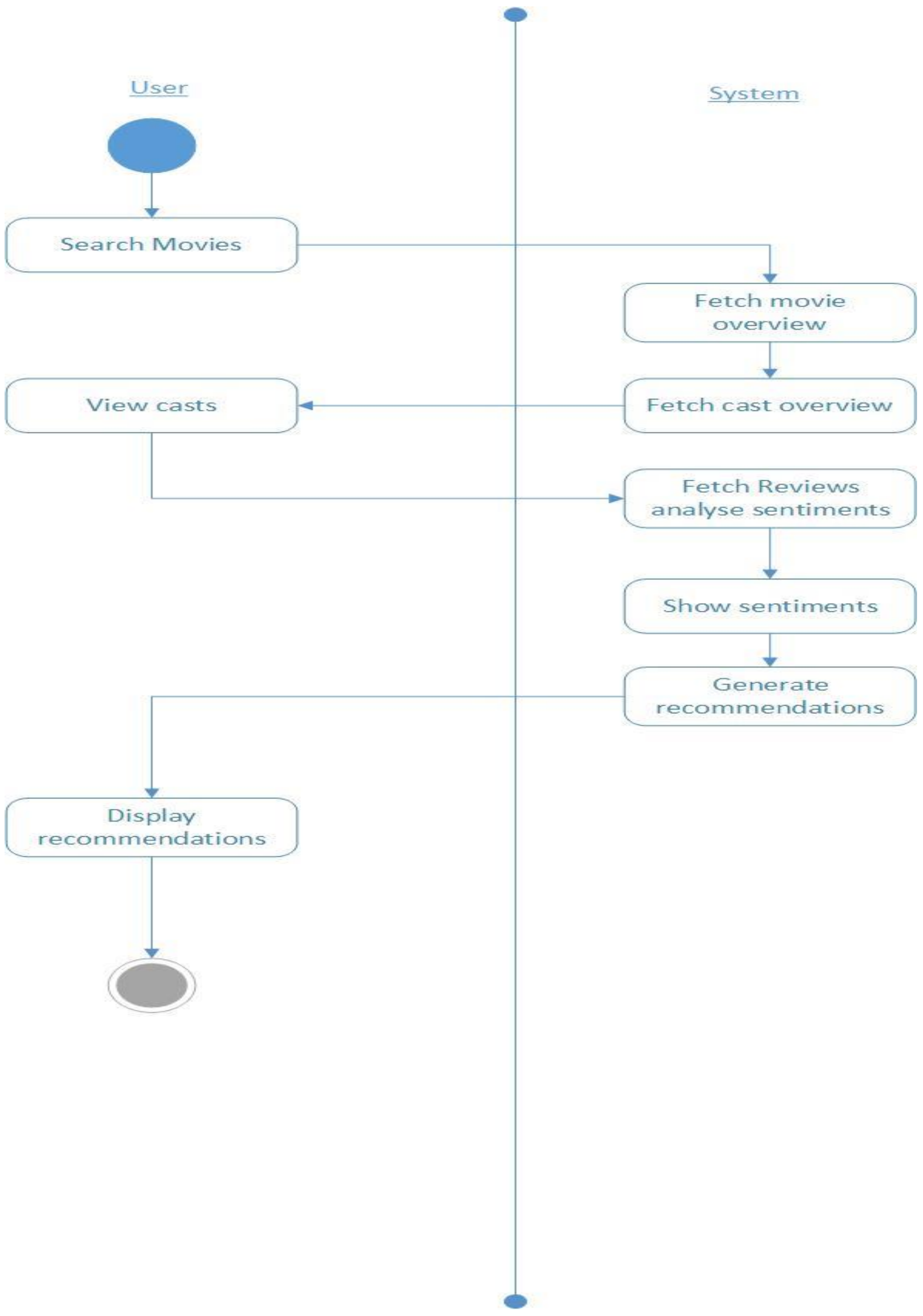


Recommend
Dataset
Scrape
Classify
Recommend Movies
Scrape relevant information from site
(info)
Device
«web app» Movie Recommendation System
Scrape
Scrape relevant information from site
Web Links
Classify Reviews
Reviews

3.5 DFD diagram:

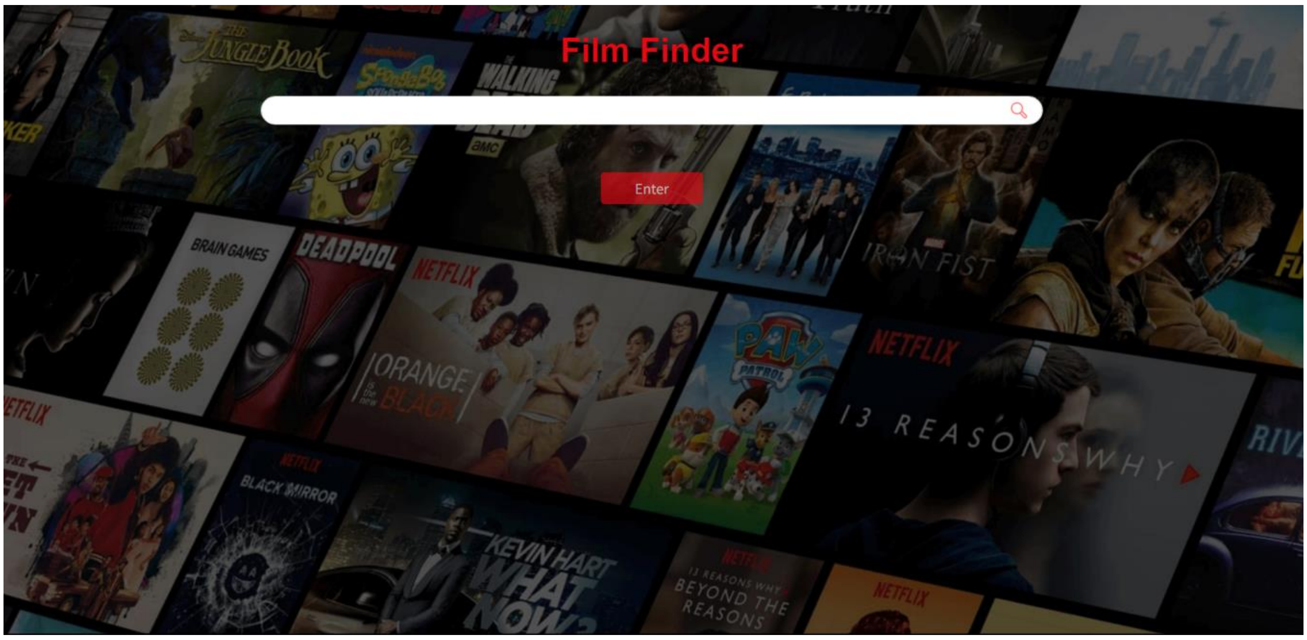


3.6 Activity Diagram:

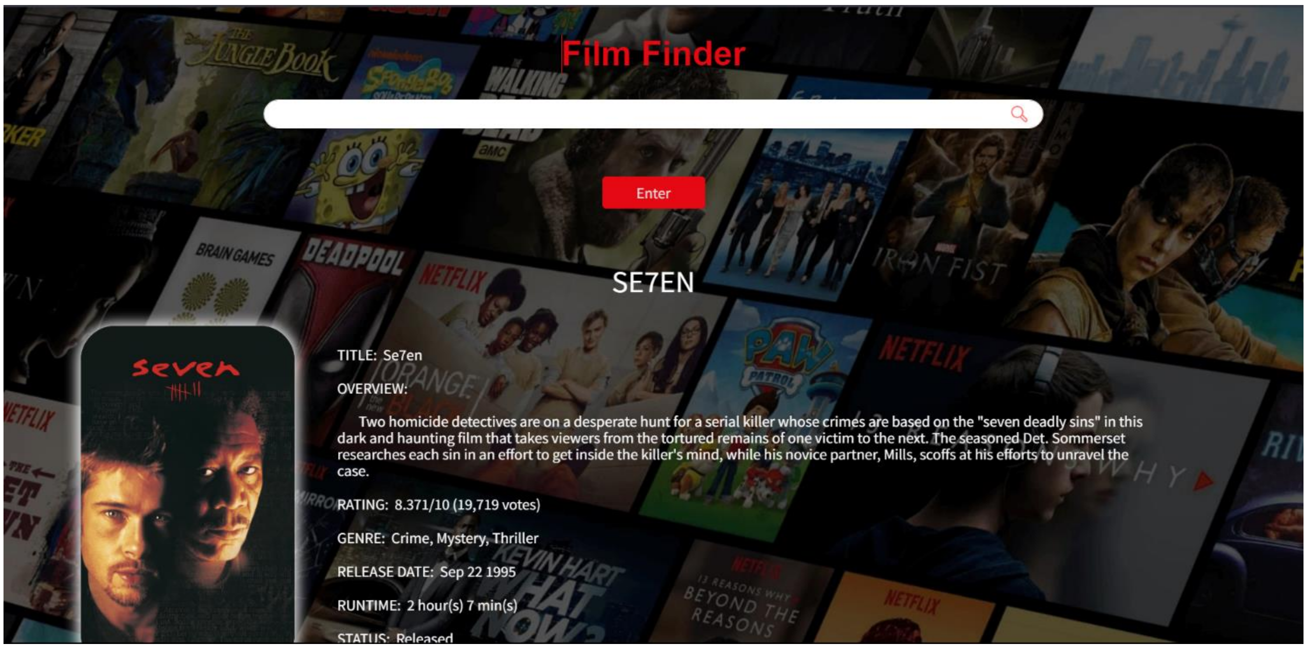


3.5 User Interface Design (Screens etc.)

<u>Home Page:</u>



Homepage with search:



SE7EN

TITLE: Se7en

OVERVIEW:

Two homicide detectives are on a desperate hunt for a serial killer whose crimes are based on the "seven deadly sins" in this dark and haunting film that takes viewers from the tortured remains of one victim to the next. The seasoned Det. Sommerset researches each sin in an effort to get inside the killer's mind, while his novice partner, Mills, scoffs at his efforts to unravel the case.

RATING: 8.371/10 (19,719 votes)

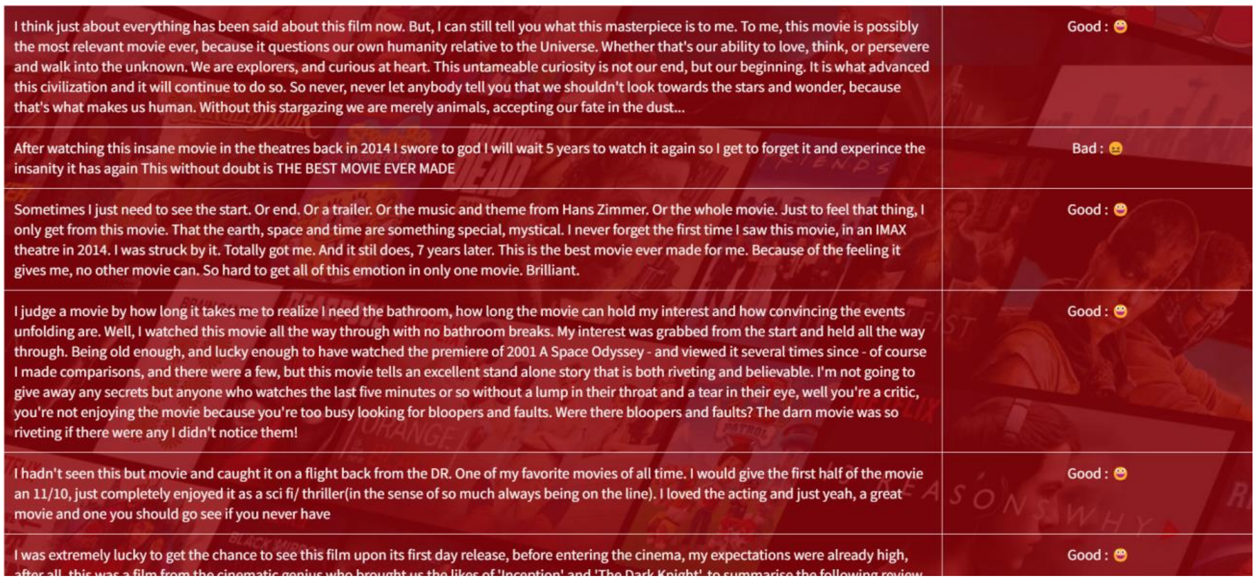
GENRE: Crime, Mystery, Thriller

RELEASE DATE: Sep 22 1995

RUNTIME: 2 hour(s) 7 min(s)

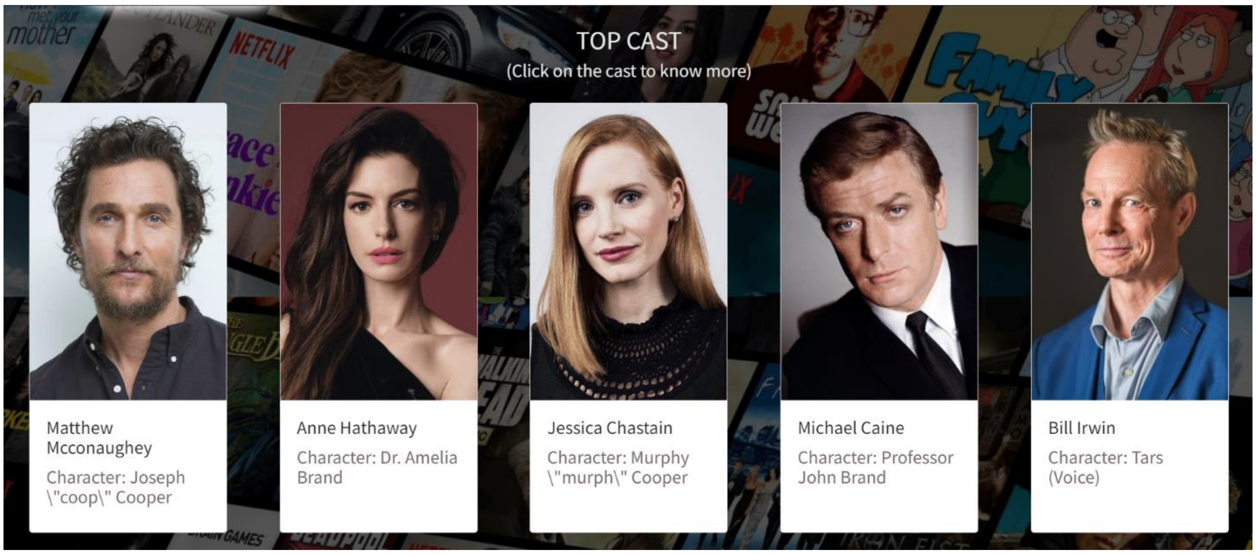
STATUS: Released

Review System:

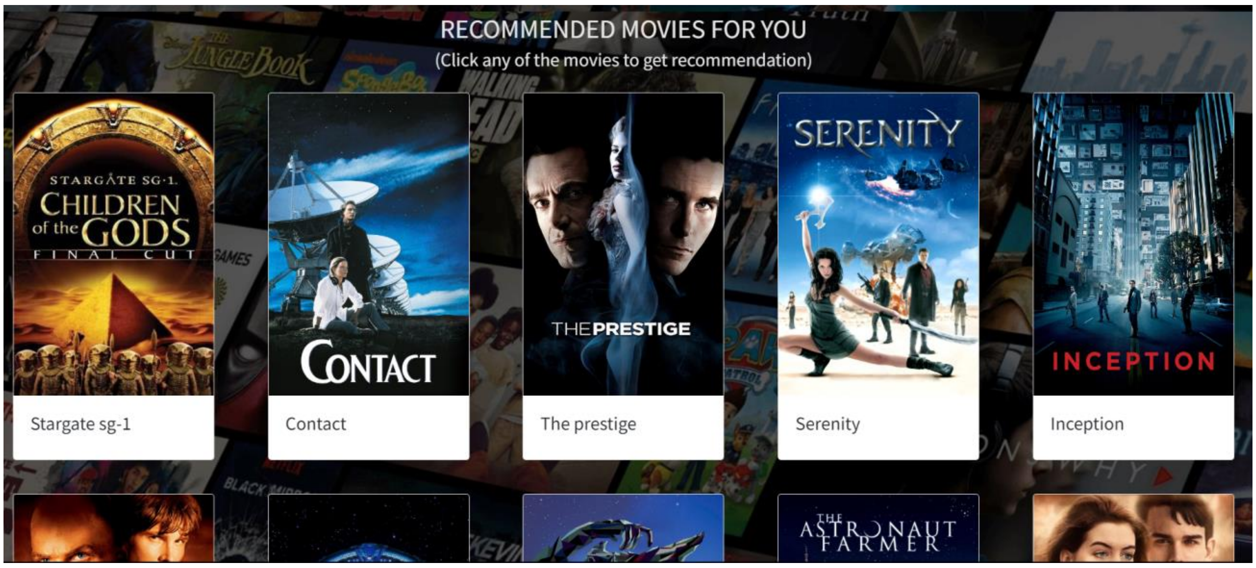


I think just about everything has been said about this film now. But, I can still tell you what this masterpiece is to me. To me, this movie is possibly the most relevant movie ever, because it questions our own humanity relative to the Universe. Whether that's our ability to love, think, or persevere and walk into the unknown. We are explorers, and curious at heart. This untameable curiosity is not our end, but our beginning. It is what advanced this civilization and it will continue to do so. So never, never let anybody tell you that we shouldn't look towards the stars and wonder, because that's what makes us human. Without this stargazing we are merely animals, accepting our fate in the dust...	Good : 😀
After watching this insane movie in the theatres back in 2014 I swore to god I will wait 5 years to watch it again so I get to forget it and experince the insanity it has again This without doubt is THE BEST MOVIE EVER MADE	Bad : 😖
Sometimes I just need to see the start. Or end. Or a trailer. Or the music and theme from Hans Zimmer. Or the whole movie. Just to feel that thing, I only get from this movie. That the earth, space and time are something special, mystical. I never forget the first time I saw this movie, in an IMAX theatre in 2014. I was struck by it. Totally got me. And it stil does, 7 years later. This is the best movie ever made for me. Because of the feeling it gives me, no other movie can. So hard to get all of this emotion in only one movie. Brilliant.	Good : 😀
I judge a movie by how long it takes me to realize I need the bathroom, how long the movie can hold my interest and how convincing the events unfolding are. Well, I watched this movie all the way through with no bathroom breaks. My interest was grabbed from the start and held all the way through. Being old enough, and lucky enough to have watched the premiere of 2001 A Space Odyssey - and viewed it several times since - of course I made comparisons, and there were a few, but this movie tells an excellent stand alone story that is both riveting and believable. I'm not going to give away any secrets but anyone who watches the last five minutes or so without a lump in their throat and a tear in their eye, well you're a critic, you're not enjoying the movie because you're too busy looking for bloopers and faults. Were there bloopers and faults? The darn movie was so riveting if there were any I didn't notice them!	Good : 😀
I hadn't seen this but movie and caught it on a flight back from the DR. One of my favorite movies of all time. I would give the first half of the movie an 11/10, just completely enjoyed it as a sci fi/ thriller(in the sense of so much always being on the line). I loved the acting and just yeah, a great movie and one you should go see if you never have	Good : 😀
I was extremely lucky to get the chance to see this film upon its first day release, before entering the cinema, my expectations were already high, after all, this was a film from the cinematic genius who brought us the likes of 'Inception' and 'The Dark Knight', to summarise the following review	Good : 😀

Cast System:


TOP CAST
(Click on the cast to know more)
Matthew Mcconaughey
Character: Joseph \"coop\" Cooper
Anne Hathaway
Character: Dr. Amelia Brand
Jessica Chastain
Character: Murphy \"murph\" Cooper
Michael Caine
Character: Professor John Brand
Bill Irwin
Character: Tars (Voice)

Recommend System:


RECOMMENDED MOVIES FOR YOU
(Click any of the movies to get recommendation)
Stargate sg-1
Contact
The prestige
Serenity
Inception

3.6 Module Specification:

Data Collection and Preprocessing:

- Module: Data Collection
- Functionality: Collect movie data from various sources, such as user ratings, movie descriptions, and reviews.
- Module: Data Preprocessing
- Functionality: Clean and preprocess the collected data, handling missing values, removing duplicates, and formatting the data for further analysis.

Content-based Filtering:

- Module: Feature Extraction
- Functionality: Extract relevant features from movie data, such as genre, director, cast, and keywords.
- Module: Vectorization
- Functionality: Transform textual data (e.g., movie descriptions) into numerical vectors using vectorization.

Module: Similarity Calculation

- Functionality: Calculate the similarity between movies based on their extracted features.
- Module: Recommendation Engine
- Functionality: Generate movie recommendations based on movie similarities.

Sentiment Analysis:

- Module: Text Sentiment Analysis
- Functionality: Perform sentiment analysis on user reviews or comments associated with movies.
- Module: Sentiment Integration
- Functionality: Integrate sentiment scores into the recommendation system to enhance personalization.

User Interface:

- Module: UI Design
- Functionality: Design and implement a user interface for users to interact with the recommendation system.

Deployment:

- Module: Deployment
- Functionality: Deploy the recommender system in a production environment, making it accessible to users.

CHAPTER 4: USER MANUAL

Drawbacks and Limitations:

Cost of Feature Extraction:

Drawback: Extracting relevant features from items can be computationally expensive.

Limitation: This could impact real-time recommendation systems or those operating with limited resources.

Scalability Issues:

Drawback: As the number of items grows, the system's ability to handle and process the data may be challenged.

Limitation: Scaling the system for a large number of movies could lead to increased computational demands.

Dependency on Feature Representation:

Drawback: Accuracy depends on the quality of the features used to represent items.

Limitation: If features don't capture the nuances of user preferences, recommendations may be less accurate.

Limited in Handling Evolving Preferences:

Drawback: Content-based systems may struggle to adapt to changes in user preferences over time.

Limitation: Preferences may shift, and the system might not easily pick up on these changes.

Proposed Enhancements:

Hybrid Recommender Systems:

Integrate collaborative filtering techniques with content-based methods to benefit from the strengths of both approaches. Hybrid systems often provide more accurate and diverse recommendations.

User Profiles:

Create more detailed user profiles, including preferences for specific actors, directors, or even visual styles.

Explainability:

Enhance the system's ability to provide explanations for recommendations, helping users understand why certain movies are being suggested.

Mobile Optimization:

Optimize the recommender system for mobile platforms, considering limited screen space and different user behaviors.

Content-based Methods for New Items:

Address the "cold start" problem by employing content-based methods to recommend new or less-reviewed movies before collaborative filtering becomes effective.

Bibliography :

1. Resnick, P., & Varian, H. R. (1997). Recommender systems. Communications of the ACM, 40(3), 56-58.
2. Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1–2), 1–135.
3. Tan, S. H., Liew, C. F., & Ong, T. S. (2017). A survey on sentiment analysis in recommender systems. Expert Systems with Applications, 88, 438-453.

Websites:

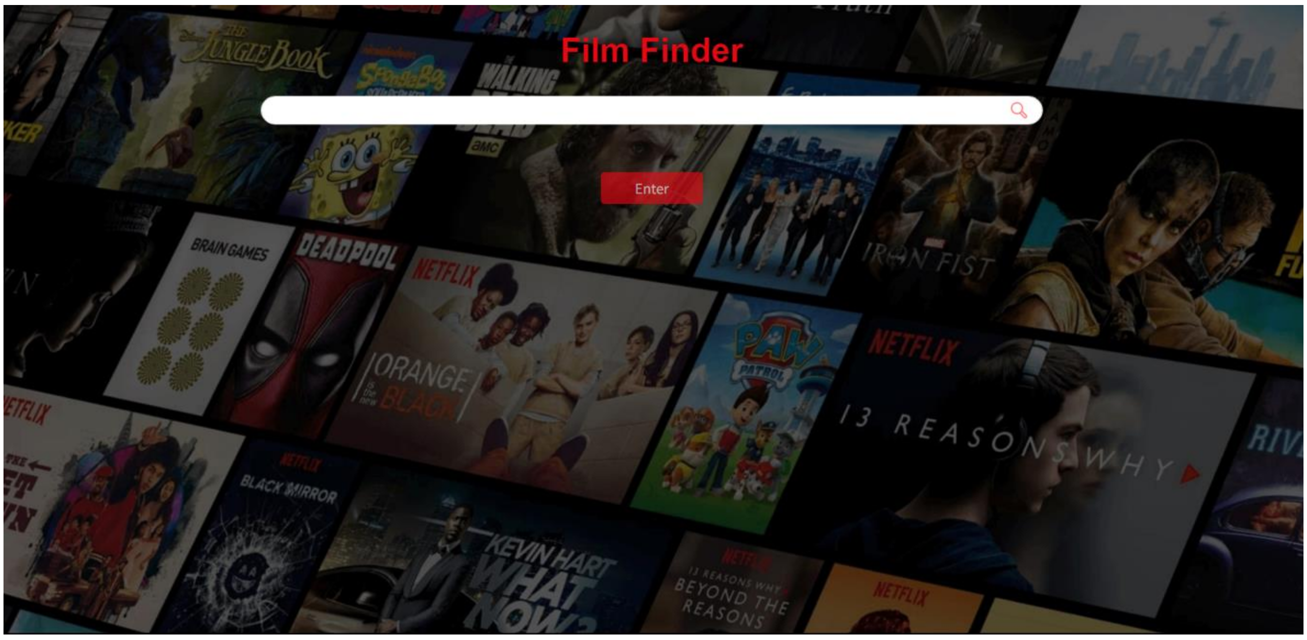
https://www.tutorialspoint.com/jsp/index.html
https://www.stackoverflow.com/
https://www.geekforgeeks.com/

https://www.youtube.com/

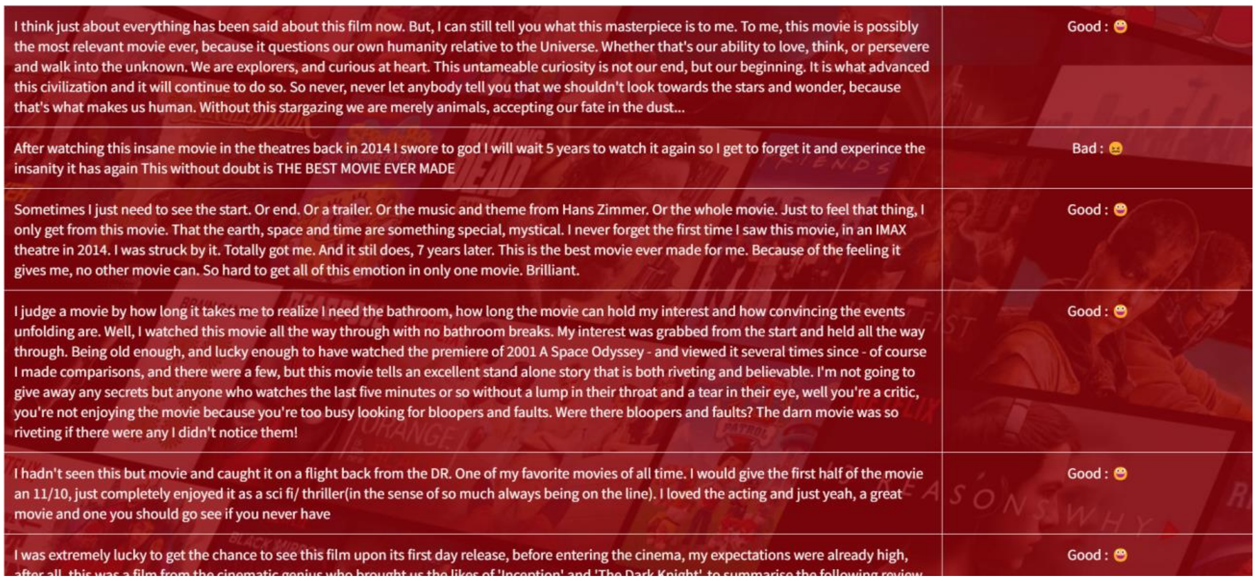
ANNEXURES:

ANNEXURE 1 : USER INTERFACE SCREENS

Homepage:

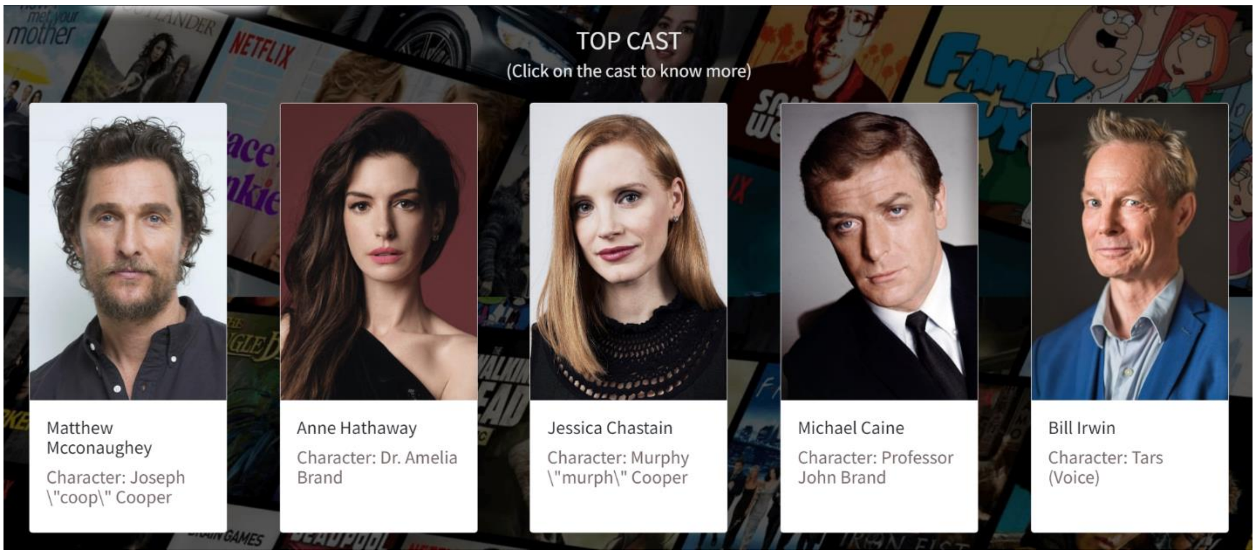


Review page:



I think just about everything has been said about this film now. But, I can still tell you what this masterpiece is to me. To me, this movie is possibly the most relevant movie ever, because it questions our own humanity relative to the Universe. Whether that's our ability to love, think, or persevere and walk into the unknown. We are explorers, and curious at heart. This untameable curiosity is not our end, but our beginning. It is what advanced this civilization and it will continue to do so. So never, never let anybody tell you that we shouldn't look towards the stars and wonder, because that's what makes us human. Without this stargazing we are merely animals, accepting our fate in the dust...	Good : 😀
After watching this insane movie in the theatres back in 2014 I swore to god I will wait 5 years to watch it again so I get to forget it and experince the insanity it has again This without doubt is THE BEST MOVIE EVER MADE	Bad : 😖
Sometimes I just need to see the start. Or end. Or a trailer. Or the music and theme from Hans Zimmer. Or the whole movie. Just to feel that thing, I only get from this movie. That the earth, space and time are something special, mystical. I never forget the first time I saw this movie, in an IMAX theatre in 2014. I was struck by it. Totally got me. And it stil does, 7 years later. This is the best movie ever made for me. Because of the feeling it gives me, no other movie can. So hard to get all of this emotion in only one movie. Brilliant.	Good : 😀
I judge a movie by how long it takes me to realize I need the bathroom, how long the movie can hold my interest and how convincing the events unfolding are. Well, I watched this movie all the way through with no bathroom breaks. My interest was grabbed from the start and held all the way through. Being old enough, and lucky enough to have watched the premiere of 2001 A Space Odyssey - and viewed it several times since - of course I made comparisons, and there were a few, but this movie tells an excellent stand alone story that is both riveting and believable. I'm not going to give away any secrets but anyone who watches the last five minutes or so without a lump in their throat and a tear in their eye, well you're a critic, you're not enjoying the movie because you're too busy looking for bloopers and faults. Were there bloopers and faults? The darn movie was so riveting if there were any I didn't notice them!	Good : 😀
I hadn't seen this but movie and caught it on a flight back from the DR. One of my favorite movies of all time. I would give the first half of the movie an 11/10, just completely enjoyed it as a sci fi/ thriller(in the sense of so much always being on the line). I loved the acting and just yeah, a great movie and one you should go see if you never have	Good : 😀
I was extremely lucky to get the chance to see this film upon its first day release, before entering the cinema, my expectations were already high, after all, this was a film from the cinematic genius who brought us the likes of 'Inception' and 'The Dark Knight', to summarise the following review	Good : 😀

Cast Page:


TOP CAST
(Click on the cast to know more)

Matthew Mcconaughey
Character: Joseph \"coop\" Cooper

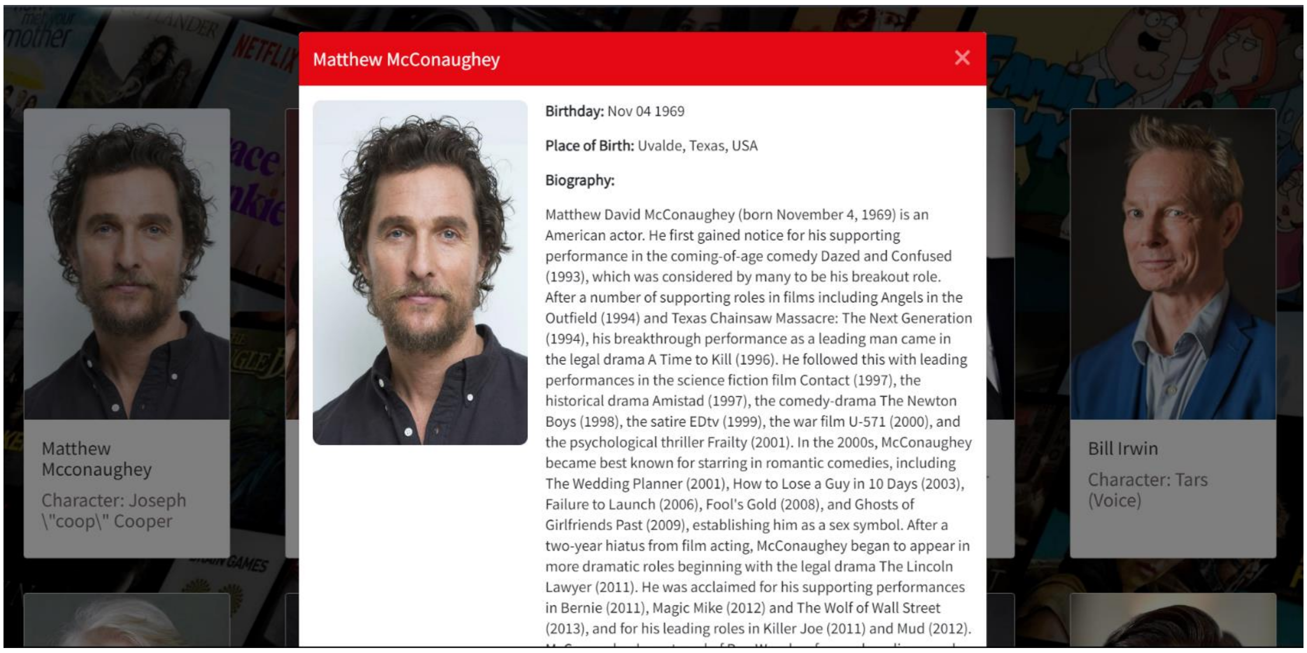
Anne Hathaway
Character: Dr. Amelia Brand

Jessica Chastain
Character: Murphy \"murph\" Cooper

Michael Caine
Character: Professor John Brand

Bill Irwin
Character: Tars (Voice)

Other Screens:


Matthew McConaughey

Birthday: Nov 04 1969

Place of Birth: Uvalde, Texas, USA

Biography:

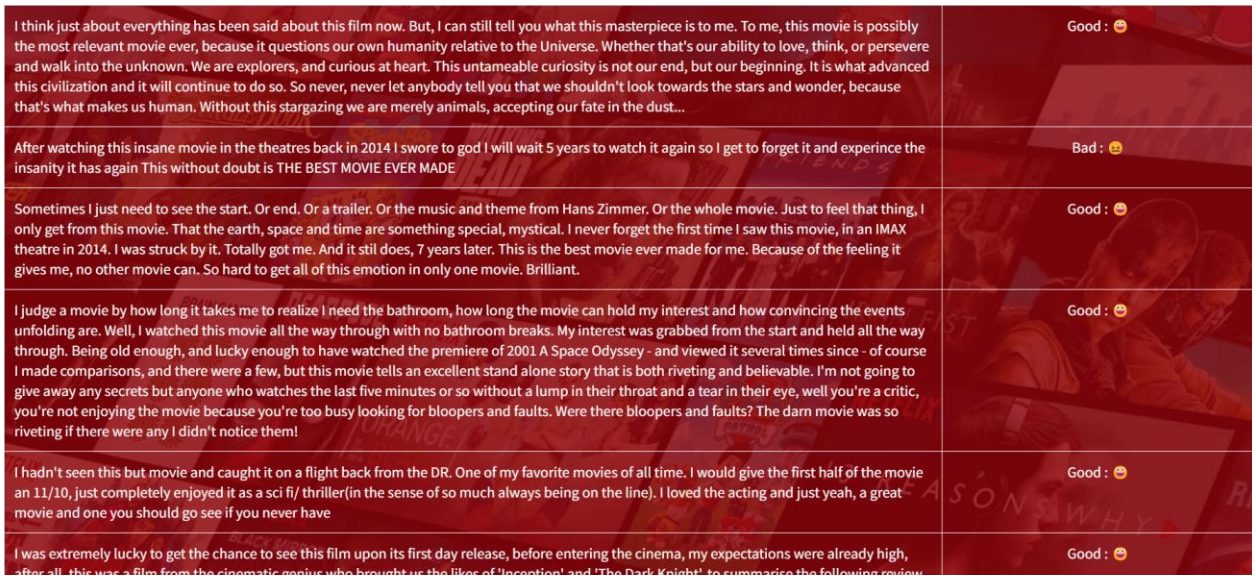
Matthew David McConaughey (born November 4, 1969) is an American actor. He first gained notice for his supporting performance in the coming-of-age comedy Dazed and Confused (1993), which was considered by many to be his breakout role. After a number of supporting roles in films including Angels in the Outfield (1994) and Texas Chainsaw Massacre: The Next Generation (1994), his breakthrough performance as a leading man came in the legal drama A Time to Kill (1996). He followed this with leading performances in the science fiction film Contact (1997), the historical drama Amistad (1997), the comedy-drama The Newton Boys (1998), the satire EDtv (1999), the war film U-571 (2000), and the psychological thriller Frailty (2001). In the 2000s, McConaughey became best known for starring in romantic comedies, including The Wedding Planner (2001), How to Lose a Guy in 10 Days (2003), Failure to Launch (2006), Fool's Gold (2008), and Ghosts of Girlfriends Past (2009), establishing him as a sex symbol. After a two-year hiatus from film acting, McConaughey began to appear in more dramatic roles beginning with the legal drama The Lincoln Lawyer (2011). He was acclaimed for his supporting performances in Bernie (2011), Magic Mike (2012) and The Wolf of Wall Street (2013), and for his leading roles in Killer Joe (2011) and Mud (2012).

Matthew
Mcconaughey
Character: Joseph \"coop\" Cooper

Bill Irwin
Character: Tars (Voice)

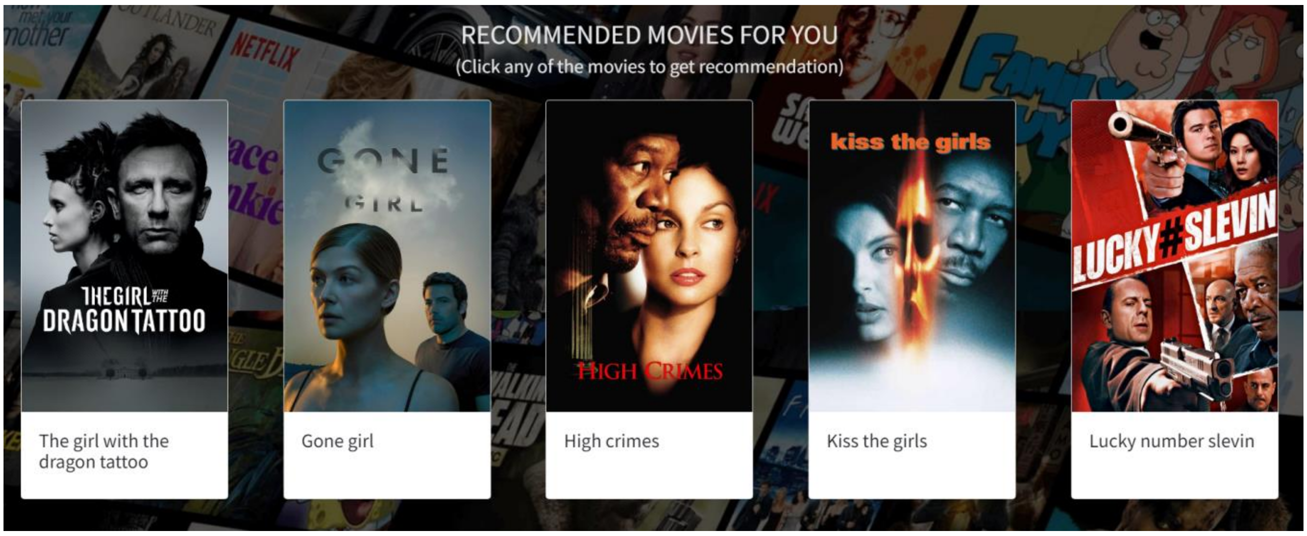
ANNEXURE 2 : OUTPUT REPORTS WITH DATA (if any)

Reviews using Sentiement anyalsis:



I think just about everything has been said about this film now. But, I can still tell you what this masterpiece is to me. To me, this movie is possibly the most relevant movie ever, because it questions our own humanity relative to the Universe. Whether that's our ability to love, think, or persevere and walk into the unknown. We are explorers, and curious at heart. This untameable curiosity is not our end, but our beginning. It is what advanced this civilization and it will continue to do so. So never, never let anybody tell you that we shouldn't look towards the stars and wonder, because that's what makes us human. Without this stargazing we are merely animals, accepting our fate in the dust...	Good : 😄
After watching this insane movie in the theatres back in 2014 I swore to god I will wait 5 years to watch it again so I get to forget it and experince the insanity it has again This without doubt is THE BEST MOVIE EVER MADE	Bad : 😖
Sometimes I just need to see the start. Or end. Or a trailer. Or the music and theme from Hans Zimmer. Or the whole movie. Just to feel that thing, I only get from this movie. That the earth, space and time are something special, mystical. I never forget the first time I saw this movie, in an IMAX theatre in 2014. I was struck by it. Totally got me. And it stil does, 7 years later. This is the best movie ever made for me. Because of the feeling it gives me, no other movie can. So hard to get all of this emotion in only one movie. Brilliant.	Good : 😄
I judge a movie by how long it takes me to realize I need the bathroom, how long the movie can hold my interest and how convincing the events unfolding are. Well, I watched this movie all the way through with no bathroom breaks. My interest was grabbed from the start and held all the way through. Being old enough, and lucky enough to have watched the premiere of 2001 A Space Odyssey - and viewed it several times since - of course I made comparisons, and there were a few, but this movie tells an excellent stand alone story that is both riveting and believable. I'm not going to give away any secrets but anyone who watches the last five minutes or so without a lump in their throat and a tear in their eye, well you're a critic, you're not enjoying the movie because you're too busy looking for bloopers and faults. Were there bloopers and faults? The darn movie was so riveting if there were any I didn't notice them!	Good : 😄
I hadn't seen this but movie and caught it on a flight back from the DR. One of my favorite movies of all time. I would give the first half of the movie an 11/10, just completely enjoyed it as a sci fi/ thriller(in the sense of so much always being on the line). I loved the acting and just yeah, a great movie and one you should go see if you never have	Good : 😄
I was extremely lucky to get the chance to see this film upon its first day release, before entering the cinema, my expectations were already high, after all, this was a film from the cinematic genius who brought us the likes of 'Inception' and 'The Dark Knight', to summarise the following review	Good : 😄

Recommended movies:


RECOMMENDED MOVIES FOR YOU
(Click any of the movies to get recommendation)
The girl with the dragon tattoo
Gone girl
High crimes
Kiss the girls
Lucky number slevin

ANNEXURE 3 : SAMPLE PROGRAM CODE

1.Main.py:

```
import numpy as np

import pandas as pd

from flask import Flask, render_template, request

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics.pairwise import cosine_similarity

import json

import bs4 as bs

import urllib.request

import pickle

import requests


# load the nlp model and tfidf vectorizer from disk

filename = 'nlp_model.pkl'

clf = pickle.load(open(filename, 'rb'))

vectorizer = pickle.load(open('tranform.pkl','rb'))


def create_similarity():

    data = pd.read_csv('main_data.csv')

    # creating a count matrix

    cv = CountVectorizer()

    count_matrix = cv.fit_transform(data['comb'])

    # creating a similarity score matrix

    similarity = cosine_similarity(count_matrix)

    return data,similarity


def rcmd(m):

    m = m.lower()

    try:

        data.head()

        similarity.shape
    except:
        data, similarity = create_similarity()
    if m not in data['movie_title'].unique():
        return('Sorry! The movie you requested is not in our database. Please check the spelling or try with some
other movies')
    else:
        i = data.loc[data['movie_title']==m].index[0]
        lst = list(enumerate(similarity[i]))
        lst = sorted(lst, key = lambda x:x[1] ,reverse=True)
        lst = lst[1:11] # excluding first item since it is the requested movie itself
        l = []
        for i in range(len(lst)):
            a = lst[i][0]
            l.append(data['movie_title'][a])
        return l


# converting list of string to list (eg. "["abc","def"]" to ["abc","def"])
def convert_to_list(my_list):
    my_list = my_list.split('","')
    my_list[0] = my_list[0].replace('["','')
    my_list[-1] = my_list[-1].replace('"]','')
    return my_list


def get_suggestions():
    data = pd.read_csv('main_data.csv')
    return list(data['movie_title'].str.capitalize())


app = Flask(__name__)


@app.route("/")
@app.route("/home")
def home():
    suggestions = get_suggestions()
    return render_template('home.html',suggestions=suggestions)

@app.route("/similarity",methods=["POST"])
def similarity():
    movie = request.form['name']
    rc = rcmd(movie)
    if type(rc)==type('string'):
        return rc
    else:
        m_str="---".join(rc)
        return m_str

@app.route("/recommend",methods=["POST"])
def recommend():
    # getting data from AJAX request
    title = request.form['title']
    cast_ids = request.form['cast_ids']
    cast_names = request.form['cast_names']
    cast_chars = request.form['cast_chars']
    cast_bdays = request.form['cast_bdays']
    cast_bios = request.form['cast_bios']
    cast_places = request.form['cast_places']
    cast_profiles = request.form['cast_profiles']
    imdb_id = request.form['imdb_id']
    poster = request.form['poster']
    genres = request.form['genres']
    overview = request.form['overview']
    vote_average = request.form['rating']
    vote_count = request.form['vote_count']
    release_date = request.form['release_date']
    runtime = request.form['runtime']
    status = request.form['status']
    rec_movies = request.form['rec_movies']
    rec_posters = request.form['rec_posters']

    # get movie suggestions for auto complete
    suggestions = get_suggestions()

    # call the convert_to_list function for every string that needs to be converted to list
    rec_movies = convert_to_list(rec_movies)
    rec_posters = convert_to_list(rec_posters)
    cast_names = convert_to_list(cast_names)
    cast_chars = convert_to_list(cast_chars)
    cast_profiles = convert_to_list(cast_profiles)
    cast_bdays = convert_to_list(cast_bdays)
    cast_bios = convert_to_list(cast_bios)
    cast_places = convert_to_list(cast_places)

    # convert string to list (eg. "[1,2,3]" to [1,2,3])
    cast_ids = cast_ids.split(',')
    cast_ids[0] = cast_ids[0].replace("[","")
    cast_ids[-1] = cast_ids[-1].replace("]","")

    # rendering the string to python string
    for i in range(len(cast_bios)):
        cast_bios[i] = cast_bios[i].replace(r'\n', '\n').replace(r'\"','\"')

    # combining multiple lists as a dictionary which can be passed to the html file so that it can be processed
easily and the order of information will be preserved
    movie_cards = {rec_posters[i]: rec_movies[i] for i in range(len(rec_posters))}

    casts = {cast_names[i]:[cast_ids[i], cast_chars[i], cast_profiles[i]] for i in range(len(cast_profiles))}

    cast_details = {cast_names[i]:[cast_ids[i], cast_profiles[i], cast_bdays[i], cast_places[i], cast_bios[i]] for i
in range(len(cast_places))}
```

```
    # web scraping to get user reviews from IMDB site
    sauce = 
urllib.request.urlopen('https://www.imdb.com/title/{}/reviews?ref_=tt_ov_rt'.format(imdb_id)).read()
    soup = bs.BeautifulSoup(sauce,'lxml')
    soup_result = soup.find_all("div",{"class":"text show-more__control"})

    reviews_list = [] # list of reviews
    reviews_status = [] # list of comments (good or bad)
    for reviews in soup_result:
        if reviews.string:
            reviews_list.append(reviews.string)
            # passing the review to our model
            movie_review_list = np.array([reviews.string])
            movie_vector = vectorizer.transform(movie_review_list)
            pred = clf.predict(movie_vector)
            reviews_status.append('Good' if pred else 'Bad')

    # combining reviews and comments into a dictionary
    movie_reviews = {reviews_list[i]: reviews_status[i] for i in range(len(reviews_list))}

    # passing all the data to the html file
    return 
render_template('recommend.html',title=title,poster=poster,overview=overview,vote_average=vote_average,
        vote_count=vote_count,release_date=release_date,runtime=runtime,status=status,genres=genres,
        movie_cards=movie_cards,reviews=movie_reviews,casts=casts,cast_details=cast_details)

if __name__ == '__main__':
    app.run(debug=True)
```

Home.html:

```
<!DOCTYPE html>
<html>
<head>
  <title>Film Finder</title>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

  <!-- Google Fonts -->
  <link href="https://fonts.googleapis.com/css?family=IBM+Plex+Sans&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css2?family=Noto+Sans+JP&display=swap" rel="stylesheet">

  <!-- Font Awesome -->
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
awesome.min.css">

  <!-- Bootstrap -->
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">

  <!-- Auto Complete -->
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/@tarekraafat/autocomplete.js@7.2.0/dist/css/autoComplete.min.css">
  <link rel= "stylesheet" type= "text/css" href= "{{ url_for('static',filename='style.css') }}">
  <script type="text/javascript">
    var films = {{suggestions|tojson}};
  </script>

</head>

<body id="content" style="font-family: 'Noto Sans JP', sans-serif;">

        <div class="ml-container" style="display: block;">
```

```
                        <!-- <a href="https://github.com/kishan0725/AJAX-Movie-Recommendation-System-with-Sentiment-Analysis" target="_blank" class="github-corner" title="View source on GitHub">

                              <svg data-toggle="tooltip"

                              data-placement="left" width="80" height="80" viewBox="0 0 250 250"

                                     style="fill:#e50914; color:#fff; position: fixed;z-index:100; top: 0; border: 0; right: 0;" aria-hidden="true">

                                     <path d="M0,0 L115,115 L130,115 L142,142 L250,250 L250,0 Z"></path>

                                     <path

                                       d="M128.3,109.0 C113.8,99.7 119.0,89.6 119.0,89.6 C122.0,82.7 120.5,78.6 120.5,78.6 C119.2,72.0 123.4,76.3 123.4,76.3 C127.3,80.9 125.5,87.3 125.5,87.3 C122.9,97.6 130.6,101.9 134.4,103.2"

                                       fill="currentColor" style="transform-origin: 130px 106px;" class="octo-arm"></path>

                                     <path

                                       d="M115.0,115.0 C114.9,115.1 118.7,116.5 119.8,115.4 L133.7,101.6 C136.9,99.2 139.9,98.4 142.2,98.6 C133.8,88.0 127.5,74.4 143.8,58.0 C148.5,53.4 154.0,51.2 159.7,51.0 C160.3,49.4 163.2,43.6 171.4,40.1 C171.4,40.1 176.1,42.5 178.8,56.2 C183.1,58.6 187.2,61.8 190.9,65.4 C194.5,69.0 197.7,73.2 200.1,77.6 C213.8,80.2 216.3,84.9 216.3,84.9 C212.7,93.1 206.9,96.0 205.4,96.6 C205.1,102.4 203.0,107.8 198.3,112.5 C181.9,128.9 168.3,122.5 157.7,114.1 C157.9,116.9 156.7,120.9 152.7,124.9 L141.0,136.5 C139.8,137.7 141.6,141.9 141.8,141.8 Z"

                                       fill="currentColor" class="octo-body"></path>

                              </svg>

                            </a> -->

    <center><h1 style=""> Film Finder</h1></center>

    <div class="form-group shadow-textarea" style="margin-top: 30px;text-align: center;color: white;">

        <input type="text" name="movie" class="movie form-control" id="autoComplete" autocomplete="off" placeholder="Enter the Movie Name" style="background-color: #ffffff;border-color:#ffffff;width: 60%;color: #181818" required="required" />

        <br>

    </div>


    <div class="form-group" style="text-align: center;">

      <button class="btn btn-primary btn-block movie-button" style="background-color: #e50914;text-align: center;border-color: #e50914;width:120px;" disabled="true" >Enter</button><br><br>

    </div>

            </div>
```

```
  <div id="loader" class="text-center">
  </div>

  <div class="fail">
    <center><h3>Sorry! The movie you requested is not in our database.
    Please check the spelling or try with other movies!</h3></center>
  </div>

	<div class="results">
    <center>
      <h2 id="name" class="text-uppercase"></h2>
    </center>
	</div>




	<script
src="https://cdn.jsdelivr.net/npm/@tarekraafat/autocomplete.js@7.2.0/dist/js/autoComplete.min.js"></script>
  <script type="text/javascript" src="{{url_for('static', filename='autocomplete.js')}}"></script>

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script type="text/javascript" src="{{url_for('static', filename='recommend.js')}}"></script>

</body>
</html>
```

Style.css:

```
.movie {
	color: #fff;
	margin-left: auto;
	margin-right: auto;
	resize: none;
}
```

```
.movie-content {
            display: flex;
            flex-wrap: wrap;
            justify-content:space-around;
}

.movie-content > div {
   margin:20px;
}

.btn-block{
            width: 15%;
            text-align: center;
            margin-left: auto;
            margin-right: auto;
            color: #e4e0e0;
}

#content {
            background-image: url("../static/image.jpg");
            background-color: #181818;
            font-family: 'Noto Sans JP', sans-serif;
}

#details {
            margin-left: 50px;
}

.footer {
            color: #e4e0e0;
            text-align:right;
            position: fixed;
            bottom: 20px;
            right: 20px;
            width: 100%;
}

h1 {
    font-family: 'Netflix Sans', 'Helvetica Neue', Helvetica, Arial, sans-serif;
    color: #e50914;
    font-weight: bold;
    margin-top: 30px;
    text-shadow: #000000 0px 0px 13px;
 }

.github-corner:hover .octo-arm {
            animation: octocat-wave 560ms ease-in-out;
}

@keyframes octocat-wave {
  0%,
  100% {
    transform: rotate(0)
  }

  20%,
  60% {
    transform: rotate(-25deg)
  }

  40%,
  80% {
    transform: rotate(10deg)
  }
```

```
}

#autoComplete {
  background-position: 98% ;
}

#name {
             color: white;
             padding: 1px;
}

h6 {
    margin-bottom: 20px;
}

@media only screen and (max-width: 650px) {
    #mcontent {
      display: block;
    }
    .poster-lg {
      display: none;
    }
    #details {
                          margin-left: 30px;
             }
    #loader {
                          display: none;
                          position: fixed;
                          z-index: 100;
                          left: 0;
                          top:0;
                          width: 100%;
				height: 100%;
				background-image: url("../static/loader.gif");
				background-size: 40%;
				background-position: 50% 50%;
				background-color: rgba(255, 255, 255, 1);
				background-repeat: no-repeat;
				-webkit-transition: background-image 0.2s ease-in-out;
				transition: background-image 0.2s ease-in-out;
			}

			#loader-text {
				vertical-align: middle;
				color:white;
			}

			#autoComplete {
			  background-position: 97% ;
			}

			svg[data-toggle=tooltip] {
				width: 50px;
				height: 50px;
			}
		}

		@media only screen and (max-width: 991px) {
			.modal-body{
				display: block;
			}
			.profile-pic {
				margin-left: auto;
				margin-right: auto;
```

```
                display: block;
                margin-bottom: 20px;
        }
}
@media only screen and (min-width: 992px) {
        .modal-body {
                display: flex;
        }
}

@media only screen and (min-width: 651px) {
  .poster-sm {
    display: none;
   }


        #mcontent {
                display: flex;
                flex-wrap: nowrap;
        }


        #loader {
                display: none;
                position: fixed;
                z-index: 100;
                left: 0;
                top:0;
                width: 100%;
                height: 100%;
                background-image: url("../static/loader.gif");
                background-size: 20%;
                background-position: 50% 50%;
                background-color: rgba(255, 255, 255, 1);
				background-repeat: no-repeat;
				-webkit-transition: background-image 0.2s ease-in-out;
				transition: background-image 0.2s ease-in-out;
		}

		#loader-text {
				vertical-align: middle;
				color:white;
		}

}

.poster{
    -webkit-box-shadow: 0px 1px 15px 4px rgba(250,250,250,1);
    -moz-box-shadow: 0px 1px 15px 4px rgba(250,250,250,1);
    box-shadow: 0px 1px 15px 4px rgba(250,250,250,1);
}

.card:hover {
		cursor: pointer;
}

.castcard:hover {
    cursor: pointer;
}

.cast-img {
		filter: brightness(100%);
		-moz-transition: all 0.75s ease;
    -webkit-transition: all 0.75s ease;
		transition: all 0.75s ease;
}
```

```
.cast-img:hover {
            filter: brightness(50%);
            -moz-transition: all 0.75s ease;
    -webkit-transition: all 0.75s ease;
    transition: all 0.75s ease;
}

.fig {
            display: flex;
            align-items: center;
            justify-content: center;
            backdrop-filter: brightness(50%);
            position: absolute;
            bottom: 0px;
            top: 0px;
            right: 0px;
            left: 0px;
            opacity: 0;
            -moz-transition: all 0.75s ease;
    -webkit-transition: all 0.75s ease;
    transition: all 0.75s ease;
}

.fig:hover {
            opacity: 1;
            backdrop-filter:br;
            -moz-transition: all 0.75s ease;
    -webkit-transition: all 0.75s ease;
    transition: all 0.75s ease;
}
```

```
.card-btn {
            border-radius: 20px;
}

.imghvr {
            position: relative;
}

.table td {
    border-color: white;
    border-style:solid;
    border-width:1px;
}

.fail {
            display: none;
            color: white;
}
```

Autocomplete.js:

```
new autoComplete({
    data: {                              // Data src [Array, Function, Async] | (REQUIRED)
      src: films,
    },
    selector: "#autoComplete",           // Input field selector              | (Optional)
    threshold: 2,                        // Min. Chars length to start Engine | (Optional)
    debounce: 100,                       // Post duration for engine to start | (Optional)
    searchEngine: "strict",              // Search Engine type/mode           | (Optional)
    resultsList: {                       // Rendered results list object      | (Optional)
        render: true,
        container: source => {
            source.setAttribute("id", "food_list");
        },
        destination: document.querySelector("#autoComplete"),
        position: "afterend",
        element: "ul"
    },
    maxResults: 5,                       // Max. number of rendered results | (Optional)
    highlight: true,                     // Highlight matching results      | (Optional)
    resultItem: {                        // Rendered result item            | (Optional)
        content: (data, source) => {
            source.innerHTML = data.match;
        },
        element: "li"
    },
    noResults: () => {                   // Action script on noResults      | (Optional)
        const result = document.createElement("li");
        result.setAttribute("class", "no_result");
        result.setAttribute("tabindex", "1");
        result.innerHTML = "No Results";
        document.querySelector("#autoComplete_list").appendChild(result);
    },
    onSelection: feedback => {           // Action script onSelection event | (Optional)
        document.getElementById('autoComplete').value = feedback.selection.value;
    }
});
```