# Movie Recommendation System

*A Project Report Submitted*

*to*

## MANIPAL ACADEMY OF HIGHER EDUCATION

*For Partial Fulfillment of the Requirement for the*

*Award of the Degree*

*Of*

**Bachelor of Technology**

*in*

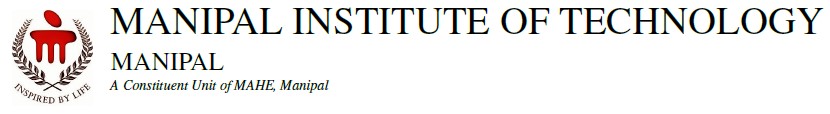
**Information Technology**

*by*

## Sai Yashwanth Reddy(220911390), Shreyansh Tandon(220911316),Ayush Bhagat(220911336), Koyyana Arya Sai(220911354)

*Under the guidance of*

|  |
| --- |
| **Mr. Akshay K C** |
| Assistant Professor |
| Department of I&CT |
| Manipal Institute of Technology |
| **Mr. Raghavendra** |
| Assistant Professor |
| Department of I&CT |
| Manipal Institute of Technology |
| Manipal, Karnataka, India  **April 2024** |



**March 2024**

# ABSTRACT

Recommendation System is a system that seeks to predict or filter preferences according to the user’s choices. Recommendation systems are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general, It is a simple algorithm whose aim is to provide the most relevant information to a user by discovering patterns in a dataset. The algorithm rates the items and shows the user the items that they would rate highly. An example of recommendation in action is when you visit Amazon and you notice that some items are being recommended to you or when Netflix recommends certain movies to you. They are also used by Music streaming applications such as Spotify and Deezer to recommend music that you might like. They gradually learn your preferences over time and suggest new products which they think you’ll love. We can make this application using python language and collaborative based filtering algorithm. Collaborative filtering tackles the similarities between the users and items to perform recommendations. We include a data set with user id, ratings, item number and time spent. With these data we use mapping technique and correlation concept to match user id and ratings. The next movie recommendation should be based on the user’s rating to watched movies

**Table of Contents**

**Table of Contents** [**ii**](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230970)

**Revision History** [**ii**](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230971)

**1. Introduction 3**

1.1 Purpose [3](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230973)

1.2 Document Conventions [3](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230974)

1.3 Intended Audience and Reading Suggestions [3](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230975)

1.4 Product Scope [4](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230976)

1.5 References [5](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230977)

**2. Overall Description** [**5**](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230978)

2.1 Product Perspective [5](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230979)

2.2 Product Functions [5](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230980)

2.3 User Classes and Characteristics [6](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230981)

2.4 Operating Environment [6](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230982)

2.5 Design and Implementation Constraints [6](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230983)

2.6 User Documentation [6](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230984)

2.7 Assumptions and Dependencies [6](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230985)

**3. External Interface Requirements 7**

3.1 User Interfaces [7](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230987)

3.2 Hardware Interfaces [7](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230988)

3.3 Software Interfaces [7](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230989)

3.4 Communications Interfaces [7](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230990)

**4. System Features** [**7**](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230991)

4.1 Sorting of Movies [8](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230993)

**5. Other Nonfunctional Requirements** [**8**](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230994)

5.1 Performance Requirements 9

5.2 Safety Requirements [9](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230996)

5.3 Security Requirements [9](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230997)

5.4 Software Quality Attributes [9](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230998)

5.5 Business Rules [10](file:///C:\Users\acer\Downloads\SRS%20(1).docx#__RefHeading___Toc441230999)

**List of Tables List of Figures Abbreviations**

**Person:**

**Id(primary key)**

**Name**

**Actor:**

**Id (primary key)**

**Movie\_id (primary key)**

**Role**

**Director:**

**Movie\_id (primary key)**

**Id (primary key)**

**Producer:**

**Movie\_id (primary key**

**id (primary key)**

**Genre:**

**Name(primary key)**

**Movie\_id (primary key)**

**Awards:**

**name(primary key)**

**movie\_id (primary key)**

**year**

**Monthly\_income:**

**Movie\_id (primary key)**

**Income**

**Movie:**

**Movie\_id (primary key)**

**Title**

**Release\_date**

**Average\_rating**

**Movie\_description:**

**Movie\_id(primary key)**

**Description**

**Rating:**

**Username(primary key)**

**Movie\_id(primary key)**

**Rating**

**Avreage\_rating (Trigger)**

**User:**

**Username (primary key)**

**Password**

# Chapter 1

## Chapter Title

## 1.1 Purpose

The purpose of the Software Requirements Specification (SRS) document is to provide a detailed overview of our software product, its parameters, and goals. This document aims to gather and analyze and give an in-depth insight into the **Hybrid Movie Recommender system** by defining the problem statement in detail. It concentrates on the capabilities required by stakeholders and their needs while defining high-level product features. The straightforward user interface, hardware, and software requirements of the **Hybrid Movie Recommender system** are provided in this document.

## Document Conventions

Bold: Used for headings, subheadings, and emphasizing important information.

Italic: Used for emphasizing specific terms or phrases.

## Intended Audience and Reading Suggestions

**Intended Audience**:

Developers: They should read the document to understand the system's functional and non-functional requirements, external interface requirements, and risks and mitigations.

Project Managers: They should read the document to understand the project's scope, stakeholders, and overall description.

Marketing Staff: They should read the document to understand the competitive landscape and how the product differentiates itself from competitors.

Users: They should read the document to understand the user characteristics and use case diagrams.

Testers: They should read the document to understand the functional requirements and external interface requirements.

Documentation Writers: They should read the document to understand the document conventions and intended audience.

**Reading Suggestions:**

Overview: Read the introduction, overall description, and system description sections to get a general understanding of the project.

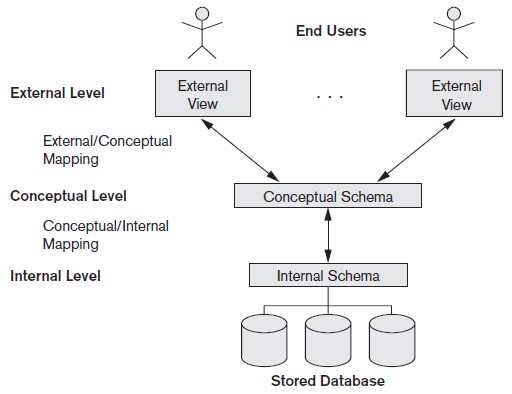
Functional Requirements: Read the functional requirements specification section to understand the system's functional requirements.

Non-Functional Requirements: Read the non-functional requirements section to understand the system's performance, security, safety, and software quality attributes.

External Interface Requirements: Read the external interface requirements section to understand the system's user interfaces, hardware interfaces, and software interfaces.

## Scope of project

This software helps users of the customer platform explore content quickly with our recommendation system’s help. The software we are developing is a Hybrid Recommendation System for Movies, which uses the combination of collaborative and content-based filtering in the context of web-based recommender systems. In particular, we will link the well-known TMDB data set. The content filtering part of the system is based on trained neural networks representing individual user preferences. Using various experiments, we will demonstrate the influence of supplementary user and item features on our proposed hybrid recommender’s prediction accuracy. To decrease system runtime and reveal latent user and item relations, we will factorize our hybrid model via singular value decomposition (SVD). Due to the enormous amount of information available online, the need for highly developed personalization and filtering systems is growing permanently. Recommendation systems constitute a specific type of information filtering that attempts to present items according to the interests expressed by a user.



**Figure 1.1: Three Tier Architecture**

**Chapter 2**

**Overall Description**

## Product Perspective

Upon navigating to the main page, if an existing user, then we need prompt the user to log in. Upon logging in with the correct credentials, they will be redirected to searching movies data to the platform.Based on which new movies are recommended

In case the user is new, a registration prompt is provided first, then the user's profile is built and the user can search for movies he likes then system. There are two types of users: normal users, who’s watch history and movie preferences are recorded, and admin users, which will be handled to the platform owners for the purpose of maintaining the platform.

The website should provide a search box, which enables the users to search movies. The admin user has the option to add and remove movies.

Our Customers are video-on-demand platform companies like Netflix, Amazon Prime, Hulu, and Hotstar. The business model that provides movies on demand to its customers enables searching for content on their website quickly and increases user satisfaction on the platform. Its success revolves around the potency of its recommendations. This System can expand to all video-on-demand platforms with the necessary data. We conceptualize similarity in a broad sense, such as the similarity between movies, members, genres. It uses phrases such as ‘Similar titles to watch instantly,’ ‘More like …’. Search is also one of the crucial aspects of any OTT Platform.

## Product Functions

There will be two major functions which includes-

* Recommend the movies to users which are trending in their region that particular week.
* It will recommend a few movies based on the genre selected by the user.

## User Classes and Characteristics

The user is expected to be Internet literate and be able to use a search engine. The main screen of the Recommendation Website will have the search function and a link to add new content for the content providers.

The Content provider is expected to be Internet literate, have login credentials to navigate through the website, and be able to log in and have full details about the content to be added so that the recommendation system works optimally.

## Operating Environment

Operating environment for the movie recommendation system is as listed below:

* Distributed database
* Client/server system
* Operating system: windows
* Database:sql+ database
* Platform:chrome/edge/firefox

## Design and Implementation Constraints

* Sql commands for above queries / applications.
* Python for the implementation of the recommender system.
* HTML,CSS,JavaScript,ReactJS for the development of the UI.

## User Documentation

**How to use?**

Set your minimum five genres which asks for the same ,and click on the button which says Recommend and the algorithm will provide you with a list of movies.

## Assumptions and Dependencies

Our assumption is that the application is used on a computer or a mobile with enough performance capability, and the use of an up-to-date internet browser.

**Chapter 3**

This project includes sorting off different movies according to the mood of the user which is based on the past experiences of the user.It is based on different past preferences of the user and what kind of films the user would like to watch.We only implement a simple user interface for showing system recommendations.In main system,user logins first and starts giving options of movies which are created on our recommendation service data.The output is shown in main application interface.

**4.1 Sorting of movies**

**4.1.1 Description And Priority**

This project includes sorting off different movies according to the mood of the user which is based on the past experiences of the user.

**4.1.2 Stimulus/Response Sequences**

It is based on different past preferences of the user and what kind of films the user would like to watch.We only implement a simple user interface for showing system recommendations. In main system,user logins first and starts giving options of movies which are created on our recommendation service data .The output is shown in main application interface.

Generate Data: User can get movie recommendations on the device.Movie information will be collected according to length , genres , actors and rating.

**4.1.3 Functional Requirements**

Recommendation:It can suggest movies as recommendations based on data set by user’s approach.The main function will show movies based on recommendation algorithm.When a user will choose movie recommendation based on the user’s past ratings of movies.

* Applicant’s dashboard page
* A page with a list of movies
* A feature to filter and organise movies based on rating,genre and other descriptions
* A feature to enable applicants to explore different movies based on their preferences

*Database:*The database based on past preferences of the user will be created.

*Algorithm:*It will capture preferences and suggest movies based on it.

*Historical Data:*It will be collected by the System.

*Searching Reporting requirement:*Will tell users how to search data.

## References

1. Abdollahpouri, Himan & Burke, Robin. (2019). Multi-stakeholder Recommendation and its Connection to Multi-sided Fairness.

1. Kunaver, Matevž & Pozrl, Tomaz & Pogacnik, Matevz & Tasic, Jurij. (2006). The evaluation of a hybrid recommender system for recommendation of movies.

[3]Rohan Nayak, Aniket Mirajkar, Jeetesh Rokade, Prof. Girish Wadhwa(2018). Hybrid Recommendation System For Movies

**Legend**

1. Name: A
2. Id : B
3. Username : C
4. Password: D
5. Role: E
6. Movie\_id : F
7. Description: G
8. Title: H
9. Average\_rating: I
10. Release\_date : J
11. Income: K
12. Genre\_name: L
13. Rating: M
14. Award\_title: N
15. Award\_year: O
16. Year: P

**Functional Dependency and Normalization:**

1. **R1-> person(A,B):**

**FD: No functional Dependency**

**N: 3NF, BCNF**

1. **R2-> Actor(B,E,F):**

**FD: id. Movie\_id -> roles**

**N: 3NF, BCNF**

1. **R3-> Director(B,F):**

**FD: No functional dependencies**

**N: 3NF,BCNF**

1. **R4-> Producer(B,F):**

**FD:No functional dependencies**

**N: 3NF, BCNF**

1. **R5-> Genre(A,F):**

**FD: No functional dependencies**

**N: BCNF, 3NF**

1. **R6-> Awards(A,F,P):**

**FD: name,movie\_id ->year**

**N: BCNF,3NF**

1. **R7 -> Monthly\_income(K,F):**

**FD: Movie\_id-> income**

**N: 3NF,BCNF**

1. **R8 -> Movie(F,H,J,I):**

**FD: Movie\_id -> Title, release\_date,Average\_rating**

**N: 3NF, BCNF**

1. **R9-> Movie\_description((F,G):**

**FD: Movie\_id-> description**

**N: 3nf,BCNF**

1. **R10 -> Rating(C,F,M,I):**

**FD: username,movie\_id -> Rating**

**N: 3NF,BCNF**

1. **R11 -> User: (C,D):**

**FD: username -> password**

**N: 3NF,BCNF**

.

A diagram on a white board

Description automatically generated