Phase-2
Submission Template

Student Name:*RANJITH.M*
Register Number: 620123106092

Institution: AVS ENGINEERING COLLEGE

Department: ELECTRONICS AND

COMMUNICATION ENGINEERING

Date of Submission: 10/05/2025

Github Repository Link: https://github.com/Ranjith1109-ops/NM-project.git

---

1. Problem Statement

Traditional movie recommendation systems often fail to capture the nuanced preferences of users. This project aims to develop an AI-driven matchmaking system that delivers personalized movie recommendations by understanding user behavior, preferences, and contextual factors.

2. Project Objectives

1. To understand what kind of movies each user likes.

2. To create a smart system that matches users with movies they may enjoy.

3. To improve recommendations by learning from what users watch and like.

4. To build a website or app where users can get movie suggestions.

5. To make sure the recommendations are personal, varied, and interesting

3.flowchart of project working flow

1. User Registration & Login

Collect basic user details and preferences.

2. User Data Collection

Gather viewing history, ratings, liked genres, etc.

3. Movie Database Setup

Use APIs (e.g., TMDB) to fetch movie details like genre, cast, director, etc.

4. Profile Analysis

Build user profiles using machine learning algorithms.

5. AI Matchmaking Engine

Match users to movies using collaborative or content-based filtering.

6. Recommendation Generation

Generate and display personalized movie suggestions.

7. Feedback Collection

User likes/dislikes and ratings improve future recommendations.

8. System Learning & Updates

AI updates user profiles and fine-tunes suggestions.

4.Data Description

1. User Data

This includes:

User ID: Unique identifier for each user

User Preferences: Favorite genres (e.g., action, comedy)

Viewing History: List of movies the user has watched

Ratings: Scores given by the user to movies

Feedback: Likes/dislikes or thumbs up/down

This data helps the system understand what each user enjoys.

---

2. Movie Data

This includes:

Movie ID: Unique identifier for each movie

Title: Name of the movie

Genre: Category like action, romance, thriller, etc.

Cast & Director: Main actors and director of the movie

Release Date: When the movie was released

Plot Summary: Short description of the movie

Ratings: Average rating from other users

5.Data Preprocessing

1. Collect Data

Get user likes, ratings, and watch history

Get movie info (name, genre, actors, etc.)

2. Clean the Data

Remove duplicates or missing values

Fix incorrect or incomplete data

3. Organize the Data

Arrange data in tables (users, movies, ratings)

Match users with the movies they've rated or watched

4. Convert Data

Change text data (like genres) into numbers so AI can understand

Create a matrix showing which user liked which movie

5. Prepare for AI

Use the clean, organized data to train the AI system

6.Exploratory Data Analysis (EDA)

1. Understand the Data

Look at the number of users and movies

See how many ratings each movie has

Check which genres are most popular

2. Visualize the Data

Use bar charts to show top-rated movies

Use pie charts for popular genres

Plot user activity (how many movies each user rated)

3. Find Patterns

Are users watching more action or comedy?

Do some movies get higher ratings than others?

Are some users more active than others?

4. Check Data Quality

Find missing values

Remove duplicate entries

Spot unusual data (e.g., a movie rated 100 in a 5-star system)

7.Feature Engineering

1. User Features

User ID: A unique number for each user

Liked Genres: Action, Comedy, Drama, etc.

Average Rating Given: How the user usually rates movies

Watch Frequency: How often the user watches movies

---

2. Movie Features

Movie ID: A unique number for each movie

Genre: One-hot encoded (e.g., Action = 1, others = 0)

Average Rating: How other users rated the movie

Cast/Director Popularity: Based on other movies' ratings

---

3. User-Movie Interaction Features

User Rating for Movie

Is Favorite Genre: Yes or No

Time Since Last Watched: How recent the movie was watched

8.Model Building

1. Choose a Model Type

You can use one of these methods:

Content-Based Filtering
Recommends movies similar to what the user already likes (based on genre, actors, etc.)

Collaborative Filtering
Recommends movies liked by similar users (based on ratings)

Hybrid Model
Combines both content and collaborative filtering for better results

---

2. Train the Model

Use the cleaned and processed data (user preferences, movie details, ratings)

Let the model learn patterns (e.g., user A likes action + high ratings = recommend similar action movies)

---

3. Test the Model

Use a small part of the data to test the model

Check how accurate the recommendations are

---

4. Improve the Model

Adjust settings (called hyperparameters)

Add more features if needed

Retrain the model to improve results

9.Visualization of Results & Model Insights

1. Top Movie Recommendations

Bar Chart: Show top 5 or 10 movies recommended to a user

Example: Movie titles on X-axis, predicted rating on Y-axis

2. Genre Preference

Pie Chart: Show the percentage of genres the user likes most

Helps to explain why certain movies were recommended

3. User Activity

Histogram: Show how many movies each user has rated

Helps identify active vs. inactive users

4. Model Accuracy

Line or Bar Chart: Compare actual vs. predicted ratings

Shows how close the model's predictions are to real user ratings

5. Heatmap of User-Movie Ratings

A grid where colors show how high or low a user rated a movie

10.Tools and Technologies Used

1. Programming Language

Python – For data processing, AI model building, and visualization

2. Libraries and Frameworks

Pandas – For handling and analyzing data

NumPy – For numerical operations

Scikit-learn – For building machine learning models

Surprise / LightFM – For building recommendation systems

Matplotlib / Seaborn – For visualizing results

TensorFlow / PyTorch (optional) – For deep learning models

3. APIs and Databases

TMDB API – To get movie information (title, genre, cast, etc.)

SQLite / MySQL / MongoDB – To store user and movie data

4. Web Interface (Optional)

Flask / Django – To create a simple website or app

HTML / CSS / JavaScript – For frontend design

5. Tools

Jupyter Notebook – For writing and testing code

Google Colab – Free online tool to run Python code with GPU support

Git – For version control

11. VS Code / PyCharm – For writing codeTeam Members and Contributions

1.Ranjith.M- Project Lead & Model Developer

Designed the project structure

Collected and cleaned data

Built and trained the AI recommendation model

2. Prem kumar.s – Data Analyst

Performed exploratory data analysis (EDA)

Created visualizations for user behavior and movie trends

Helped with feature engineering

3. Tharun.K – Frontend Developer

Designed and developed the user interface

Integrated the movie recommendation display

Handled user input forms (preferences, feedback)

4. Moulidharan .E – Backend Developer

Managed data storage and retrieval

Connected the AI model with the frontend

Ensured smooth flow between user input and model output