

MOVIE RECOMMENDATION SYSTEM USING ITEM BASED COLLABORATIVE FILTERING

**Tribhuvan University
Institute of Science and Technology**



A FINAL YEAR PROJECT REPORT SUBMISSION In
Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science in Computer Science and Information Technology

**UNDER THE SUPERVISION OF
SATYA BAHADUR MAHARJAN**

COORDINATOR

SUBMITTED BY

ANUJ POKHAREL (T.U. Exam Roll No. 11082/073)

DHIRAJ KAFLE (T.U. Exam Roll No. 11087/073)

DIPENDRA LAL SHRESTHA (T.U. Exam Roll No. 11090/073)

RESUKA MANANDHAR (T.U. Exam Roll No. 11103/073)

**SUBMITTED TO
TRINITY INTERNATIONAL COLLEGE
DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION
TECHNOLOGY
DILLIBAZAR, KATHMANDU, NEPAL**

July 2021

MOVIE RECOMMENDATION SYSTEM USING ITEM BASED COLLABORATIVE FILTERING

**Tribhuvan University
Institute of Science and Technology**



A Final Year Report Submission In
Partial Fulfillment of the Requirements for the degree of
Bachelor of Science in Computer Science and Information Technology

Submitted by

Anuj Pokharel (T.U. Exam Roll No. 11082/073)

Dhiraj Kafle (T.U. Exam Roll No. 11087/073)

Dipendra Lal Shrestha (T.U. Exam Roll No. 11090/073)

Resuka Manandhar (T.U. Exam Roll No. 11103/073)

July 2021

DECLARATION

Project entitled “**Movie Recommendation System Using Item Based Collaborative Filtering**” which is being submitted to the Department of Computer Science and Information Technology, Dillibazar, Kathmandu, Nepal for the fulfillment of the Seventh semester under the supervision of **Mr. Satya Bahadur Maharjan**.

This project is original and has not been submitted earlier in part or full in this or any other form to any university or institute, here or elsewhere, for the award of any degree.

By

Anuj Pokharel (11082/073)

Dhiraj Kafle (11087/073)

Dipendra Lal Shrestha (11090/073)

Resuka Manandhar (11103/073)

RECOMMENDATION

This is to recommend that **Anuj Pokharel, Dhiraj Kafle, Dipendra Lal Shrestha** and **Resuka Manandhar** have carried out research entitled "**Movie Recommendation System Using Item Based Collaborative Filtering**" for the fulfillment of seventh semester in **Final Year Project** under my supervision. To our knowledge, this work has not been submitted for any other degree.

He/She has fulfilled all the requirements laid down by the Trinity International College Department of Computer Science and Information Technology, Dillibazar, Kathmandu.

Mr. Satya Bahadur Maharjan

Supervisor

Program Coordinator

Department of Computer Science and Information Technology,

Trinity International College,

Dillibazar, Kathmandu, Nepal

July 2021



Dillibazar Height, PO Box: 26111, Kathmandu, Nepal

Tel: +977 1 4445955/4445956, Fax: 4437867

Email: info@trinitycollege.edu.np

LETTER OF APPROVAL

Date: 27/07/2021

On the recommendation of Mr. Satya Bahadur Maharjan, this Project Report submitted by Anuj Pokharel, Dhiraj Kafle, Dipendra Lal Shrestha and Resuka Manandhar entitled “**Movie Recommendation System Using Item Based Collaborative Filtering**” in partial fulfillment of the requirement for the award of bachelor’s degree in Computer Science and Information Technology is a bonafide record of work carried out under my/our guidance and supervision at Trinity International College, Dillibazar, Kathmandu.

EVALUATION COMMITTEE

Mr. Satya Bahadur Maharjan
Head of Department
/ Program Coordinator
Trinity International College
Dillibazar, Kathmandu, Nepal

Mr. Satya Bahadur Maharjan
Project Supervisor
/ Program Coordinator
Trinity International College
Dillibazar, Kathmandu, Nepal

External Examiner

Date: _____

ACKNOWLEDGEMENTS

We would like to thank **Mr. Satya Bahadur Maharjan** (Program Coordinator for CSIT, Trinity International College) for his inputs during the course of this project. He also kept us up to date with relevant notices and deadlines, which helped us stay on track to complete this project.

A warm thank you to the Trinity International College Bachelor's Administrative, **Mr. Abhishek Dewan** and **Mr. Shoyam Bhattarai** for logistical support during the project. They provided us with the required formats and documents that supplemented this final project.

We would also like to thank **Mr. Jitesh Tuladhar** for his invaluable help for the technical aspects of this project. Due to his proper maintenance of the labs and network, we were able to get valuable technical and logistical support for the project.

ABSTRACT

The project Movie Recommendation System Using Item Based Collaborative Filtering is web application. A recommendation system is a system used to present personal experience to the user by presenting information filtered to them, specifically for each user through several channels of communication. The main propose is to develop a movie recommendation system using collaborative filtering. Collaborative filtering is the most successful algorithm in the recommender system's field.

This application implements item-based collaborative filtering for recommendation system. The rating patterns are studied for recommending the movie with similar movies, that the user may like. For this purpose, Pearson correlation coefficient algorithm is implemented in recommendation system.

Keywords: *Item-based Collaborative Filtering, Recommendation System, Pearson correlation algorithm*

TABLE OF CONTENTS

DECLARATION	iii
RECOMMENDATION	iv
LETTER OF APPROVAL	v
ACKNOWLEDGEMENTS.....	vi
ABSTRACT.....	vii
TABLE OF CONTENTS.....	viii
CHAPTER 1	1
INTRODUCTION	1
1.1 Introduction	1
1.2 Problem Statement	2
1.3 Objectives	2
1.4 Limitations.....	3
CHAPTER 2	4
RESEARCH METHODOLOGY	4
2.1 Literature Review	4
2.2 Methodology	5
2.2.1 Phases of Recommendation Process.....	5
2.2.2 Collaborative Filtering Recommendation System.....	7
2.2.3 Collaborative Filtering Process	8
2.3 System Flowchart	10
CHAPTER 3	12
SYSTEM DEVELOPMENT	12
3.1 System Development and Design.....	12
3.2 System Requirement.....	12
3.2.1 Functional Requirement	12
3.2.2 Non-Functional Requirement.....	13
3.3 Feasibility Study	14

3.3.1 Technical Feasibility.....	14
3.3.2 Economic Feasibility	14
3.3.3 Operation Feasibility	14
3.3.4 Schedule Feasibility.....	14
3.4 System Design.....	16
3.4.1 Sequence Diagram.....	16
3.4.2 Activity Diagram	18
3.3.3 Class Diagram.....	19
3.4.4 Database design	20
3.5 System Implementation.....	20
3.5.1 Implementation.....	20
CHAPTER 4	21
Data Collection	21
CHAPTER 5	24
TESTING.....	24
5.1 Unit Testing.....	24
5.2 Integration Testing	26
5.3 Compatibility Testing.....	27
CHAPTER 6	28
RESULT AND DISCUSSION	28
6.1 Result Analysis	28
CHAPTER 7	31
CONCLUSION.....	31
7.1 Conclusion.....	31
7.2 Future Enhancements	31
REFERENCE.....	32
APPENDIX.....	33
Appendix I.....	33
Appendix II.....	37

LIST OF ABBREVIATIONS

CF	Collaborative Filtering
CSS	Cascade Style Sheet
HTML	Hyper Text Markup Language
IBCF	Item Based Collaborative Filtering
JS	JavaScript
MVT	Model View Template
ORM	Object Relational Mapping
UBCF	User Based Collaborative Filtering
IMDb	Internet Movie Database

LIST OF SYMBOLS

Σ	Summation
$\sqrt{}$	Square root

LIST OF TABLES

Table 1: Item-Item Rating Matrix	9
Table 2: Movies.csv	21
Table 3: Ratings.csv	22
Table 4: Web scraped data	22
Table 5: Sign Up	24
Table 6: Add Movie	24
Table 7: Delete User	25
Table 8: Rate Movie.....	25
Table 9: Search Movie	25
Table 10: Filter Movies.....	25
Table 11: General Recommendation	26
Table 12: Recommendation	26
Table 13: Integration Testing.....	27
Table 14: Recommended Movies List	30

LIST OF FIGURES

Figure 1: Recommendation Phases	5
Figure 2: Classification of recommendation techniques.....	6
Figure 3: User-based Collaborative Filtering	7
Figure 4: Item-based Collaborative Filtering.....	8
Figure 5: Collaborative Filtering Process	9
Figure 6: Flowchart of Movie Recommendation System	11
Figure 7: System Architecture of Movie Recommendation System	12
Figure 8: Use Case Diagram	13
Figure 9: Gantt Chart	15
Figure 10: Admin Sequence Diagram.....	16
Figure 11: Client Sequence Diagram.....	17
Figure 12: User Interaction with Recommendation System Activity Diagram.....	18
Figure 13: Class Diagram	19
Figure 14: Database Design.....	20
Figure 15: Item Similarity Heatmap	28
Figure 16: Correlation Coefficient Line Graph	29

CHAPTER 1

INTRODUCTION

1.1 Introduction

Recommendation system are information filtering system that aids user in predicting movies or preference of an item under user's consideration. Recommendation system can be implemented using item-based collaborative filtering and user based collaborative filtering. Recommender system usually uses collaborative filtering algorithms or a combination of the collaborative filtering algorithms and even other filtering algorithm. A recommendation system is a system used to present personal experienced to the user by presenting information filtered to them, specifically for each user through several channels of communication [1].

Collaborative filtering is the most successful algorithm in the recommender system's field. Generally, recommendation contains four parts: Database, Human-computer, Interface and Algorithm. Collaborative filtering is a technique that predicts and recommends the items based on user preferences in the system. Collaborative filtering is dependent on correlation rate to find users feature that are nearest to all user. It is very important to produce accurate result in order to predict the preference of the user. Collaborative filtering will calculate the correlation rate between items or users within the cluster. Collaborative filtering is the most successful and popular technique in the recommender systems field. It helps customer to make a better decision by recommending interesting item [1].

A Recommender System is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item. These are the information filtering systems that deal with the problem of information overload by filtering vital information fragment out of large amount of dynamically generated information per user's preferences, interest, or observed behavior about item. Recommender systems typically produce a list of recommendations in one of the two ways - through collaborative filtering or through content-based filtering. These approaches are often combined as hybrid recommendation systems [7].

The similarity of the item is computed based on the features associated with the items being compared. Movie Recommendation System is a web application designed to provide users with the information and recommendations about movies based upon user's rating. In the project user can give rating from 1 to 5 stars. The highest rating is 5 and the lowest is 1. The ratings less than or equals to 2 is considered to be negative rating and the ratings greater than or equals to 3 is considered to be positive. Based upon these negative and positive ratings movies are recommended to the users.

1.2 Problem Statement

As the advancement in the internet technology began couple of decades back, millions of the movies are found in movie sites. Movie sites has become one of the most visited sites these days. Back in the day's lots of movies with different genres are widely available and the qualitative movies was always questionable since the other user who has already seen the movies do not have the appropriate means of recommending the movies. Then rating and the review system came along that change the way users and the customers go through movie over the internet.

Recommendation system gives relief on that sector of the movies sites for the users. So, the main motive is to make assure that the system takes the ratings of the movies that has been given by the user and has access to give the recommendation of the similar movies that they might like and interested in looking to their past choices of the movies they have made. The ratings that the user has given for the movie are taken and using the collaborative filtering technique the appropriate recommendation is made.

1.3 Objectives

The objective of the project is to take the movie rating from the users and analyze them using collaborative filtering method to recommend the appropriate movie for the users from the large community of users. The objective of the project is given as follows:

- To create movie rating system where user can rate a movie.
- To develop recommendation system that suggest user specific interesting movies based on the rating given by the user.

1.4 Limitations

The system provides movie recommendation to the user as per their reviews or ratings on movies. The system is not all fully fledged so there are limitations.

The limitations of this system are:

- It does not perform sentiment analysis which would give more accurate results.
- It doesn't provide user to user-based recommendation.
- With the addition of new movie, there occurs cold start problem for recommendation to those newly added latest movies.

CHAPTER 2

RESEARCH METHODOLOGY

2.1 Literature Review

In the paper entitle ‘Movie’s recommendation system using collaborative filtering and k-means’ authors have described about the rapid development of technology, increment in dissemination of knowledge, also leads to the complexity and find difficulty in offering products and services as per the need. These complex information causes overload problems which in turn are time consuming. To increase the accuracy, the recommendation system is developed using k-means clustering followed by collaborative filtering technique. The system performs k-means clustering using euclidean distance formula to divide the users in the groups before entering into the collaborative filtering process. Collaborative filtering algorithm uses Pearson correlation coefficient to find users who have similar tastes and suggest items. Authors have evaluated the traditional collaborative filtering and their approach to compare them. Authors have chosen data with 500 users, 80 movies records and rating. Their results show the proposed algorithm is more accurate than the traditional existing one [1].

In the paper entitles ‘Movie Recommendation System Using Collaborative Filtering’ authors have described that the large amount of increase in information available over the internet has created a greatest challenge in searching useful information. As a result, an intelligent approach such as recommendation system is used that can recommend things like movies, books, music, restaurant, news and jokes from web. The paper aims to describe the implementation of a movie recommender system via two collaborative filtering algorithms using Apache Mahout. The movies dataset used in their project is obtained from the Yahoo Research Webscope database. The Yahoo! Movies Users Ratings file contains 2,11,231 records and consist of User ID, Movie ID and Ratings. Movie file contains 54,058 records and consists Movie ID, Title, Genre, Directors, Actors. Furthermore, this paper also focusses on analyzing the data to gain insights into the movie dataset using Matplotlib libraries in Python [2].

The paper that entitles ‘A Survey on Recommendation System with Collaborative Filtering using Big Data’ aims to analyze the existing systems i.e., collaborative filtering approach and found that they are not scalable enough. Collaborative filtering is primary approach of any recommendation system. Only collaborative filtering approach cannot provide enough scalability and accuracy. In their work author have presented a model that combines recommendation system method such as collaborative filtering with association rule mining. The main focus of their paper is to provide a scalable and robust recommendation system that can provide good accuracy. Authors have concluded that using association rule mining and collaborative filtering can ensure good scalability and strength of the recommendation system [3].

2.2 Methodology

Recommendation systems are information filtering system that aids users in predicting rating or preference of an item under user’s consideration.

2.2.1 Phases of Recommendation Process

The recommendation process generally consists of the following phases, as shown in Figure 1[7].

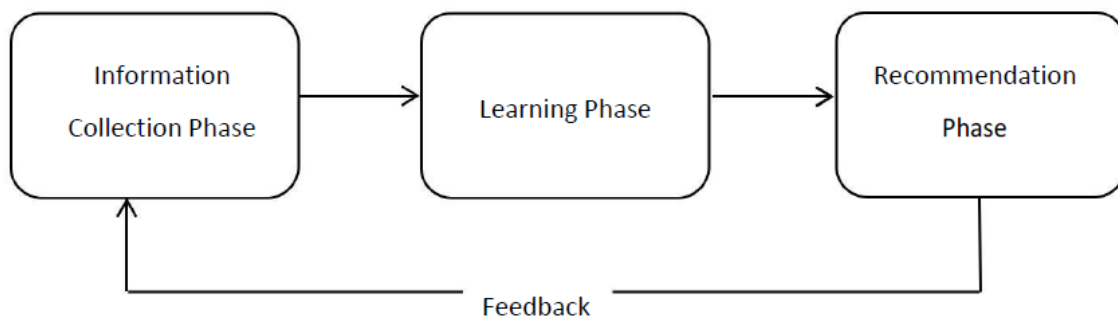


Figure 1: Recommendation Phases

I. Information collection phase

This is the process of collecting relevant information of users to generate a user profile or model for the prediction tasks including user’s attribute, behaviors or content of the resources the user accesses. A recommendation agent cannot function accurately until the user profile or model has been well constructed. The system needs to know as much as possible from the

user in order to provide reasonable recommendation right from the onset. Recommender systems rely on different types of input such as the most convenient high quality explicit feedback, which includes explicit input by users regarding their interest in item or implicit feedback by inferring user preferences indirectly through observing user behavior. Hybrid feedback can also be obtained through the combination of both explicit and implicit feedback.

II. Learning phase

It applies a learning algorithm to filter and exploit the user's features from the feedback gathered in information collection phase. Algorithms is discussed in more detail in following sections.

III. Recommendation phase

It recommends or predicts what kind of items the user may prefer. This can be made either directly based on the dataset collected in information collection phase which could be memory based or model based or through the system's observed activities of the user.

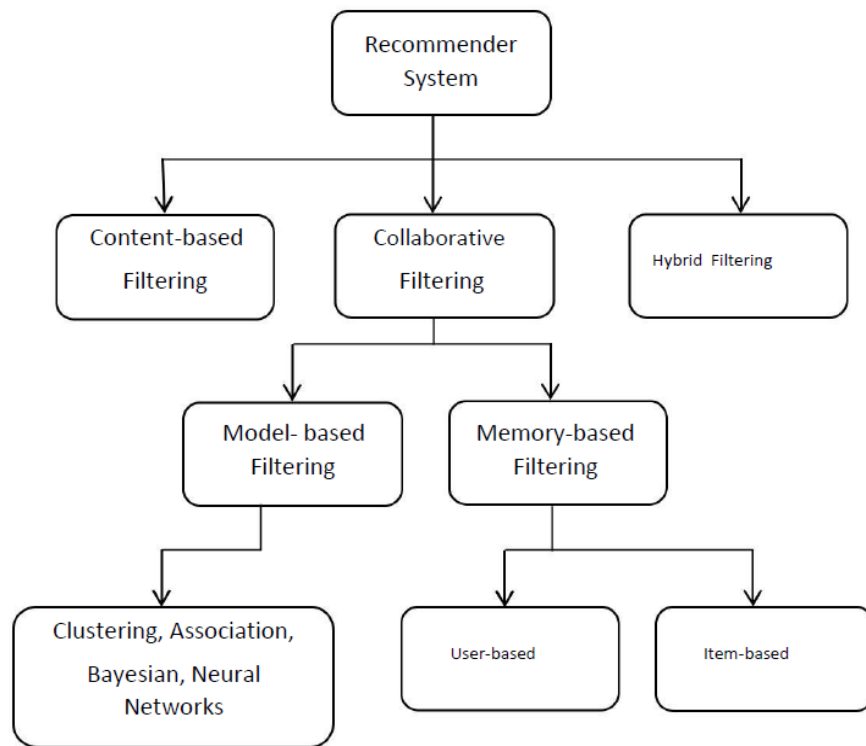


Figure 2: Classification of recommendation techniques

2.2.2 Collaborative Filtering Recommendation System

Collaborative filtering has been one of the most successful and well-studied recommendation algorithms, which only relies on the user-item interaction data to make recommendations [4].

Collaborative filtering can be categorized into two main methods. They are:

- i. User-based Collaborative Filtering
- ii. Item-based Collaborative Filtering

i. User-based Collaborative Filtering

User-based collaborative filtering approach is to predict items to the target user that are already items of interest for other users who are similar to the target user. For example, as seen [Figure 3] [5].

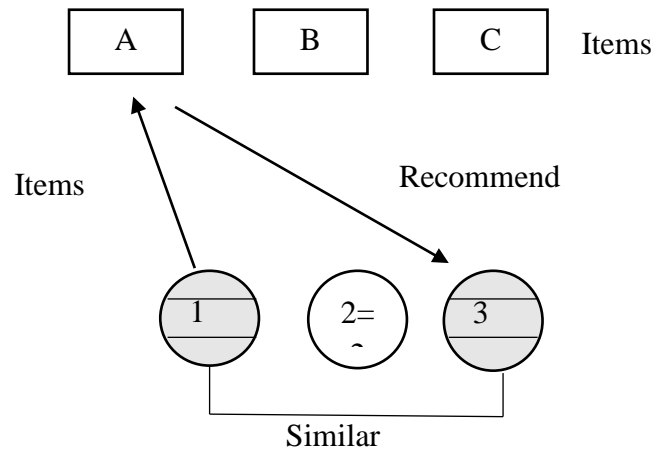


Figure 3: User-based Collaborative Filtering

Let user 1 and user 3 have very similar preference behavior. If user 1 likes Item A, UBCF can recommend Item A to user 3. UBCF needs the explicit rating scores of items rated by users to calculate similarities between users and exploits k-nearest neighbor algorithms to find the nearest neighbors based on user similarities. And then, it generates prediction in terms of items by combining the neighbor user's rating scores based on similarity weighted averaging.

ii. Item-based Collaborative Filtering

Item-based collaborative filtering approach is to predict items by inquiring into similarities between the items and other items that are already associated with the user. For example, as shown in [Figure 4] [5].

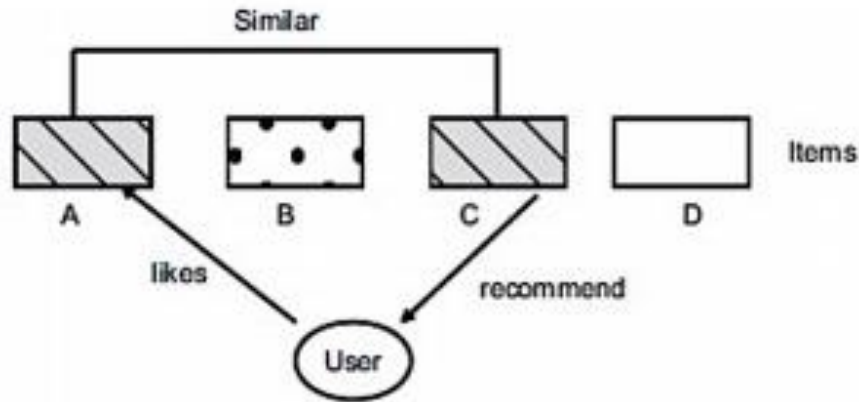


Figure 4: Item-based Collaborative Filtering

Let's say Item A and Item C are very similar. If a user likes Item A, IBCF can recommend Item C to the user. IBCF needs a set of items that the target user has already rated to calculate similarities between items and a target item. And then, it generates prediction in terms of the target item by combining the target user's previous preferences based on these item similarities. In IBCF, users' preference data can be collected in two ways. One is that user explicitly gives rating score to item within a certain numerical scale. The other is that it implicitly analyzes user's records or click-through rate.

2.2.3 Collaborative Filtering Process

Collaborative filtering (CF) processing can be mainly divided into three steps [6]:

Step 1) Collecting user ratings data matrix,

Step 2) Selecting similar neighbors by measuring the rating similarity, and then

Step 3) Generating rating prediction for items based on rating of all similar neighbors for specific item.

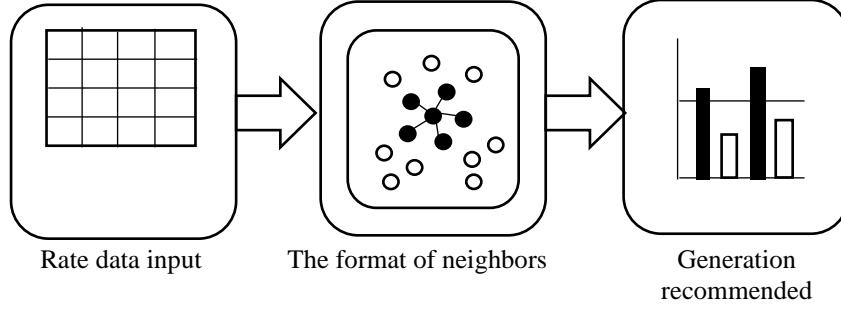


Figure 5: Collaborative Filtering Process

Item-based filtering method is implemented in the project and Pearson's correlation coefficient formula is used to compare the items and calculate correlation value among them. These correlation values show the relationship between the items.

2.2.3.1. User Rating Score Data Inputs

Generally, input data in recommendation system based on the CF technology consists of items, and user opinions on observed items as a matrix $m \times n$ as shown in [Table 1]. Symbol m symbolizes the total number of items I_m and n symbolizes the total number of items I_n . $R_{m,n}$ is the score of item I_n rated by user. In this project items are the movies.

Table 1: Item-Item Rating Matrix

Items Items	I_1	I_2	\dots	I_{n-1}	I_n
I_1	$R_{1,1}$	$R_{1,2}$	\dots	$R_{1,n-1}$	$R_{1,n}$
I_2	$R_{2,1}$	$R_{2,2}$	\dots	$R_{2,n-1}$	$R_{2,n}$
\dots	\dots	\dots	\dots	\dots	\dots
I_{m-1}	$R_{m-1,1}$	$R_{m-1,2}$	\dots	$R_{m-1,n-1}$	$R_{m-1,n}$
I_m	$R_{m,1}$	$R_{m,2}$	\dots	$R_{m,n-1}$	$R_{m,n}$

2.2.3.2. Algorithm

The Pearson coefficient correlation has a high statistical significance. It looks at the relationship between two variables. It seeks to draw a line through the data of two variables to show their relationship.

The correlation coefficient formula finds out the relation between the variables. It returns the values between -1 and 1. A correlation coefficient that is greater than zero indicates a positive relationship. A value that is less than zero signifies a negative relationship. Finally, a value of zero indicates no relationship between the variables [2].

Pearson correlation coefficient formula

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}} \quad (1)$$

Where:

x_i = x item values

y_i = y item values

\bar{x} = Mean of x values

\bar{y} = Mean of y values

$(x_i - \bar{x})^2$ = Deviation Squared

$(y_i - \bar{y})^2$ = Deviation Squared

2.3 System Flowchart

Flowchart for the movie recommendation system is as shown in Figure 6. It represents the flow of the web application built for the users. This application provides services for two types of users, one is admin and the other is client. They should register first in order to sign in.

Users are allowed to rate the movies and use recommendation services, only if they logged in. Similarly, admin is allowed to add or manage movies and other admins on server database after they sign in. All the information of the movies along with the users rating are stored on the database.

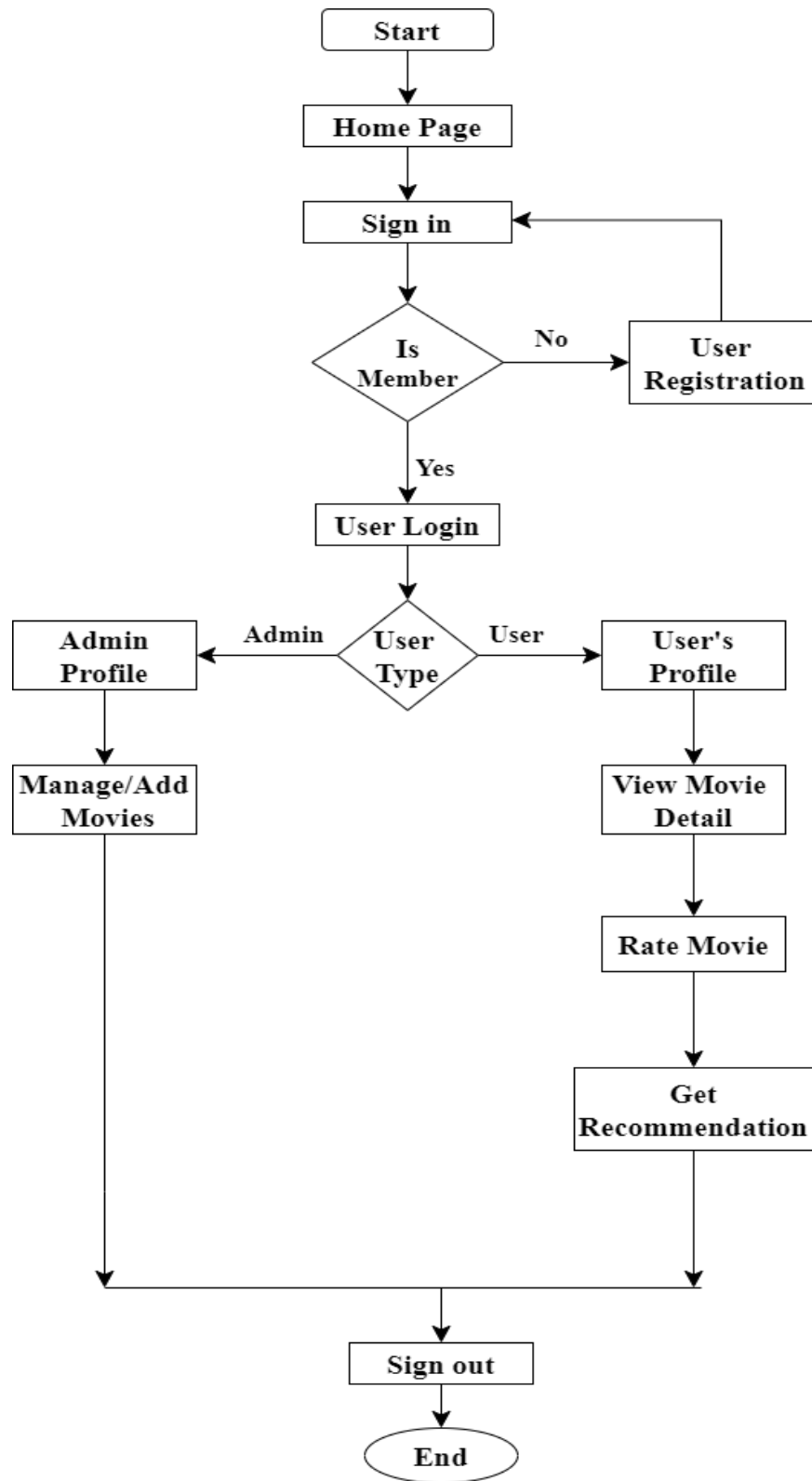


Figure 6: Flowchart of Movie Recommendation System

CHAPTER 3

SYSTEM DEVELOPMENT

3.1 System Development and Design

This system consists of major two parts, web application and recommendation system.

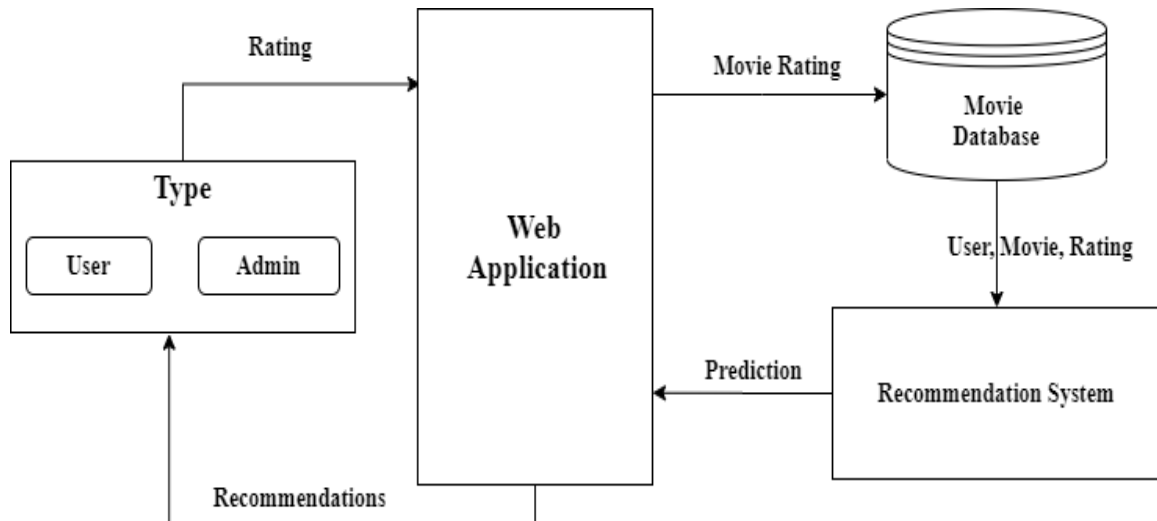


Figure 7: System Architecture of Movie Recommendation System

Web application is for the end users for the reviews and viewing the recommendations whereas the recommendation system is the actual part where the algorithm works.

3.2 System Requirement

3.2.1 Functional Requirement

Functional requirements define the functions of the system and describe what the system must be able to accomplish. The functional requirements of this system are:

- i. Different platform with different accessibility features for admin and user.
- ii. Recommendation should be done based on the user rating and movie genre.

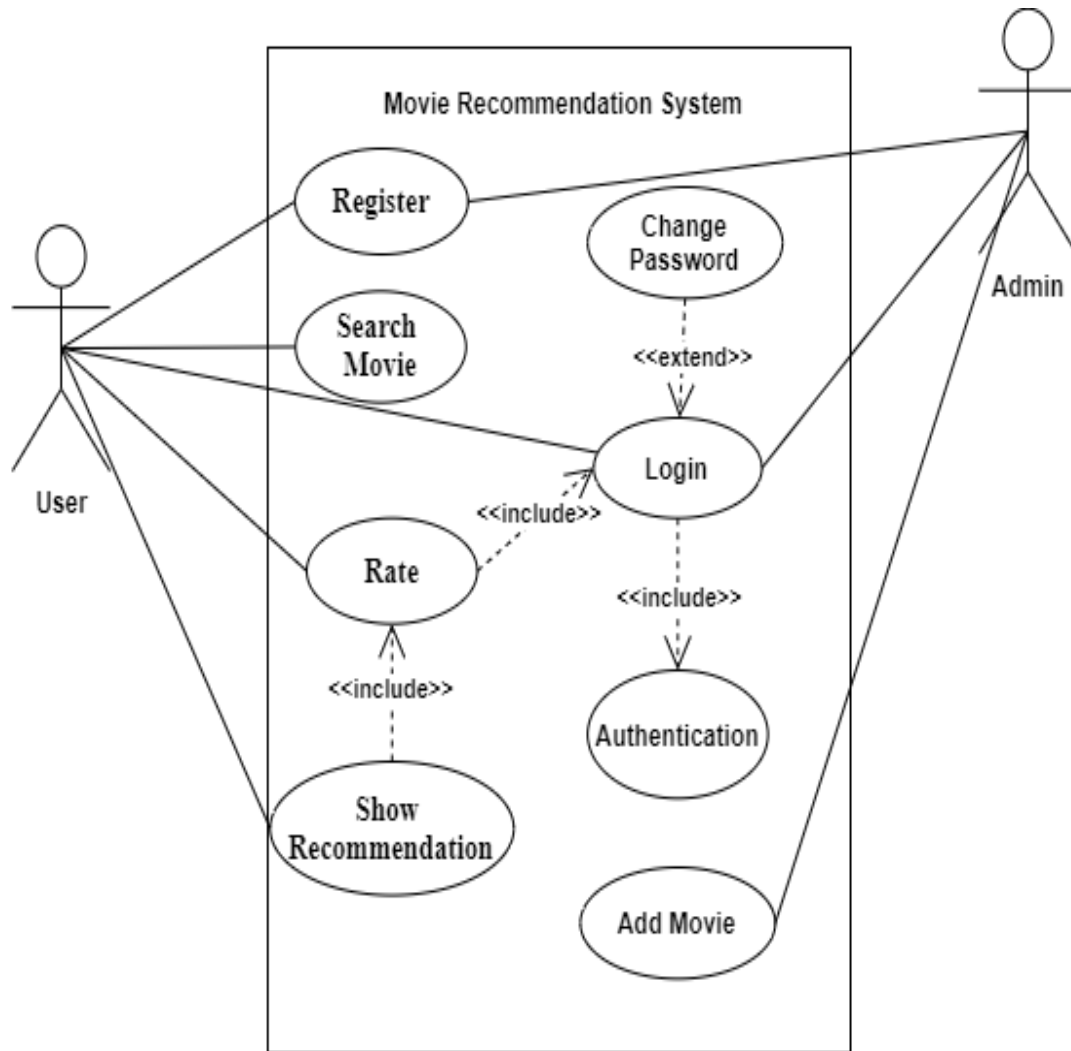


Figure 8: Use Case Diagram

Figure 8 shows the use case diagram of the system. The users initially get home page of the system where they can register and login. Once the user is registered and logged in, they gain access to the system. The user browse/search for movies available in the database. The user rate the movies which aids the system in generating recommendations which are later viewed by the user.

3.2.2 Non-Functional Requirement

The non-functional requirements are constraints upon the system behavior or quality attributes of a system.

- i. System should be developed to be simple and efficient for the end users and should be easy to understand.

- ii. System should be able minimize the rate of errors generated by users.
- iii. System should be compatible to any web browser and different web platforms.
- iv. System should be able to upgrade without disturbance to the service.

3.3 Feasibility Study

The feasibility study of the project is as follows:

3.3.1 Technical Feasibility

The project is an implementation of various web development tools. The application is built in Python platform with Django as framework. The user interface is created using CSS, JavaScript and Bootstrap frameworks. The acquired data are analyzed in Python language. The materials and dataset required for this project are also freely available on the 'Movie Lens' website. All being open-source software so development of this project is technically feasible.

3.3.2 Economic Feasibility

The project will be cheaper in respect to the monetary expenses during the development phase as the required software is open-source software. Along with a laptop and internet services, the project is economically feasible.

3.3.3 Operation Feasibility

The project is aimed to be user friendly and have simple yet attractive interfaces so that the users can operate it at ease. A user with a computer and internet service can easily access the application making it operation feasible.

3.3.4 Schedule Feasibility

The schedule feasibility of this project is described by the Figure 9 Gantt Chart shows the time-frame for each individual process for the completion of the project within the given schedule.

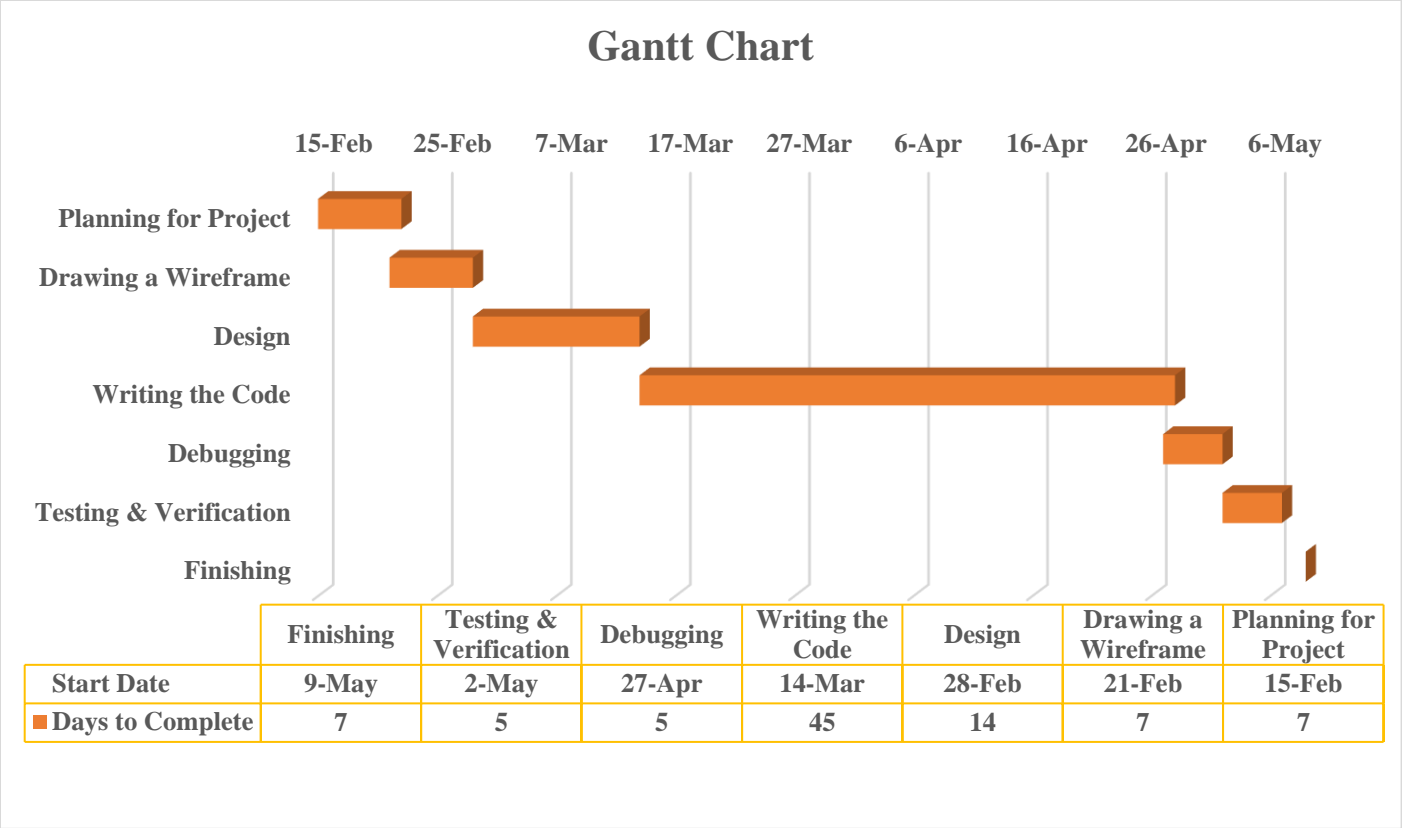


Figure 9: Gannt Chart

3.4 System Design

3.4.1 Sequence Diagram

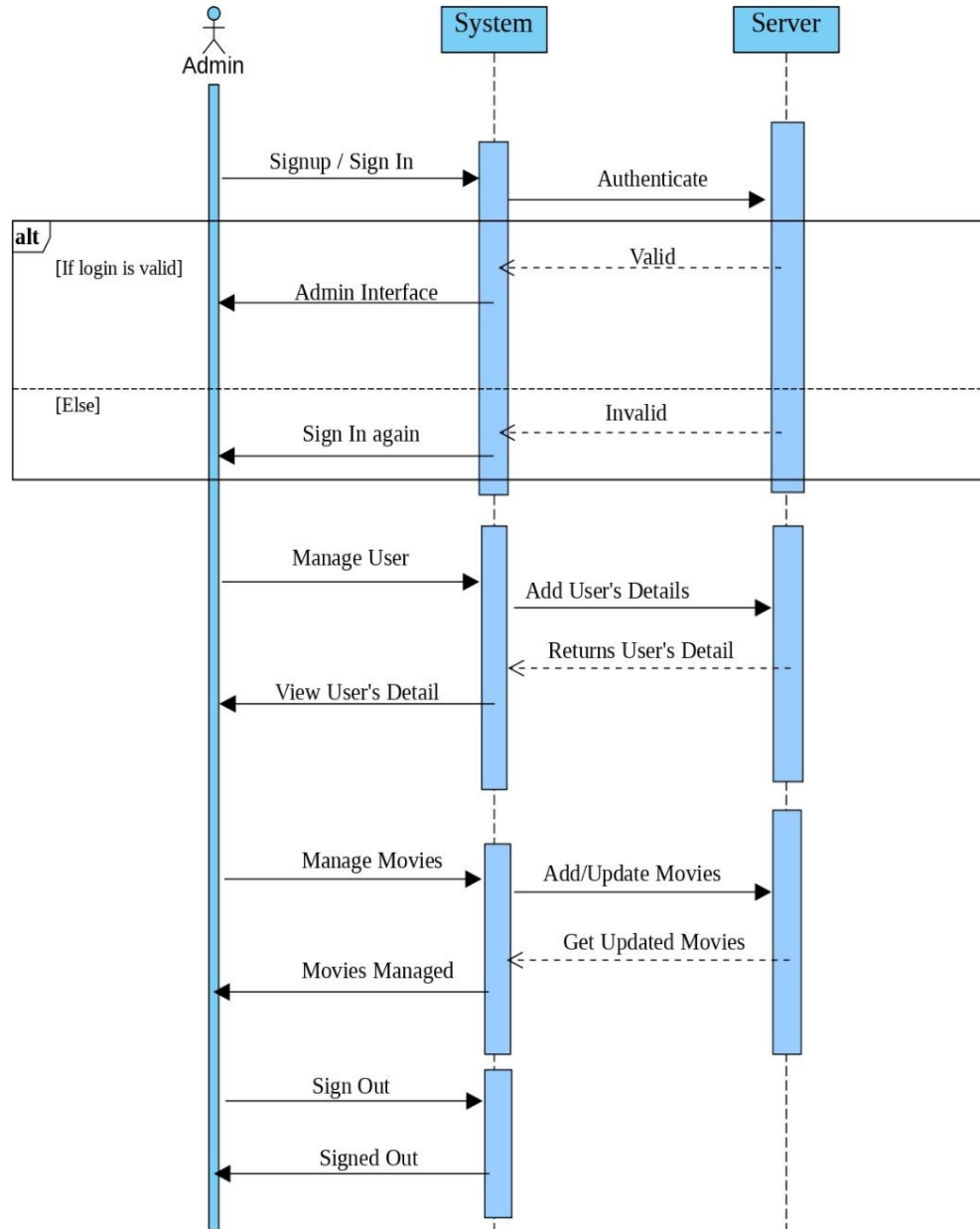


Figure 10: Admin Sequence Diagram

Figure 10 shows the sequence diagram for admin, describing the sequence of activities that he/she can perform while interacting with the web application of the Movie Recommendation System.

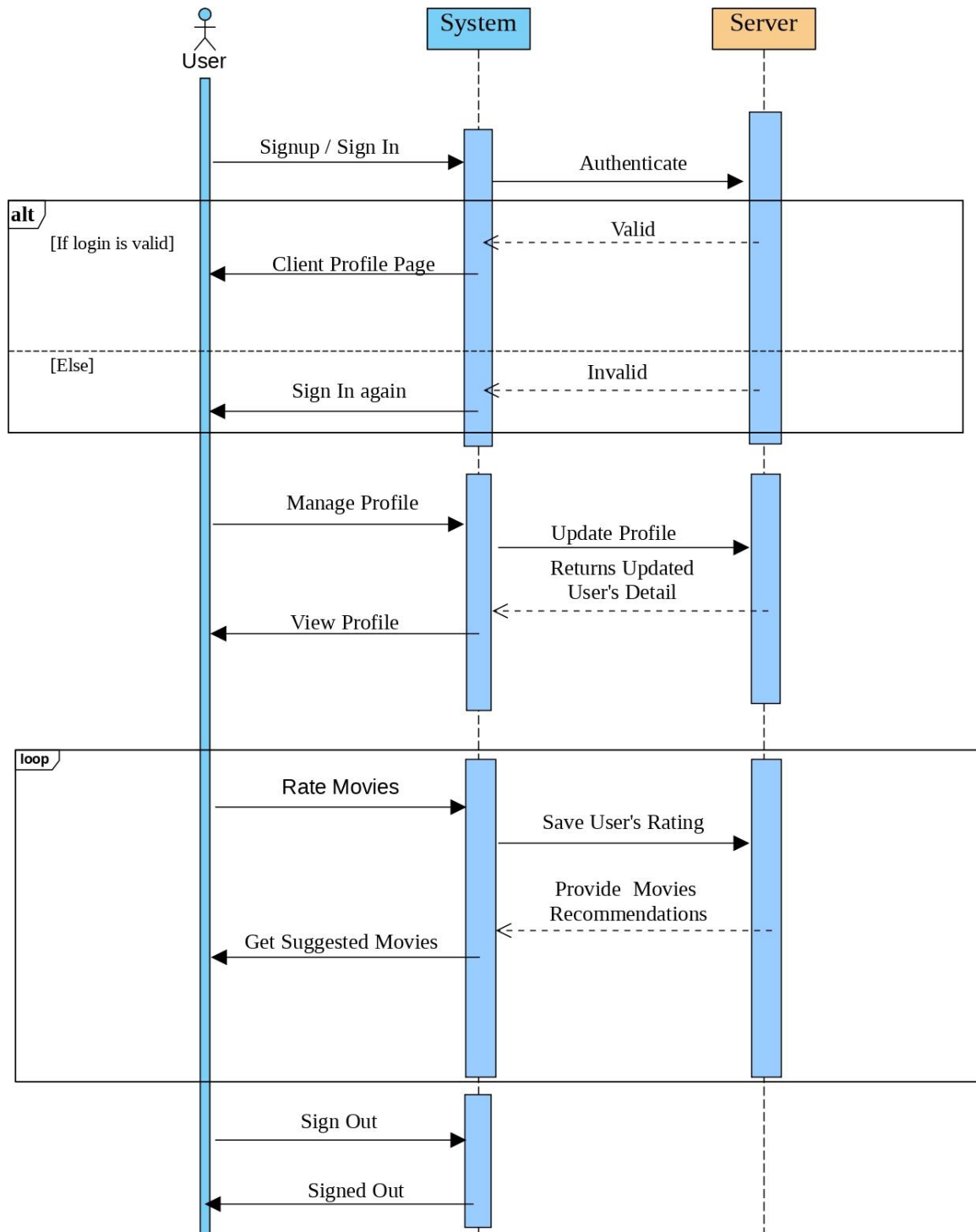


Figure 11: Client Sequence Diagram

Figure 11 shows the client sequence diagram that describes the sequence of activities performed by the client while interacting with the web application of the Movie Recommendation System.

3.4.2 Activity Diagram

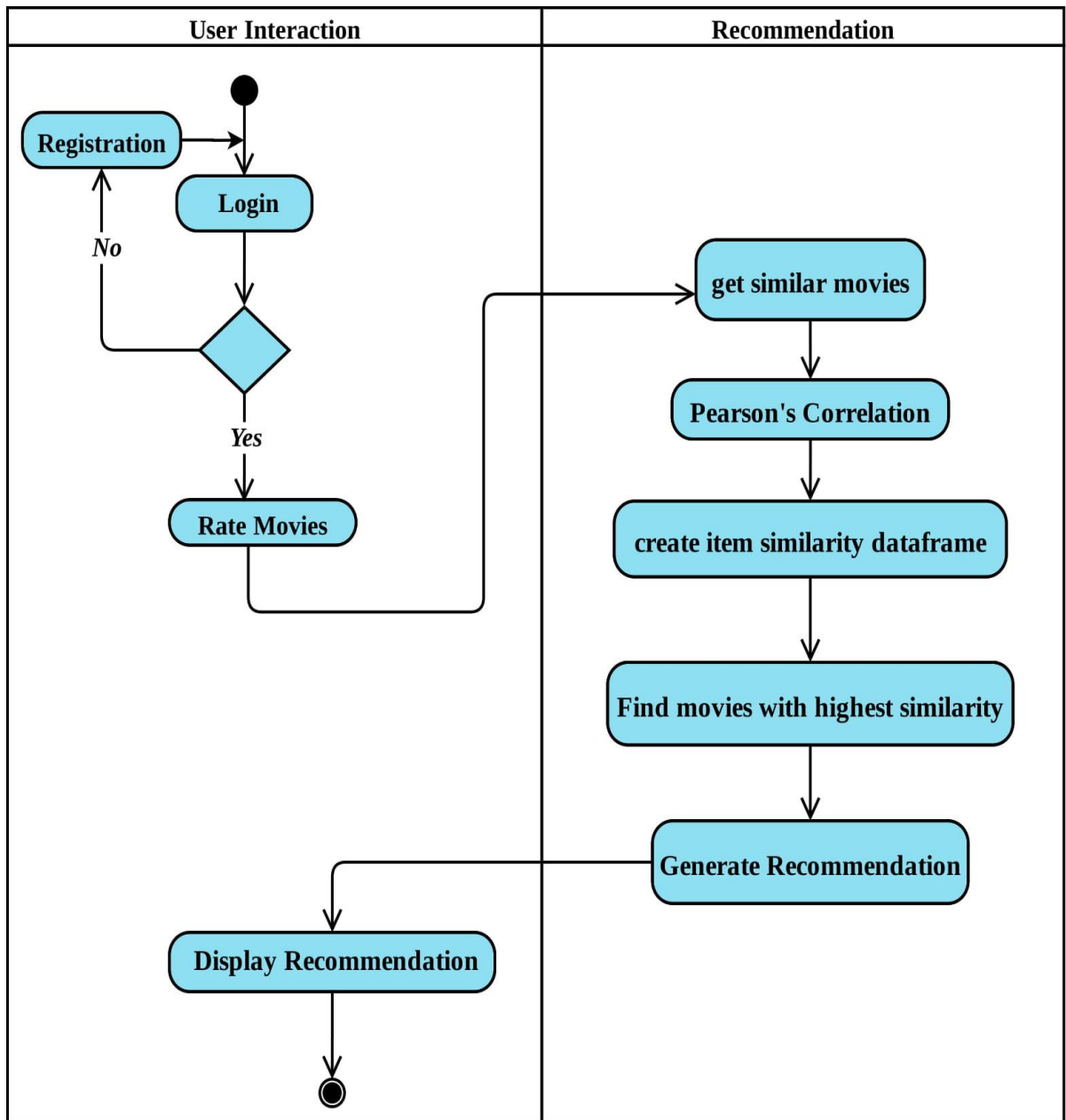


Figure 12: User Interaction with Recommendation System Activity Diagram

An activity diagram illustrates flow of activity between user and recommendation system. It describes how the recommendation system works along with users for receiving the rating input and providing a recommendation based on those users' input. Recommendation system goes through several activity to generate recommendation as shown in right part of figure 12.

3.3.3 Class Diagram

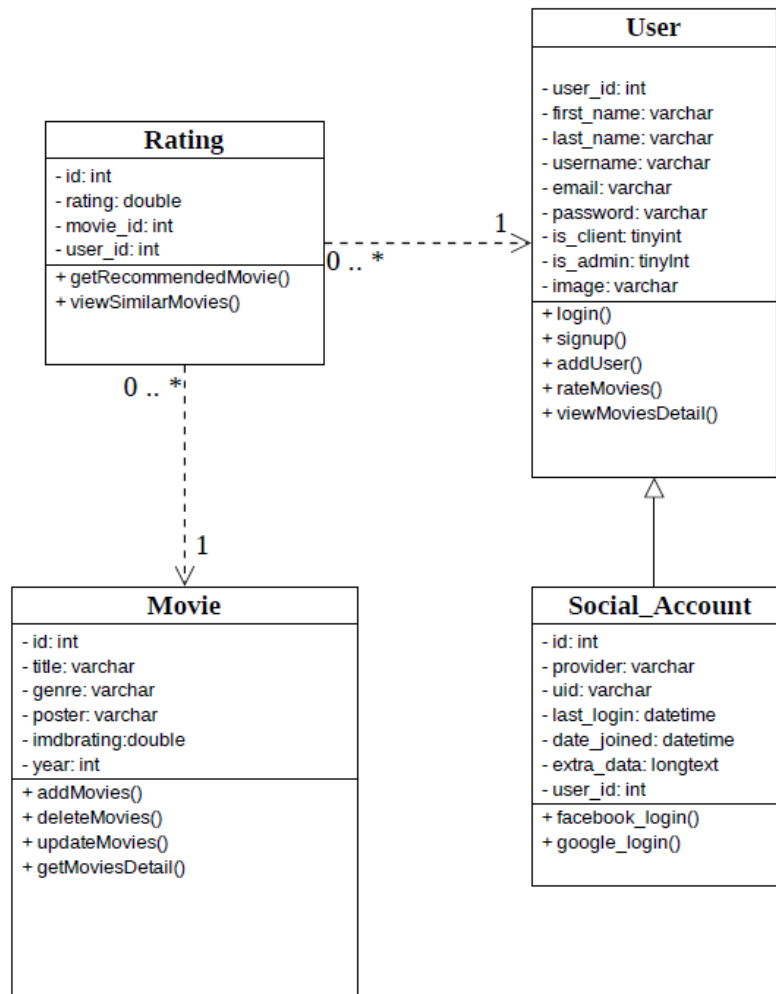


Figure 13: Class Diagram

Figure 13 represents the class diagram for a movie recommendation web application. It shows all the important classes along with their interaction. It shows the attributes and methods available for all the classes.

In figure 13, rating class has dependency relationship with user class and movie class. `0..*` number in dependency relationship between rating and user represents that single user can rate any number of movies whereas in dependency relationship between rating and movie represents that a movie can have either no ratings or may have multiple ratings. Social account class has generalization relationship with user class.

3.4.4 Database design

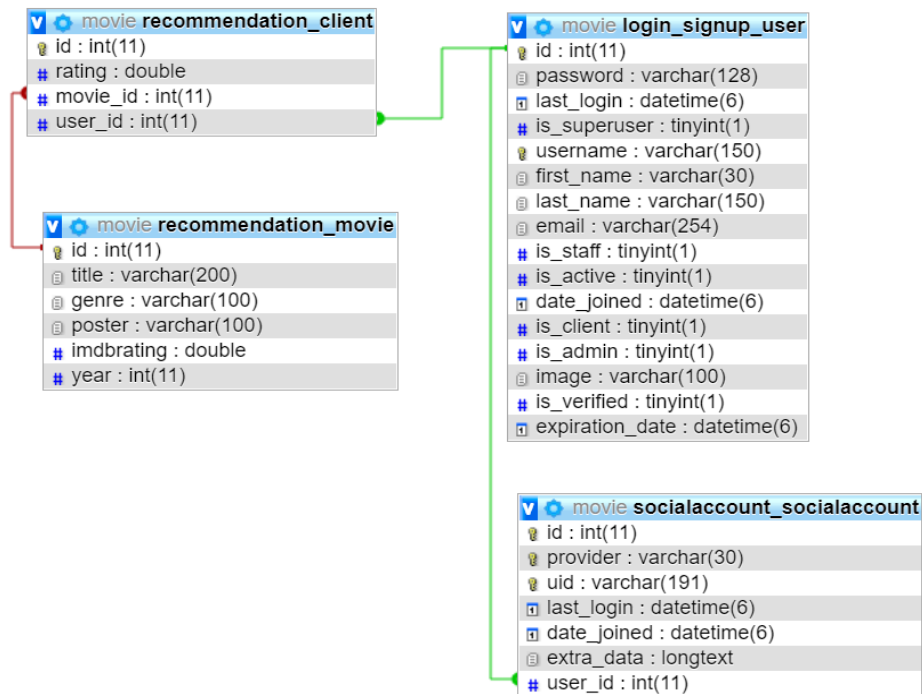


Figure 14: Database Design

Figure 14 represents the database design of a movie recommendation web application. Several entities along with their corresponding attributes and datatypes, as well as the relationship among them are defined as required for the project development. The models for the project are built and implemented in a movie recommendation web application based on this design.

3.5 System Implementation

3.5.1 Implementation

The following software and framework are implemented:

- Python 3.9.4 - Core Language
- Django Framework 2.2 - Python Web framework
- Pandas 1.1.5 - Python library for data manipulation and analysis
- HTML, CSS, Bootstrap and JS - Front end
- MySQL - Database
- Beautiful Soup - Python library used for web scrapping

CHAPTER 4

Data Collection

Primarily, the rating datasets for the movie recommendation project are from ‘**Movie Lens**’, which is officially provided from **grouplens.org** site. These datasets have 100,000 real world user’s ratings to 9,000 movies, which is rated by 600 real world users. Since, these datasets are not clean, so, in the data cleaning phase some movies which doesn’t have enough & precise information has been filtered out. Finally, these filtered datasets were saved in the **ratings.csv** & **movies.csv** format which is further used in making machine learning model. The link for dataset is: <https://grouplens.org/datasets/movielens>. Thus, **movies.csv** & **ratings.csv** obtained after filtering looks like as shown in Table 2 and Table 3:

Table 2: Movies.csv

movieId	Title	genres
1	Toy Story	Adventure Animation Children Comedy
2	Jumanji	Adventure Children Fantasy
3	Grumpier Old Men	Comedy Romance
4	Waiting to Exhale	Comedy Drama Romance
5	Father of the Bride Part II	Comedy
6	Heat	Action Crime Thriller
7	Sabrina	Comedy Romance
8	Tom and Huck	Adventure Children
9	Sudden Death	Action
10	GoldenEye	Action Adventure Thriller
11	The American President	Comedy Drama Romance
12	Dracula Dead and Loving It	Comedy Horror
13	Balto	Adventure Animation Children
14	Nixon	Drama
15	Cutthroat Island	Action Adventure Romance
16	Casino	Crime Drama
17	Sense and Sensibility	Drama Romance
18	Four Rooms	Comedy
19	Ace Ventura When Nature Calls	Comedy
20	Money Train	Action Comedy Crime Drama Thriller

Table 3: Ratings.csv

userId	movieId	rating
1	1	4
1	3	4
1	6	4
1	47	5
1	50	5
1	70	3
1	101	5
1	110	4

Secondarily, the static data (i.e., movie image, title, genre, imdb rating for movie) for displaying purpose in this project is obtained by performing web scraping of the IMDb website by automating with python coding. Thus, data obtained after performing web scraping looks like this:

Table 4: Web scraped data

Title	Imdb rating	Year	Genre	Link
Aliens	8.3	1986	Action Sci-Fi	https://www.imdb.com/title/tt0090605
Batman Begins	8.2	2005	Action	https://www.imdb.com/title/tt0372784
Cloverfield	7.0	2008	Horror Sci-Fi	https://www.imdb.com/title/tt1060277
Deadpool	8.0	2016	Action Comedy	https://www.imdb.com/title/tt1431045
Everest	7.1	2015	Action Adventure	https://www.imdb.com/title/tt2719848
Flight	7.3	2012	Drama Thriller	https://www.imdb.com/title/tt1907668
Ghostbusters	7.8	1984	Action Comedy	https://www.imdb.com/title/tt0087332
Hacksaw Ridge	8.1	2016	Drama History	https://www.imdb.com/title/tt2119532
Inception	8.8	2010	Action Sci-Fi	https://www.imdb.com/title/tt1375666
John Wick	7.4	2014	Action Crime	https://www.imdb.com/title/tt2911666
Kung Fu	7.5	2008	Animation	https://www.imdb.com/title/tt0441773

Panda			Action	
Role Models	6.8	2008	Comedy	https://www.imdb.com/title/tt0430922
Skyfall	7.7	2012	Action Thriller	https://www.imdb.com/title/tt1074638
Titanic	7.8	1997	Drama Romance	https://www.imdb.com/title/tt0120338
X-Men Origins: Wolverine	6.6	2009	Action Adventure Sci-Fi	https://www.imdb.com/title/tt0458525

During the web scraping, images have been obtained and the obtained images looks like:



CHAPTER 5

TESTING

5.1 Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the tester to test that the unit they have implemented is producing expected output against given input. Each of the function modules are thoroughly checked and validated. The purpose of unit testing is to ensure the working of each module properly without any errors. Following steps are taken in performing unit tests:

Table 5: Sign Up

Objective	Sign Up for new user
Input	<ul style="list-style-type: none">• Username• E-Mail Address• Password• Re-Password
Expected Output	Sign Up & able to access the System
Error Info	<ul style="list-style-type: none">• Please fill in this field.• The two password fields didn't match.• A user with that username already exists.• This password is too short. It must contain at least 8 characters.• This password is too common.

Table 6: Add Movie

Objective	Add new movie by admin
Input	<ul style="list-style-type: none">• Title• Genre• Imdb rating• Year• Poster

Expected Output	Movie added successfully!
Error Info	<ul style="list-style-type: none"> • Please fill in this field. • This movie already exists.

Table 7: Delete User

Objective	Able to delete User
Expected Output	User Deleted successfully!

Table 8: Rate Movie

Objective	Rate Movie
Input	Rate any star to the movie.
Expected Output	Your rating has been submitted successfully!

Table 9: Search Movie

Objective	Search Movie
Input	Movie Name
Expected Output	Display particular searched movie.
Error Info	Movie not found

Table 10: Filter Movies

Objective	Filter Movies
Input	<ul style="list-style-type: none"> • Genre • Year • Imdb rating
Expected Output	Display filtered movies
Error Info	Movies not found.

Table 11: General Recommendation

Objective	To provide Similar Movies
Expected Output	Obtain Similar Movies
Error Info	No similar movies found

Table 12: Recommendation

Objective	To provide specific movie recommendation for particular users.
Input	Rate the movie
Expected Output	Obtain personalized recommendation or suggestion.
Error Info	To get recommendation, rate movie first.

5.2 Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Here the unit tested component are integrated together and tested as a whole to see whether the components function as a system with less error as possible.

Table 13: Integration Testing

Objective	Check Integration Testing
Expected Output	<ul style="list-style-type: none">• Sign Up: user signed up successfully• Login: user login successfully.• Add Movies: New Movies added successfully by admin• Delete User: Able to delete fake user by admin• Rate Movie: User movie ratings submitted successfully• Search Movie: Successfully able to search movie• Filter Movies: Filter out movies on basis of various filtering options• General Recommendation: Obtain similar movies with respect to single movie• Recommendation: Providing personalized recommendation as per user's previous movie ratings.
Success Info	Movie Recommendation web application runs successfully.
Error Info	Exception (If any)

5.3 Compatibility Testing

Compatibility testing is done by running the project into different browsers and even in live server, the performance of the project is compatible. The project was tested in live server with domain <https://trinityproject.khullasikshya.com/>.

CHAPTER 6

RESULT AND DISCUSSION

6.1 Result Analysis

The movie recommendation system is set to work perfectly. To get recommendation, user need to rate the movies and in order to give rating for the movies user must sign up & log in. Users can rate movies they have watched as per their preference between one star to five stars, rating lesser or equals to two stars is assumed as negative rating and rating greater or equals to three stars is considered as positive rating in the system.

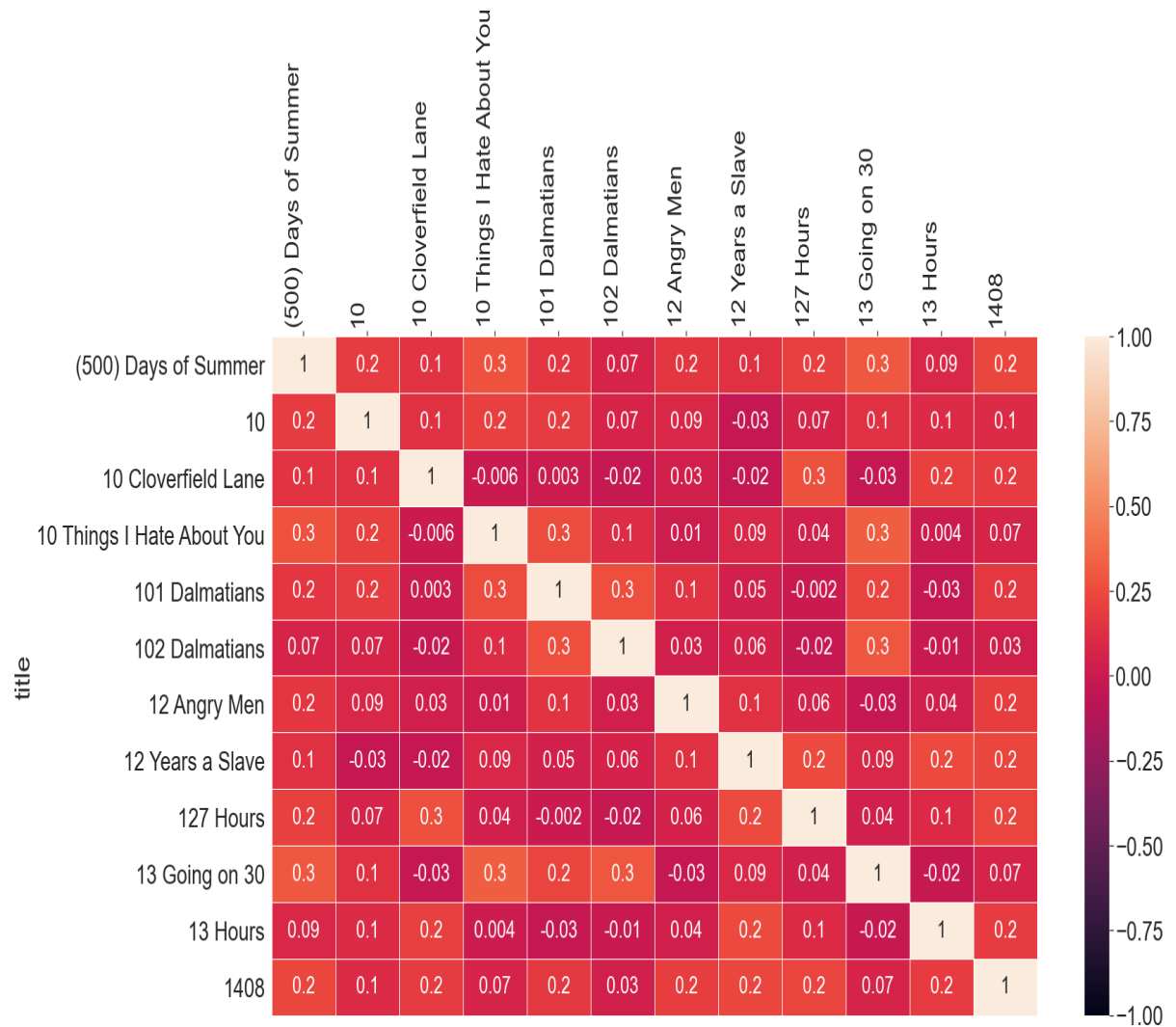


Figure 15: Item Similarity Heatmap

In the project, the Pearson correlation coefficient is evaluated between the movies. Figure 15 shows a part of the item similarity matrix which consists of the correlation coefficient among the items i.e., movies. When users rate the movies, depending upon the correlation coefficient values movies are recommended to the users.

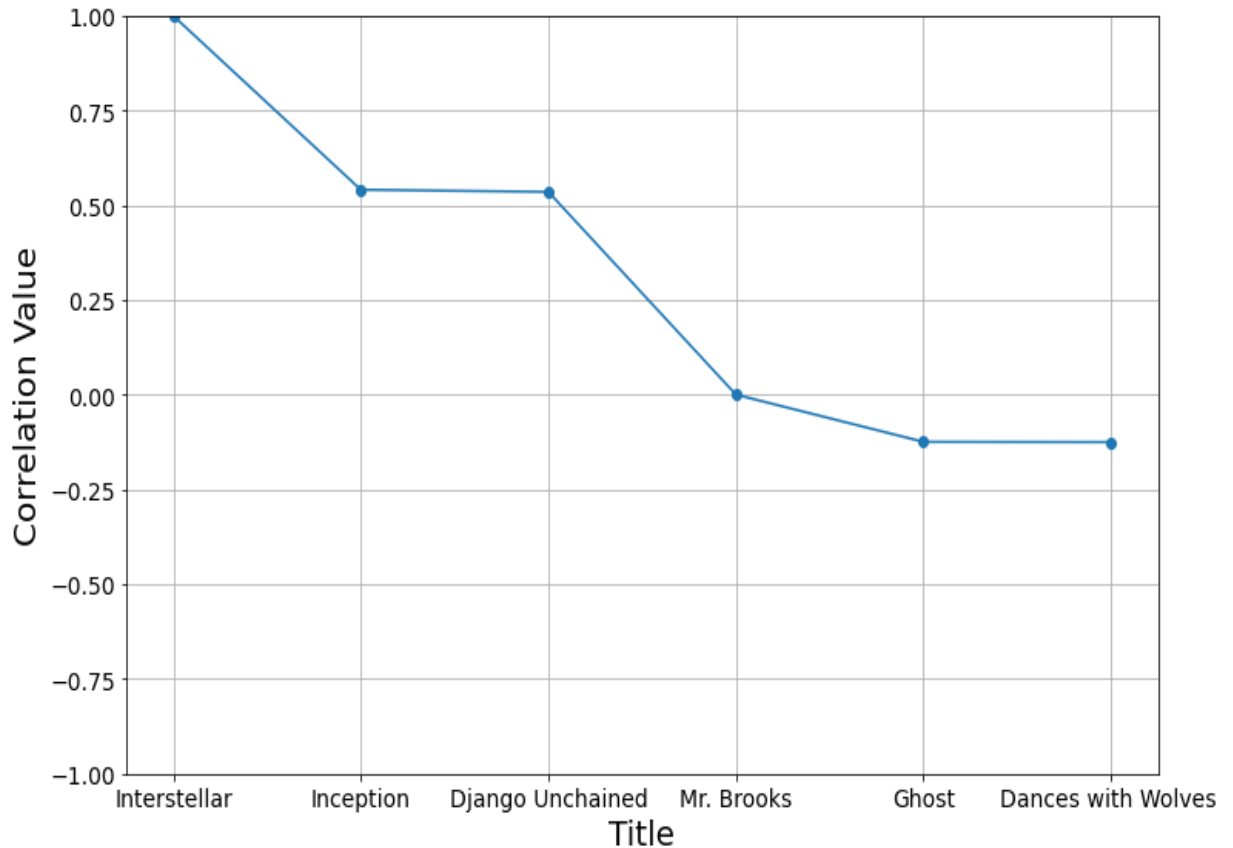


Figure 16: Correlation Coefficient Line Graph

Figure 16 illustrates the recommended movies for Interstellar as per the ratings received from the user. If the user provides a rating of five stars, then the user gets the recommended movies like Inception, Django Unchained as these movies have positive correlation values and if a user provides the rating with one star, then the user gets the recommended movies as Ghost, Dances with wolves as these movies have negative correlation values.

Table 14: Recommended Movies List

Chosen Movie	User Rating	Top 6 Recommended Movies	Correlation Coefficient	Execution Time
Interstellar	5	Inception	0.5413	0.5 second
		Django Unchained	0.5353	
		Edge of Tomorrow	0.5296	
		The Dark Knight Rises	0.5283	
		The Imitation Game	0.5236	
		Deadpool	0.5077	

Table 14 represents the top 6 movies recommended for Interstellar as 5 stars ratings received from the users along with the correlation coefficient values with the chosen movie.

Even though recommendation model is set to work perfectly, the system faced cold start problem, because the new movies added to the database doesn't have real word user ratings, so due to this, it doesn't recognize the movie and becomes unable to give its similar movies, thus cold start problem occurs.

CHAPTER 7

CONCLUSION

7.1 Conclusion

In this project, Movie Recommendation System, the recommendations for the users are based on the user's ratings for the specific movies by using the filtering technique such as item based collaborative filtering approach. All the movies of specific users are filtered & system recommends similar movies on the basis of user's rated movies.

By the date, the system is managed to filter and recommend movies based on ratings for the movies.

7.2 Future Enhancements

In future, excluding the mentioned limitations can make the system more accurate and effective among users. Some future enhancement that can be implemented in the project are listed below:

- With the incorporation of sentiment analysis, the system can get more precise.
- Hybrid recommendation technique i.e., user to user-based and item based collaborative filtering can be used to get accurate recommendation.
- Input parameter such as review, comments can be added to the recommendation engine which can provide more precise and accurate results.

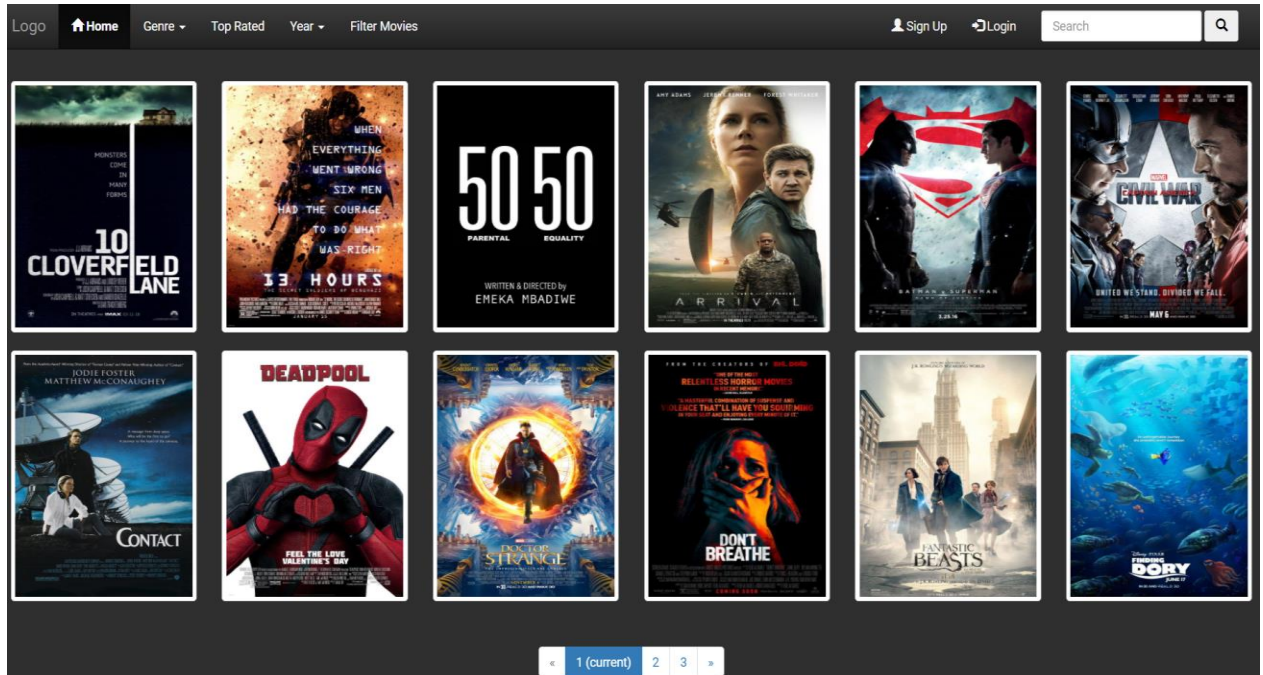
REFERENCE

- [1] P. Phorasim and L. Yu, “Movies recommendation system using collaborative filtering and k-means,” *International Journal of Advanced Computer Research*, vol. 7, no. 29, pp. 52–59, 2017.
- [2] C.-S. M. Wu, D. Garg, and U. Bhandary, “Movie Recommendation System Using Collaborative Filtering,” 2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS), 2018.
- [3] S. R. Gandhi and J. Gheewala, “A survey on recommendation system with collaborative filtering using big data,” 2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), 2017.
- [4] Z. Huang, X. Li and H. Chen, “Link Prediction Approach to Collaborative Filtering,” IEEE, Tucson, 2005.
- [5] S. G. Walunj and K. Sadafale, “An online recommendation system for e-commerce based on apache mahout framework,” *Proceedings of the 2013 annual conference on Computers and people research*, ACM (2013), Cincinnati, 2013.
- [6] Y. Lee, “Recommendation System Using Collaborative Filtering,” San José: Master Thesis and Graduation Research; The Faculty of the Department of Computer Science, 2015.
- [7] F.O Isinkaye , Y.O.Folajimi and B.A. Ojokoh , “Recommendation systems: Principles, methods and evaluation”. *Egyptian Informatics Journal*, 16(3):261– 273, 2015

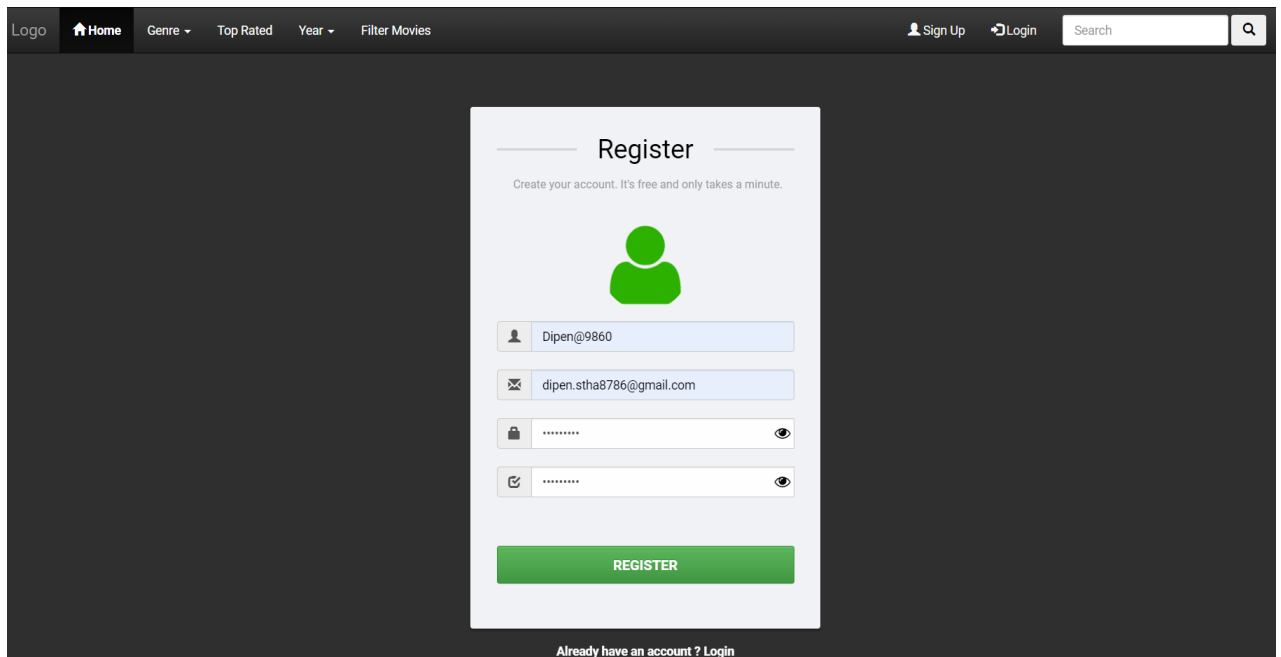
APPENDIX

Appendix I Screenshots of Movie Recommendation Web Application

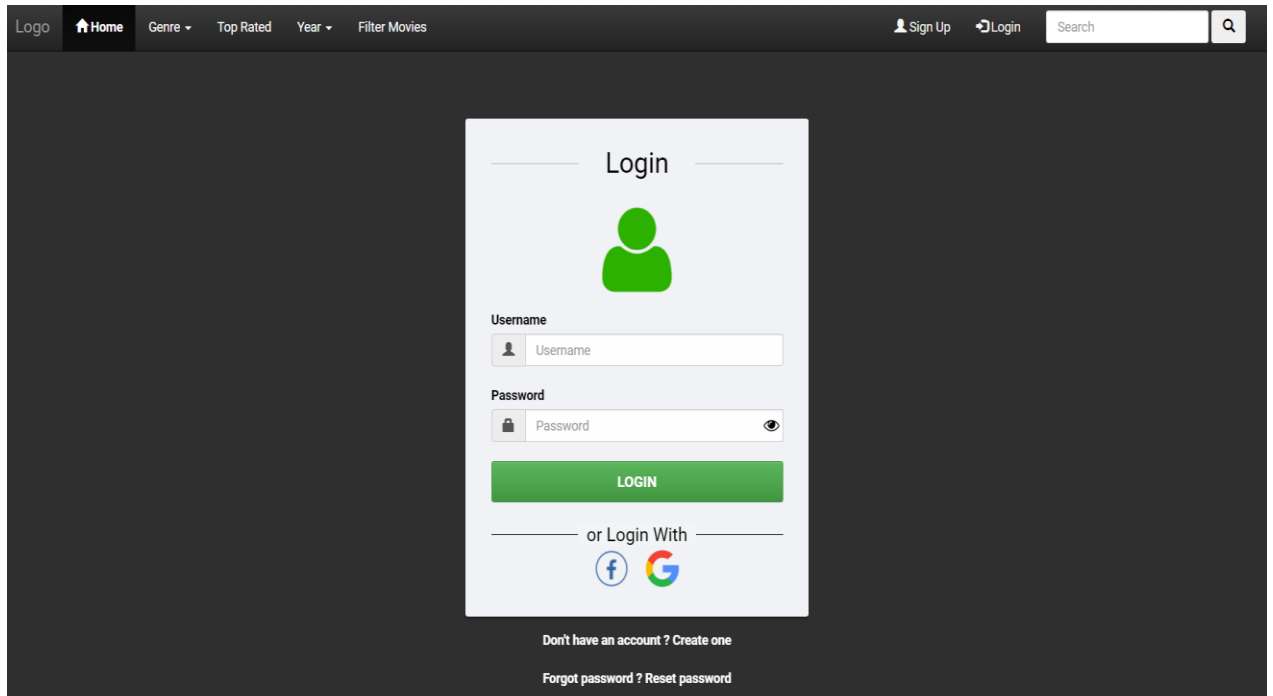
• Home Page



• Signup



• Login



The login page features a dark header with navigation links: Logo, Home, Genre, Top Rated, Year, and Filter Movies. On the right, there are links for Sign Up and Login, and a search bar. The main content area is a light gray box with the title 'Login' and a green user icon. Below the icon are input fields for Username and Password, each with a corresponding icon (person and lock). A green LOGIN button is positioned below the password field. Underneath the button, there is a link 'or Login With' followed by Facebook and Google icons. At the bottom of the box, there are two links: 'Don't have an account ? Create one' and 'Forgot password ? Reset password'.

Logo Home Genre Top Rated Year Filter Movies Sign Up Login Search

Login

Username

Password

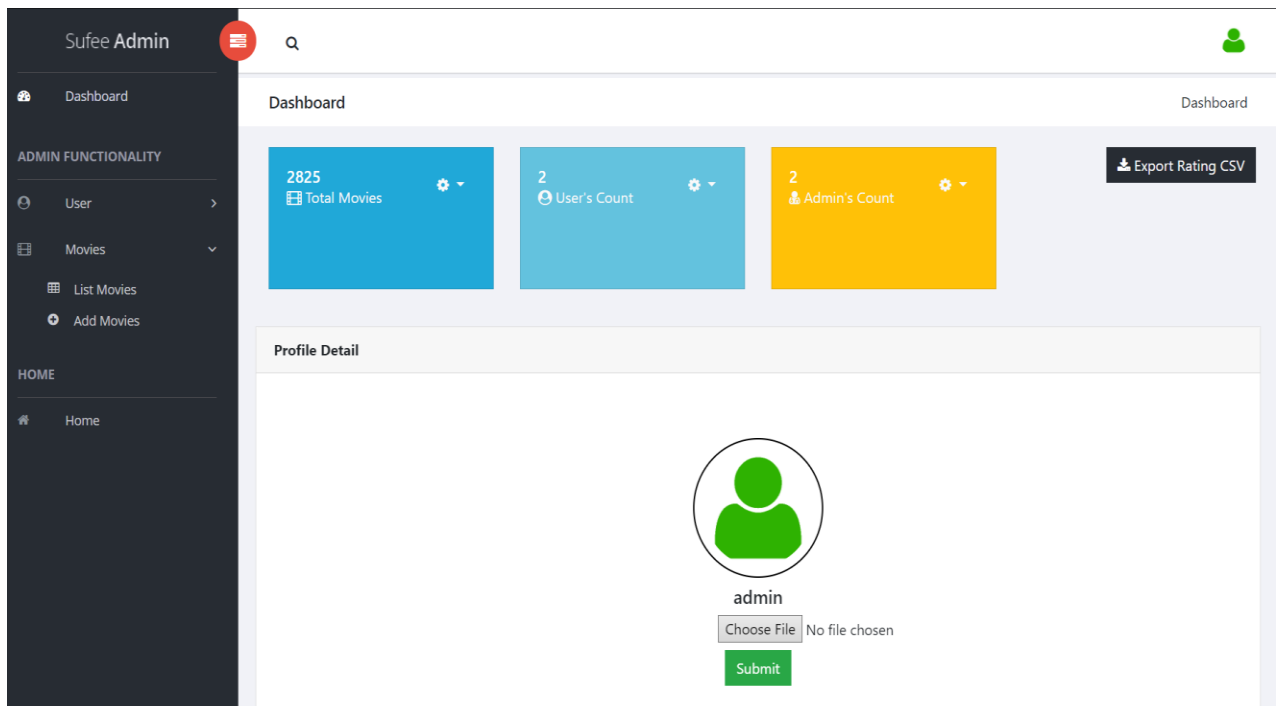
LOGIN

or Login With

Don't have an account ? Create one

Forgot password ? Reset password

• Admin Profile Page



The admin profile page has a dark sidebar with the title 'Sufee Admin' and a search bar. The sidebar contains a menu with 'Dashboard', 'ADMIN FUNCTIONALITY' (User, Movies, List Movies, Add Movies), and 'HOME' (Home). The main content area is titled 'Dashboard' and features three summary cards: '2825 Total Movies' (blue), '2 User's Count' (light blue), and '2 Admin's Count' (yellow). Each card has a settings gear icon. An 'Export Rating CSV' button is in the top right. Below the cards is a 'Profile Detail' section with a green user icon, the name 'admin', a 'Choose File' button, and a 'Submit' button. The text 'No file chosen' is next to the file button.

Sufee Admin Dashboard

ADMIN FUNCTIONALITY

User Movies List Movies Add Movies

HOME

Home

Dashboard

2825 Total Movies

2 User's Count

2 Admin's Count

Export Rating CSV

Profile Detail

admin


Choose File No file chosen

Submit

• User Profile Page

Logo
Home
Genre
Top Rated
Year
Filter Movies
Suggestions
Dipen@9860
Logout
Search


Dipen@9860's Profile



Username : Dipen@9860
Email : dipen.stha8786@gmail.com
User Type : User
Joined : June 30, 2021
Last Seen : June 30, 2021
No. of Movie Rated : 0

Profile Setting

Rated Movies




Change Image

Username:	Dipen@9860
Email:	dipen.stha8786@gmail.com
User Created at:	June 30, 2021, 9:13 a.m.
Status:	Active

Logo
Home
Genre
Top Rated
Year
Filter Movies
Suggestions
Dipen@9860
Logout
Search

Dipen@9860's Profile





Username : Dipen@9860
Email : lakhe8786@gmail.com
User Type : User
Joined : June 30, 2021
Last Seen : June 30, 2021
No. of Movie Rated : 2

Profile Setting

Rated Movies

Show
10
entries
Search:

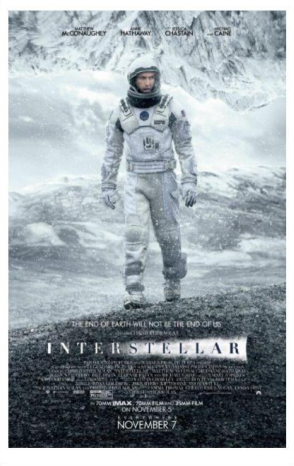
Movie Title	Rating
 Edge of Tomorrow	★ 5.0
 The Martian	★ 4.0

Showing 1 to 2 of 2 entries

Previous 1 Next

• Movie Detail Page

Logo
Home
Genre
Top Rated
Year
Filter Movies
Suggestions
Admin Panel
Logout
Search



Interstellar

Sci-Fi

IMDb 8.6 / 10


2014

★ Rate Movie

★★★★★

★ Rate

Official Trailer





Interstellar - Trailer - Official Warner Bros. ...



1:51 / 2:55

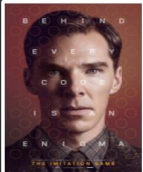

YouTube

Download

Similar Movies

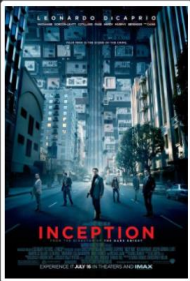


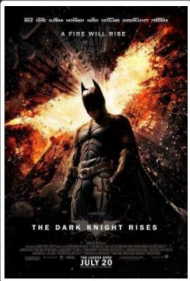
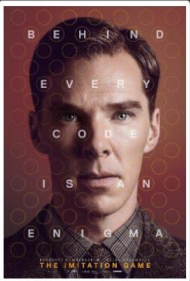




• Recommendation Page

Logo
Home
Genre
Top Rated
Year
Filter Movies
Suggestions
Admin Panel
Logout
Search

Recommended Movies

Appendix II

models.py

```
from django.db import models

from login_signup.models import User

from django.core.validators import MaxValueValidator, MinValueValidator


class Movie(models.Model):
    GENRE = (
        ('Action', 'Action'),
        ('Animation', 'Animation'),
        ('Comedy', 'Comedy'),
        ('Crime', 'Crime'),
        ('Horror', 'Horror'),
        ('Sci-Fi', 'Sci-Fi'),
        ('War', 'War'),
        ('Action Sci-Fi', 'Action Sci-Fi'),
        ('Comedy Crime', 'Comedy Crime'),
        ('Drama Thriller', 'Drama Thriller'),
    )

    YEAR = (
        (2010, '2010'), (2011, '2011'),
        (2012, '2012'), (2013, '2013'),
        (2014, '2014'), (2015, '2015'),
        (2016, '2016'), (2016, '2017'),
        (2016, '2018'), (2016, '2019'),
        (2016, '2020'), (2016, '2021'),
    )

    title = models.CharField(max_length=200)
    genre = models.CharField(max_length=100, choices=GENRE)
    poster = models.FileField()
    imdbrating = models.FloatField(default=1.0)
    year = models.IntegerField(null=True, choices=YEAR)

    def __str__(self):
        return self.title


class Client(models.Model):
    user = models.ForeignKey(User, null=True, blank=True,
                             on_delete=models.CASCADE)

    movie = models.ForeignKey(Movie, null=True, blank=True,
                              on_delete=models.CASCADE)

    rating = models.FloatField(default=1, validators=[MaxValueValidator(5),
                                                       MinValueValidator(0)])

    def __str__(self):
        return self.user.username
```

views.py

```
from django.shortcuts import render, get_object_or_404, redirect
from django.http import Http404, HttpResponseForbidden
from .models import *
from recommendation.models import Movie
from django.db.models import Q
from django.core.cache import cache
import pandas as pd
from .forms import RatingForm
from django.http import JsonResponse

def rate_movie(request, movie_id):

    if request.method == "POST":
        rating = request.POST.get('rating')
        try:
            clientObject = Client.objects.get(Q(user_id=request.user.id) &
                                              Q(movie_id=movie_id))

            form = RatingForm(request.POST or None, instance=clientObject)

            if rating != '0':
                edit = form.save(commit=False)
                edit.save()
                return JsonResponse({'status': 'true'}, status=200, safe=False)
            except:
                form = RatingForm(request.POST)
                if rating != '0':
                    form.save()
                    return JsonResponse({'status': 'true'}, status=200, safe=False)

            return JsonResponse({'status': 'false'}, status=400, safe=False)

def suggestion(request):

    if not request.user.is_authenticated:
        return redirect("login")

    if not request.user.is_active:
        raise Http404

    clients = Client.objects.filter(Q(user=request.user))
                                .filter(Q(rating__gte=0.1)).order_by('-id')[:10]

    movie_id = []
    title = []
    rate = []
    genre = []
    year = []

    suggested_movies = list()

    try:
        for client in clients:
            movie_id.append(client.movie_id)
            title.append(client.movie.title)
            rate.append(client.rating)
            genre.append(client.movie.genre)
            year.append(client.movie.year)

        movies_list = myengine(movie_id, title, rate, genre, year)
```

```

        for i in range(len(movies_list)):

            queryset_movie_suggestion = Movie.objects.
                filter(Q(title=movies_list[i]))

            suggested_movies += queryset_movie_suggestion
    except:
        pass

    context = {'suggested_movies': suggested_movies}

    return render(request, 'recommendation/suggestion.html', context)

def myengine(movie_id, name, rating, genre, year):

    ratings = pd.read_csv('ratings.csv')

    movies = pd.read_csv('movies.csv')

    ratings = pd.merge(movies, ratings).drop(['genres', 'timestamp'], axis=1)

    user_ratings = ratings.pivot_table(index=['userId'], columns=['title'],
                                         values='rating')

    user_ratings = user_ratings.dropna(thresh=3, axis=1).fillna(0)

    item_similarity_df = user_ratings.corr(method='pearson')

    '''
    Creating item_similarity_df dataframe each time is a time consuming task so
    inorder to optimize execution time we use feather file format for creating
    item_similarity_df.feather and cached it for the efficient reading.'''

    item_similarity_df = cache.get('cleaned_data')
    if item_similarity_df is None:
        item_similarity_df = feather.read_dataframe('item_similarity_df.feather')
        item_similarity_df.index = item_similarity_df.columns
        cache.set('cleaned_data', item_similarity_df, timeout=36000)

    movielist = []

    try:
        def get_similar_movies(movie_name, user_rating):
            user_rating = float(user_rating)
            similar_score = item_similarity_df[movie_name] * (user_rating - 2.5)
            similar_score = similar_score.sort_values(ascending=False)
            return similar_score

        def check_seen(movie, seen_movies):
            for item in seen_movies:
                if item == movie:
                    return True
            return False

```

```

length = len(name)
all_movies = Movie.objects.all()
all_movies_list = []

for movie in all_movies:
    all_movies_list.append(movie.title)

all_recommended_movies = []
for i in range(0, length):
    try:
        temp_recommended_movie_list = []
        all_suggested_movies = get_similar_movies(name[i], rating[i])
        temp_recommended_movie_list =
            list(all_suggested_movies.index)

        if rating[i] == 5.0:

            for j in range(1, len(temp_recommended_movie_list)):

                if temp_recommended_movie_list[j] in all_movies_list:

                    all_recommended_movies.
                        append(temp_recommended_movie_list[j])

                if len(all_recommended_movies) >= 6 * (i+1):
                    break

        elif rating[i] == 4.0:

            count = 0
            for j in range(len(temp_recommended_movie_list)):

                if temp_recommended_movie_list[j] in all_movies_list:

                    count +=1
                    if count >= 8:
                        all_recommended_movies.
                            append(temp_recommended_movie_list[j])

                if len(all_recommended_movies) >= 6 * (i + 1):
                    break

        elif rating[i] == 3.0:

            count = 0
            for j in range(len(temp_recommended_movie_list)):

                if temp_recommended_movie_list[j] in all_movies_list:

                    count += 1
                    if count >= 14:
                        all_recommended_movies.
                            append(temp_recommended_movie_list[j])

                if len(all_recommended_movies) >= 6 * (i + 1):
                    break

```

```

elif rating[i] == 2.0:

    count = 0
    for j in range(len(temp_recommended_movie_list)):

        if temp_recommended_movie_list[j] in all_movies_list:

            count += 1
            if count >= 7:

                all_recommended_movies.
                    append(temp_recommended_movie_list[j])

            if len(all_recommended_movies) >= 6 * (i + 1):
                break

elif rating[i] == 1.0:

    for j in range(0, len(temp_recommended_movie_list)):

        if temp_recommended_movie_list[j] in all_movies_list:

            all_recommended_movies.
                append(temp_recommended_movie_list[j])

        if len(all_recommended_movies) >= 6 * (i+1):
            break

except:
    pass

i = 0
for movie in all_recommended_movies:

    if not check_seen(movie, name):
        movielist.append(movie)

    i = i + 1
    if i >= 100 + length:
        break

return movielist

except:
    pass

```