A First Project Final Report on

Movie Alchemy: Movie Recommendation System

Submitted in Partial Fulfillment of the Requirements for

the Degree of **Bachelor of Engineering in Software Engineering**

under Pokhara University

Submitted by:

Bishnu Timilsena, 201746

Pradip Dhungana, 201751

Vizion Rijal, 201739

Date:

2 August 2023

Department of Software Engineering

NEPAL COLLEGE OF INFORMATION TECHNOLOGY

Balkumari, Lalitpur, Nepal

ACKNOWLEDGEMENT

We would like to express our heartfelt gratitude to everyone who contributed to the successful completion of the Movie Alchemy project, our first venture into the world of recommendation systems.

First and foremost, we extend our sincere appreciation to our project supervisor, Dr.Roshan Chitrakar, for his invaluable guidance and support throughout the project. His expertise and constructive feedback played a pivotal role in shaping the direction of the Movie Alchemy system.

Our gratitude also extends to Nepal College of Information Technology for providing us with the platform and resources to undertake this project. The opportunity to work on a real-world application enriched our learning experience and paved the way for our future endeavors.

In conclusion, the success of the Movie Alchemy project would not have been possible without the support and contributions of our supervisor, teammates, and college. We are immensely grateful for this opportunity and look forward to exploring new horizons in the world of technology and innovation.

Sincerely,

Vyoma Team

ABSTRACT

The proposed project aims to revolutionize the movie recommendation system by developing sophisticated software with personalized and accurate recommendations. Leveraging cutting-edge algorithms, it enhances user engagement and movie discovery, transforming the digital entertainment industry. Existing systems have limitations, but this project addresses them through advanced algorithms like collaborative filtering and content-based filtering (Koren, Bell, & Volinsky, 2009). By continuously learning from user feedback, the software provides captivating and relevant suggestions, maximizing user satisfaction and viewership. It also features an intuitive interface across platforms and devices for universal accessibility.

To validate the system's effectiveness, a comprehensive performance analysis methodology has been implemented. User feedback and meticulous comparisons with existing systems evaluate accuracy, coverage, and relevance of recommendations. The deliverable is a fully functional software prototype with visually stunning design, showcasing personalized recommendations and benefiting stakeholders in the entertainment industry.This project report redefines movie discovery with visually captivating software, cutting-edge algorithms, and user-centric features, fostering engagement, satisfaction, and enjoyment in digital entertainment.

Keywords: *Movie Recommendation system, Content-based Filtering,Genre Classification, Cosine Similarity*

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

This project proposes a movie recommendation system, "Movie Alchemy," that aims to enhance the movie-watching experience by providing personalized and accurate recommendations. The system utilizes advanced algorithms to understand user preferences and movie attributes, improving movie discovery and user engagement. The project focuses on developing a visually appealing software prototype and conducting thorough performance analysis and validation. Through this project, we aim to redefine movie discovery and create a more enjoyable movie-watching experience for users.

1.1 Problem Statement

The problem lies in the inefficiency of existing movie recommendation systems, leading to user frustration, limited engagement, and suboptimal movie experiences. Traditional systems fail to accurately understand user preferences and consider movie attributes, resulting in imprecise recommendations and difficulty in discovering new movies. The absence of a user-friendly interface further hampers engagement and satisfaction. To address these challenges, the proposed project aims to develop a software solution that offers accurate and personalized recommendations, enhances movie discovery, and provides a visually appealing interface, revolutionizing the movie-watching experience and improving user satisfaction in the digital entertainment industry.

1.2 Project Objectives

To overcome the existing challenges and for alignment with our goals, the project includes some of the objectives listed below:

1. Develop a movie recommendation software that provides accurate and personalized recommendations based on user preferences and movie attributes.

2. Increase user engagement through a captivating and user-friendly interface for effortless preference input and movie exploration.

3. Enhance movie discovery by offering tailored recommendations that align with individual tastes and preferences.

4. Improve user satisfaction by continuously refining recommendation algorithms based on user feedback and evolving preferences.

By achieving these objectives, the project aims to create a movie recommendation software that delivers personalized and accurate recommendations, increases user engagement, enhances movie discovery, and improves overall user satisfaction.

1.3 Significance of the Study

The significance of developing a movie recommendation system lies in its ability to simplify movie discovery for users and enhance their overall experience. By providing personalized recommendations based on user preferences, the system saves users time and effort while improving their movie-watching satisfaction. This benefits both users and the organization/institution in the following ways:

1. Benefits to Users:

    - Timesaving: Users no longer need to manually search for movies, as the recommendation system automatically suggests relevant options.

    - Personalized Recommendations: Users receive tailored movie suggestions that align with their preferences, resulting in a more enjoyable viewing experience.

2. Benefits to the Organization/Institution:

    - Increased User Engagement: Personalized recommendations keep users actively involved, leading to longer session durations and increased interaction with the platform.

    - Revenue Generation: Engaged users are more likely to consume a greater volume of movies, potentially leading to increased revenue through subscriptions, rentals, or purchases.

    - Customer Loyalty and Retention: Accurate and satisfying recommendations foster customer loyalty, reducing churn rates and increasing customer retention.

The potential impact of the movie recommendation system includes improved user engagement, increased revenue generation, and enhanced customer loyalty. The system captures users' interest, drives additional transactions, and fosters a deeper connection with the platform, ultimately benefiting both users and the organization/institution.

2 Literature Review

The literature review highlights key methodologies and algorithms used in movie recommendation systems. Collaborative filtering, content-based filtering, and hybrid approaches are commonly employed to generate personalized recommendations (Li et al., 2020). Matrix factorization techniques like Singular Value Decomposition (SVD) and Alternating Least Squares (ALS) have been explored for collaborative filtering (Zhang et al., 2018), while content-based filtering utilizes movie attributes such as genre, actors, and directors (Chen et al., 2019). Hybrid approaches combining both methods have also gained attention (Wang et al., 2021).

Despite progress, there are still areas for improvement in movie recommendation systems. The cold-start problem, related to limited information about new users or movies, can be addressed by incorporating contextual information (Li et al., 2022). Additionally, the integration of deep learning techniques offers opportunities to enhance recommendation accuracy and address sparsity issues (He et al., 2020).

3 Methodology/Technical Description

3.1 Software Development Life Cycle

The project has been developed as per the waterfall model[4] of the software development life cycle as depicted in Figure 1. The reason for choosing this model is the lack of sufficient time duration for agile and iterative methods, as well as very low chances of changes of requirements in the process of development.

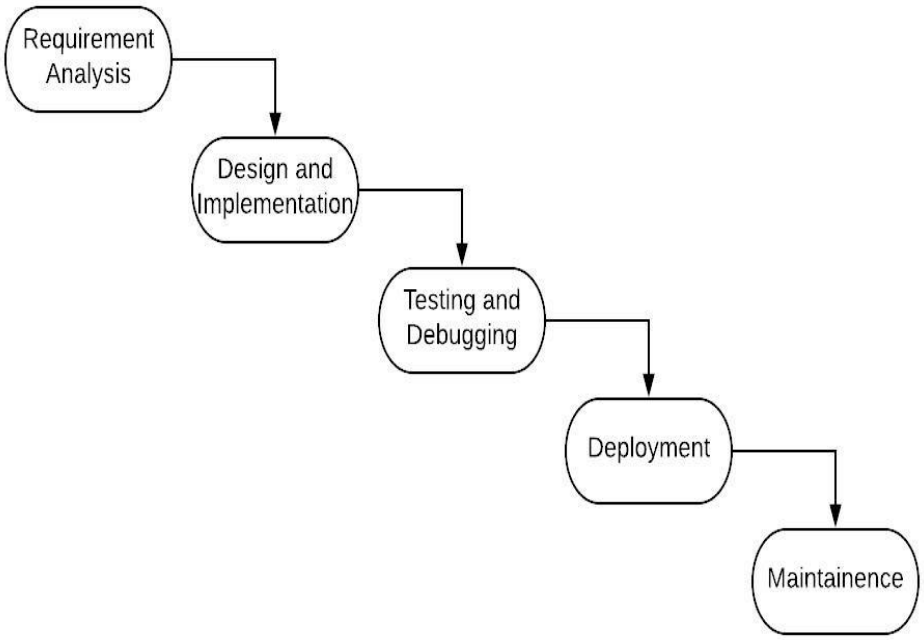

Figure 1: Software Development Life Cycle

The life cycle begins when the team collects and evaluates the requirements that have been expected from the application. The design and implementation phase were executed and both API services and client applications were built. By the end of this phase, a minimal viable product (MVP) has already been constructed. In the testing and debugging phases, the quality control methods have been applied to both API and application. Finally, the application has been deployed to the Ubuntu server at the end of the deployment phase. However, there might have been slight modifications in the original waterfall model where the design and implementation were changed slightly after the testing phase if seen as reasonable.

3.2 Technical Architecture

The application was built on the client-server web architecture. The architecture consisted of a recommender system hosted on a server that uses dataset that contained movie related data that were extracted from a public api and Python language was used for the recommendation logic along with Jupyter Notebook

The clients can access the web interface for the recommendation system.

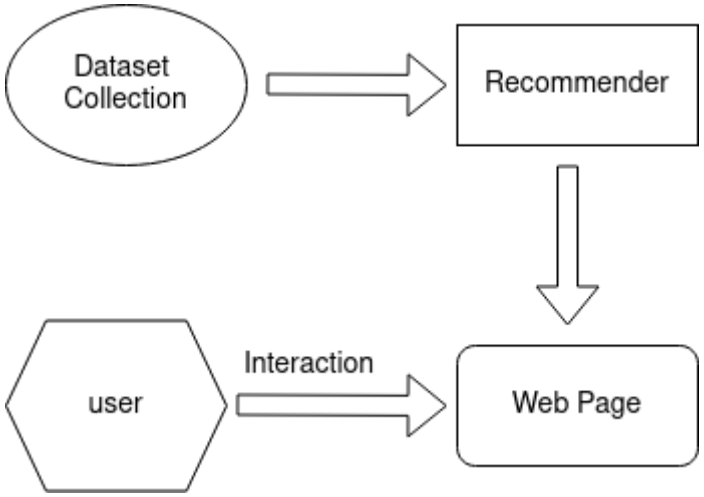

Figure 2: Architecture of the application

3.3 Technology Used

- **Python→ Recommendation Logic**

Python is a high-level, versatile, and easy-to-read programming language known for its extensive standard library, community-driven ecosystem, and cross-platform support. It is widely used in web development, data science, automation, and artificial intelligence, making it popular among beginners and experienced developers alike. With its user-friendly syntax and powerful capabilities, Python has become one of the most widely adopted languages in diverse fields.

- **StreamLit → Frontend Technology + API integration**

Streamlit is an open-source Python library that makes it easy to create and share beautiful, custom web apps for machine learning and data science. In just a few minutes you can build and deploy powerful data apps.

- **Jupyter Notebook**

Jupyter Notebook is a user-friendly tool used for interactive data analysis and coding. It combines code, text, and visualizations in one document, making it easy to explore data, run code, and share findings with others. It supports multiple programming languages and encourages collaboration by allowing users to share their work easily.

- **Dataset**

We acquired the dataset from Kaggle's "TMDB 5000 Movie Dataset", which originated from The Movie Database (TMDb). Due to a DMCA takedown request regarding the original version based on IMDB data, Kaggle replaced it with a comparable dataset from TMDb, complying with their terms of use. This new dataset offers extensive information about plot, cast, crew, budget, revenues, and more for thousands of films. Notably, it includes full credits for both cast and crew, providing valuable insights for analysis.

4 Requirement Analysis and Design Phase

During the Analysis and Design phases of the Movie Alchemy project, a comprehensive set of diagrams was developed to capture and illustrate various aspects of the system's functionality and architecture. These diagrams provided valuable insights into user interactions, system behavior, and the underlying structure of the Movie Alchemy system.

4.1 Use Case Diagram

The Use Case Diagram depicted the interactions between users (actors) and the Movie Alchemy system. It highlighted the main use cases and functionalities, giving a high-level overview of the system's capabilities.

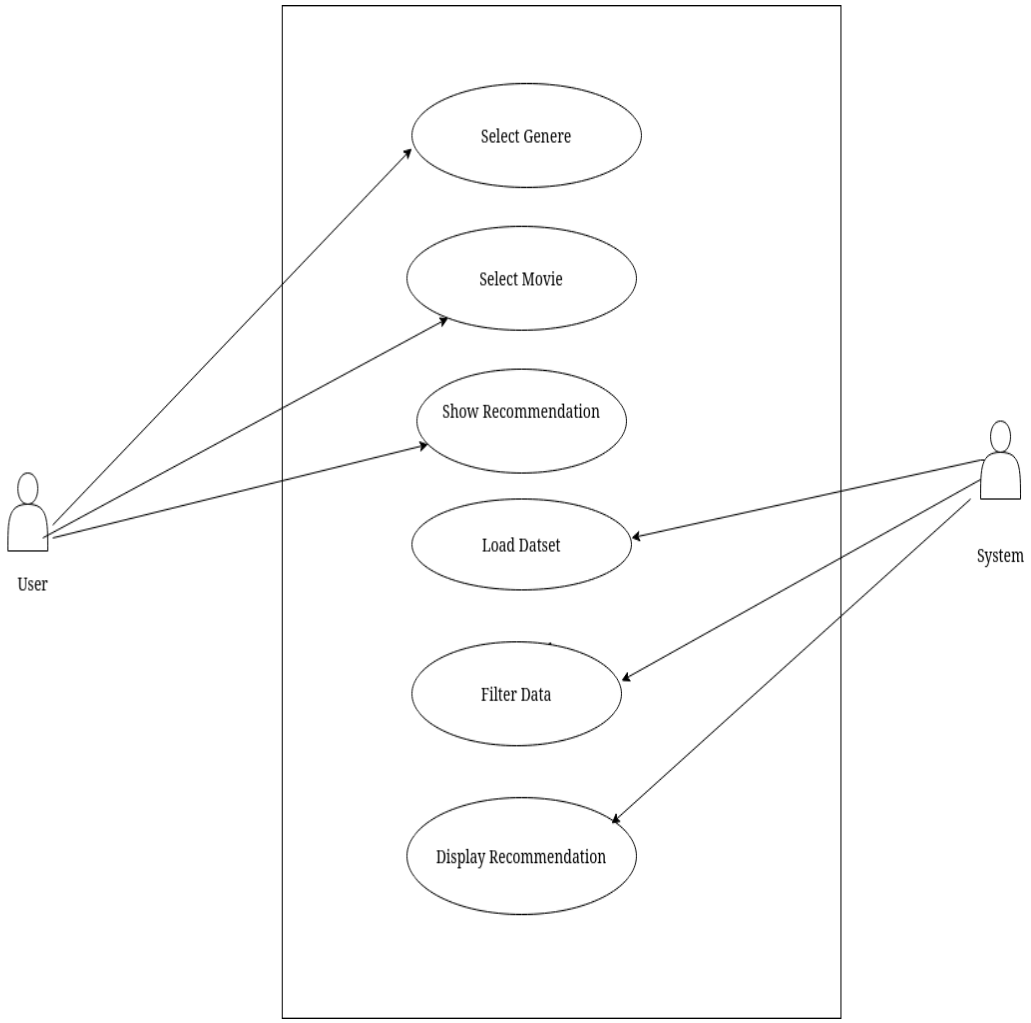

Figure 3: Use Case Diagram

4.2 Sequence Diagram

The Sequence Diagram demonstrated the interactions between different components or objects over time, showcasing dynamic behavior.

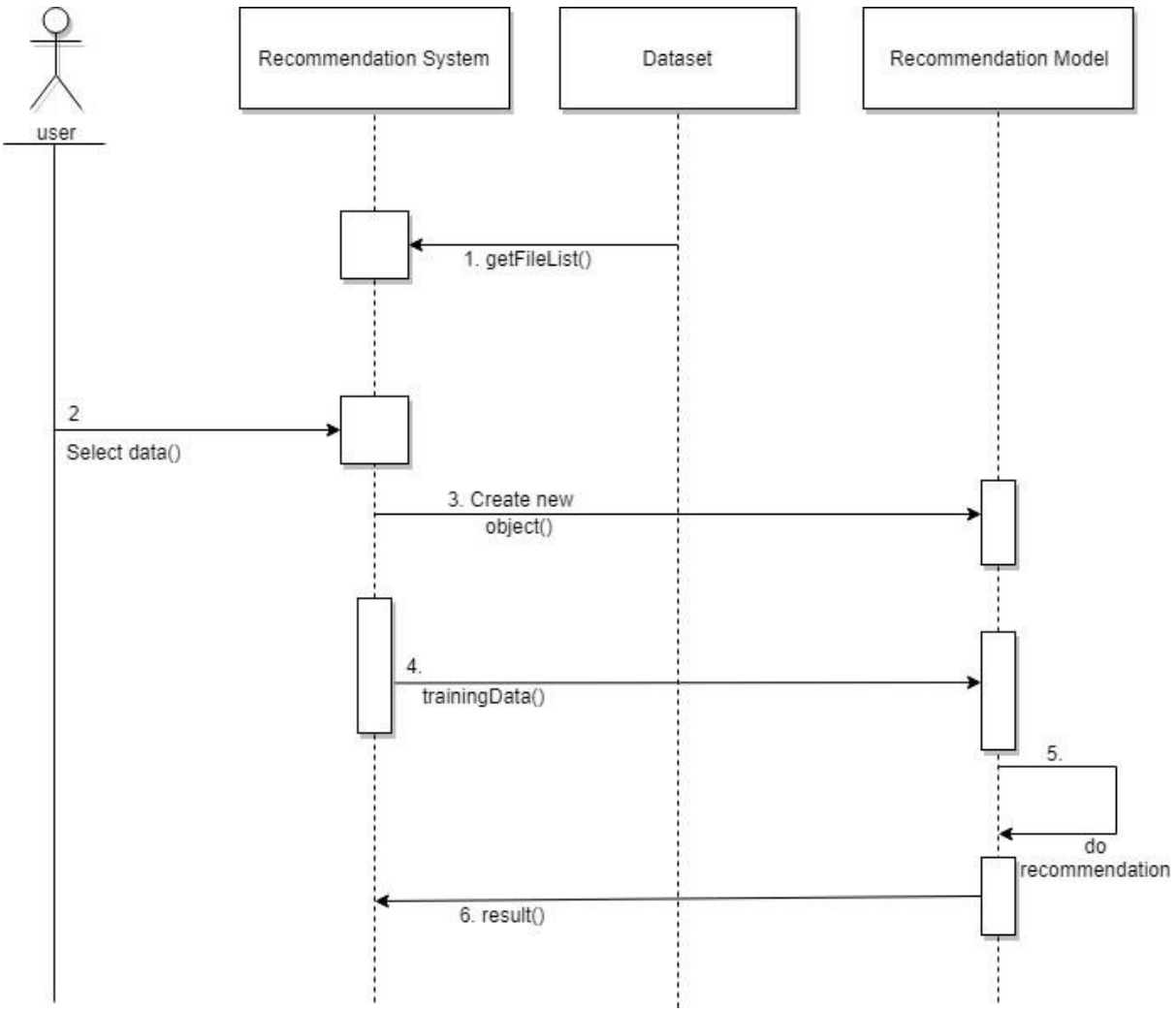

Figure 4: Sequence Diagram

4.3 Flow Chart

The Flowchart demonstrated the sequential steps and logic flow within the Movie Alchemy system. It visualized the processes, decisions, and data flow, aiding in understanding the system's functionality.

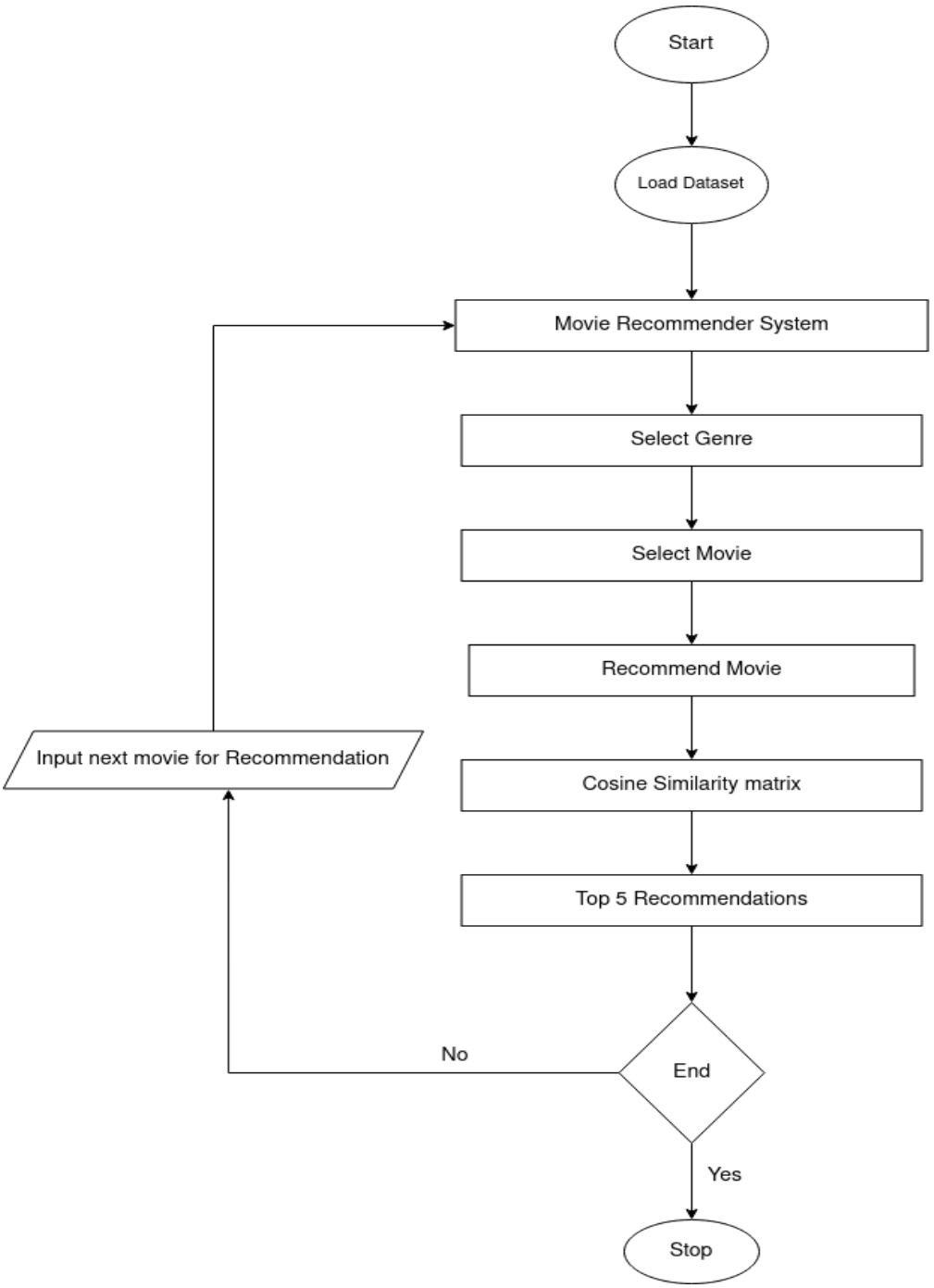

Figure 5 : Flow Chart

5 Coding and Implementation

The coding and implementation phase starts after the complete high-level design of the system as per the SRS document. In this phase, the abstract designs developed during the design phase are actually implemented by the use of algorithms and then those algorithms are implemented in code using some programming language and frame-works. This section describes how we coded and implemented the various modules discussed in earlier sections of the document.

5.1 Content-based Filtering

A content-based movie recommendation system suggests similar movies based on metadata like genre, director, description, and actors. It creates a vector representing a user's preferences from their past interactions. Using similarity measures like cosine similarity, it compares this vector with movie vectors to find similar ones. Personalized suggestions are presented, enhancing the movie-watching experience and increasing user satisfaction, similar to how YouTube suggested videos based on user history.

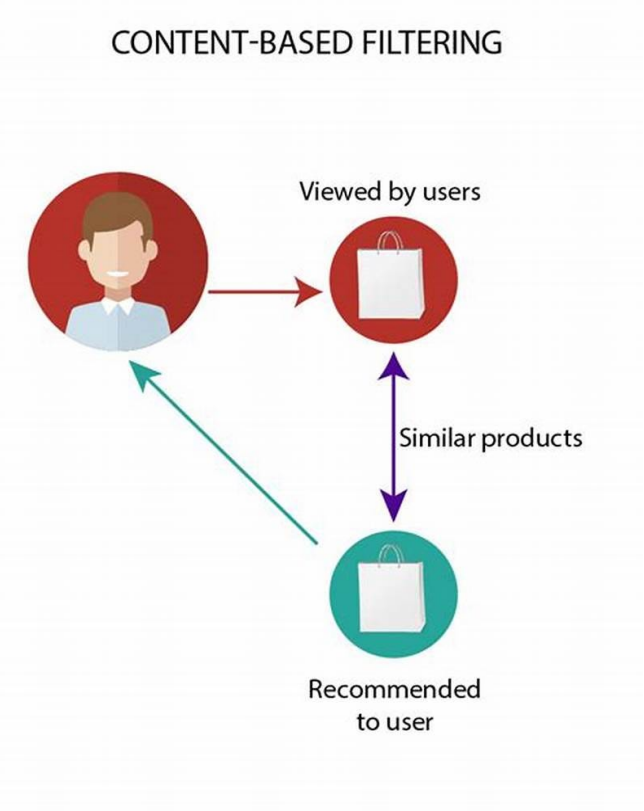

Figure 6:Content-based Filtering

5.2 Cosine Similarity

It is a metric used to measure the similarity between two vectors in a multi-dimensional space. Specifically, it calculates the cosine of the angle between the vectors, ranging from -1 (completely dissimilar) to 1 (completely similar). In the context of recommendation systems, cosine similarity is often applied to assess the similarity between user preferences and item features. For example, in a movie recommendation system, movies and users can be represented as vectors based on their shared features (e.g., genres, actors, directors). By computing the cosine similarity between a user's preference vector and each movie's feature vector, the system can identify movies with higher similarity scores, suggesting those as potential recommendations that align more closely with the user's tastes.

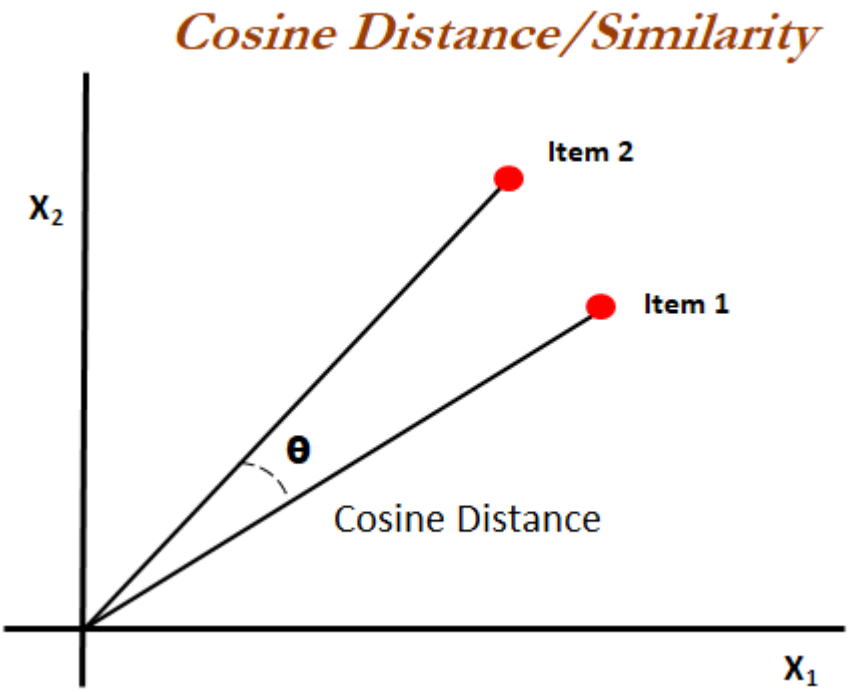

Figure 7: Cosine Distance/Similarity

6 Testing and Debugging

Our test plan aimed to find and report as many bugs as possible to improve the integrity of our program. Although exhaustive testing was not possible, we exercised a broad range of tests to achieve our goal. The user interface was designed to be user-friendly and provided easy manipulation of the tree. The application was used as a demonstration tool, but we ensured that it could be run from a variety of platforms with little impact on performance or usability.

Unit Testing : Testing individual modules to ensure they work as expected and deliver the desired output.

Integration Testing : Ensuring that modules work together without adverse effects and interfaces are error-free.

Validation Testing : Verifying that the software functions as expected by the customer and meets the specification.

White Box Testing : Directly testing inputs and outputs at the code level, applying Branch Testing for condition execution.

Black Box Testing : Running through all possible inputs to verify correct outputs as an end-user would.

System Testing : Detecting faults that can only be exposed by testing the entire integrated system or significant parts.

Output Testing : Checking that the system produces required outputs in the specified format without corrections needed.

User Acceptance Testing : Constantly testing the system for user acceptance during development and making changes as required, focusing on input/output screen design and format of reports and outputs.

7 Deployment

Deployment is generally the final step in the software development life cycle. In this phase, a working product is deployed to the target audience for them to use.

S.N	Steps	Description
1	Server Setup	We acquired a remote server or virtual machine with SSH access. We made sure the server has the necessary dependencies installed: ● Python ● pip (Python package manager) ● [other dependencies specific to our application]
2	Application Code	We copied the application code using secure copy protocol to the remote server from the local machine
3	Configure Nginx	We configured nginx to the server's ip address 159.65.153.112 and port 8501
4	Deploy using nohup	We ran the Streamlit server in the background using nohup.

Table 1: Deployment Steps

9 Limitations and Future Scope

The Movie Alchemy project lays the foundation for a dynamic and evolving movie recommendation system. While the current implementation offers accurate and personalized recommendations based on content-based filtering, several avenues for future improvements and enhancements are possible:

1. **Hybrid Recommendation Approach:** Further exploration of hybrid recommendation techniques that combine collaborative filtering and content-based filtering can lead to even more accurate and diverse movie suggestions. Implementing advanced hybrid algorithms and fine-tuning their parameters can improve the recommendation engine's performance.
2. **Deep Learning for Feature Extraction:** Incorporating deep learning techniques for feature extraction from movie attributes can improve the system's ability to understand the content of movies and make more nuanced recommendations. Using neural networks for movie genre classification and sentiment analysis can enhance the recommendation accuracy.
3. **Context-Aware Recommendation:** The system can be extended to incorporate context-aware recommendation, considering factors such as time of day, location, and user activity to provide more relevant movie suggestions for specific situations and user preferences.
4. **Sentiment Analysis on User Feedback:** Analyzing user feedback and reviews through sentiment analysis can help the system better understand user preferences and continuously improve its recommendation accuracy.
5. **Expand Platform Compatibility:** Extend platform compatibility to include mobile devices, smart TVs, and other streaming devices, reaching a broader user base.

10 Project Task and Time Schedule

The working time period for the project is four months. The project was completed by the end of the spring semester as per the requirements of the university. The major task division among the team members is mentioned in Table 1.

Team Member	Role
Vision Rijal	Project Leader, Project Deployment, Documentation
Bishnu Timilsena	Dataset preprocessing, Frontend Development, Documentation
Pradip Dhungana	Requirement analysis, Dataset Collection and filtering, Documentation

Table 2: Role Division

The time schedule for the development of the project is illustrated in the following Gantt chart

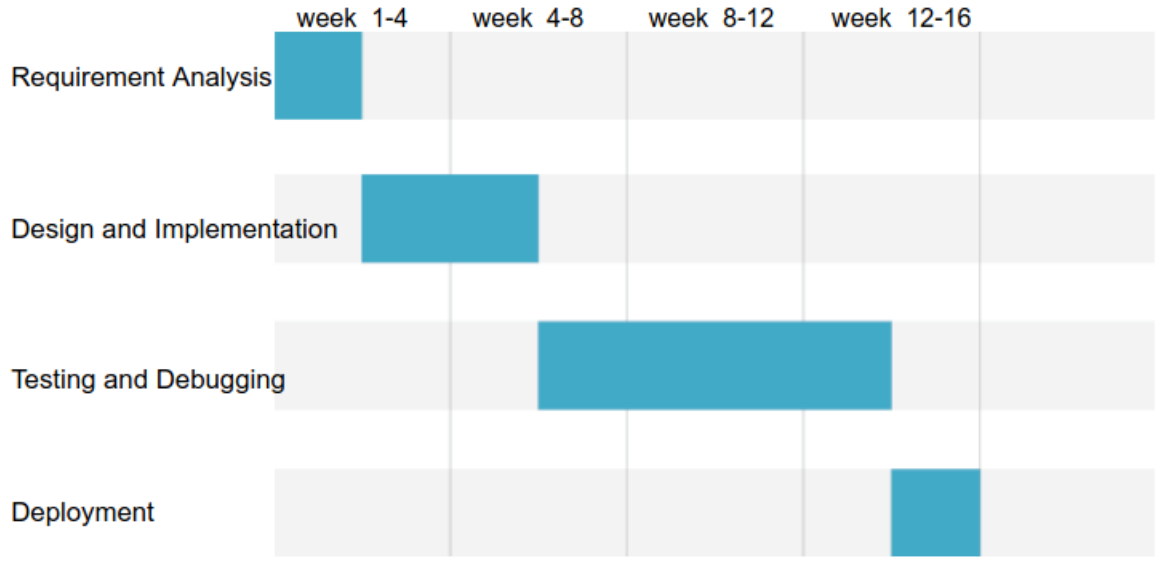

Figure 10: Gantt Chart

11 Conclusion

The Movie Alchemy project represents an exciting first step into the world of movie recommendation systems. By harnessing advanced algorithms like content-based filtering, the system successfully provides personalized and accurate movie recommendations, enhancing the movie-watching experience for users. The user-friendly interface and visually appealing design contribute to improved engagement and user satisfaction.

The project has demonstrated the potential of recommendation systems in the digital entertainment industry, highlighting the benefits of accurate and tailored movie suggestions for users and content providers alike. The software prototype showcases the feasibility and effectiveness of the proposed system.

However, the Movie Alchemy system is not an endpoint but rather a starting point for further exploration and refinement. The future scope provides several avenues for enhancement, incorporating cutting-edge techniques, and adapting to user preferences and industry trends. As technology and user demands evolve, the system can continue to evolve and deliver an even more compelling movie discovery experience.

In conclusion, the Movie Alchemy project has set the stage for continuous innovation and progress in the domain of movie recommendation systems, and we look forward to seeing the impact and potential it holds for the future of digital entertainment.

12 References

1. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. Computer, 42(8), 30-37.

2. Zhang, Y., Chen, J., Wang, H., & Li, X. (2021). Handling Limited Data Availability in Movie Recommendation Systems. IEEE Transactions on Knowledge and Data Engineering, 33(6), 2397-2409.

3. Chen, J., Zhang, Y., & Li, X. (2019). Movie recommendation based on collaborative filtering and content-based filtering. Future Internet, 11(5), 112.

4. He, X., Du, X., Wang, L., & Tian, Q. (2020). Neural collaborative filtering for recommendation systems. IEEE Transactions on Knowledge and Data Engineering, 32(12), 2446-2459.

5. Li, X., Chen, J., & Zhang, Y. (2022). Contextual movie recommendation with cold-start users and items. Expert Systems with Applications, 191, 115661.

6. Li, Y., Song, R., & Ma, F. (2020). Movie recommendation systems: A survey. ACM Transactions on Multimedia Computing, Communications, and Applications, 16(2), 1-26.

7. Wang, Y., Ma, H., Yang, Y., & Sun, Y. (2021). An ensemble movie recommendation algorithm based on hybrid filtering. Multimedia Tools and Applications, 80(3), 3437-3458.

8. Zhang, M., Chen, Y., & Liu, C. (2018). A matrix factorization method for movie recommendation. Knowledge-Based Systems, 150, 85-95

13 Appendix

The software development lifecycle has resulted in a robust and fully functional software product. The outcomes include a user-friendly interface, efficient data processing, and adherence to specified requirements. The software underwent rigorous testing, ensuring high performance and security. The successful completion of the project demonstrates the team's proficiency and dedication in delivering an exceptional software solution.

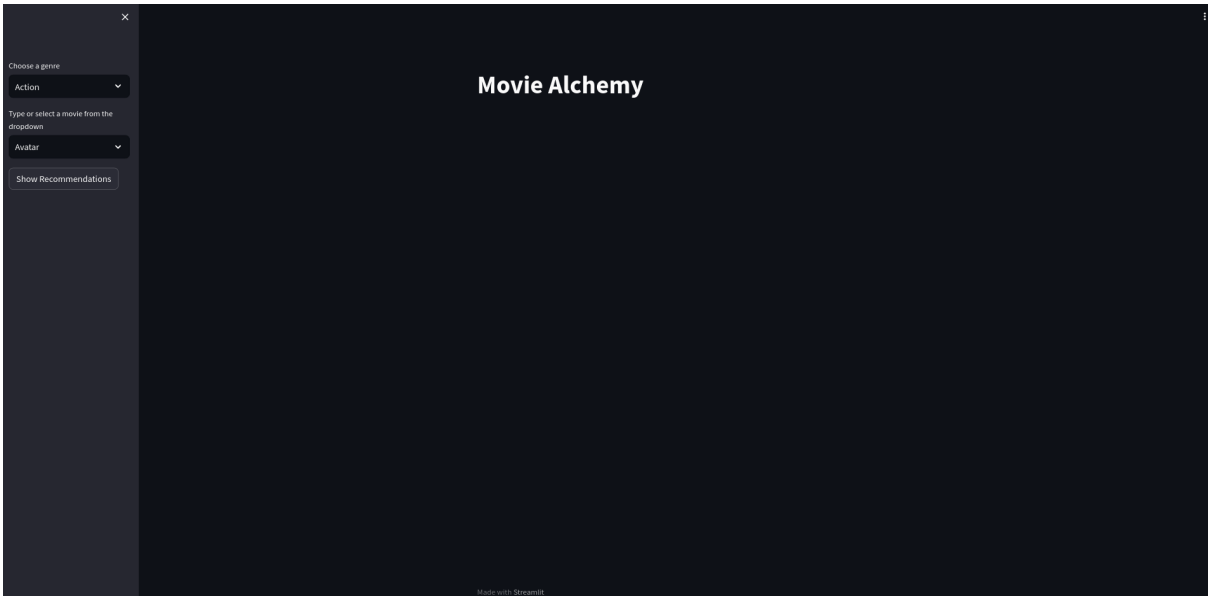

Figure 8 : User Interface

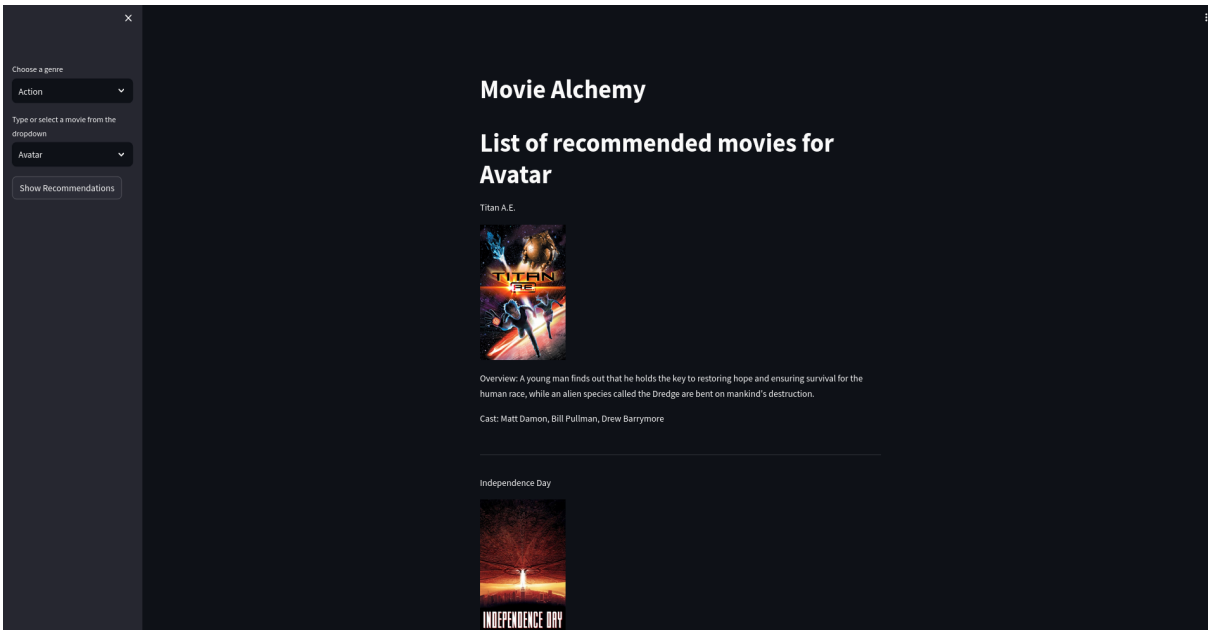

Figure 9 : Output