



A Project Report
On
“Filmlane-A Movie Recommendation System”

In Partial Fulfillment of requirements for the Bachelor Degree in Computer
Science and Information Technology

Submitted to:
Institute of Science and Technology
Mechi Multiple Campus
Bhadrapur, Jhapa

Submitted by:
Ashok Basnet [21373/075]
Dipesh Rajbanshi [21379/075]
Konika Sitoula [21381/075]

Under the Supervision of
Mr. Pradip Khatiwada

Date:
26th April, 2023



MECHI MULTIPLE CAMPUS (MMC)

Bhadrapur, Jhapa

SUPERVISOR'S RECOMMENDATION

I hearby recommend that this project prepared under my supervision by the team of **Ashok Basnet, Dipesh Rajbanshi & Konika Sitaula** entitled “**Flimlane – A Movie Recommendation System**” is accepted as fulfilling in partial requirements for the degree of Bachelor of Science in Computer Science and Information Technology. In my best knowledge, this is an original work in Computer Science by them.

.....

Mr. Pradip Khatiwada

Supervisor

Department of Computer Science and Information Technology

Mechi Multiple Campus

Bhadrapur, Jhapa



Tribhuvan University
Institute of Science and Technology
MECHI MULTIPLE CAMPUS
Bhadrapur, Jhapa

CERTIFICATE OF APPROVAL

This is to certify that this project prepared by the team of **Ashok Basnet, Dipesh Rajbanshi, Konika Sitoula** entitled “**Flimlane – A Movie Recommendation System**” in partial fulfillment of the requirement for the degree of Bachelors of Science in Computer Science and Information Technology has been well studied and prepared. In our opinion, it is satisfactory in the scope and quality as a project for the required degree. The Project was carried out under special supervision and within the time frame prescribed by the syllabus.

Signature of Supervisor	Signature of HOD/Coordinator
Signature of External Examiner	Signature of Internal Examiner

ACKNOWLEDGEMENT

We would like to extend our heartfelt thanks to **Mechi Multiple Campus** for providing us with the platform to undertake this project work. The college has been instrumental in shaping our learning experience and has provided us with the resources and support necessary for the successful completion of this project. We would like to express our profound gratitude to our Head of Department, **Mr. Om Kanta Koirala** for his unwavering support and guidance. His expertise and experience were invaluable in the development of this project. His constant encouragement, motivation and constructive feedback helped us to overcome the challenges we faced during the course of the project. We would also like to express our deepest appreciation to our project supervisor, **Mr. Pradip Khatiwada**, for his unwavering support and guidance. His vast knowledge and experience in the field of project work helped us to understand the intricacies of the subject. His dedication and commitment to our project have been instrumental in its successful completion. Lastly, we would like to acknowledge the challenges that we faced during the course of this project and thank everyone who has helped us overcome them. This project would not have been possible without the support and guidance of our college, Head of Department, supervisor and everyone who has helped us along the way.

Project Members

Ashok Basnet[21373/075]

Dipesh Rajbanshi [21379/075]

Konika Sitoula [21381/075]

ABSTRACT

The movie industry has been growing rapidly over the years, leading to an exponential increase in the number of movies available to the audience. This vast range of options can make it difficult for movie lovers to find movies that match their taste. Movie recommendation systems have emerged as a solution to this problem, providing personalized recommendations to users based on their preferences. This paper presents a movie recommendation system that can provide movie suggestions to both new and existing users based on their preferences and interests. This movie recommendation system uses data mining techniques to gather information from a database of movies and compares this information with user input to filter and present movies with similar characteristics.

Keywords: Content-based filtering, Data Mining, Machine Learning

Table of Contents

ACKNOWLEDGEMENT	i
ABSTRACT.....	ii
List of Abbreviations	v
List of Figures	vi
List of Tables	vii
CHAPTER 1: INTRODUCTION	1
1.1. Introduction	1
1.2. Problem Statement:	2
1.3. Objectives:.....	2
1.4. Scope and limitation:.....	2
1.5. Development Methodology	3
1.6. Report Organization:	5
CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW	6
2.1. Background Study	6
2.2. Literature Review	6
2.3. Related Works	8
2.3.1. IMDb:	8
2.3.2. Rotten Tomatoes:.....	9
CHAPTER 3: SYSTEM ANALYSIS	10
3.1. Requirement Analysis	10
3.1.1. Functional Requirements	10
3.1.2. Non Functional Requirements	12
3.2. Feasibility Analysis	12
3.2.1. Technical Feasibility.....	13
3.2.2. Operational Feasibility	13
3.2.3. Schedule Feasibility.....	13
3.3. Analysis.....	15
3.3.1. STATE Diagram.....	15
3.3.2. CLASS Diagram.....	16
3.3.3. ACTIVITY Diagram	17
CHAPTER 4: SYSTEM DESIGN	18
4.1 System Design.....	18

4.1.1. UML Diagram	18
4.1.1.1. STATE Diagram.....	18
4.1.1.2. CLASS Diagram.....	20
4.1.1.3. ACTIVITY Diagram	21
4.2. Algorithm Details.....	22
CHAPTER 5: IMPLEMENTATION AND TESTING	24
5.1. Implementation.....	24
5.1.1. Tools Used.....	24
5.1.2. Data Collection	24
5.2. Testing.....	26
5.2.1. Unit Testing	26
5.2.2. Integration Testing.....	29
5.2.3. System Testing	30
5.3. Result Analysis	31
CHAPTER 6: CONCLUSION AND FUTURE RECOMMENDATIONS	32
6.1. Conclusion.....	32
6.2. Future Recommendations.....	32
References	33
Appendices	34

List of Abbreviations

API	Application Programming Language
CSS	Cascading Style Sheet
HTTP	Hyper Text Transfer Protocol
JS	JavaScript
ML	Machine Learning
UML	Unified Modeling Language
ER	Entity Relationship
HTML	Hyper Text Markup Language
IMDb	Internet Movie Database

List of Figures

Figure 1: Agile Development Model.....	4
Figure 2: Use Case Diagram.....	11
Figure 3: Gantt chart.....	14
Figure 4: STATE Diagram.....	15
Figure 5: CLASS Diagram.....	16
Figure 6: ACTIVITY Diagram.....	17
Figure 7: STATE Diagram.....	19
Figure 8: CLASS Diagram.....	20
Figure 9: ACTIVITY Diagram.....	21
Figure 10: Content Based Filtering.....	22
Figure 11: Home Page.....	34
Figure 12: Top Rated Movies.....	34
Figure 13: Movie Details with Recommended Movies.....	35
Figure 14: Registration Page.....	36
Figure 15: LogIn Page.....	36
Figure 16: Main Code for Recommendation.....	37

List of Tables

Table 1: Companies benefit though recommendation system according to 2016 A.D.....	2
Table 2: Unit testing for Registration Module.....	27
Table 3: Unit testing for Login Module.....	28
Table 4: Unit testing for Search Module.....	28
Table 5: For Successful Integration.....	29
Table 6: For Unsuccessful Integration.....	30
Table 7: System Testing.....	31

CHAPTER 1: INTRODUCTION

1.1. Introduction

Film lovers are always on the hunt for their next favorite movie. With so many options available, it can be difficult to choose which movie to watch next. In the era of online streaming platforms such as Netflix, Amazon Prime, YouTube and many more which provides millions of videos, it can be challenging to pick the suitable one that matches user's interests and preferences. This is where movie recommendation systems come into play, and one of the most popular methods for recommendation is content-based filtering. Content-based filtering uses an algorithm that analyzes the features of movies such as genre, director, actors, and plot, and recommends similar movies to the user. One such system that utilizes content-based filtering is "Filmlane", a movie recommendation system that provides personalized recommendations to its users based on their movie preferences.

"Filmlane" is a web-based tool that helps in discovering movies and their descriptions. Its primary purpose is to assist users in finding a movie and suggesting similar ones based on their search. With the rise of advanced recommender systems, users now have high expectations for receiving quality recommendations, and they quickly lose interest in services that cannot deliver. Technology companies are, therefore, placing significant emphasis on enhancing their recommendation systems. However, the issue is more complicated than it appears, as each user has unique preferences influenced by various factors, such as mood, season, or activity. Consequently, relying on a simplistic approach of suggesting popular and top rated movies may not be sufficient. For instance, a user's preferred music genre while upset may differ significantly from the one they like while in excitement.

Table 1: Companies benefit though recommendation system according to 2016 A.D.

Netflix	2/3 rd of the movies watched are recommended
Google News	Recommendations generate 38% more click-through
Amazon	35% sales from recommendations

1.2. Problem Statement:

Due to the endless options of online movies available, it can be challenging to locate a movie that matches with one's preferences. People invest their valuable time browsing on the internet to find a suitable movie that matches their interests. As movie preferences vary from person to person, it is necessary to read reviews and seek feedback from others before watching a film, which can be a troublesome process.

1.3. Objectives:

The objectives of the movie recommendation system are:

- To provide an online platform where movie lovers can find movies of their interests.
- To apply Content-based Filtering technique for precise recommendations.

1.4. Scope and limitation:

The scope of recommendation system in today's world is increasing day by day. With so many movies online, it becomes quite hard for the user to search the movie based on his interest. This is where the recommendation system becomes useful.

The scopes of this project are:

- It can be used to recommend movies based on the user's search content.
- It makes searching of similar movies fast with the help of recommendation

algorithm.

- It can be used as a platform to engage different types of users.
- Content based recommendations are highly relevant to the user.

Though the demand of recommendation system is very high, it has certain limitation.

Some of the limitations of this project are:

- It does not allow users to rate movies.
- It is based on already saved dataset.
- Content loading takes a long time.

1.5. Development Methodology

There are various types of software development lifecycle model used while developing software. Among them the most popular methodology is the agile model. In this project, we have used the agile model to develop our system. Agile methodology is a software development process that involves the collaborative effort of self-organizing and cross-functional teams, as well as customers, to evolve demands and solutions. It is a highly effective and straightforward approach to transforming a business need into software solutions. It involves iterative and incremental development, delivering working software in small, frequent releases. Key principles of Agile include delivering working software frequently, embracing change, and collaborating with customers and stakeholders. Agile is a flexible and adaptable methodology that emphasizes collaboration with customers and stakeholders to respond quickly to changes and deliver working software. The Agile process involves breaking the project down into smaller chunks of work called "sprints." The Agile process also involves continuous testing, integration, and delivery of the software, ensuring that it is always in a usable state.

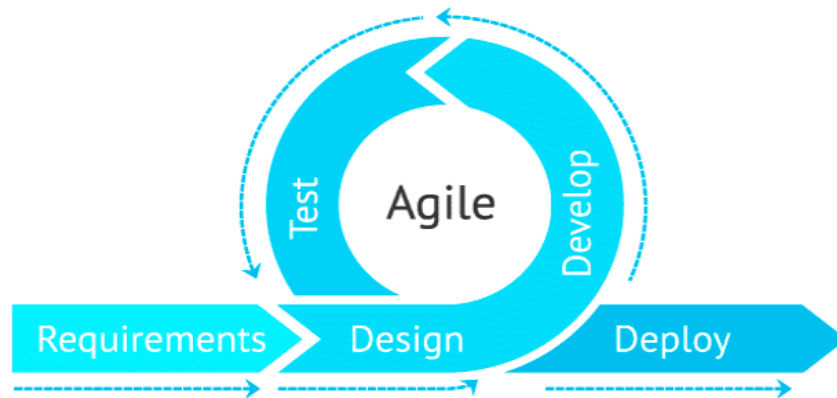


Figure 1: Agile Development Model

To develop a movie recommendation system using agile development Model, different phases involved are as follows:

Requirement Gathering: At first, all the requirements related to the project are gathered.

Iteration: Iteration includes design, develop and test. In this phase, design of different components of the system is carried out. After designing elements, they are developed and tested.

Deploy: The final phase is deploy where the program is deployed on a server for everyone to access.

The Agile method is a good way to develop a movie recommendation system for a few reasons. Firstly, it encourages the team to make changes often and keep updating the system with user feedback. Secondly, it helps different team members, like data analysts, machine learning experts, and software developers, work together to make sure the system works well. Lastly, it allows the team to easily adjust the system to keep up with changes in the movie industry and make sure it stays useful for users.

1.6. Report Organization:

This document is categorized into several chapters and further divided into sub chapters including all the details of the project.

- Chapter 1: It is the introduction section which includes short introduction of the system, scope and limitations and objectives of the system.
- Chapter 2: This chapter covers research methods, as well as a background study and a review of relevant literature.
- Chapter 3: It is all about system analysis. It also includes feasibility study and requirement analysis.
- Chapter 4: It includes the System Design and details of the Algorithm.
- Chapter 5: This chapter explains the implementation and testing procedures. It contains the details about the tools that are required to design the system. Different testing processes are included in this section.
- Chapter 6: It includes conclusion of the whole project. It also provides details regarding additional possibilities that can be achieved through this project.

CHAPTER 2: BACKGROUND STUDY AND LITERATURE REVIEW

2.1. Background Study

In the last decade, numerous recommendation systems have been created and are being utilized in various domains. These systems use diverse approaches like content-based, collaborative, knowledge-based, utility-based, and hybrid techniques. Online recommendation systems typically rely on previous user ratings to provide suggestions to current users who share similar interests. An example of such a system was developed by Jung, Harris, Webster, and Herlocker in 2004, which aimed to enhance search results. This system motivates users to input longer and more informative search queries and collects ratings from them regarding whether the search results met their information needs or not. These ratings are then employed to recommend relevant results to later users who share similar interests.

2.2. Literature Review

The main focus of this research paper is to explore the recommendations generated through Content-based Filtering by using multiple techniques for converting text to vectors. The researchers combined several algorithms to produce a final list of recommended movies. The method used in this study involved preprocessing the data set obtained from kaggle.com and consolidating relevant features into a single feature. The text from this feature was then converted into vectors to determine similarity, and recommendations were generated based on these vectors. The results showed that the final recommendations were slightly better than those generated by individual algorithms such as Content-based Recommendation using CountVectorizer and Cosine Similarity (Algorithm 1) and Content-based Recommendation using TfidfVectorizer and Cosine Similarity (Algorithm 2). The researchers concluded that it is beneficial to combine different algorithms to obtain the best results. [1]

The aim of this research is to improve the CineMatch algorithm by 10% by incorporating modern collaborative filtering techniques. Various machine learning techniques are employed to build a movie recommendation system, utilizing movie ratings to construct a Similarity matrix and Sparse matrix. The User-Item Sparse Matrix, User-User Similarity Matrices, Item-Item Similarity Matrix, and Cold Start Problem are among the methods employed in this system. Different algorithms, including XGBoost, Matrix Factorization SVD, Surprise Baseline, and Surprise KNNBaseline Predictor, are tested on the Netflix Prize dataset. The SVDpp algorithm performed the best, achieving a Test RMSE of 1.0675. [2]

This article outlines the creation of a movie recommendation system that utilizes information on cast, keywords, crew, and genres. The system combines these attributes into a single column, which serves as the main factor for the recommender. The approach used in this content-based recommendation system predicts what movies a user will enjoy based on their previously liked films and their attributes. Unlike collaborative filtering, which relies on user ratings and preferences, content-based systems are not affected by unrated items and can provide personalized recommendations without considering other user profiles. [3]

In this study, a movie recommendation system is developed using a content-based filtering approach with a hybrid deep learning model that combines Monarch Butterfly Optimization and Deep Belief Network. The goal is to recommend movies based on the user's Facebook profile. The system uses feature extraction, vector creation, filtering, weighting, and feature selection methods to classify movies. The Facebook dataset and the MovieLens dataset are used for performance evaluation, and the proposed model's performance is compared to that of other state-of-the-art algorithms. The results demonstrate that the proposed model outperforms other algorithms in terms of MAE, RMSE, Precision, and Recall. The MAE and RMSE values for the proposed model are 0.716 and 0.915, respectively, and its precision and recall are 97.35% and 96.60%, respectively. The study concludes that the proposed model is effective in recommending movies to users based on their Facebook profile. [4]

In this study, a new approach was proposed that measures the cosine similarity and levenshtien distance which together becomes a hybrid model which recommends movies based on features extracted such as titles, genres, directors, actors, and plots content. Two clear steps were separated to deploy the measurement. The first step was to transform all extracted features from the movies to the vectors. Specially, the Vector Space Model was applied for each plot content of a movie. In the second step, measurement was done for each feature by using Cosine-based Similarity and Levenshtien distance then a set of similar movies was obtained. For content-based recommender system specifically, attempt was done to find a new way to improve the accuracy of the representative of the movie. Anew approach was introduced for setting weight for those features, the movie can be represented more accurately by TF-IDF algorithm also sentiment analysis was done with the help of TF-IDF Vectorizer and multinomial Naïve-Bayes classifier. [5]

2.3. Related Works

Some of the related sites are:

2.3.1. IMDb:

IMDb is an online database that provides comprehensive information related to films, TV shows, video games, and streaming content. The website offers data about the cast, production crew, plot summaries, personal biographies, trivia, ratings, and reviews from both fans and critics. Users can register on the site and rate any movie on a scale of 1 to 10. The system then calculates the weighted mean-rating for each title. IMDb was launched in 1990 and has become one of the world's largest and most popular movie and TV databases. The website's success can be attributed to its vast collection of content, user-friendly interface, and advanced search engine. Additionally, the website offers various features such as personalized recommendations, watchlists, and movie and TV show rankings. IMDb also has a mobile app that allows users to access the site's content on their smartphones or tablets. Overall, IMDb is an essential resource for movie enthusiasts and industry professionals alike, providing extensive information about the world of film and TV.

2.3.2. Rotten Tomatoes:

Rotten Tomatoes is a popular American review-aggregation website for films and television. Its primary objective is to provide access to reviews from a variety of critics. Initially, the team at Rotten Tomatoes gathers online reviews from writers who are certified members of different writing guilds or associations of film critics. In order to become a recognized critic on the website, the critic's reviews must receive a certain number of "likes" from users. Rotten Tomatoes then monitors all the reviews collected for each film and use them to determine the percentage of positive reviews. It then categorizes each movie as "fresh" or "rotten" based on the percentage of positive reviews. The website also displays an overall score for each movie, calculated as a weighted average of all the reviews it has received. In addition to reviews, Rotten Tomatoes offers movie and TV show trailers, news articles, and interviews with actors, directors, and other industry professionals. The site's community features allow users to rate and review movies and TV shows, create personalized watch lists, and engage in discussion forums. Overall, Rotten Tomatoes is a comprehensive resource for movie and TV enthusiasts seeking to discover new content and make informed viewing decisions.

CHAPTER 3: SYSTEM ANALYSIS

3.1. Requirement Analysis

Requirement analysis is a process that involves frequent communication with system users to identify their specific feature expectations. It also includes resolving conflicts or ambiguity in requirements as per the diverse users' demands, avoiding feature creep, and documenting all aspects of the project development process from start to finish. The primary objective of the system analyst is to determine the requirements of the new system that needs to be developed, for that the study of specification of the requirements is very essential. The development of the new system starts with a preliminary survey of the existing system, and an investigation is conducted to determine whether upgrading the system to an application program can address the problems and eliminate the inefficiencies of the existing system.

Two types of requirements are analyzed while developing a system. They are:

3.1.1. Functional Requirements

A functional requirement refers to the description of the service that the software should provide, which encloses the software system or its components. This requirement outlines the entire workflow that the developed system is expected to perform.

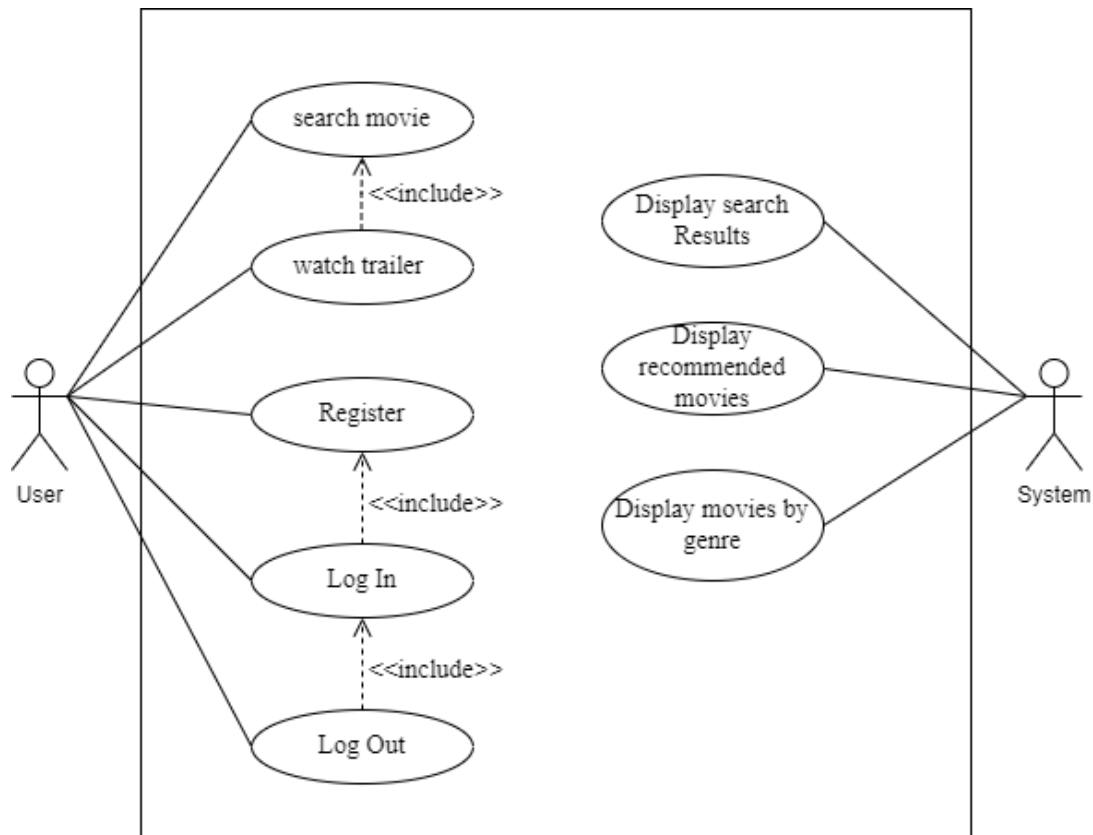


Figure 2: Use Case Diagram

The functional requirements of this system are:

3.1.1.1. Search Movie:

The user searches a movie on the search bar. When the search movie request is being made to the server, the other similar movies recommendation request is also made to the server.

3.1.1.2. Watch Trailer:

The client application provides an interface where the user can watch trailer of the searched movie.

3.1.1.3. Handle user movie requests:

The server application receives and handles the requests made by the users for movies details.

3.1.1.4. Recommend using Content-Based Filtering:

The server application is capable of producing recommendations by interpreting contextual information given by the user. The server consist an algorithm called 'Content Based Filtering' which uses Cosine Similarity Measure to recommend similar movies based on the searched content by the user. For example: if a user searches "Titanic" of genre 'romance' then other similar movies of action genre are recommended by the algorithm.

3.1.2. Non Functional Requirements

Non-functional requirements are rules that are used to evaluate how well a system works. These rules define things like safety, how dependable the system is, how fast it performs, and how easy it is to use. They limit how the system can be designed across different stages. Some non-functional requirements are described below:

3.1.2.1. Accessibility:

This project is a web-based application, which means that any user can access the system regardless of their device, operating system, or platform.

3.1.2.2. Maintainability:

New features can be added in the project based on the user requirements. Since the code is very simple and well organized, it is easier to find and correct the defects and to make the changes in the project.

3.1.2.3. Portability:

This project can be executed under different operation conditions provided it meets its minimum configuration. Only system files and dependent assemblies would have to be configured in such case.

3.2. Feasibility Analysis

Feasibility studies attempt to rationally and objectively identify the positive and negative aspects of an existing or proposed system, as well as the opportunities and risks presented by the outside world, the resources needed to implement the plan, and finally the

likelihood of success. In this stage, the project's feasibility is analyzed, and a business proposal is presented with a very basic project plan and some cost projections. The proposed system's practicality must be studied during system analysis. This will guarantee that the proposed system won't burden the business.

There are three main types of feasibilities to be considered under this analysis. They are:

- Technical Feasibility
- Operational Feasibility
- Schedule Feasibility

3.2.1. Technical Feasibility

The current system that we are building is web-based portal. It uses HTML, CSS, and JavaScript as front-end and Python as a back-end language. It is based on client server architecture and needs internet connection to access the information. It supports windows and Linux platform for its operation. All the technology required by the application are available and can be accessed freely, hence it is technically feasible.

3.2.2. Operational Feasibility

The success of this project relies on the practical implementation of a system that meets the users' needs. This system provides a simple user interface in web platform, which can be easily used by any type of users having basic ideas of using smart phones and PC's. This system provides correct results according to the way the systems need to do. Hence this will be operationally feasible too.

3.2.3. Schedule Feasibility

The project was estimated to be fully developed in 6 months. The total project development task is sub divided into various phases and time was allocated as per requirements. Time Schedule / Gantt chart used during the project development phase according to the methodology used is given below:

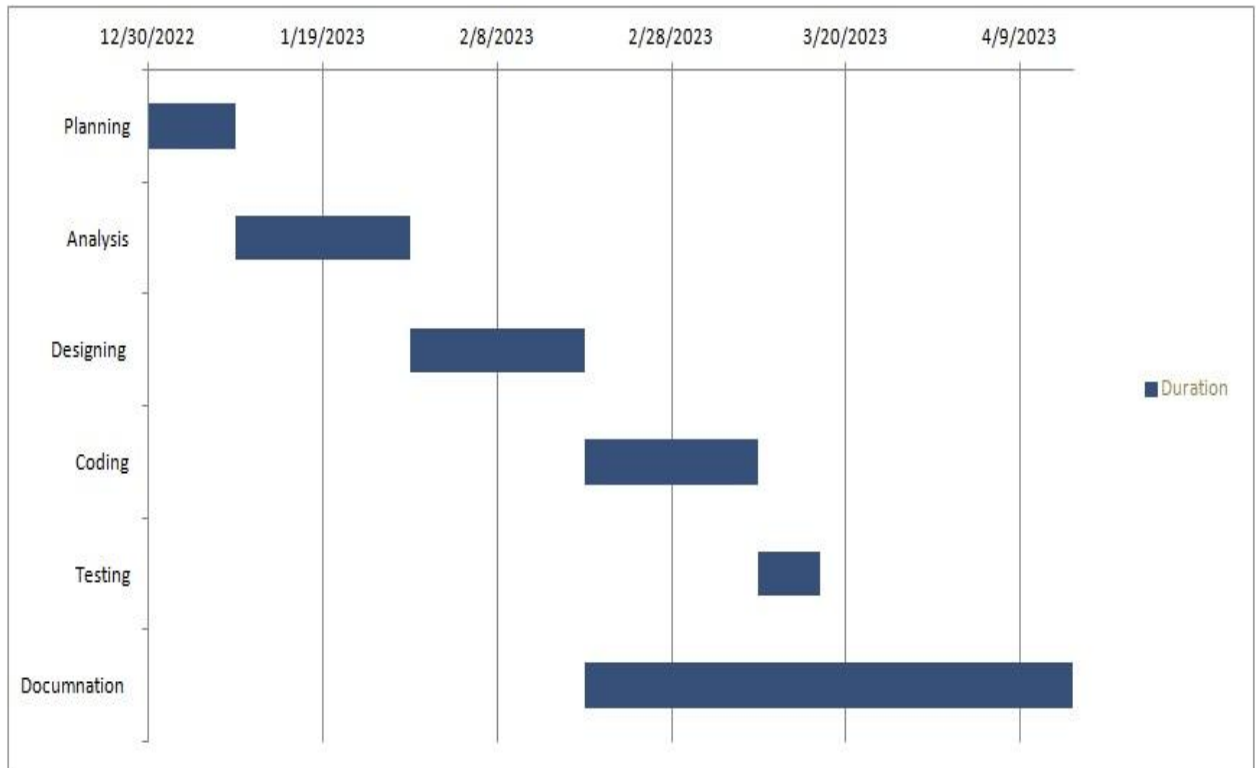


Figure 3: Gantt chart

3.3. Analysis

3.3.1. STATE Diagram

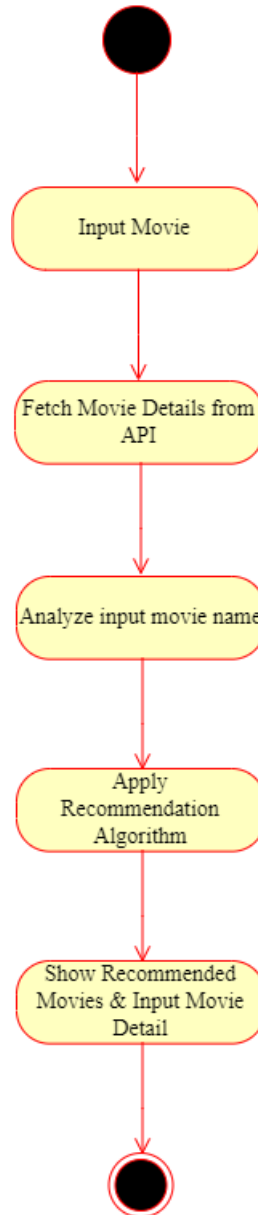


Figure 4: STATE Diagram

3.3.2. CLASS Diagram

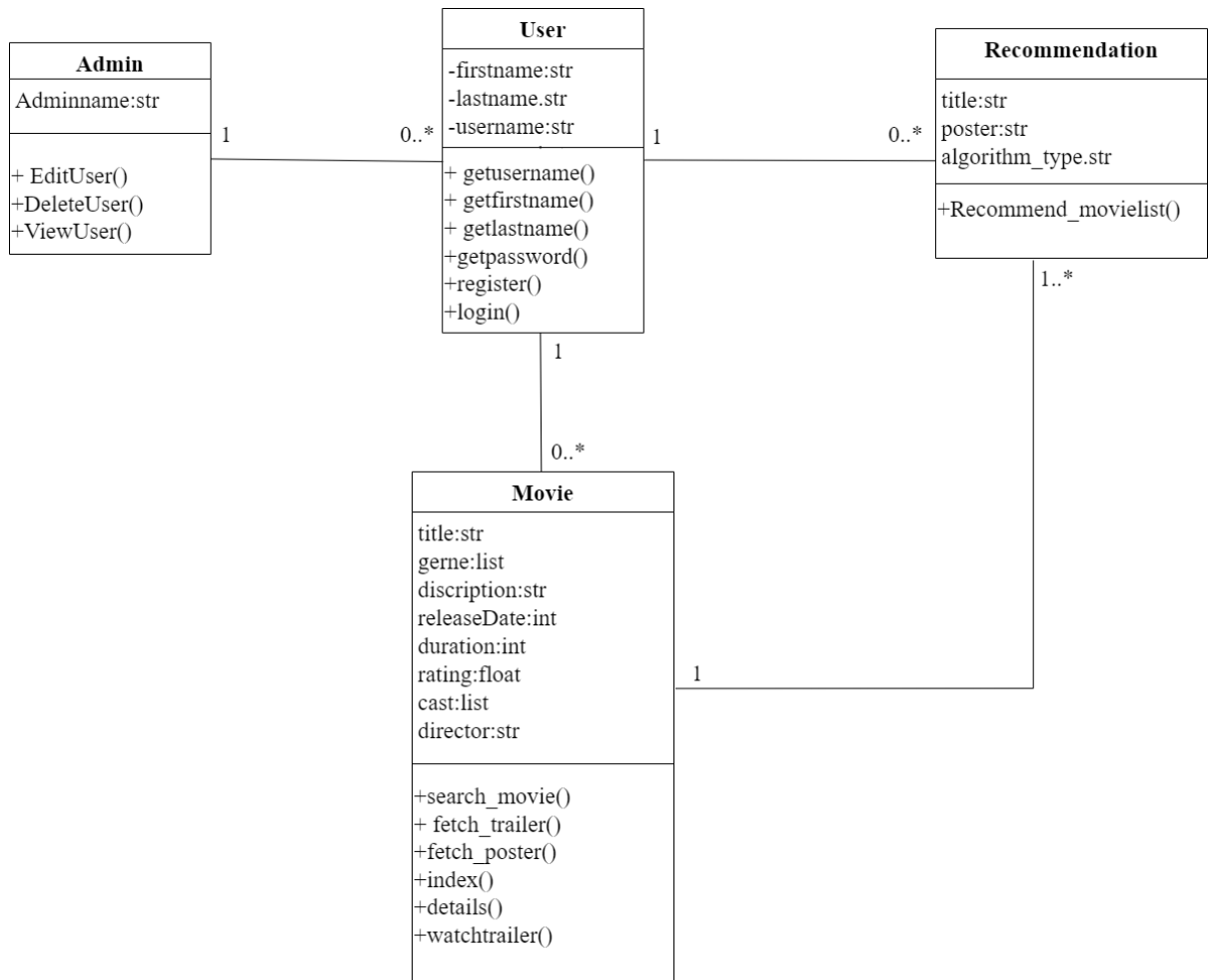


Figure 5: CLASS Diagram

3.3.3. ACTIVITY Diagram

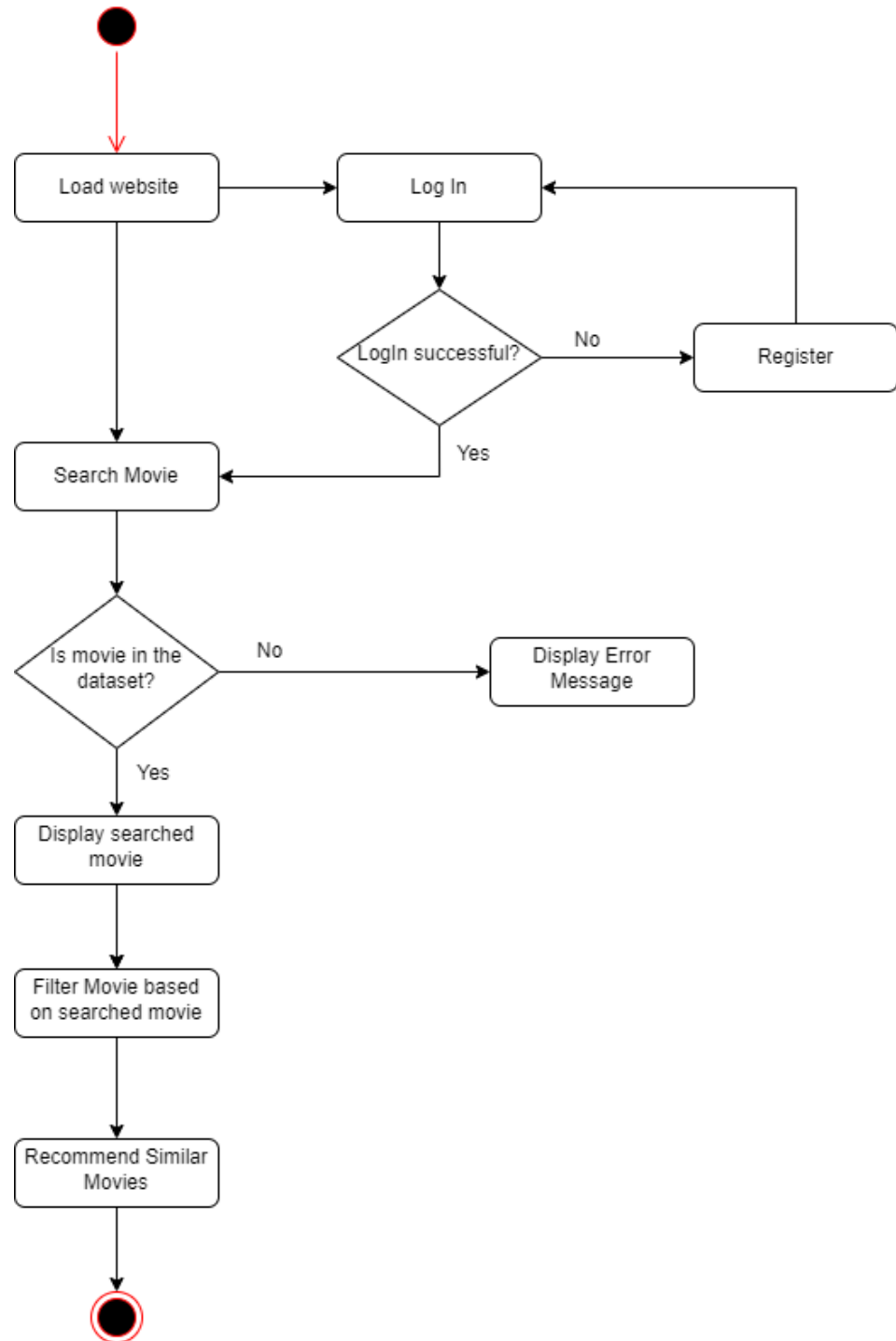


Figure 6: ACTIVITY Diagram

CHAPTER 4: SYSTEM DESIGN

4.1 System Design

System design is an essential step in software development that requires significant effort and cost. To minimize errors and reduce costs, a prototyping model is often used to test software. The goal of system design is to plan and implement a practical solution to a problem, accomplished through two phases: logical design and physical design. The logical design phase determines how the system should operate in specific situations, including input, output, datasets, and procedures. The physical design phase involves implementing the logical design, focusing on the system's output and coding to produce a functional system that meets user needs.

4.1.1. UML Diagram

A UML diagram for a movie recommendation system includes Use Case, Class, Sequence, and Activity diagrams. The Class diagram shows objects such as User, Movie, Genre, and Recommendation. The Sequence diagram demonstrates interactions and messages exchanged during the recommendation process. The Activity diagram shows the flow of activities or processes, like generating recommendations. The UML diagram provides a clear representation of the system's structure and functionality, aiding developers in designing, implementing, and testing the system.

4.1.1.1. STATE Diagram

A state diagram is a type of diagram used in software engineering to represent the behavior of a system. It shows the different states that the system can be in, how it can transition between these states, and what events cause these transitions. State diagrams are commonly used to model the behavior of complex systems, such as software applications, hardware devices, and communication protocols.

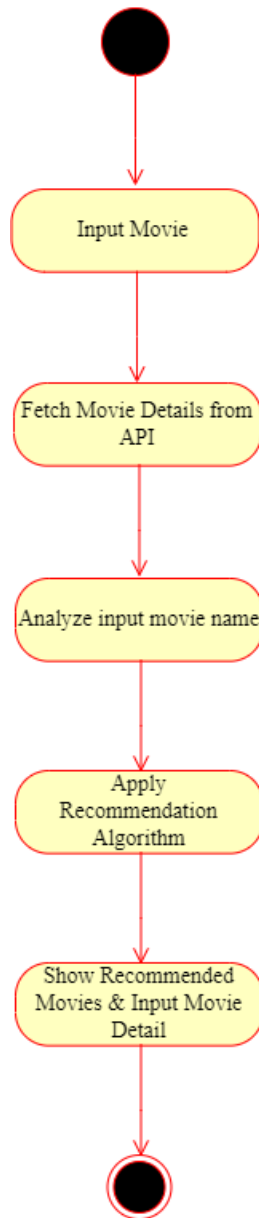


Figure 7: STATE Diagram

4.1.1.2. CLASS Diagram

A class diagram for a movie recommendation system illustrates the various classes and their relationships, such as Movie Input Data, Algorithm, Movie Selection, and Movie Suggestion. The User class may have attributes like Name, Age, and Preferences, while the Movie class may have attributes like Hindi Movie Details, English Movie Details, English Movie Names, Poster. The Recommendation class may have methods like Analysis() and Gerne_Selection(), Movie_Name_Selection(). Overall, the diagram provides a visual representation of the key components of the system and their interactions.

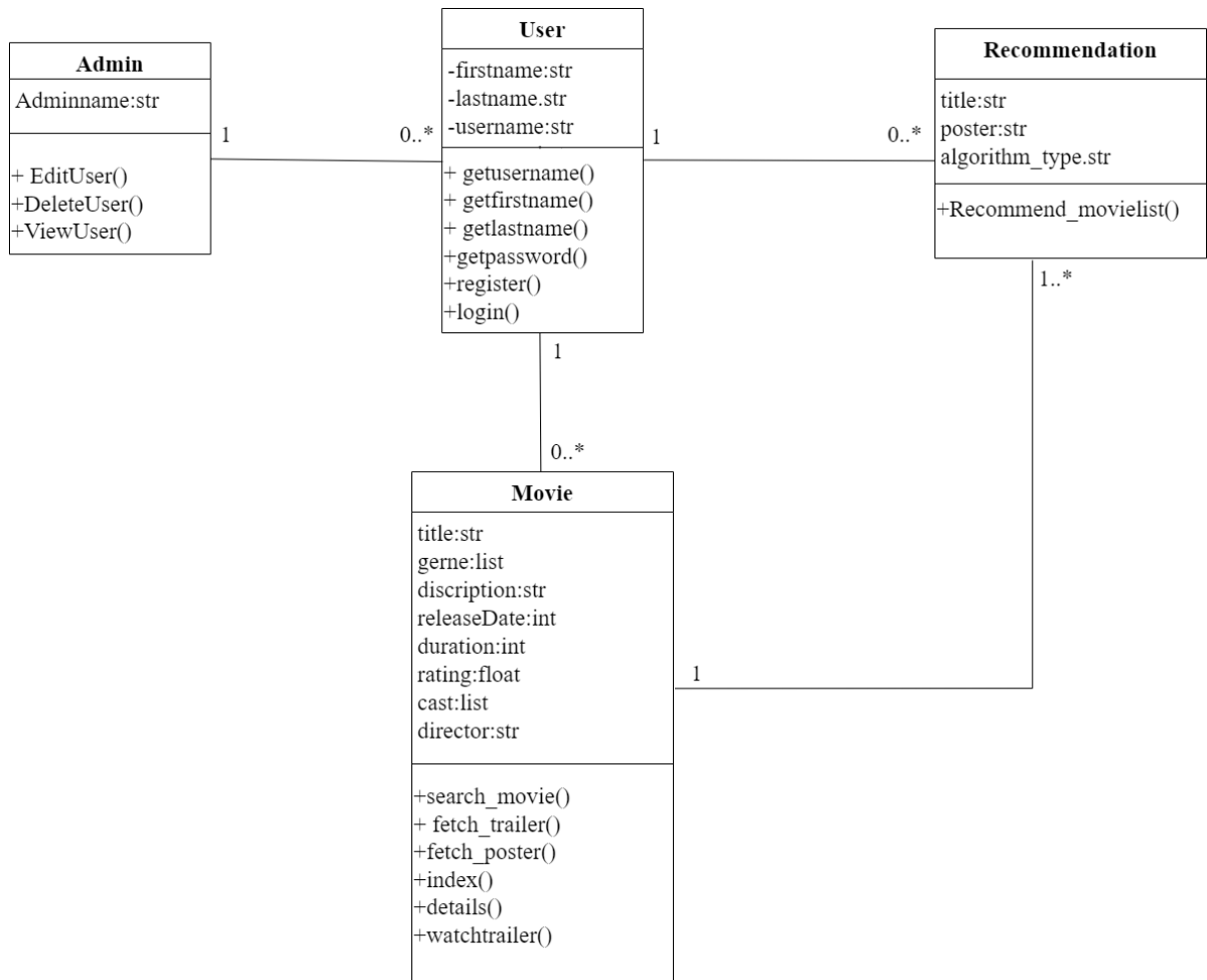


Figure 8: CLASS Diagram

4.1.1.3. ACTIVITY Diagram

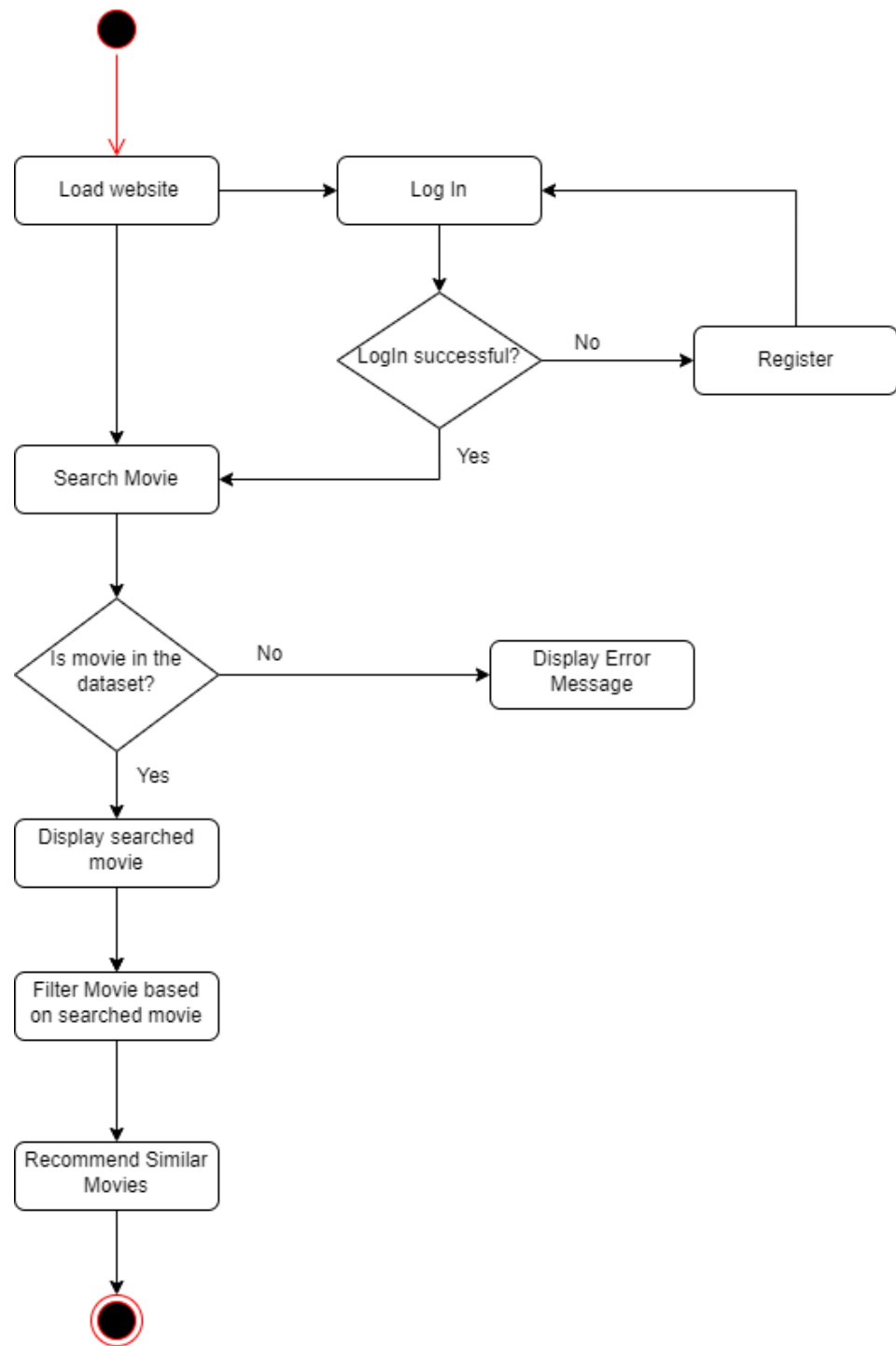


Figure 9: ACTIVITY Diagram

4.2. Algorithm Details

The project uses Content Based Filtering algorithm to recommend similar movies based on the search content of the user. Content-based filtering uses item gestures to recommend other items similar to what the user likes, based on their previous actions or explicit feedback.

Content-based filtering algorithm for movie recommendation system:

Step 1: Get user's watched history and movie preferences.

Step 2: Preprocess the movie data by extracting relevant features (such as genre, actors, director, year of release, etc.).

Step 3: Calculate the similarity score between each movie and the user's preferences based on the extracted features.

Step 4: Sort the movies in descending order of their similarity score with the user's preferences.

Step 5: Recommend the top N movies to the user.

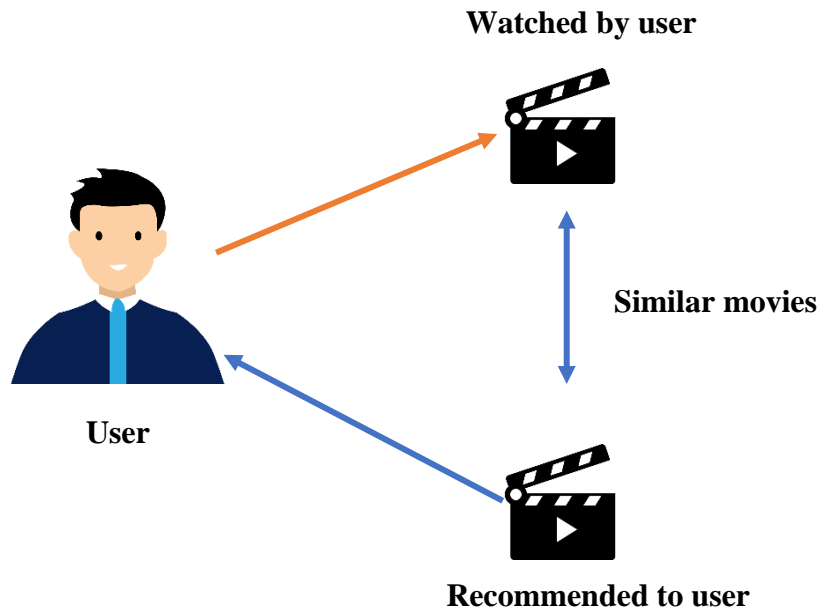


Figure 10: Content Based Filtering

Content-based filtering uses cosine similarity measure to recommend similar movies to the user. This algorithm calculates a similarity score between the attributes of the items in the system and the user's preferences. This score is computed using cosine similarity, which measures the similarity between two vectors. The algorithm compares the attributes of the items and the user's preferences by calculating the cosine similarity score between them. This score ranges from -1 to 1, where 1 means the items and user preferences are identical, 0 means they are completely different, and -1 means they are completely dissimilar. The algorithm then generates recommendations based on the top-ranked items that are most similar to the user's preferences.

$$u(c, s) = \cos(\vec{w_c}, \vec{w_s}) = \frac{\vec{w_c} \cdot \vec{w_s}}{\|\vec{w_c}\| \times \|\vec{w_s}\|}$$

$$= \frac{\sum_{i=1}^K w_{i,c} w_{i,s}}{\sqrt{\sum_{i=1}^K w_{i,c}^2} \sqrt{\sum_{i=1}^K w_{i,s}^2}}$$

In information retrieval, a scoring heuristic is often used to represent the utility function $u(c, s)$ based on vectors \vec{c} and \vec{s} , which can be measured using a cosine similarity measure. The number of keywords in the system is represented by K . By using cosine similarity or a similar measure, a recommender system can assign higher utility $u(c, s)$ to articles with more heavily weighted bioinformatics terms in \vec{s} and lower utility to articles with less heavily weighted bioinformatics terms.

CHAPTER 5: IMPLEMENTATION AND TESTING

5.1. Implementation

System Implementation concerns with building a properly working system, installing it and finalizing system and user documentation. Implementation also involves closing down of project. After finalizing the conceptual design of the system, we started coding to create the necessary functionality, in order to make it practical. The idea behind this project was always to develop a system that should be ease in availability, usability and access. Web Application was selected because it closely matches the thought of the system begun with. Although the process of documentation proceeds throughout the life cycle, it receives formal attention during the implementation phase. This form of document is prepared to reveal of the information accumulated about the system during development and implementation of this system. Finally, this phase ensured that the system meet all the specifications and objectives developed in earlier project phases. The tools and technologies used to implement this project are briefly discussed on the following section.

5.1.1. Tools Used

Some of the tools that are used in our project are listed below:

- CASE tools: Visual Studio Code, draw.io, Git, GitHub
- Programming Languages: HTML, CSS, Bootstrap, JavaScript, Python, Django
- Database Platforms: SQLite, API

5.1.2. Data Collection

Collecting data is an essential step in building a movie recommendation system. The system needs to accumulate a broad and varied range of information related to the movies, including data on the movie titles, genres, release dates, top 3 casts, director,

description and ratings. For this particular project, the data is not the primary focus, and it has been sourced from kaggle.com to train and test the system. The data has been extracted from two CSV dataset files, namely, `tmdb_5000_credits.csv` and `tmdb_5000_movies.csv`, which contain a total of 4806 data points. These files have been pre-processed and combined to form a final dataset file named "`finaldataset.csv`" that will be used for our recommendation system.

5.1.3. Implementation Details of Modules

- **Accounts:** It consists of attributes: user name, first name, last name, e-mail and password. It consists of register method which allows a user to register a user and login method that allows a registered user to login to the system.
- **Search Data:** It allows a user to input a movie name and search the movie details as well as recommends other similar movies to the user.
- **Movie class:** It consists of attributes like movie title, released date, director, duration, cast, genre, rate, origin language, trailer, description. It consists of methods: `index`, `search`, `details`, `fetch_poster`, `fetch_trailer` and `recommendation`.
 - **Index method:** This method returns the movie poster using movie id.
 - **Search method:** This method is used to search movies as well as to fetch the movie details and to recommend other similar movies.
 - **Details method:** This method returns the details of the searched movie like: movie title, genre, duration, rate, cast, etc.
 - **Recommendation method:** This method uses Content-Based filtering algorithm to recommend movies that are similar to searched movie.
 - **Fetch_poster method:** This method is used to fetch movie poster from `themoviedb.org` API using movie id.
 - **Fetch_trailer method:** This method is used to fetch movie trailer from `themoviedb.org` API using movie id.

5.2. Testing

Testing is a process that involves examining a software application or system to identify any issues or defects between expected and actual results. The main goal of testing is to make sure that the software meets the specified requirements, performs as expected, and is free from bugs. There are various types of testing, such as unit, integration, system, acceptance, and regression testing. Testing can be performed manually or with automated tools and methods. The testing process involves creating test cases, executing tests, analyzing results, and reporting issues.

5.2.1. Unit Testing

Unit testing is a type of software testing where individual units or components of a software application are tested in isolation from the rest of the system. The goal of unit testing is to verify that each unit or module of code is functioning correctly and producing expected results. Unit tests are typically automated and focus on testing a specific function, method, or class, using a set of inputs and comparing the output to an expected result. Unit testing helps to detect errors and bugs early in the development cycle, which makes it easier and less expensive to fix them. It is an important part of ensuring the quality and reliability of software applications.

Table 2: Unit testing for Registration Module

Test case	Precondition	Steps	Expected Result	Actual Result
Test if user is able to sign up successfully	User must enter all the details according to format and precautions.	1. Correct information is passed in the all fields while registering. 2. Click on sign in Button. Example: Username: Messi Email: messi10@gmail.com Password: messi10 Confirm password: messi10	The registration is successful.	The new user is Successfully registered in the system.
Test if user is able to signup successfully	User must enter all the details according to format and precautions.	1. Correct information is passed in the all fields while registering. 2. Click on sign in Button Example: Username: Messi Email: Messi10@gmail.com Password: messi10 Confirm password: messi	The registration is unsuccessful	New user registration process is unsuccessful.

Table 3: Unit testing for Login Module

Test case	Precondition	Steps	Expected Result	Actual Result
Login for users.	The user should be already registered.	1. Enter correct user name and password 2. Click Login	User must successfully login to user profile.	User is successfully logged in.
Login for users.	The user should be already registered.	1. Enter correct user name and incorrect password 2. Click Login .	User cannot access user profile.	Login unsuccessful.

Table 4: Unit testing for Search Module

S. No.	Test Case	Steps	Test Data	Test Result
1	Search Movie.	1. Enter a movie name.	Spider-Man 2	Search successful.
2	Search Movie.	1. Enter a movie name.	Spider	Search successful.
3	Search Movie.	1. Enter a movie name.	Kabaddi	Search Unsuccessful.

5.2.2. Integration Testing

Integration testing of this system involves testing of registration, login and search modules by combining them together. The goal of integration testing is to ensure that the components of the system are functioning correctly and produces the desired result.

Table 5: For Successful Integration

S No.	Test Case	Steps	Test Data	Expected Result
1	Register the user.	1. Go to registration page.	User name: Konika First name: Konika Last name: Sitoula Email:konika23@gmail.com Password: 001122 Confirm Password: 001122	Registration successful.
2	Verify Login.	1. Go to login page. 2. Enter correct username and password. 3. Click Login.	User name: Konika Password: 001122	Login successful.
3	Search Movie.	1. Enter a movie name.	Titanic	Search successful.

Table 6: For Unsuccessful Integration

S No.	Test Case	Steps	Test Data	Expected Result
1	Register the user	1. Go to registration page. 2. Enter username, first name, last name, email, password, confirm password. 3. Click on Register.	User name: Konika First name: Konika Last name: Sitoula Email:konika23@gmail.com Password: 001122 Confirm Password: 1122	Registration unsuccessful.

5.2.3. System Testing

After the successful completion of unit testing and integration testing, the system as a whole was tested to ensure whether the developed system was working well and that the system meets its requirements or not. The test results were positive which means that the system was working perfectly fine and has met the system requirements which were analyzed during the analysis and requirement phase.

Table 7: System Testing

S No.	Test Case	Steps	Test Data	Expected Result
1	Register the user.	1. Go to registration page.	User name: Konika First name: Konika Last name: Sitoula Email:konika23@gmail.com Password: 001122 Confirm Password: 001122	Registration successful.
2	Verify Login.	1. Go to login page. 2. Enter correct username and password. 3. Click Login.	User name: Konika Password: 001122	Login successful.
3	Search Movie.	1. Enter a movie name.	Spider-Man 2	Search successful.

5.3. Result Analysis

After the movie name was given as an input by the user, the corresponding details of the input movie like title, genre, cast, released date, description, rating, etc. were shown and right below all those details the list of recommended movies using Content Based Filtering algorithm was shown. The main part of the project was to recommend other similar movies based on the search content of the user using the Content Based Filtering algorithm and the system successfully recommended the movies to the user.

CHAPTER 6: CONCLUSION AND FUTURE RECOMMENDATIONS

6.1. Conclusion

This system successfully recommends movies to the user based on the content they search. By using Content based filtering algorithm that uses cosine similarity to analyze the content of movies such as genre, cast, director, and plot, the system was able to recommend 5 similar movies to the user based on their input. With the implementation of this system, the users are able to find movies of their preferences in a short period of time. The expected output of recommending similar movies to the user based on user's searched content and providing details about the movie, cast, release date, ratings, descriptions, etc. is successfully achieved.

During the entire project development period we were able to develop our skills. This project enabled us to manage time and resource besides of different constraints.

6.2. Future Recommendations

We can optimize this system by using different techniques to improve its performance and enhance user experience. This can be achieved by reducing the web-app load time, providing movie rating options, and enabling users to add movies to their wish list. By implementing these features and techniques, web applications can become faster, more efficient, and provide a better overall experience for users.

References

- [1] M. B. Yeole, M. D. Rokade and S. S. Khatal, "Movie Recommendation System using Content based Filtering," vol. 7, no. 4 , pp. 633-648, 2021.
- [2] M. C. Keshava, S. Srinivasulu, P. N. Reddy and B. D. Naik, "Machine Learning Model for Movie Recommendation System," *International Journal of Engineering Research & Technology (IJERT)*, vol. 9, no. 4, pp. 800-805, 2020.
- [3] N. Pradeep, R. K. Mangalore, B. Rajpal, N. Prasad and R. Shastri, "Content Based Movie Recommendation System," *International Journal of Research in Industrial Engineering*, vol. 9, no. 4, pp. 337-348, 2020.
- [4] S. Sridhar, D. Dhanasekaran and P. C. Latha, "Content-Based Movie Recommendation System Using MBO with DBN," *Intelligent Automation & Soft Computing*, vol. 35, no. 3, pp. 3241-3257, 2022.
- [5] O. Kunde, O. Gaikwad, P. Kelgandre, R. Damodhar and P. M. M. M. Swami, "The Movie Recommendation System using Content Based Filtering with TF-IDF Vectorization and Levenshtein Distance," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, vol. 2, no. 2, pp. 257-263, 2022.
- [6] P. Sharma and L. Yadav, "Movie Recommendation System Using Item Based Collaborative Filtering," *International Journal of Innovative Research in Computer Science & Technology (IJIRCST)*, vol. 8, no. 4, pp. 266-270, 2020.
- [7] M. B. Yeole, M. D. Rokade and S. S. Khatal, "MOVIE RECOMMENDATION SYSTEM APPROACH USING CLASSIFICATION TECHNIQUE," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 3, no. 7, pp. 1767-1771, 2021.
- [8] R. Marappan and S. Bhaskaran, "Movie Recommendation System Modeling Using Machine Learning," *International Journal of Mathematical, Engineering, Biological and Applied Computing*, vol. 1, no. 1, pp. 12-16, 2022.
- [9] B.-B. CUI, "Design and Implementation of Movie Recommendation System Based on Knn Collaborative Filtering Algorithm," *ITM Web of Conferences*, no. 12, pp. 1-5, 2017.

Appendices

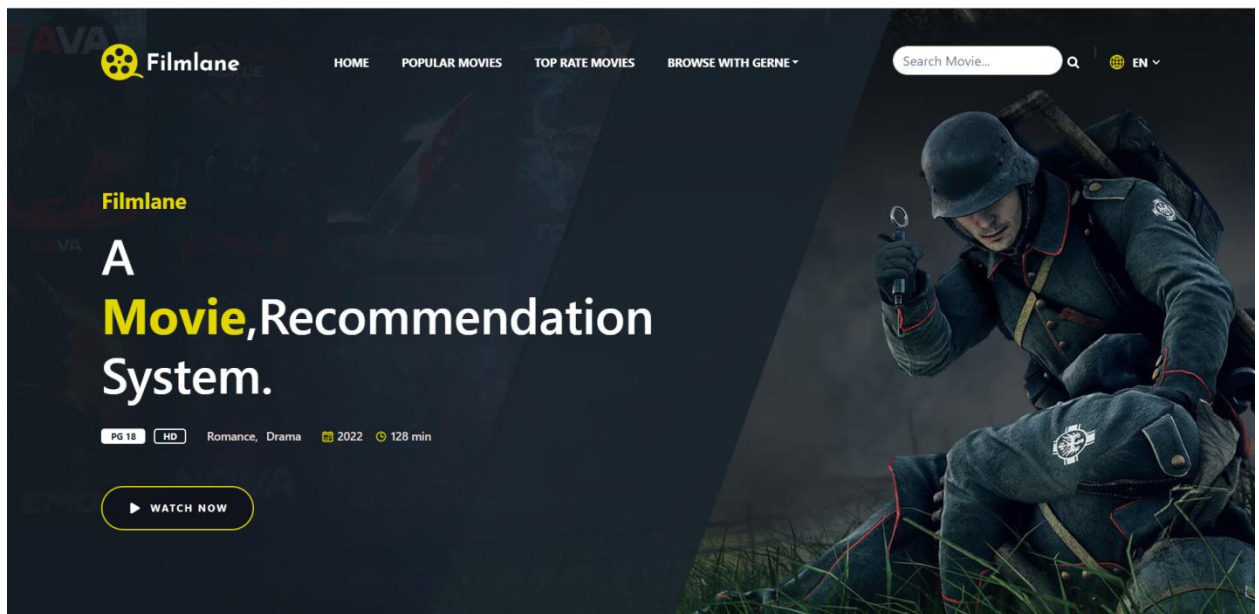


Figure 11: Home Page

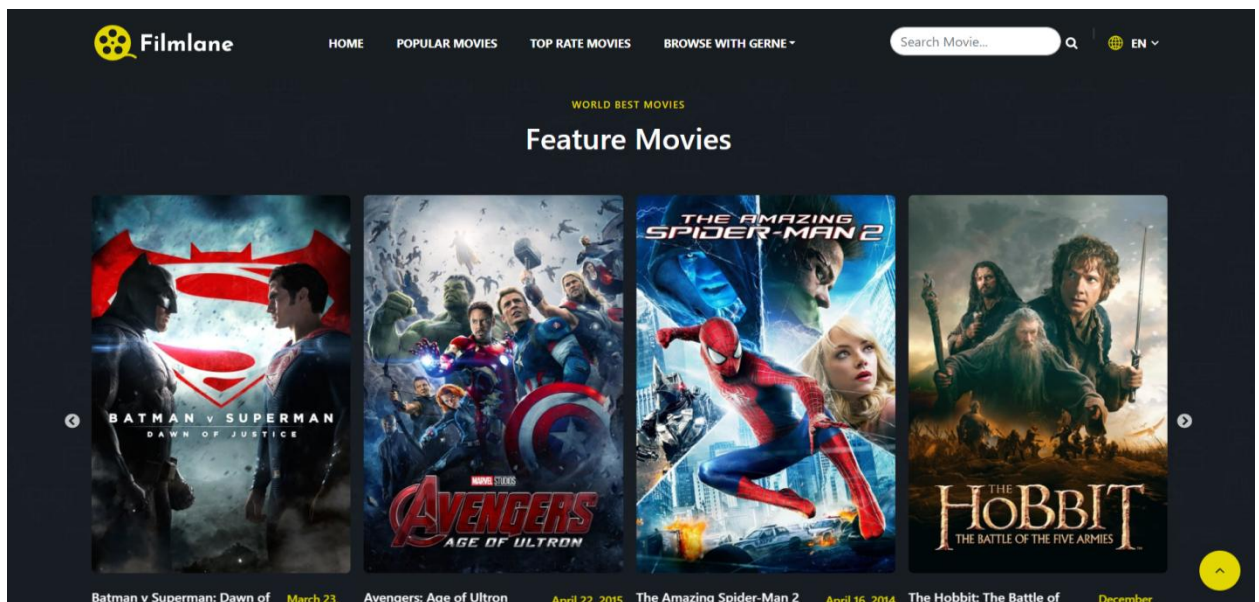


Figure 12: Top Rated Movies

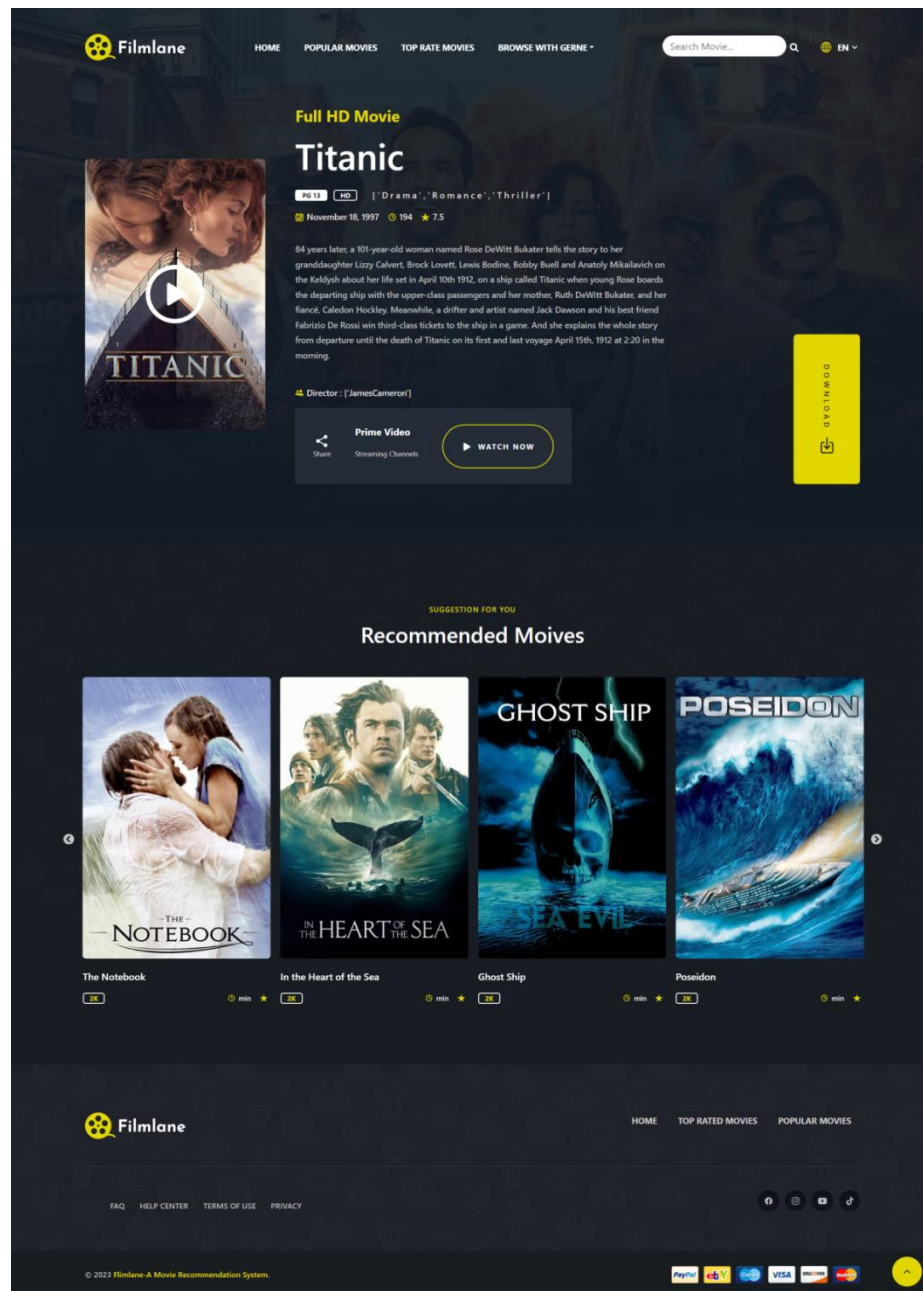


Figure 13: Movie Details with Recommended Movies

CREATE ACCOUNT

Username

*

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

First name

Last name


Email address

Password

*

REGISTER

Figure 14: Registration Page

 Filmlane

HOMEPOPULAR MOVIESTOP RATE MOVIESBROWSE WITH GENRE

Search Movie... Login

LOG IN

Username

*

Password

*

LOGIN

Not registered?Create Account

Figure 15: LogIn Page

```

#Recommended Movies
movies_dict = pickle.load(open('movie_list.pkl','rb'))
movies = pd.DataFrame(movies_dict)
similarity = pickle.load(open('similarity.pkl','rb'))

def recommendation(title):
    movie = Movie.objects.get(title=title)
    # context = {'movie':movie}
    index=movies[movies['title']== title].index[0]
    similarity_score=sorted(list(enumerate(similarity[index])),reverse=True,key=lambda x : x[1])

    res = []
    resid = []
    recommended_movie_posters = []
    for i in similarity_score[1:6]:
        movie_id = movies.iloc[i[0]].movie_id
        recommended_movie_posters.append(fetch_poster(movie_id))
        res.append(movies.iloc[i[0]].title)
        resid.append(movie_id)
        # print(movie_id)
        # res.append(movies.iloc[i[0]].movie_id)

    result = zip(res,recommended_movie_posters,resid)
    # data = {
    #     # 'res':res,
    #     'result':result,
    #     # 'recommended_movie_posters':recommended_movie_posters,
    #     'movie':movie,
    # }
    return result

```

Figure 16: Main Code for Recommendation