

University of Mumbai

Movie Recommendation System

Submitted at the end of semester VI in partial fulfillment of requirements

For the degree of

Bachelors in Technology

by

Name: Nakul Chamariya

Roll No: 1813068

Name: Merediya Shabdarali

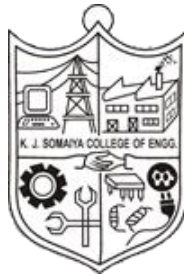
Roll No: 1813101

Name: Smit Mehta

Roll No: 1813104

Guide

Prof. Deepak Kulkarni



Department of Electronics and Telecommunication Engineering

K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Batch 2018 -2022

K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Certificate

This is to certify that the dissertation report entitled **Movie Recommendation System** submitted by **Nakul Chamariya, Marediya Shabdarali** and **Smit Mehta** at the end of semester VI of TY B. Tech is a bonafide record for partial fulfillment of requirements for the degree of Bachelors in Technology in Electronics and Telecommunication Engineering of University of Mumbai



Dipak Kulkarni

Guide/Examiner1

Expert/Examiner2

Date: 18-05-2021

Place: Mumbai-77

K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

Certificate of Approval of Examiners

We certify that this dissertation report entitled **Movie Recommendation System** is bonafide record of project work done by **Nakul Chamariya, Marediya Shabdarali** and **Smit Mehta** during semester VI.

This project work is submitted at the end of semester VI in partial fulfillment of requirements for the degree of Bachelors in Technology in Electronics and Telecommunication Engineering of University of Mumbai.

Guide/Examiner1

Expert/Examiner2

Date: 18-05-2021

Place: Mumbai-77

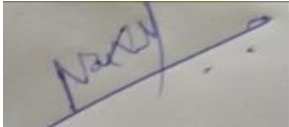
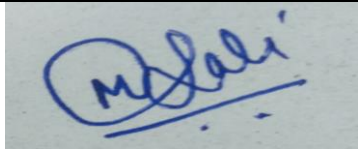

K. J. Somaiya College of Engineering, Mumbai-77

(Autonomous College Affiliated to University of Mumbai)

DECLARATION

We declare that this written report submission represents the work done based on our and / or others' ideas with adequately cited and referenced the original source. We also declare that we have adhered to all principles of intellectual property, academic honesty and integrity as we have not misinterpreted or fabricated or falsified any idea/data/fact/source/original work/ matter in my submission.

We understand that any violation of the above will be cause for disciplinary action by the college and may evoke the penal action from the sources which have not been properly cited or from whom proper permission is not sought.

 _____ Signature of the Student _____ Roll No. 1813068	 _____ Signature of the Student _____ Roll No. 1813101
 _____ Signature of the Student _____ Roll No. 1813104	

Date: 18-05-2021

Place: Mumbai-77

Abstract

The Covid 19 pandemic traditional methods of entertainment were not possible such as cinemas, concerts etc. which sparked an increase in Online Entertainment Industry with sudden surge in number of daily users on OTT platforms. Apps such as Netflix, Hotstar were on high demand. They became an important source of entertainment as they provided vast content and the content that user prefer. Rather than just getting bored people turned towards such sites for enjoyment and relaxation during such times.

OTT platforms provide users with wide variety of content ranging from Movies, TV shows, Exclusive Shows, Reality Shows, Podcasts etc. These services are available for cheaper price than watching movies in theaters or getting cable connections. Due to these advantages the increase in demand of such platforms is seen. This platform uses various algorithms for better functionality of application. One of the main functionalities is recommendation of a movie/series/shows which are preferred by user according to their search.

Movie recommendation system designed by us provides recommendation to the user as per their choice of movie he/she searches which is free of cost. All of data provided about movie and the recommendations is up-to date and recommendations keep on improving as the data increases.

This recommendation system is designed on basis of similarity model and provides user with best possible recommendations.

Contents

1	Introduction	1
1.1	Background	1
1.2	Motivation	2
1.3	Scope of the project	2
1.4	Brief description of project undertaken	3
1.5	Organization of the report	5
2	Literature Survey	6
3	Project design	8
3.1	Introduction	8
3.2	Problem statement	15
3.3	Block diagram / system diagram	15
3.4	Objectives	16
4	Implementation and experimentation	17
4.1	Steps in implementing	17
4.2	Procedure	24
5	Conclusions and scope for further work	25
5.1	Conclusions	25
5.2	Scope for further work	25
	References	27
	Acknowledgements	28

List of Figures

1. Content based filtering	4
2. Collaborative based filtering	4
3. Data Frame	11
4. Vector	11
5. Cosine similarity formula	12
6. Cosine similarity vector representation	12
7. Data Frame	13
8. Vectors	13
9. Vectors in3-	14
10. Calculation of cosine similarity for 3-d vector	14
11. Block diagram	15
12. Home page	17
13. Search bar	18
14. Trending movies	18
15. Movie information	19
16. Actors of the movie	20
17. Directors of the movie	20
18. Writers of the movie	22
19. Recommendations of the movie	23

CHAPTER 1

Introduction

This chapter present about the background of movie recommendation, its relation with machine learning, types of recommendations system used. Further we discussed about motivation and scope of the project. And finally, gave brief description about the project

1.1 Background

A recommendation system is a type of information filtering system which attempts to predict the preferences of a user, and make suggests based on these preferences. There are a wide variety of applications for recommendation systems. These have become increasingly popular over the last few years and are now utilized in most online platforms that we use. The System recommends the same movies to users with similar demographic features. Since each user is different, this approach is considered to be too simple. The basic idea behind this system is that movies that are more popular and critically acclaimed will have a higher probability of being liked by the average audience.

Advancement in technology is reaching new heights every day and due to which we can see enormous growth in information. To deal with such large data we use machine learning that automates analytical model building. The early classification of machine learning is divided into three broad categories: Supervised learning, Unsupervised learning and Reinforcement learning. We use computers to make predictions to help us achieve better results using various computational statistics. Tasks can be performed without being explicitly programmed to do so. It becomes a tedious task to extract the relevant information. Search engines solve the problem to some extent but it does not solve the personalization problem. Recommendation System framework plays a vital role in today's internet surfing, be it buying a product from an e-commerce site or watching a movie on some video-on-demand service. In our everyday life, we depend on recommendations given by other people either by word of mouth or reviews of general surveys. People often use recommender systems over the web to make decisions for the items related to their choice. Recommendation systems are software tools and techniques whose goal is to make useful and sensible recommendations to a collection of users for items or products that might interest them. In other

words, the recommender system or recommendation systems belongs to a class of information filtering system that aims at predicting the preference or rating given to an item. In content-based filtering, we do profiling based on what type of content any user is interested in and using the collected information, it recommends items. Another one is collaborative filtering, where we make clusters of similar users and use that information to make recommendations. Hybrid systems are the one which takes into account both above stated approaches to deal with operational data more concisely. Our goal is to provide accurate recommendations with less computational complexity

1.2 Motivation

The Covid-19 pandemic has led to sudden drift of users on OTT platforms and hours of content consumed increased drastically. So, keep the user's using their services OTT platforms had to recommend new content to the users but any then problem arises which content will make the user to continue so recommendation systems play's major role in recommending right content for user.

The topic we have chosen for mini project is: Movie Recommendation System. This project gave us a great opportunity to learn and implement concepts which helps OTT platforms such a great success.

1.3 Scope of the Project

A recommender system, or a recommendation system (sometimes replacing 'system' with a synonym such as platform or engine), is a subclass of information filtering system that seeks to predict the "rating" or "preference" a user would give to an item.

Recommender systems are used in a variety of areas, with commonly recognized examples taking the form of playlist generators for video and music services, product recommenders for online stores, or content recommenders for social media platforms and open web content recommenders. These systems can operate using a single input, like music, or multiple inputs within and across platforms like news, books, and search queries. There are also popular recommender systems for specific topics like restaurants and online dating. Recommender systems have also been developed to explore research articles and experts, collaborators, and financial services.

Recommender systems can be a very powerful tool in a company's arsenal, and future developments are going to increase business value even further. Some of the applications include being able to anticipate seasonal purchases based on recommendations, determine important

purchases, and give better recommendations to customers which can increase retention and brand loyalty.

1.4 Brief description of project undertaken

Well, here we are, using the Cosine Similarity (the dot product for normalized vectors) to build a Movie Recommend System! Recommendation Systems work based on the similarity between either the content or the users who access the content. There are several ways to measure the similarity between two items. We used Similarity Matrix to recommend the next most similar product to the user. Here, we will build a machine learning algorithm that would recommend movies based on a movie the user likes. This Machine Learning model would be based on Cosine Similarity which uses Collaborative filtering or content-based filtering. This model combined with flask framework will work as a Movie Recommendation website.[3]

Approach:

1. Collaborative filtering-based systems:

Our content-based engine suffers from some severe limitations. It is only capable of suggesting movies which are close to a certain movie. That is, it is not capable of capturing tastes and providing recommendations across genres. Also, the engine that we built is not really personal in that it doesn't capture the personal tastes and biases of a user. Anyone querying our engine for recommendations based on a movie will receive the same recommendations for that movie, regardless of who she/he is. Therefore, in this section, we will use a technique called Collaborative Filtering to make recommendations to Movie Watchers. It is basically of two types: -

- **User based filtering-** These systems recommend products to a user that similar users have liked. For measuring the similarity between two users we can either use 16 pearson correlation or cosine similarity. Although computing user-based CF is very simple, it suffers from several problems. One main issue is that users' preference can change over time. It indicates that precomputing the matrix based on their neighboring users may lead to bad performance. To tackle this problem, we can apply item-based CF.

- Item Based Collaborative Filtering - Instead of measuring the similarity between users, the item-based CF recommends items based on their similarity with the items that the target user rated. Likewise, the similarity can be computed with Pearson Correlation or Cosine Similarity. The major difference is that, with item-based collaborative filtering, we fill in the blank vertically, as oppose to the horizontal manner that user-based CF does. [3]



Fig-1 : Collaborative filtering[2]

2. Content-based Filtering Systems:

In content-based filtering, items are recommended based on comparisons between item profile and user profile. A user profile is content that is found to be relevant to the user in form of keywords (or features). A user profile might be seen as a set of assigned keywords (terms, features) collected by algorithm from items found relevant (or interesting) by the user. A set of keywords (or features) of an item is the Item profile. For example, consider a scenario in which a person goes to buy his favorite cake 'X' to a pastry. Unfortunately, cake 'X' has been sold out and as a result of this the shopkeeper recommends the person to buy cake 'Y' which is made up of ingredients similar to cake 'X'. This is an instance of content-based filtering.[3]

CONTENT-BASED FILTERING

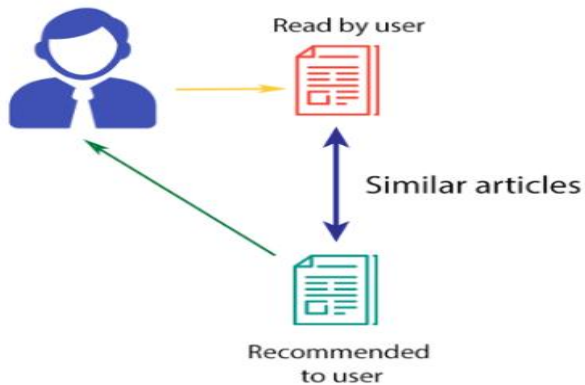


Fig-2: Content-based filtering[2]

Technical Requirements:

The web app can only run-on web browsers supporting HTML5.

Compatibility:

The platform should be compatible with multiple browsers including:

- Chrome version 87.0.4280.141
- Firefox version 84
- Safari version 11.1.2
- Internet Explorer 11.0

Availability:

System must be available for user when ever they want recommendation for their movies.

As it requires less maintenance its easy to make sure it's up and running all the time.

Performance:

- The system should be lag free and fast for best user experience.
- Should be easy to understand and use to make it more user-friendly.

1.5 Organization of the project

We elaborated till now all the essential parts of our system, its scope, motivation, requirements, compatibility and performance of the project.

In the follow up part we have discussed the background details regarding the program that were going on continuously throughout the course of our project. We discuss about the technologies, programming languages that we have used to create the Web Application. Nonetheless we also discuss about the methodology to our approach, algorithms that we have used to make the Web Application more robust. We have presented the implementation of various web pages. We later on discuss about our results and conclude with our learning and describe the scope for future work. We also added few references.

CHAPTER 2

Literature Survey

This chapter presents about requirements or skills you need to build the movie recommendation system using python framework

MOVREC is a movie recommendation system presented by D.K. Yadav et al. based on collaborative filtering approach. Collaborative filtering makes use of information provided by user. That information is analyzed and a movie is recommended to the users which are arranged with the movie with highest rating first. Luis M Capos et al has analyzed two traditional recommender systems i.e., content based filtering and collaborative filtering. As both of them have their own drawbacks he proposed a new system which is a combination of Bayesian network and collaborative filtering. A hybrid system has been presented by Harpreet Kaur et al. The system uses a mix of content as well as collaborative filtering algorithm. The context of the movies is also considered while recommending. The user - user relationship as well as user - item relationship plays a role in the recommendation.[4]

For building the web application using the machine learning model based on python we need to learn about Flask and jinja

1. Flask and its applications

Flask is a micro-framework designed to create a web application in a short time. It only implements the core functionality giving developers the flexibility to add the feature as required during the implementation. It is a lightweight, WSGI application framework. This framework can either be used for pure backend as well as frontend if need be. The former provides the functionality of the interactive debugger, full request object, routing system for endpoints, HTTP utilities for handling entity tags, cache controls, dates, cookies etc.

2. Jinja and how to use it to render data on html page?

The Jinja, however, is another dependency of the Flask. It is a full-featured template engine. Sandboxed execution, powerful XSS prevention, template inheritance, easy to do debug, configurable syntax is it's few of many features. In addition, the code written in the HTML template is compiled as python code

3. How an TMDB API works?

API is available for everyone to use. A TMDB user account is required to request an API key. Professional users are approved on a per application basis. As always, you must attribute TMDB as

the source of your data. API service is for those of you interested in using our movie, TV show or actor images and/or data in your application. Our API is a system we provide for you and your team to programmatically fetch and use our data and/or images.

Documentation of TMDB API - developers.themoviedb.org

CHAPTER 3

Project Design

This chapter presents in brief the tools required to build the movie recommendation system. It also emphasize on how to use cosine similarity to recommend movies to the user and flow of the project in form of block diagram

3.1 Introduction

Technologies Used –

Html:

The Hyper Text Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items.[5]

CSS:

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .CSS file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the

page load speed between the pages that share the file and its formatting.[6]

Python:

Python is an interpreted, high-level and general-purpose programming language. Python's design philosophy emphasizes code readability with its notable use of significant indentation. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects. Python is dynamically typed, and garbage collected. It supports multiple paradigms, including structured (particularly, procedural), object-oriented and functional programming. Python is often described as a "batteries included" language due to its comprehensive standard library. In our project we have used python for our recommendation system based on cosine similarity.[7]

Flask:

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.[8]

How to get the Dataset?

The first step to build a movie recommendation system is getting the appropriate data. You may download the movies dataset from the web, or from the link below which contains a 22MB CSV file titled “**movie_dataset.csv**”

CSV file contains a total of **4802 movies** and **24 columns**: index, budget, genres, homepage, id, keywords, original_language, original_title, overview, popularity, production_companies, production_countries, release_date, revenue, runtime, spoken_languages, status, tagline, title,

vote_average, vote_count, cast, crew and director

Among all these different features, the ones we are interested in to find the similarity for making the next recommendation are the following:

- 1)genre
- 2)cast
- 3)plot
- 4)year of release

A user who likes a horror movie will most probably like another horror movie. Some users may like seeing their favourite actors in the cast of the movie. Others may love movies directed by a particular person. Combining all of these aspects, our shortlisted 4 features are sufficient to train our recommendation algorithm.

2)Converting text to vectors

We will use count vectorizer to convert genre, plot, cast to numerical vectors

The Count Vectorizer provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary.

You can use it as follows:

1. Create an instance of the Count Vectorizer class.
2. Call the fit() function in order to learn a vocabulary from one or more documents.
3. Call the transform() function on one or more documents as needed to encode each as a vector.

An encoded vector is returned with a length of the entire vocabulary and an integer count for the number of times each word appeared in the document.

Because these vectors will contain a lot of zeros, we call them sparse. Python provides an efficient way of handling sparse vectors in the scipy . sparse package.

The vectors returned from a call to transform() will be sparse vectors, and you can transform

them back to numpy arrays to look and better understand what is going on by calling the `toarray()` function.

3) Compute cosine similarities between all words

To compute similarity between two movies we use cosine similarity function in sklearn

How cosine similarity works for building Recommenders?

Cosine similarity is a method to measure the difference between two non zero vectors of an inner product space. See the example below to understand.

Suppose I want to check if Bernard and Clarissa have similar movie preferences, and I only have two movie reviews. The reviews are scores from 1 to 5, where 5 is the best score and 1 the worst, and 0 means that a person has not watched the movie.[1]

Name	Iron Man (2008)	Pride & Prejudice (2005)
Bernard	4	3
Clarissa	5	5

Fig-3: DataFrame [1]

I can represent each person's reviews in a separate vector.

$$\vec{b} = \begin{bmatrix} 4 \\ 3 \end{bmatrix} \quad \vec{c} = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

Fig-4: Vectors[1]

Vector b represents Bernard and vector c Clarissa.

The cosine similarity will measure the similarity between these two vectors which is a measurement of how similar are the preferences between these two people.[1]

$$\text{similarity} = \cos \theta = \frac{b \cdot c}{\|b\| \|c\|}$$

$b \cdot c \Rightarrow$ Is the Dot product of the two vectors

$\|b\| \|c\| \Rightarrow$ Is the product of each vector's magnitude

Fig-5: Cosine similarity formula[1]

In the image, below each vector represents a person's preferences and they have an angle θ between them. Similar vectors will have a lower angle θ , and dissimilar vectors (different film preferences) will have bigger θ . [1]

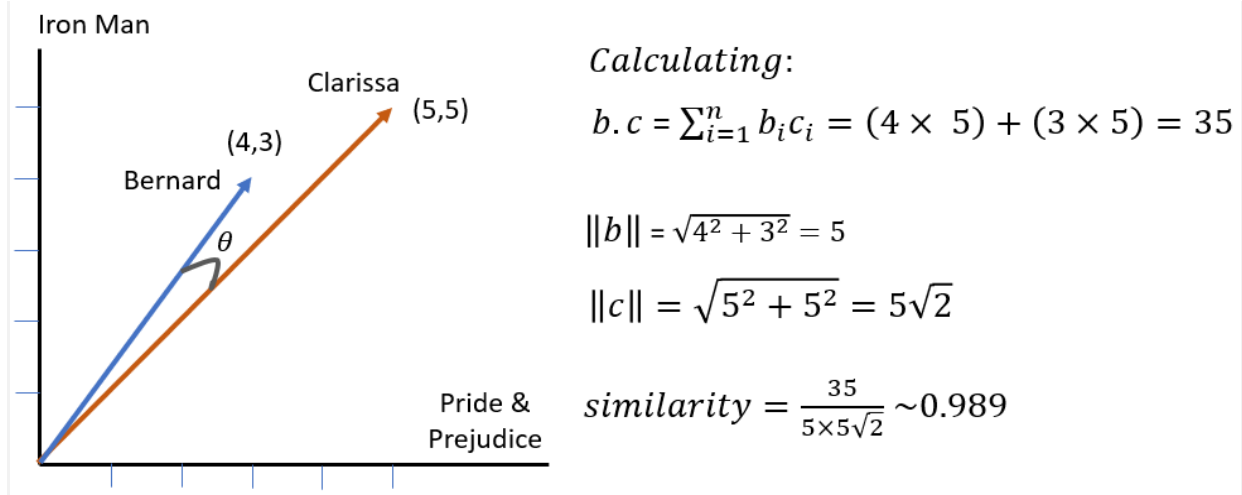


Fig-6: Cosine similarity vector representation[1]

In the example above the similarity 0.989 is close to the maximum value of 1, this means that given only two movie reviews the two users have similar preferences.

Theoretically, the cosine similarity can be any number between -1 and +1 because of the image of the cosine function, but in this case, there will not be any negative movie rating so the angle θ will be between 0° and 90° bounding the cosine similarity between 0 and 1. If the angle $\theta = 0^\circ \Rightarrow$ cosine similarity = 1, if $\theta = 90^\circ \Rightarrow$ cosine similarity = 0. [1]

The cosine similarity can be calculated for more than 2 movies. In the example below I will add the ratings for a movie that Bernard liked and Clarissa disliked this should decrease cosine similarity value.[1]

Name	Iron Man	Pride & Prejudice	DitRICT 9
Bernard	4	3	5
Clarissa	5	5	1

Fig-7: DataFrame[1]

The new vectors are:

$$\vec{b} = \begin{bmatrix} 4 \\ 3 \\ 5 \end{bmatrix} \quad \vec{c} = \begin{bmatrix} 5 \\ 5 \\ 1 \end{bmatrix}$$

Fig-8 : Vectors[1]

Vector b represents Bernard and vector c Clarissa.

The plot has 3 dimensions now:

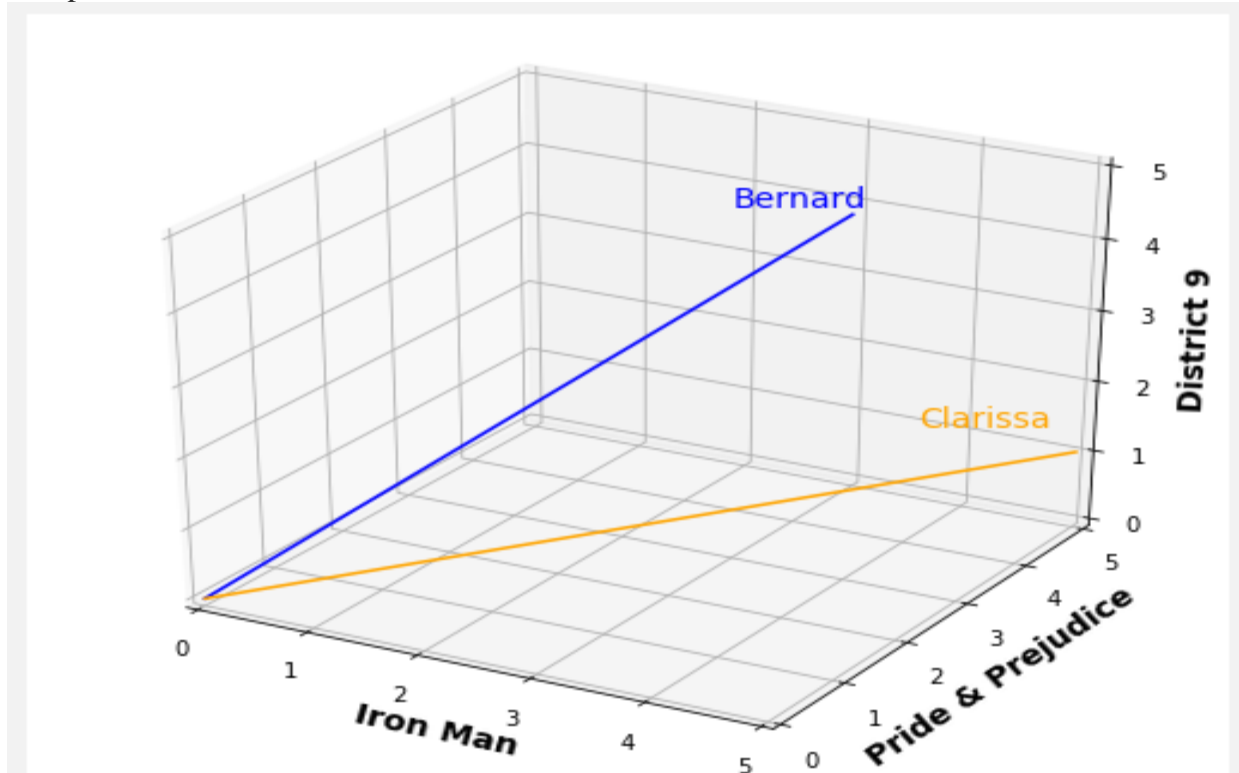


Fig-9 : Vectors in 3-d[1]

Calculating the similarity:

$$b \cdot c = (4 \times 5) + (3 \times 5) + (5 \times 1) = 40$$

$$\|b\| = \sqrt{4^2 + 3^2 + 5^2} = 5\sqrt{2}$$

$$\|c\| = \sqrt{5^2 + 5^2 + 1^2} = \sqrt{51}$$

$$\text{similarity} = \frac{40}{5\sqrt{2} \times \sqrt{51}} \sim 0.792$$

Fig-10 : Calculation of cosine similarity for 3-d vector[1]

The similarity has reduced from 0.989 to 0.792 due to the difference in ratings of the District 9 movie. The cosine can also be calculated in Python using the Sklearn library.[1]

4)Getting the recommendation of the movies

Now since we have similarities of movie we can show user top 10 movies which are similar to the movie

3.2 Problem Statement

Given the movie by user recommend top ten movies to user based on genre, plot and cast similarity.

Also show information about the movie to user like movie thumbnail, plot, genre, top cast, top writers/directors, etc.

3.3 Block diagram / system diagram

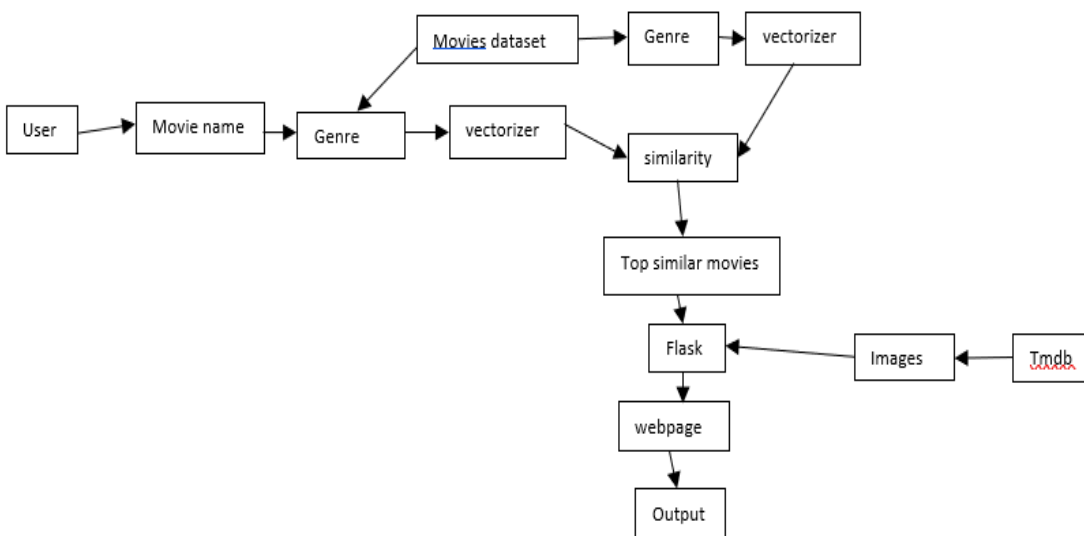


Fig-11 : Block diagram

3.5 Objective

The **objective of recommender systems** is to provide **recommendations** based on recorded information on the users' preferences. These **systems** use information filtering techniques to process information and provide the user with potentially more relevant items.

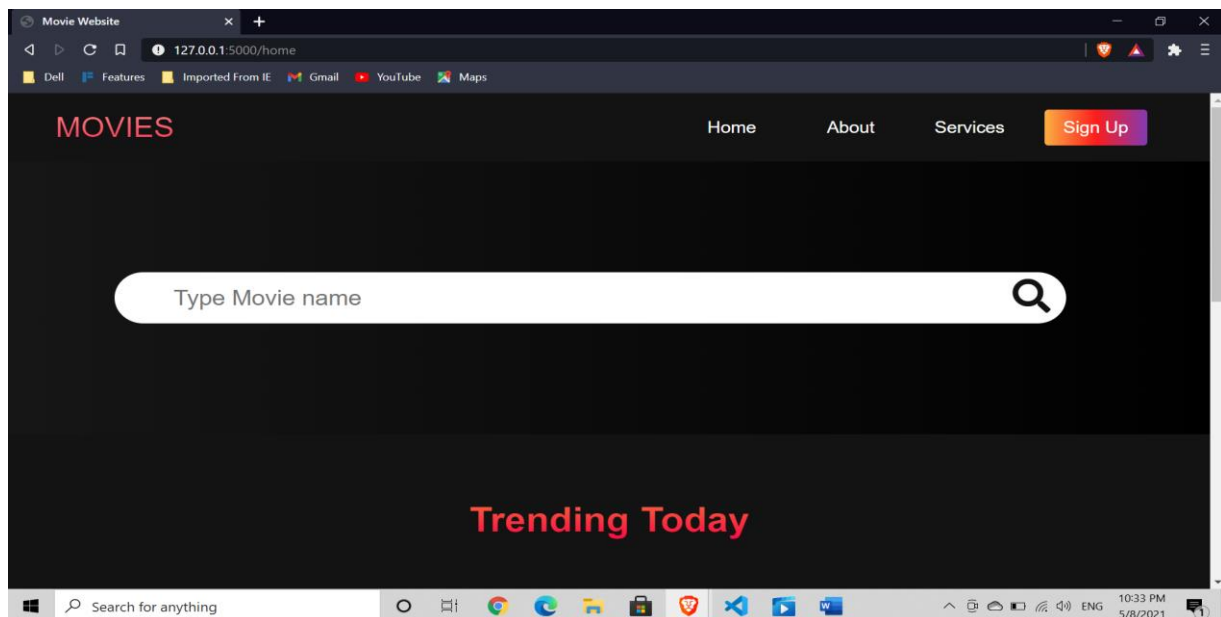
CHAPTER 4

Implementation and Experimentation

This chapter presents how to use the website of movie recommendation and what are the features user will get along with movie recommendation

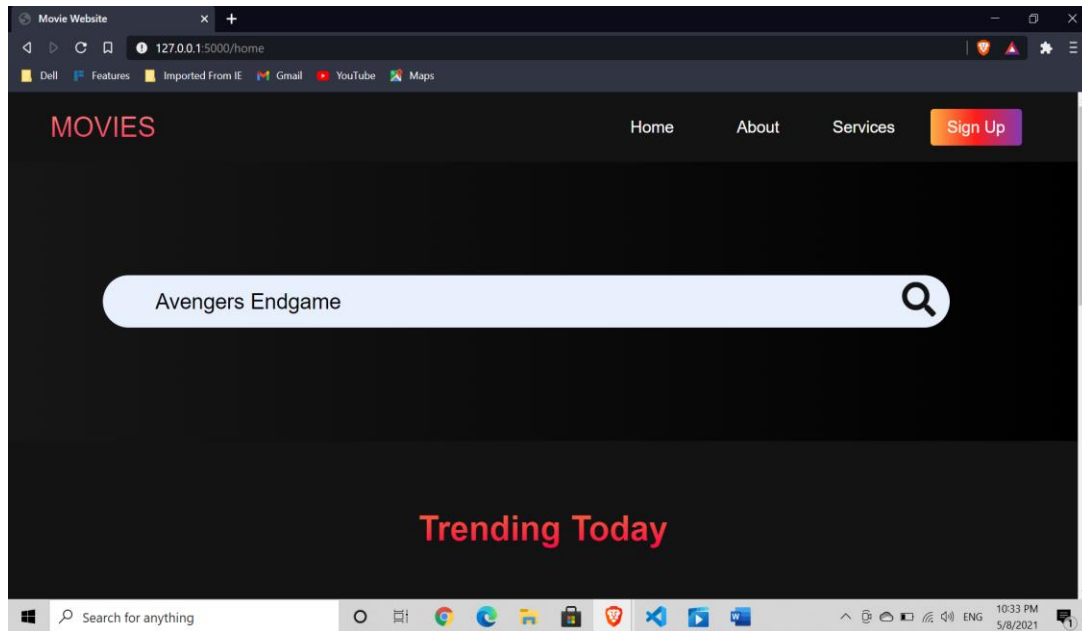
4.1 Steps in Implementing

Fig-12 : Home Page



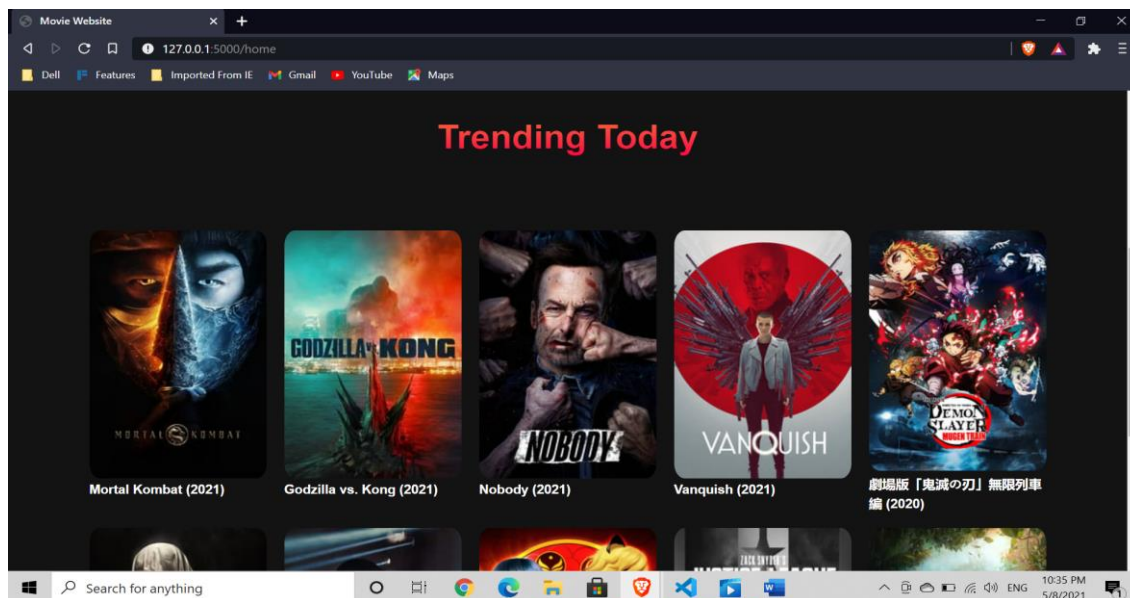
The home page of our Web application with Navbar, Searchbar and Trending today list of movies.

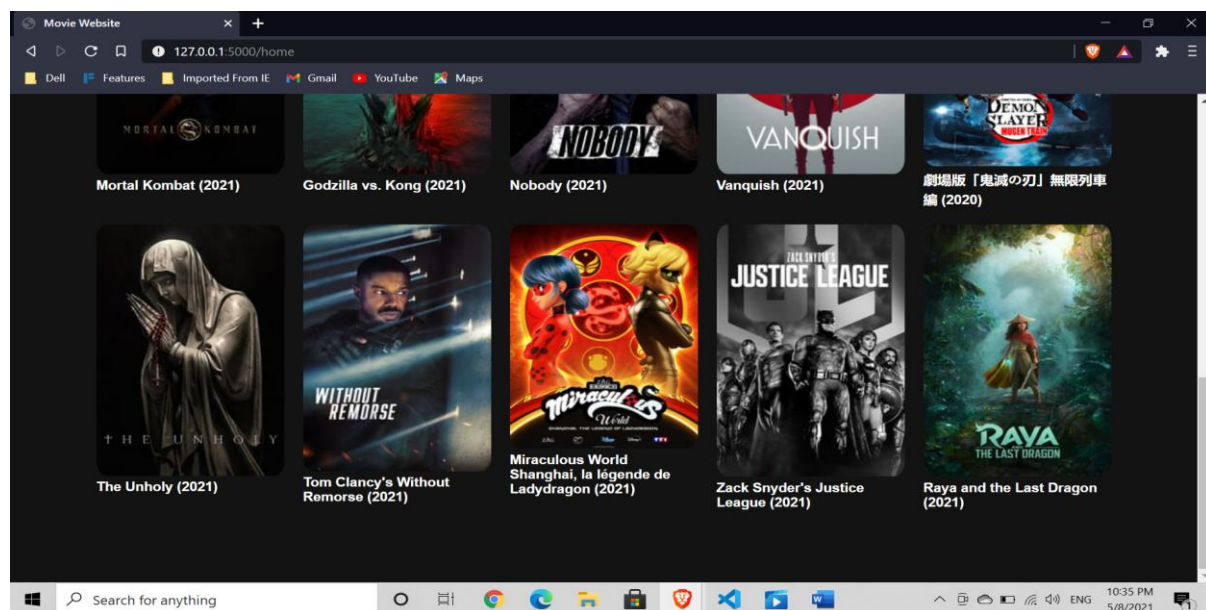
Fig-13 : Search bar



Search bar of application takes the name of the movie for which recommendations are expected.

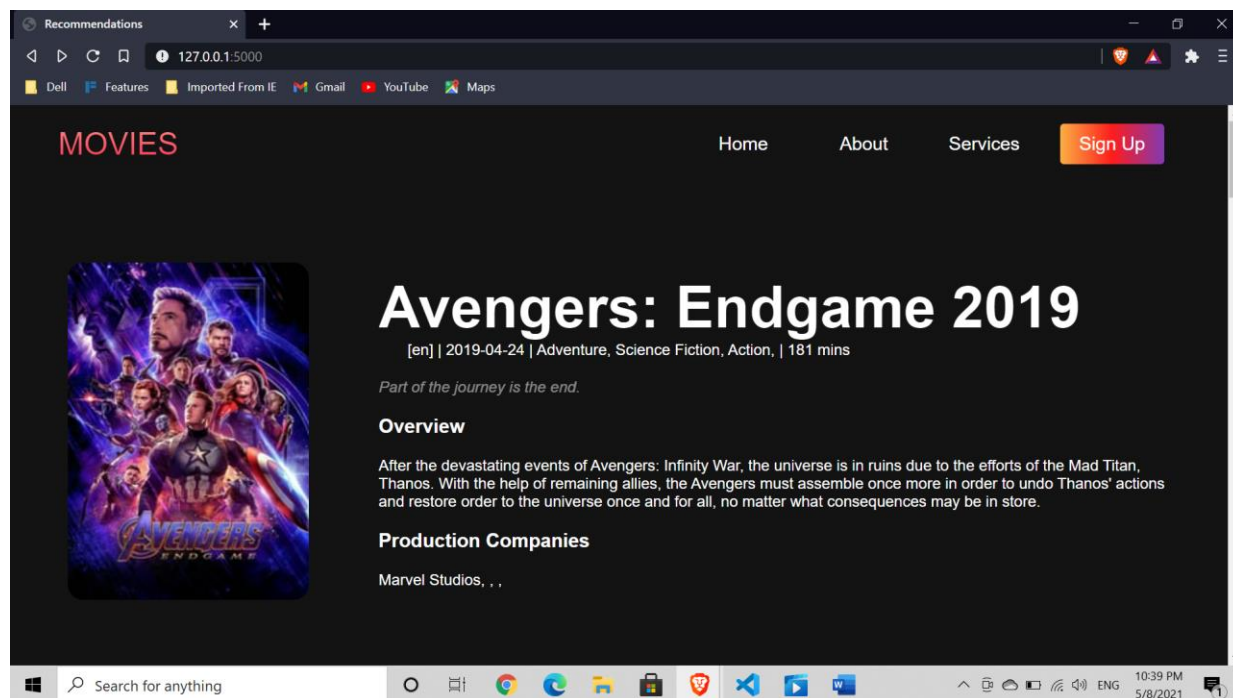
Fig-14 : Trending Today





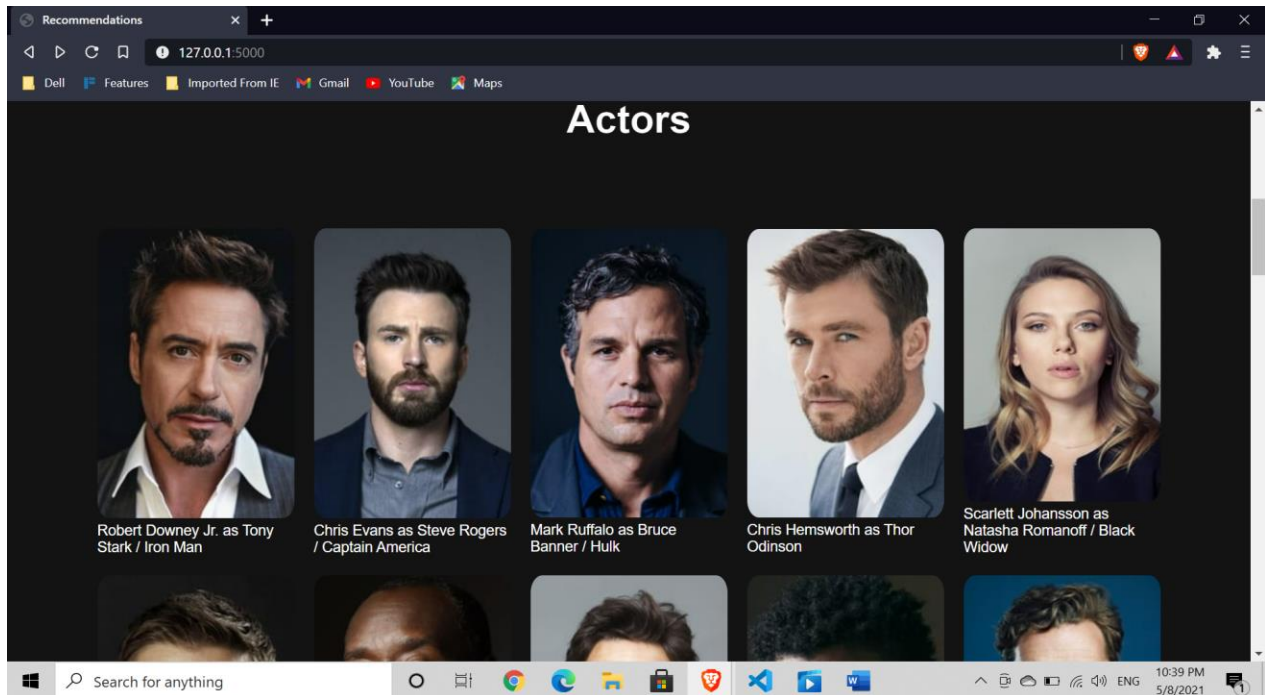
Trending today gives the list of movies which trending for the day. List keeps on changing as per trending movies.

Fig-14 : Movie Information



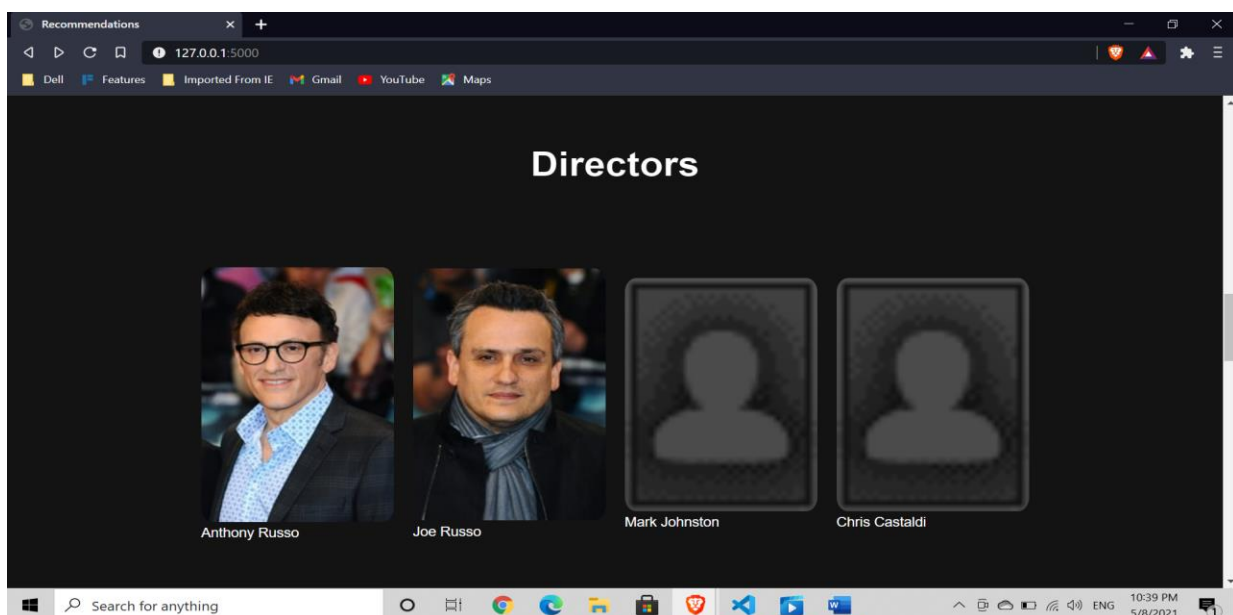
This page gives the information of the movie typed in the search bar.

Fig-16: Actors



This is the list of top 10 actors associated with the movie.

Fig-17 : Directors



List of top 4 directors associated with the movie.

Blank images are shown because no data writes exist but images are not available. If only one director is listed then blanks don't show-up. Eg- Mortal Kombat.

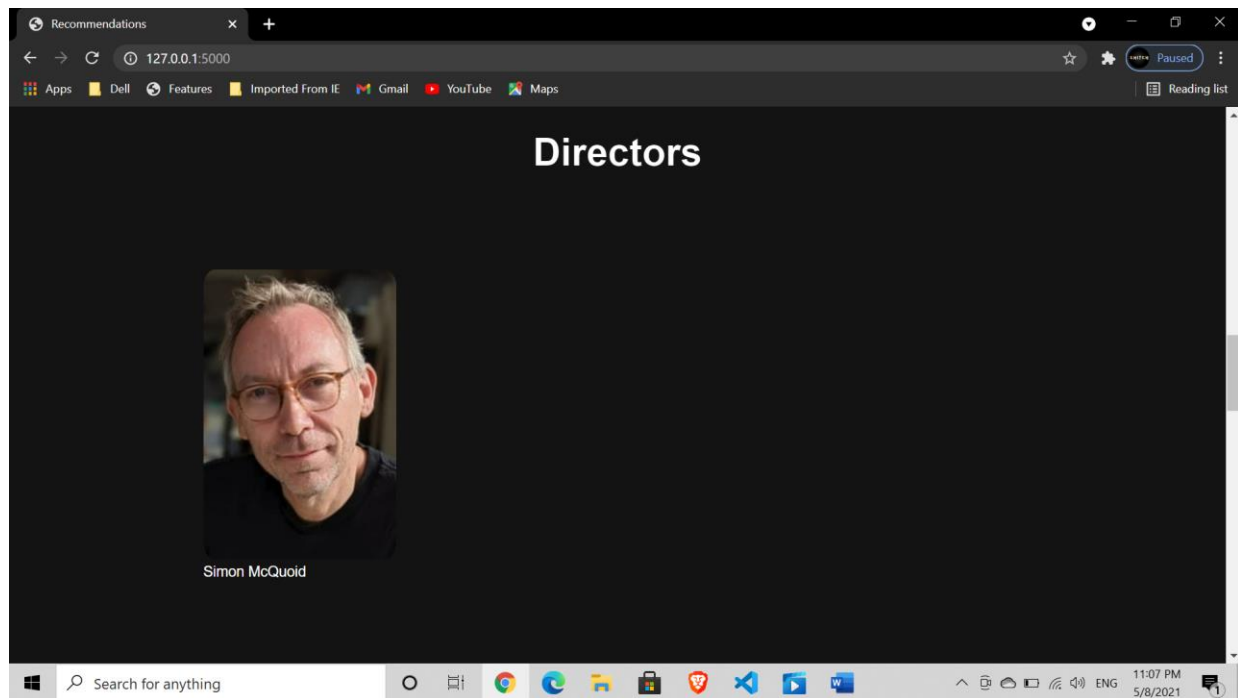
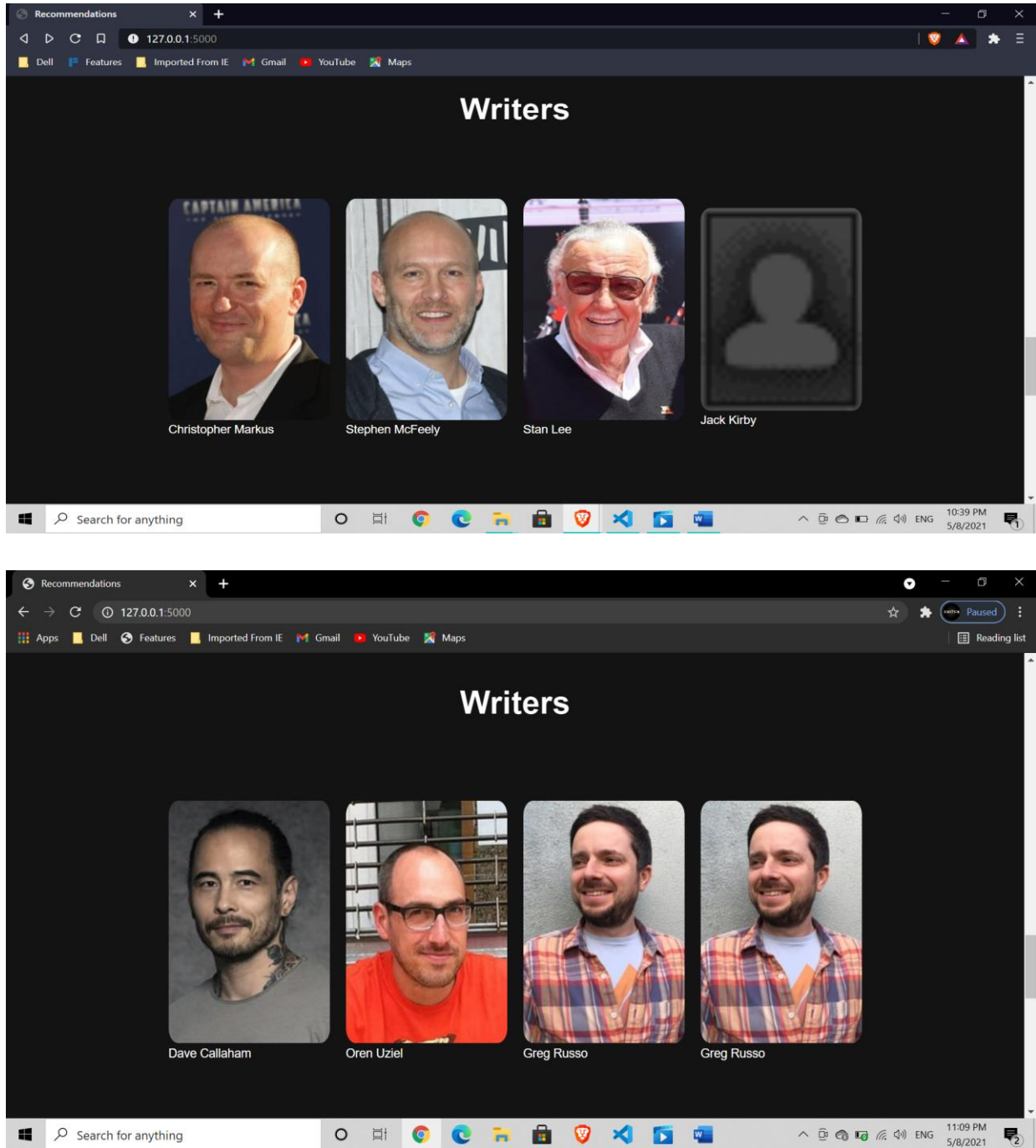


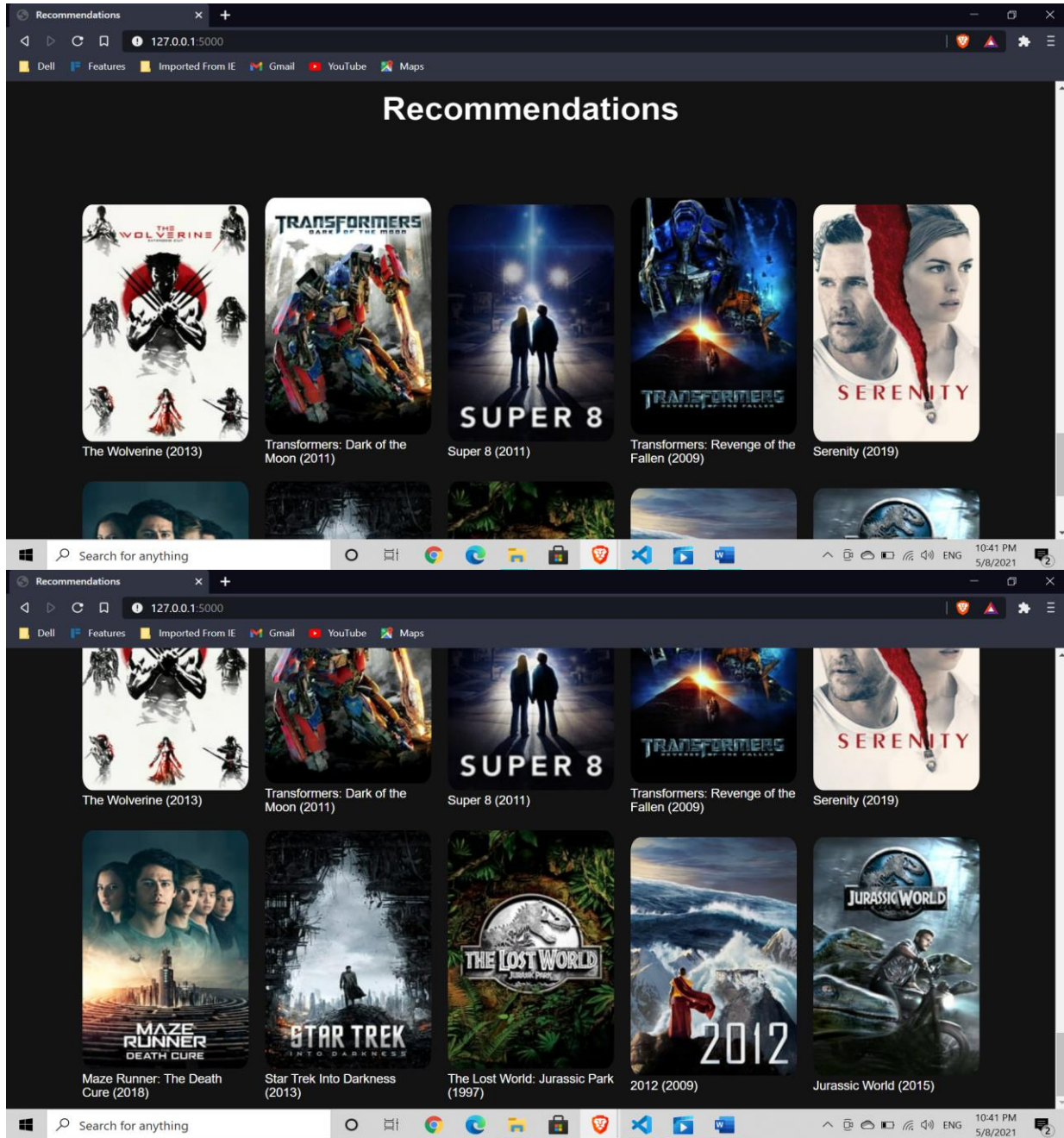
Fig-18: Writers



List of top 4 Writers associated with the movie.

Same as director's blanks only show if images are not present in data. But if name of same director is repeated then shows as per below image. Eg- Mortal Kombat.

Fig-19: Recommendations



This the recommendation page where top 10 recommendations of the typed movie will be shown.

4.2 Procedure

- User goes on to our website.
- User can see the trending movies on home page if they want to watch a trending movie.
- User can type the name of the movie they want to search for recommendations and press enter or click on search button.
- User is redirected to the recommendations page.
- There they can find information of their movie they searched for.
- Scrolling below will give more information such as Actors, Directors and Writers of the searched movie.
- At the end the recommendations for the searched movie are shown.

CHAPTER 5

Conclusion and Scope of future work

This chapter leads to final conclusion and scope of the movie recommendation to our future along with reference and acknowledgement

5.1 Conclusions

The Movie Recommendation Application was successfully implemented which allowed us to explore and integrate various technologies.

1. Managing and simplifying the data from API and make a proper use of it.
2. Integrating machine learning in web application via the use of Flask framework.
3. Gain a deeper understanding of web development techniques.
4. Learnt about the algorithms used in OTT platforms or any recommendations platform.

5.2 Scope for further work

Recommendation systems has a great future as the Online media industry is going to increase. The Applications which have best recommendation systems will provide best experience to user which will help in further growth. Movie Recommendation System will play a major role as a fierce competition between apps like Netflix, Hotstar, Hulu etc.

In present we don't have perfect recommendation systems we are faced by various problems such as Cosine similarity calculation do not work well when we don't have enough rating for movie or when user's rating for some movie is exceptionally either high or low. As an improvement on this project some other methods such as adjusted cosine similarity can be used to compute similarity.

Adjusted cosine similarity, which is similar to cosine similarity, is measured by normalizing the user vectors U_x and U_y and computing the cosine of the angle between them. However, unlike cosine

similarity, when computing the dot product of the two user vectors, adjusted cosine similarity uses the deviation between each of the user's item ratings, denoted R_{ui} , and their average item rating, denoted \bar{R}_u , in place of the user's raw item rating. The main advantage of this approach is that in item-based collaborative filtering, the item vectors consist of ratings from different users who often have varying rating scales.

In future recommendation systems will keep on improving and providing users with best recommendations.

References

- [1] <https://medium.com/@bkexcel2014/building-movie-recommender-systems-using-cosine-similarity-in-python-eff2d4e60d24>
- [2] <https://www.analyticsvidhya.com/blog/2020/11/create-your-own-movie-movie-recommendation-system/>
- [3] <https://www.raco.cat/index.php/ELCVIA/article/download/373942/467477/>
- [4] https://easychair.org/publications/preprint_download/hM2Z
- [5] <https://www.w3schools.com/html/>
- [6] <https://www.w3schools.com/css/>
- [7] <https://www.w3schools.com/python/>
- [8] <https://flask.palletsprojects.com/en/2.0.x/>
- [9] <https://youtu.be/3mwFC4SHY-Y>
- [10] <https://medium.com/bhavaniravi/build-your-1st-python-web-app-with-flask-b039d11f101c>
- [11] <https://programminghistorian.org/en/lessons/creating-apis-with-python-and-flask>
- [12] <https://jinja.palletsprojects.com/en/3.0.x/>

Acknowledgements

The completion of this undertaking couldn't have been possible without the participation and assistance of numerous individuals. The group members would like to express their sincere appreciation to Professor Dipak Kulkarni, from the **Electronics & Telecommunication Department**. We would also like to extend our gratitude towards K. J. Somaiya College of Engineering.