A

PROJECT REPORT

ON

“ONESELF MOVIE RECOMMENDER”

SUBMITTED TO

SHIVAJI UNIVERSITY, KOLHAPUR

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE AWARD OF DEGREE BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

MR. VALLABH VINAYAK SANGAR 23UAD308

UNDER THE GUIDANCE OF

Mr. S. P. Pise

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE ENGINEERING

DKTE SOCIETY’S TEXTILE AND ENGINEERING INSTITUTE, ICHALKARANJI

(AN EMPOWERED AUTONOUMOUS INSTITUTE)

2024-2025

D.K.T.E. SOCIETY'S

TEXTILE AND ENGINEERING INSTITUTE, ICHALKARANJI
(AN EMPOWERED AUTONOUMOUS INSTITUTE)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE ENGINEERING

CERTIFICATE

This is to certify that, project work entitled

"ONESELF MOVIE RECOMMENDER"

is a bonafide record of project work carried out in this college by

MR. VALLABH VINAYAK SANGAR 23UAD308

is in the partial fulfillment of award of degree Bachelor of Technology in Artificial Intelligence and Data Science Engineering prescribed by Shivaji University, Kolhapur for the academic year 2024-2025.

MR. S. P. PISE
(PROJECT GUIDE)

PROF. (DR.) T. I. BAGBAN
(HOD AI & DS DEPT.)

PROF.(DR.) L.S.ADMUTHE
(DIRECTOR)

EXAMINER: _____

DECLARATION

We hereby declare that, the project work report entitled "ONESELF MOVIE RECOMMENDER" which is being submitted to D.K.T.E. Society's Textile and Engineering Institute Ichalkaranji, affiliated to Shivaji University, Kolhapur is in partial fulfillment of degree B.Tech.(AI & DS). It is a bonafide report of the work carried out by us. The material contained in this report has not been submitted to any university or institution for the award of any degree. Further, we declare that we have not violated any of the provisions under Copyright and Piracy / Cyber / IPR Act amended from time to time.

Title	Name of the Student	PRN	Signature
MR.	VALLABH VINAYAK SANGAR	23UAD308	

ACKNOWLEDGEMENT

With great pleasure we wish to express our deep sense of gratitude to Mr. S. P. Pise for his valuable guidance, support, and encouragement in the completion of this project report.

Also, we would like to take the opportunity to thank our head of department Dr. T. I. Bagban for his cooperation in preparing this project report.

We feel gratified to record our cordial thanks to other staff members of the Artificial Intelligence and Data Science Department for their support, help, and assistance which they extended as and when required.

Thank you,

Title	Name of the Student	PRN	Signature
MR.	VALLABH VINAYAK SANGAR	23UAD308	

ABSTRACT

The project employs a Machine Learning-based content filtering system for personalized movie recommendations. It leverages textual features extracted from movie descriptions in a dataset of 5,000 movies. The ML model identifies key descriptive words to generate relevant movie suggestions. Recommendations are based solely on content similarities derived from the input text data. The user interface is developed using Python with Streamlit for dynamic interactivity. CSS and JavaScript are utilized to enhance the aesthetic appeal and functionality of the GUI. The home page features banners, platform details, an About Us section, and contact information. Secure access is ensured through dedicated login and registration pages. Upon authentication, users are directed to a recommendation page where they can search for movies. The system retrieves corresponding movie data from the dataset while fetching posters via the TMDB API. Users can update their profiles, change passwords, and log out from their accounts. Overall, the project seamlessly integrates ML with a user-friendly interface to enrich movie discovery.

INDEX

1. Introduction

1. Introduction

In the digital era, where the consumption of online content is growing rapidly, personalized recommendations have become essential to enhance user engagement and satisfaction. Movie recommendation systems are a key part of many streaming platforms, helping users discover relevant content based on their preferences. The *ONESELF Movie Recommender System* is designed to provide a smart, user-friendly movie suggestion experience by leveraging machine learning techniques and web technologies.

This project employs a content-based recommendation system built using Python and the Streamlit framework. It analyzes textual features such as movie descriptions, genres, cast, and keywords to identify similar movies and recommend them to the user. The application also integrates the TMDB API to fetch movie posters, offering a visually appealing interface for users.

The platform is structured to include user authentication, a responsive and interactive homepage, and a recommendation dashboard. It caters to a seamless and secure user experience while delivering accurate and dynamic movie recommendations.

a. Problem Definition

In a vast library of available movies, users often find it difficult to decide what to watch next. With thousands of options to choose from, manually searching and selecting content becomes overwhelming and time-consuming. Existing platforms may not always provide personalized recommendations tailored to a user's interests or past preferences, leading to poor user satisfaction and platform engagement.

b. Aim and Objective of the Project

The main aim of the project is to build a user-friendly, content-based movie recommender system using machine learning and modern web development tools. The key objectives are:

- To develop a movie recommendation model based on content similarity (description, genre, etc.).
- To create an interactive and modern web application using Streamlit.
- To provide a secure user authentication system with login and registration features.
- To fetch dynamic poster content using the TMDB API for better visual appeal.
- To allow users to search for movies and view recommendations instantly.
- To implement user account features like profile display, password update, and logout.

c. Scope and Limitation of the Project

Scope:

- Recommends movies based on textual metadata such as descriptions, genres, and keywords.
- Uses a dataset of 5000 movies for training the recommendation model.
- Provides a clean and professional GUI using Streamlit.
- Enables real-time interaction and movie poster fetching using the TMDB API.
- Supports user login, registration, and profile management features.

Limitations:

- The recommendation system is limited to the 5000 movies present in the dataset; it does not include real-time scraping of newly released movies.
- Only content-based filtering is used, which may lead to repetitive recommendations if the user's input is narrow.
- No collaborative filtering or hybrid techniques are applied.
- Performance may vary depending on internet connectivity due to API calls for fetching posters.

2. Background study and literature overview

a. Literature Overview

Recommendation systems have been a topic of extensive research in the field of machine learning and data science. These systems are primarily categorized into three types: content-based filtering, collaborative filtering, and hybrid approaches.

- Content-Based Filtering recommends items similar to those a user has liked in the past, based on features like genre, keywords, or descriptions. This approach is effective when user profiles are limited but item information is rich.
- Collaborative Filtering relies on the behavior and preferences of similar users. While effective in many cases, it requires a large user base and historical data, and suffers from the cold-start problem.
- Hybrid Models combine both techniques to enhance accuracy and mitigate individual limitations.

Several research papers and applications have implemented content-based models using techniques like TF-IDF (Term Frequency–Inverse Document Frequency), cosine similarity, and natural language processing (NLP) to analyze movie plots and tags. These methods help in identifying the semantic similarity between movies.

In this project, a content-based recommendation approach is adopted using textual metadata such as movie overviews, genres, and keywords. The machine learning model is built using TF-IDF vectorization and cosine similarity to find and rank similar movies.

b. Investigation of Current Projects and Related Work

Many popular streaming platforms like Netflix, Amazon Prime Video, and Hulu use advanced recommendation engines that combine collaborative and content-based techniques. Their systems are backed by large-scale datasets and user behavior analytics to offer highly personalized suggestions.

Academic and open-source implementations have also contributed significantly to this domain:

- MovieLens is a well-known dataset used in research to test recommendation algorithms. It includes user ratings, tags, and metadata for movies.
- TMDB (The Movie Database) provides APIs for accessing movie metadata and posters, which are useful for visual enhancements in movie recommendation interfaces.
- Several GitHub repositories demonstrate implementations of movie recommenders using content-based filtering, often relying on NLP techniques to process textual data.

The ONESELF Movie Recommender System builds upon these ideas by:

- Using a dataset of 5000 movies enriched with textual features.
- Implementing a TF-IDF-based model to compute similarity.
- Creating a user interface with Streamlit for real-time interaction.
- Enhancing the UI by integrating TMDB API for fetching posters and movie details.
- Providing basic user authentication features like login, registration, and user profile management.

This project stands out in its simplicity, user-friendliness, and clean interface while maintaining the core functionalities of a content-based recommendation system.

3. Requirement Analysis

a. Requirement Gathering

The requirement gathering process involves identifying the functional and non-functional needs of the system. The following methods were used for gathering the requirements:

- User Feedback and Observation: Understanding what users expect from a movie recommendation platform.
- Domain Research: Studying how existing movie platforms function.
- Technical Feasibility Analysis: Assessing the tools and technologies available.

Key gathered requirements:

- A user-friendly movie recommendation system.
- Content-based filtering using movie descriptions.
- User login and registration functionality.
- A responsive and visually appealing GUI.
- Integration with an external API for fetching movie posters (TMDB).
- Secure password handling and session management.
- Ability for users to view/update their information and log out.

b. Requirement Specification

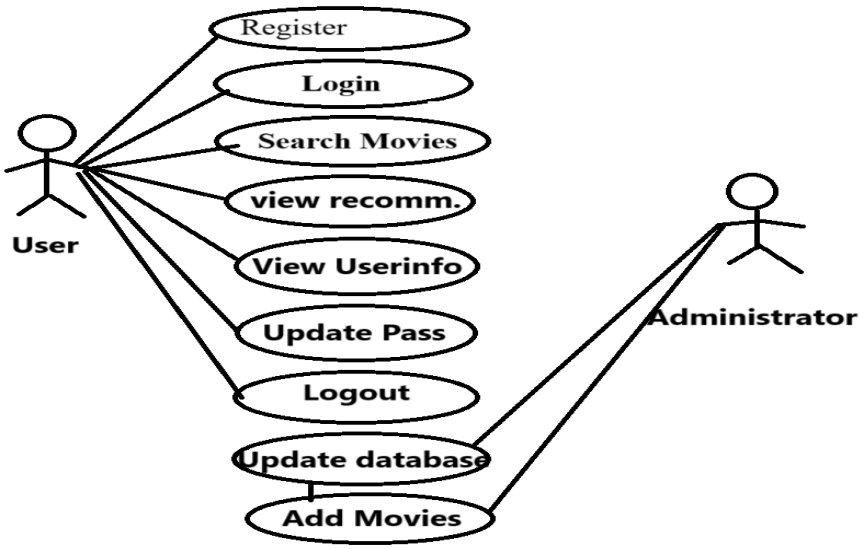
Functional Requirements:

1. User Authentication:
    - Users must register and login to access movie recommendations.
    - Password should be securely handled.
2. Recommendation System:
    - Users should be able to search for a movie and receive similar movie recommendations based on textual similarity.
    - The system must fetch poster images using the TMDB API.
3. User Interface:
    - Home page displaying platform name, banners, About Us, Contact, and a footer.
    - Login page for registered users.
    - Registration page for new users.
    - Recommendation page with movie search and results.
    - User info, update password, and logout options in a dropdown menu.

Non-Functional Requirements:

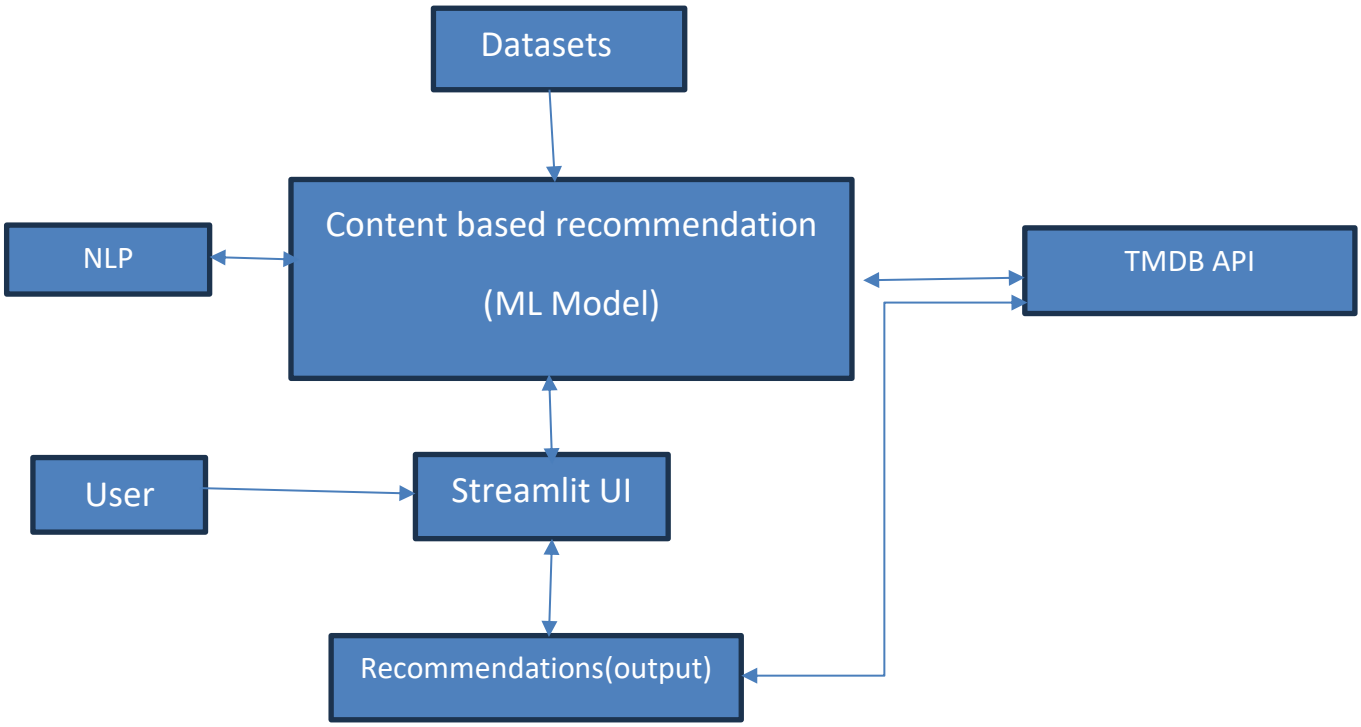
- Performance: The system should respond to user actions with minimal delay.
- Scalability: Should be able to handle additional movies or users if the dataset/API grows.
- Security: Proper password hashing and session control to protect user data.
- Maintainability: Code should be modular and documented for future updates.

c. Use case Diagram

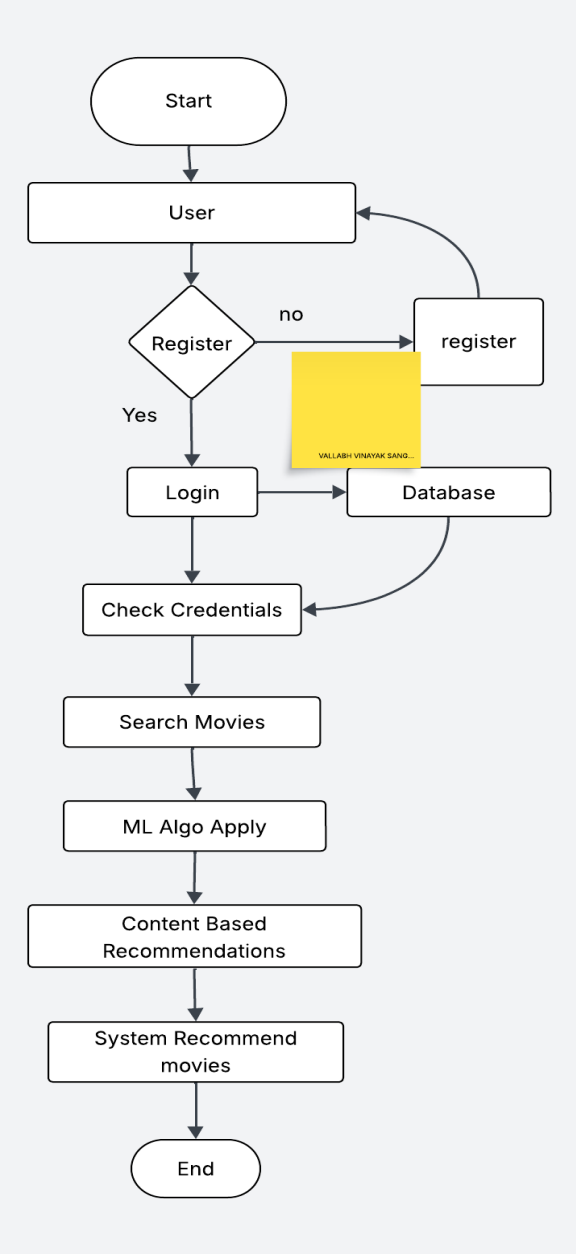


4. System design

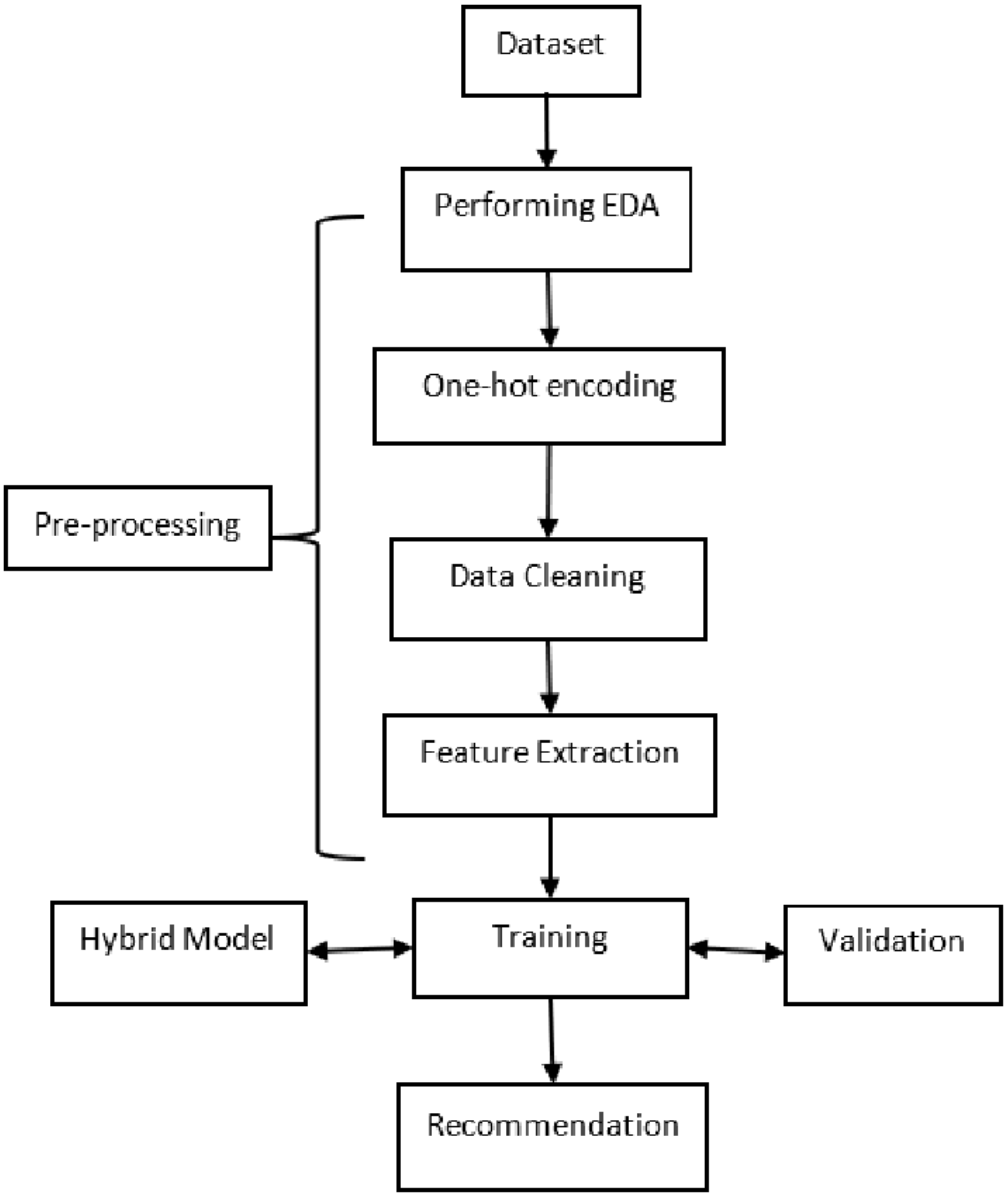
a. Architectural Design



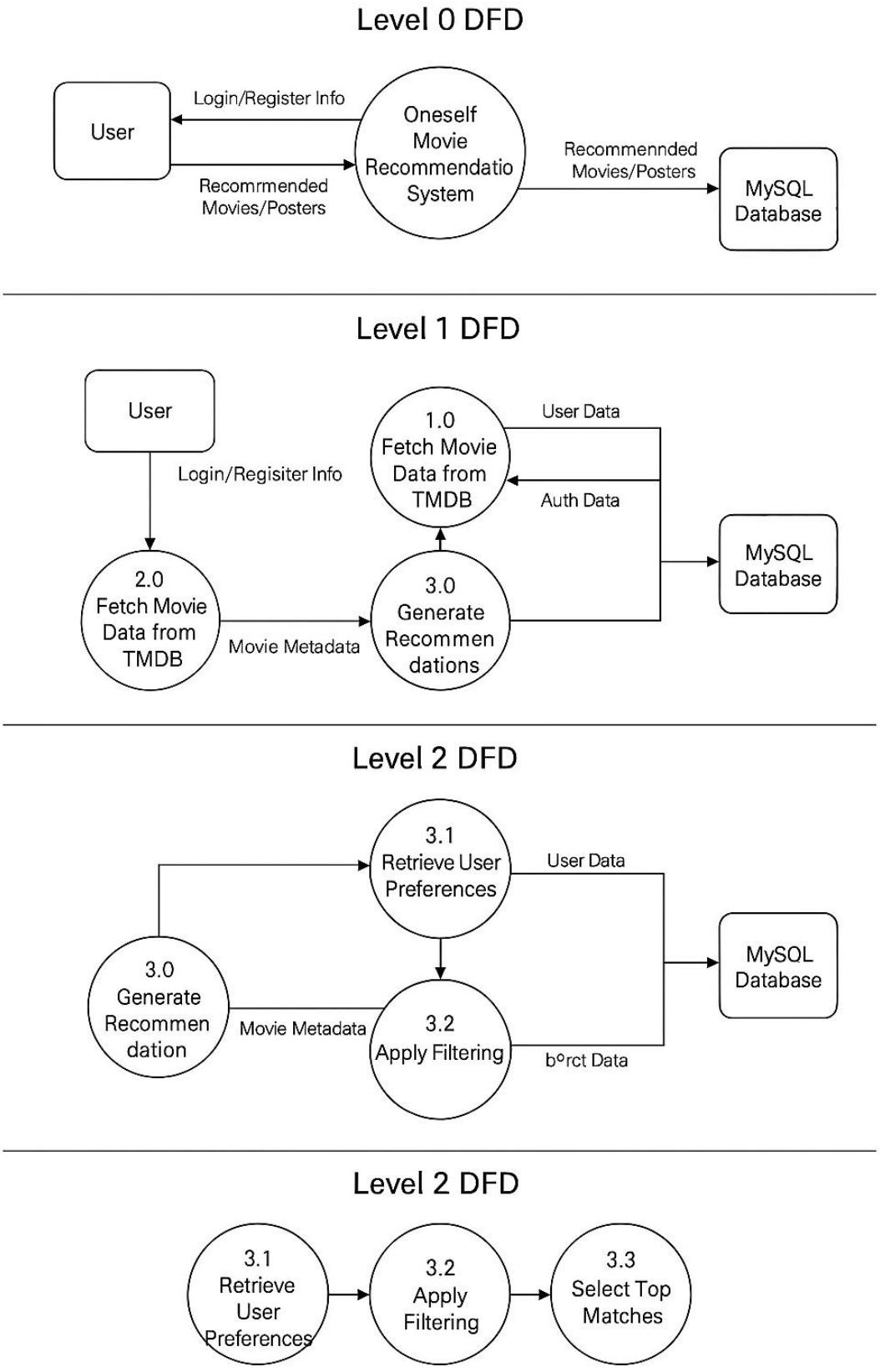
b. Flow Chart


Start
User
Register
no
register
Yes
Login
Database
Check Credentials
Search Movies
ML Algo Apply
Content Based Recommendations
System Recommend movies
End

c. System Modeling



Dataflow Diagram



5. Implementation

a. Agile Methodologies

For the implementation of the ONESELF Movie Recommender System, an Agile approach was adopted to ensure iterative development, regular feedback, and continuous improvement. The Agile methodology was chosen for its flexibility and effectiveness in handling evolving user requirements and technology updates. Key aspects included:

- **Iterative Development:**
  The project was divided into multiple short development cycles (sprints), each focused on implementing specific functionalities such as the user interface, authentication module, recommendation engine, and TMDB API integration. This allowed for regular testing and feedback, ensuring that each module met the desired requirements before proceeding to the next phase.
- **Regular Feedback:**
  Throughout development, feedback was collected from potential users and stakeholders. This input was used to refine both the UI/UX and underlying algorithms. Sprint reviews and demos were held at the end of each cycle, which helped in identifying improvements and adjustments early in the process.
- **Flexibility:**
  The Agile approach facilitated quick adjustments in response to challenges, such as changes in TMDB API response times or alterations in the movie dataset. Priorities were re-assessed continuously, ensuring that the most critical features for user experience were implemented first.
- **Collaboration and Communication:**
  Daily stand-ups and regular retrospectives encouraged transparent communication among team members. This close collaboration ensured that issues were resolved quickly and that everyone maintained a clear understanding of the project's goals and current progress.

b. Development Model

The development model for the ONESELF Movie Recommender System was built on an **Iterative and Incremental** approach, which aligns well with Agile methodologies. This model emphasized delivering a usable system in increments, with each iteration focusing on a subset of features. The key elements of the development model included:

- **Requirements Analysis and Planning:**
  Initial requirements were gathered through user interviews and research into existing recommendation systems. This formed a baseline for the application's core functionalities, such as content-based movie recommendation, user authentication, and poster retrieval via the TMDB API.
- **Prototyping:**
  Early prototypes were developed using Streamlit for the graphical user interface. These prototypes focused on key screens, including the homepage, login, registration, and recommendation pages. User feedback from these prototypes was integral to subsequent improvements.
- **Module-Based Development:**
  The application was divided into modules:
  - **User Authentication Module:** Implementing login, registration, and session

management using MySQL.
    - **Recommendation Engine Module:** Utilizing a content-based machine learning model to generate movie recommendations by analyzing textual movie data.
    - **UI/UX Module:** Building a modern, responsive interface with Streamlit, enriched with custom CSS and JavaScript to create an engaging and professional user experience.
- **Integration and Testing:** Each module was rigorously tested in isolation before integrating into the main system. Integration testing ensured that the communication between modules (e.g., between the recommendation engine and the UI, and between the TMDB API and the poster fetching logic) was smooth.
- **Continuous Improvement:** Feedback from initial deployments was used to refine algorithms and enhance the user interface. Features were prioritized based on user experience, leading to iterative enhancements in both functionality and design.

6. Future Scope

While the current system effectively provides personalized movie recommendations based on content-based filtering, several avenues exist for future enhancements to further enrich the user experience and system performance:

1. Hybrid Recommendation Model:

- **Integration of Collaborative Filtering:**
  Future iterations may combine content-based and collaborative filtering techniques. By incorporating user behavior data (e.g., ratings, watch history), the system could offer even more accurate and diverse recommendations.
- **Dynamic User Profiling:**
  Enhancing the system with real-time user feedback and behavior analysis could personalize recommendations more effectively by continuously updating user profiles.

2. Advanced Natural Language Processing (NLP):

- **Deep Learning Models:**
  Incorporate advanced NLP models, such as transformers, to extract more nuanced features from movie descriptions. This could improve the quality of recommendations by better understanding context and sentiment.
- **Semantic Similarity:**
  Use semantic embeddings (e.g., BERT, GPT) to compute movie similarity, which could capture deeper contextual relationships between movies.

3. Enhanced User Interface (UI) and Experience (UX):

- **Responsive Design:**
  Further refining the design to ensure optimal viewing across all devices (desktop, tablet, mobile) will improve accessibility.
- **Interactive Elements:**
  Incorporate interactive features such as real-time animations, hover effects, and transitions to create a more engaging experience.
- **Personalized Dashboards:**
  Provide users with personalized dashboards that display trending movies, favorite genres, and recently watched recommendations.

4. Data Enrichment:

- **Larger Datasets:**
  Expand the dataset to include more movies and additional metadata (e.g., cast, director, user reviews) to improve recommendation quality.
- **Real-Time Data Integration:**
  Integrate live data feeds for new releases and trending movies to keep recommendations fresh and up-to-date.

5. Scalability and Performance Improvements:

- **Optimized Algorithms:**
  Refine and optimize the recommendation algorithms to handle larger datasets and a higher number of concurrent users.
- **Cloud Deployment:**
  Move the deployment to scalable cloud platforms like AWS, Google Cloud, or Azure to support a growing user base and enhance performance.

6. Additional Features:

- **Social Integration:**
  Allow users to share recommendations, create watchlists, and interact with a community of movie enthusiasts.
- **User Feedback System:**
  Implement a feedback mechanism (such as ratings and reviews) to continuously improve the recommendation accuracy.
- **Content Moderation:**
  Integrate advanced content-moderation techniques to handle inappropriate or low-quality content effectively.

7. References (public repository GitHub source code links)

Github Repository - https://github.com/vallabhsangar12/movie_recommendation