A PROJECT REPORT

on

"MOVIE RECOMMENDATION SYSTEM"

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN INFORMATION TECHNOLOGY

BY

VINAYAK MISHRA	20051184
ARYAN BHARDWAJ	20051928
DEEP DUBEY	20051865
AYUSH KUMAR	2005507

UNDER THE GUIDANCE OF
MRS. RONALI PADHY

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024

May 2020

A PROJECT REPORT

on

"MOVIE RECOMMENDATION SYSTEM"

Submitted to

KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
INFORMATION TECHNOLOGY

BY

VINAYAK MISHRA	20051184
ARYAN BHARDWAJ	20051928
DEEP DUBEY	20051865
AYUSH KUMAR	2005507

UNDER THE GUIDANCE OF

MRS.RONALI PADHY

SCHOOL OF COMPUTER ENGINEERING

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

BHUBANESWAE, ODISHA -751024

May 2022

KIIT Deemed to be University

School of Computer Engineering

Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

"MOVIE RECOMMENDATION SYSTEM."

submitted by

VINAYAK MISHRA	20051184
ARYAN BHARDWAJ	20051928
DEEP DUBEY	20051865
AYUSH KUMAR	2005507

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering OR Information Technology) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2022-2023, under our guidance.

Date: 03/05/2023

(MRS. RONALI PADHY)
Project Guide

Acknowledgements

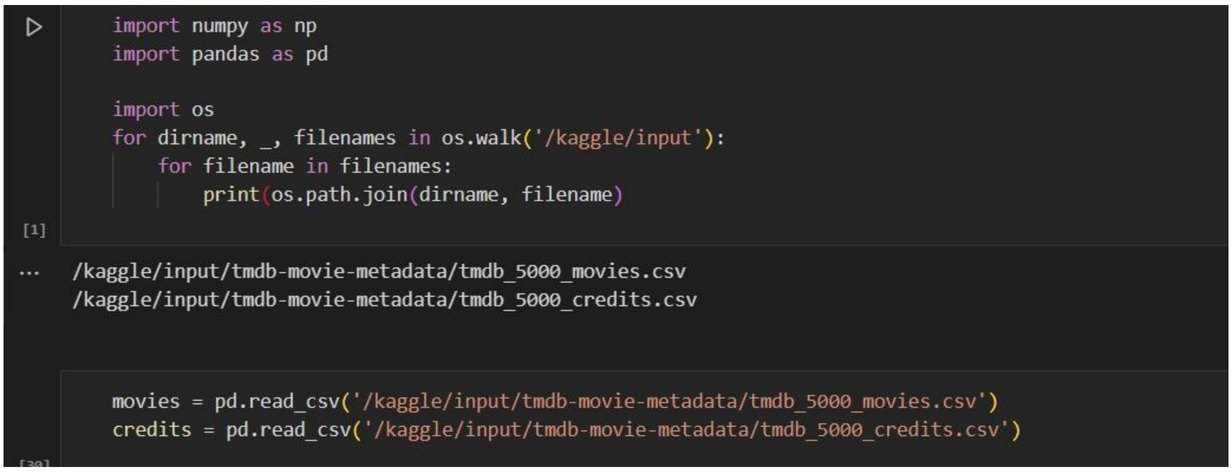
We are profoundly grateful to **MRS. RONALI PADHY** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. ....

VINAYAK MISHRA
ARYAN BHARDWAJ
DEEP DUBEY
AYUSH KUMAR

ABSTRACT

In this paper, we consider ML(MACHINE LEARNING) for movie recommendation.Movie recommendation systems using machine learning have become increasingly popular in recent years. These systems utilize algorithms to analyze a user's viewing history and preferences to provide personalized recommendations for movies and TV shows.

The primary goal of these systems is to provide a more engaging and satisfying viewing experience for users. By analyzing user behavior, such as which movies or TV shows they watch and how frequently they watch them, these systems can make accurate predictions about the types of content users are likely to enjoy.

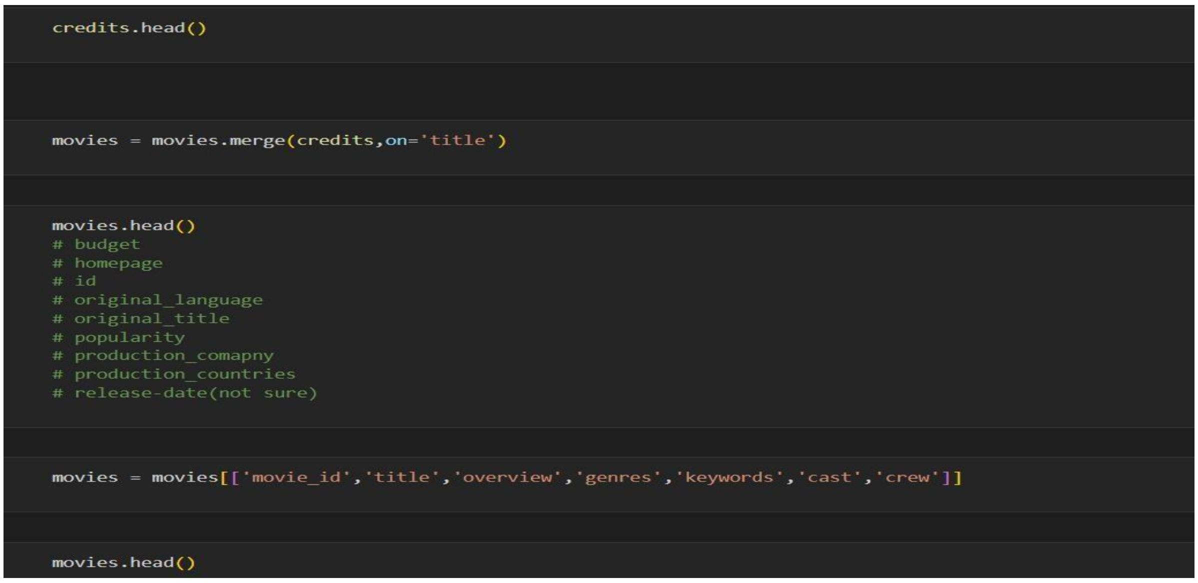


```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/tmdb-movie-metadata/tmdb_5000_movies.csv
/kaggle/input/tmdb-movie-metadata/tmdb_5000_credits.csv
```

```
movies = pd.read_csv('/kaggle/input/tmdb-movie-metadata/tmdb_5000_movies.csv')
credits = pd.read_csv('/kaggle/input/tmdb-movie-metadata/tmdb_5000_credits.csv')
```



```
credits.head()
```

```
movies = movies.merge(credits,on='title')
```

```
movies.head()
# budget
# homepage
# id
# original_language
# original_title
# popularity
# production_comapny
# production_countries
# release-date(not sure)
```

```
movies = movies[['movie_id','title','overview','genres','keywords','cast','crew']]
```

```
movies.head()
```

Contents

List of Figures

Chapter 1

Introduction

Movie recommendation systems have become increasingly popular in recent years due to the growing demand for personalized content recommendations. One approach to building a movie recommendation system is by using cosine similarity, a metric commonly used in machine learning for measuring the similarity between two vectors.

The cosine similarity approach involves representing movies as vectors based on their feature attributes such as genre, director, actors, and ratings. Each movie is then represented as a vector in a high-dimensional space, where the features are the dimensions. The similarity between two movies can be calculated using the cosine of the angle between their vectors, with a cosine similarity score of 1 indicating that two movies are identical and a score of 0 indicating that they are completely dissimilar.

To generate recommendations for a particular user, the cosine similarity approach identifies movies that are similar to the ones the user has already watched and liked. These similar movies are then ranked based on their cosine similarity score and presented to the user as recommendations.

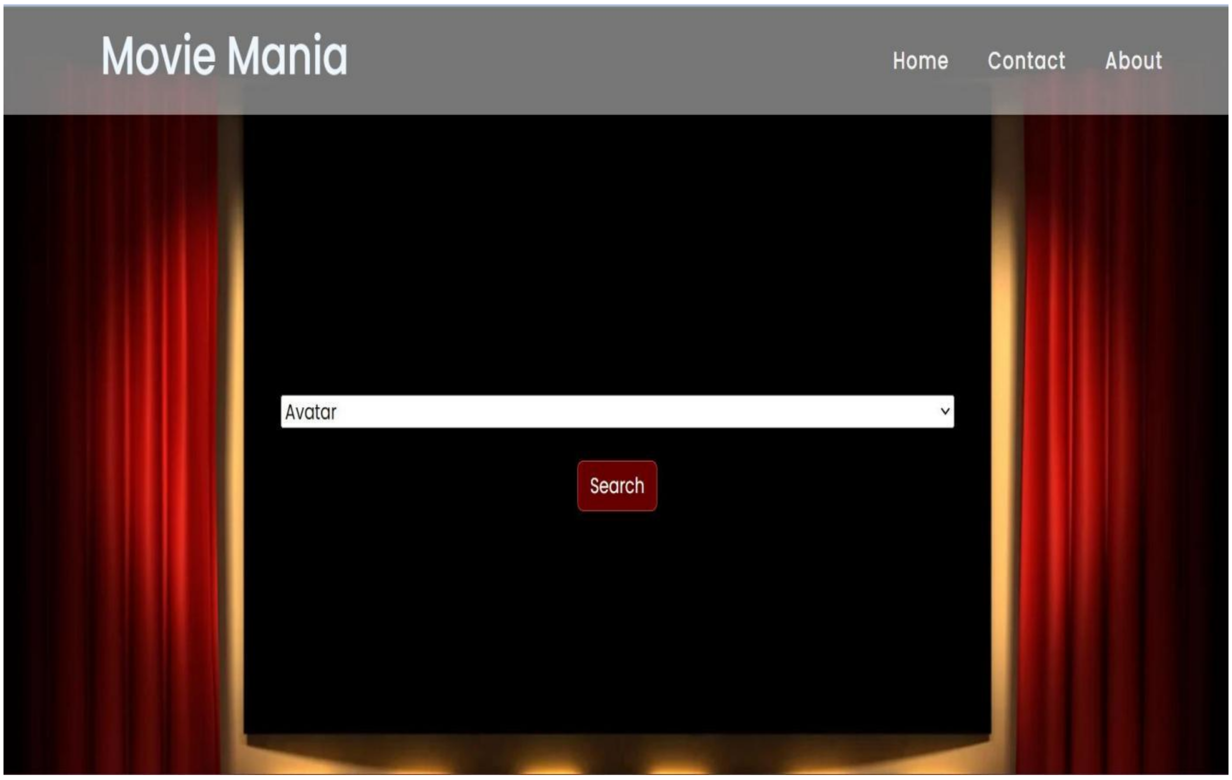


FIG1.1:- HOME PAGE

Chapter 2

Basic Concepts/ Literature Review

Movie recommendation systems using cosine similarity rely on several basic concepts. These include:

Vectors: Movies are represented as vectors, which are mathematical representations of the movie's features such as genre, director, actors, and ratings. Each feature is considered a dimension in a high-dimensional space, and the combination of all features creates the vector.

Cosine similarity: Cosine similarity is a metric used to measure the similarity between two vectors. It measures the cosine of the angle between the two vectors and produces a value between 0 and 1. A cosine similarity score of 1 indicates that two vectors are identical, while a score of 0 indicates that they are completely dissimilar.

User profiles: User profiles are created by analyzing a user's viewing history and preferences. The user's viewing history is represented as a vector, and recommendations are generated by finding movies that have high cosine similarity scores with the user's profile vector.

Similarity threshold: A similarity threshold is set to determine the minimum cosine similarity score required for a movie to be recommended. Movies with cosine similarity scores below this threshold are not recommended to the user.

2.1 Sub section …………

By using cosine similarity to compare the similarities between movies and user profiles, movie recommendation systems can provide personalized recommendations to users. This approach is effective in handling large and sparse datasets, making it suitable for movie recommendation systems with a large number of users and movies. It is also relatively simple to implement and provides accurate recommendations.

Chapter 3

Problem Statement / Requirement Specifications

---> You have to provide users with personalized content recommendations based on their viewing history and preferences. With the overwhelming amount of movie and TV show options available on various streaming platforms, users may struggle to find content that they are interested in. Additionally, users may not be aware of certain movies or TV shows that they would enjoy but have not yet discovered.

3.1 Project Planning

Define project scope and goals: The first step is to clearly define the scope and goals of the project. This includes identifying the target audience, defining the types of content to be recommended, and establishing the success criteria for the project.

Data collection and preprocessing: The next step is to collect and preprocess data for the recommendation system. This includes collecting user behavior data, such as viewing history and ratings, as well as metadata about movies and TV shows, such as genre, director, and cast.

Feature engineering: Feature engineering involves selecting relevant features from the collected data and transforming them into a format suitable for machine learning algorithms. This may involve normalizing data, encoding categorical variables, and creating user and item profiles.

Model selection and training: The next step is to select an appropriate machine learning algorithm for the recommendation system and train the model using the preprocessed data. Common algorithms for movie recommendation systems include collaborative filtering, content-based filtering, and hybrid approaches.

Model evaluation and tuning: Once the model is trained, it is evaluated using metrics such as accuracy, precision, and recall. The model may then be tuned to improve its performance, such as adjusting hyperparameters or selecting a different algorithm.

Deployment and testing: The final step is to deploy the recommendation system and test it with real users. This involves monitoring user feedback and behavior to ensure the system is providing relevant recommendations.

3.2 Project Analysis

project analysis for a movie recommendation system in machine learning involves a thorough understanding of the data, the domain, and the machine learning algorithms, as well as the ability to interpret and evaluate the results of the model.

3.3 System Design

3.3.1 Design Constraints

The design constraints of a movie recommendation system in machine learning involve balancing the need for accuracy and efficiency with the need for privacy and transparency, while also providing a diverse set of recommendations that are relevant to the user's preferences and interests.

3.3.2 System Architecture **OR** Block Diagram

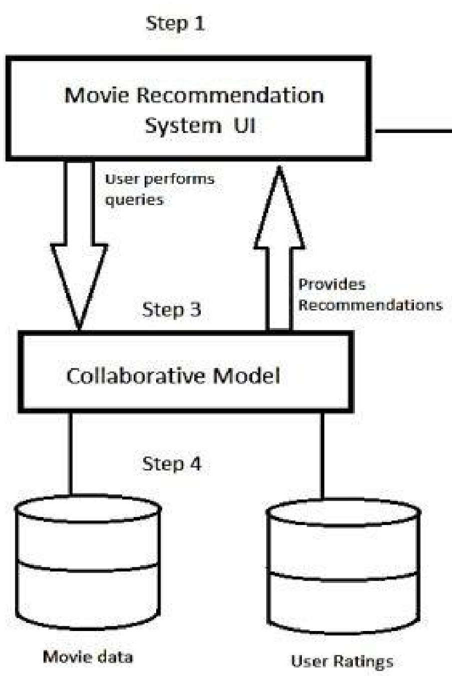


FIG 3.1 BLOCK DIAGRAM

Chapter 4

Implementation

In this movie recommendation site we used Flask for GUI and used pickle files to store movie names and their data , then on user input movie the system will give closest 3 similar recommended movie as per the algorithm.

4.1 Methodology OR Proposal

- ***FLASK:***Flask is a Python web framework used for building web applications. It is a lightweight framework that is easy to use and can be integrated with various libraries to build a variety of web applications ranging from small personal projects to large enterprise applications

- ***PICKLE:***Pickle files are a way to serialize and store Python objects in a file format that can be used later. The pickling process converts a Python object into a byte stream representation, which can then be stored in a file or transferred over a network. The unpickling process takes this byte stream and converts it back into a Python object. Pickle files can be useful for a variety of purposes, such as caching expensive computations, storing and loading trained machine learning models, or transferring data between different parts of a distributed system.

- ***ALGORITHM:****Cosine similarity* is a commonly used technique in movie recommendation systems to measure the similarity between movies. In this approach, the system represents each movie as a vector in a high-dimensional space based on various features such as genre, director, actors, and plot summary. The cosine similarity between two movies is then calculated as the cosine of the angle between their respective feature vectors.

Firstly we collected data on movies, including information such as title, genre, director, actors, and plot summary. Then we preprocess the data by removing stop words, stemming, and converting the text into a vector representation.Once the data is preprocessed, the system can calculate the cosine similarity between pairs of movies. This can be done by computing the dot product between their respective feature vectors and dividing by the product of their magnitudes.After computing the cosine similarity between pairs of movies, the system can rank the movies based on their similarity scores and recommend the top-rated movies to the user.

4.2 ***Testing OR Verification Plan***

The system can be evaluated by comparing its recommendations to those made by human experts or through user feedback. This can help refine the system and improve its accuracy over time.

Test ID	Test Case Title	System Behavior	Expected Result
T01	Superman Returns	• Superman 2 • Superman 3 • Superman 4	• Superman 2 • Superman 3 • Superman 4
T02	Pirates of the Caribbean:At World's End	• Pirates of the Caribbean:Dead Man's Chest • Pirates of the Caribbean:The Curse of the Black Pearl • Pirates of the Caribbean:On Stranger Tides	• Pirates of the Caribbean:Dead Man's Chest • Pirates of the Caribbean:The Curse of the Black Pearl • Pirates of the Caribbean:On Stranger Tides
T03	WALL-E	• The Black Hole • Bicentennial Man • Galaxina	• Up • Ratatouille • Inside Out

4.3 ***Result Analysis OR Screenshots***

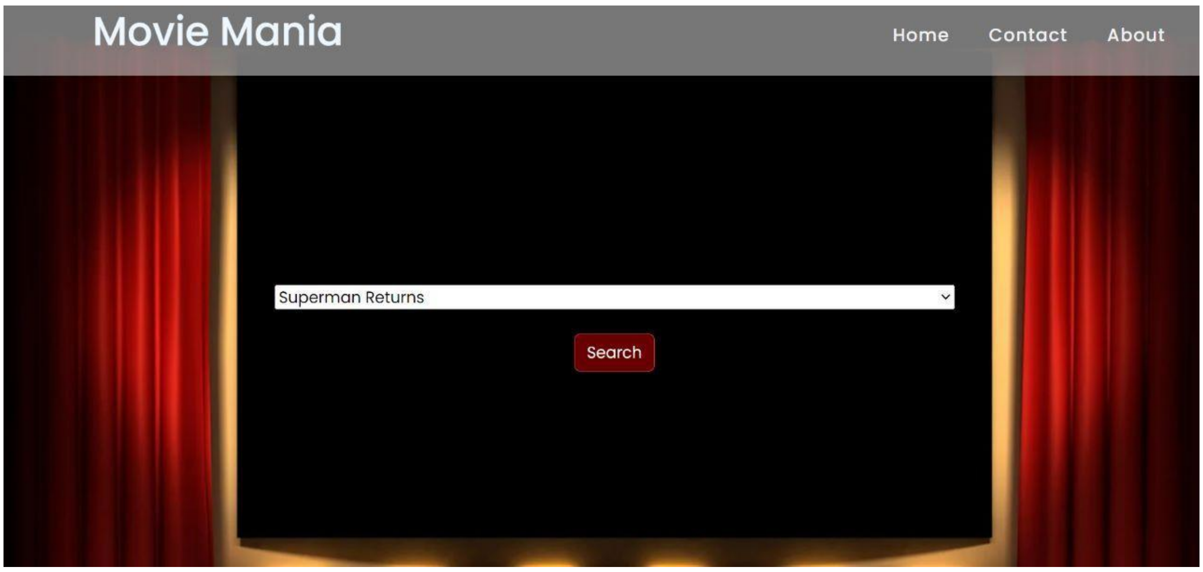


Fig 4.1 Home Page Content

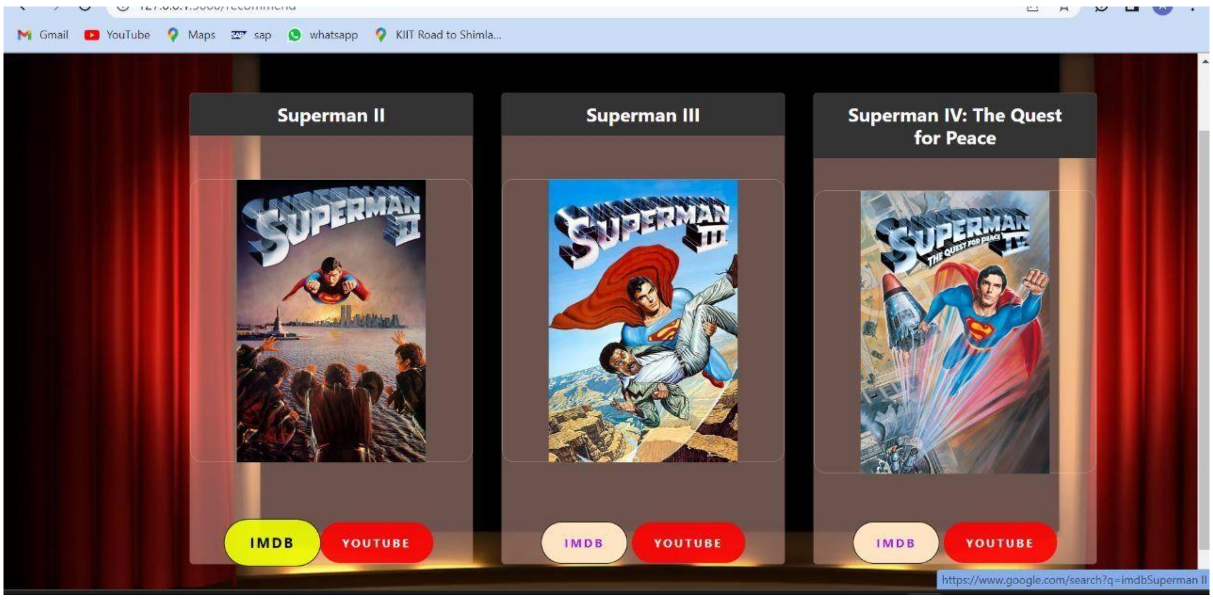
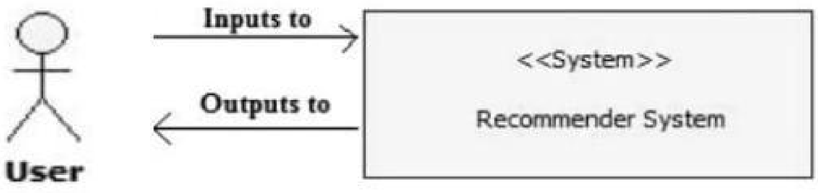


Fig 4.2 Working of website

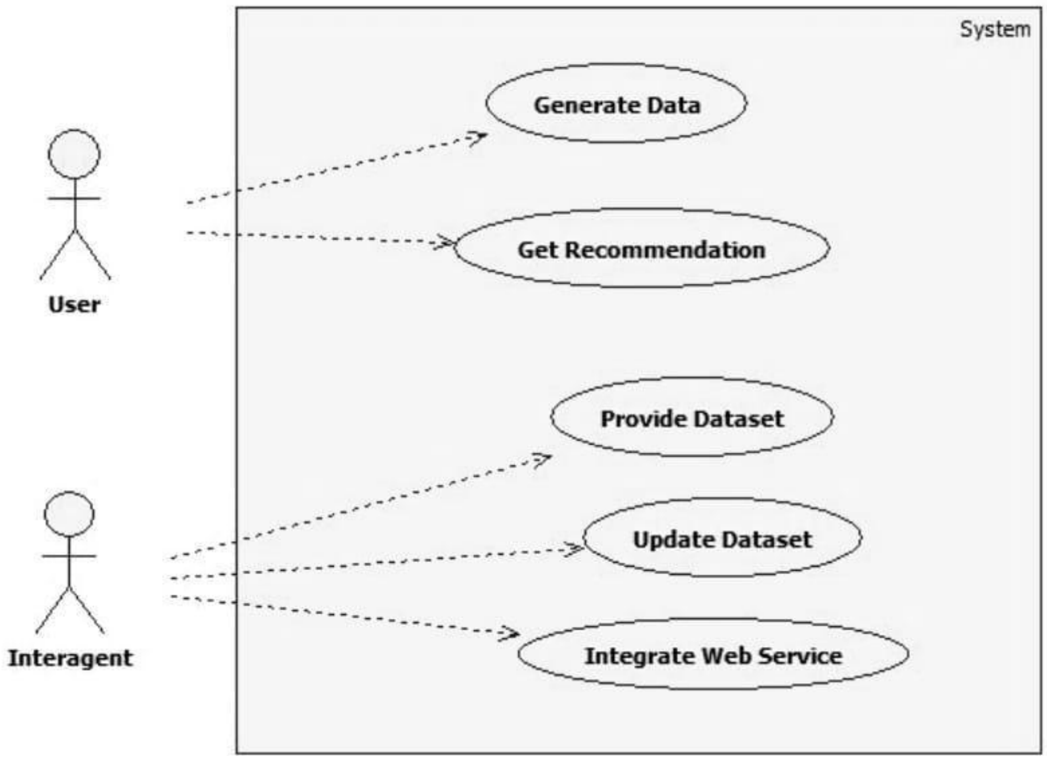
Chapter 5

Standards Adopted

5.1 Design Standards



5.1(a) System Context Diagram



5.1(b) Use Case Diagram

5.2 Coding Standards

Coding standards are collections of coding rules, guidelines, and best practices.
Few of the coding standards are:
1. Write as few lines as possible.
2. Use appropriate naming conventions.
3. Segment blocks of code in the same section into paragraphs.
4. Use indentation to marks the beginning and end of control structures. Clearly specify the code between them.
5. Don’t use lengthy functions. Ideally, a single function should carry out a single task.

5.3 Testing Standards

- Functional Testing: This involves testing the system to ensure that it meets all the functional requirements outlined in the project specifications. The system should be tested against use cases and scenarios to ensure that it performs as expected.

- Performance Testing: This involves testing the system to ensure that it performs optimally, even under high loads. This is important for ensuring that the system can handle a large number of users and requests without slowing down or crashing.

- Usability Testing: This involves testing the system to ensure that it is easy to use and navigate. This is important for ensuring that users can quickly and easily find the movies they want to watch.

- Compatibility Testing: This involves testing the system to ensure that it works correctly on different devices, platforms and browsers. This is important for ensuring that users can access the system regardless of the device or platform they are using.

- Regression Testing: This involves testing the system after any changes or updates have been made to ensure that it still functions correctly and that new changes have not introduced any new bugs or issues.

- User Acceptance Testing: This involves testing the system with actual users to ensure that it meets their needs and expectations. This is important for ensuring that the system is user-friendly and useful to its intended audience.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

In conclusion, building a movie recommendation app using machine learning and Flask can provide users with personalized movie recommendations based on their movie preferences. By collecting and preprocessing a large dataset of movie information, training a machine learning model, deploying it using Flask, and building a user interface, you can create an app that provides relevant recommendations to users.

6.2 Future Scope

- *Integration with streaming services:* You can integrate the app with popular streaming services like Netflix, Amazon Prime, Hulu, etc., to provide users with seamless access to recommended movies.

- *Enhancing recommendation algorithms*: As users interact with the app, you can continuously train and improve the recommendation model to provide more accurate and relevant recommendations.

- *Adding social features*: You can add social features like sharing recommendations with friends, creating watchlists, and commenting on movies.

- *Expanding to other domains:* Once the app is built and validated for the movie recommendation domain, it can be extended to other domains like music, books, or products.

Overall, there is a vast potential to expand and enhance a movie recommendation app using machine learning. With the advancements in machine learning and data science, this app can provide personalized and engaging experiences to users in the future.

References

https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset

https://www.sciencedirect.com/topics/computer-science/cosine-similarity

INDIVIDUAL CONTRIBUTION REPORT:

MOVIE RECOMMENDATION SYSTEM

ARYAN BHARDWAJ
20051928

Abstract: The aim of the project is to propose a movie recommendation system using cosine similarity based on the user's interests and previous watching experiences.

Individual contribution and findings: As an individual my role included designing and developing the front end of the system, and deploying the model as a web application using Python's Flask web framework. My contribution included utilizing the Flask framework to develop a user interface that was user-friendly and easy to navigate.

Additionally, I used pickle to read two pickle files: the movie list and similarity. These files contained important information that was used to generate recommendations for users. I then created a function that utilized this data to recommend movies based on a user's preferences and viewing history.

To provide a seamless user experience, I designed the user interface with two main pages: home.html and recommend.html. On the home page, users could input their preferences and select their favorite genres. On the recommend page, users received personalized movie recommendations based on their input.

Furthermore, I deployed the model as a web application using Flask, making it easy for users to access the system from any device with internet access. This helped to improve the accessibility and usability of the system.

Individual contribution to project report preparation: In summary, my contribution to the movie recommendation system project was in designing and developing the front end of the system, and deploying the model as a web application using Python's Flask web framework. My work enabled users to easily access the system and receive personalized movie recommendations based on their viewing history and preferences, providing an excellent user experience.

DEEP DUBEY
20051865

Abstract: The aim of the project is to propose a movie recommendation system using cosine similarity based on the user's interests and previous watching experiences.

Individual contribution and findings: As an individual my contribution included calculating the cosine similarity between movies and writing the code for it. The goal of the project was to recommend movies to users based on their movie preferences. To achieve this, we needed a way to measure the similarity between movies.

I suggested using cosine similarity as a metric because it is a popular method in natural language processing and has proven to be effective in measuring similarity between documents. In the context of movie recommendation, we can treat each movie as a document and use cosine similarity to measure the similarity between two movies based on their plot, cast, genre, and other features.

I wrote the code to calculate the cosine similarity using Python's sk-learn library. The code takes in a dataset of movies and their features, and outputs a similarity matrix that shows the similarity score between each pair of movies. This matrix is then used to recommend similar movies to users based on their user movie preferences.

Individual contribution to project report preparation: Overall, my contribution helped to improve the accuracy and effectiveness of our movie recommendation system. By using cosine similarity as a metric and writing the code for it, we were able to provide more personalized and relevant movie recommendations to our users.

VINAYAK MISHRA
20051184

Abstract: The aim of the project is to propose a movie recommendation system using cosine similarity based on the user’s interests and previous watching experiences.

Individual contribution and findings: To achieve this, we needed clean and reliable data that was suitable for analysis. I began by cleaning the datasets, removing duplicates and irrelevant information, and ensuring that the data was in a consistent format. This helped to eliminate errors and inconsistencies that could have affected the accuracy of our recommendations.

Next, I performed dimensionality reduction. This allowed us to extract important features from the data and reduce the dimensionality of the data, making it more manageable for analysis.

Finally, I merged two datasets of movie names and credits using Python's Pandas library. The merged dataset contained information on movie names, cast and crew, and other important features that were used to generate recommendations for users.

Individual contribution to project report preparation: Overall, my contribution to the project helped to ensure that we had clean, accurate, and well-prepared data for analysis. By combining my skills in data cleaning, preprocessing, and merging datasets, I helped to lay the foundation for a successful movie recommendation system that provided personalized and relevant recommendations to users based on their interests.

AYUSH KUMAR
2005507

Abstract: The aim of the project is to propose a movie recommendation system using cosine similarity based on the user's interests and previous watching experiences.

Individual contribution and findings: My contribution was in fetching data from two pkl files containing movie names and similarity data, and using that data to fetch the posters for the recommended movies using an API from the Movie Hub.

I utilized my skills in data manipulation and web scraping to fetch the movie titles from the data and connect them with IMDb and YouTube buttons that linked to the respective pages for the movie on those platforms. I also created a function to fetch the posters for the recommended movies using the API from the Movie Hub, which helped to provide a more visually appealing and user-friendly interface for the system.

By fetching the movie titles and linking them to IMDb and YouTube, I helped to improve the user experience and provide users with more information about the recommended movies. Additionally, by fetching the posters using the API from the Movie Hub, I added an extra level of detail to the recommendations, making them more visually appealing and informative.

Individual contribution to project report preparation: Overall, my contribution to the movie recommendation project was crucial in fetching data from the pkl files, connecting the movie titles to IMDb and YouTube, and fetching the posters for the recommended movies using an API from the Movie Hub. My work helped to create a more user-friendly and visually appealing system, enhancing the user experience and improving the overall quality of the recommendations.