LUCKNOW PUBLIC COLLEGE OF PROFESSIONAL STUDIES



SUMMER TRAINING PROJECT REPORT

ON

“Movie recommended search system”

OF

## BACHELOR OF COMPUTER APPLICATION

FROM

UNIVERSITY OF LUCKNOW

lucknow

DEPARTMENT OF COMPUTER APPLICATION

Submitted by : Submitted TO :

tanishq gupta Dr. Karuna Shnakar Awasthi

(ROLL NO. 200820040088)

For the Academic Session : 2022-2023

##### INDEX

1. Project Model

2. Introducation

2.1 relevance of the project

2.2 Problem Statement

2.3 Objective of the Projects

2.4 Scope of the Project

2.5 Methodology for Movie Recommendation

3. Methodology

4. System Requirements Specification

4.1 Hardware Requirements

4.2 Software Requirements

4.Feasibility Report

4.1 TECHNICAL FEASIBILITY

4.**2** DELIVERABLES

4.3 SOFTWARE MODEL

4.4 FINANCIAL FEASIBILITY

4.5 TECHNICAL FEASIBILITY TOOLS

4.6RESOURCE & PERFORMANCE FEASIBILITY

4.7 RISK FEASIBILITY

4.8 SOCIAL/LEGAL FEASIBILITY

4.9 OUTLINE PLAN

5. USER MODULE DESCRIPTION

6. RESULTS AND DISCUSSION

6.1 SCREEN SHOT

7. DATA PREPROCESSING

7.1 PREPROCESSING INFORMATION

7.2 PREPROCESSING CODE

8. IMPLEMENTATION

8.1 CODE INFORMATION

9. SOFTWARE INTERFACE DESIGN

10. SYSTEM ANALYSIS AND DESIGN

10.1 DATA FLOW DAIGRAM(DFD)

11. TESTING

11.1 UNIT TESTING

11.2 INTEGRATION TESTING

11.3 SYSTEM TESTING

12. PLANNING AND SCHEDULING

12.1 PERT CHART

12.2 GANTT CHART

13. CONCLUSION AND FUTRURE SCOPE

13.1 FUTURE SCOPE

13.2 CONCLUSION

14. REFERENCES

## 

**ACKNOWLEDGEMENT**

The completion of any task is the reward of not only the persons actively involved in accomplishing it, but also the people involved in inspiring, guiding and helping these people. We take the opportunity here to thank all those people who have helped us in this project. Yes this project wouldn’t have been possible without the unrelenting care and support of many people.

First and foremost, I would like to express my deepest gratitude to Dr. Karuna Shnakar Awasthi for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project.

I would like to express my gratitude towards my parents and members of **LPCPS** for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to industry persons for giving me such attention and time. My thanks and appreciation also go to my colleagues in developing this project and people who have willingly helped me out with their abilities.

Above all, I thank the almighty God, for giving me the immense strength and wisdom to be dedicated in my work and give my best.

THANK YOU.

## CERTIFICATE OF ORIGINALITY



## This is To Certify That The Project Report Entitled

Submitted To Lucknow Public College Of Professional Studies Affliated To The University Of Lucknow In Partial Fulfillment Of Requirement For The Award Of The Degree Of Bachelor Of Computer Application (BCA) , Is An Authentic And Original Work Carried Out By

Miss , \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

With Enrolment No. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Under My Guidance .The Matter Embodied In This Project Is A Genuine Work Done By The Student And Has Not Been Submitted To The University Or To Any University / Institute For The Fulfillment Of Any Course Of Study

## 

Signature of the Student Signature of Guide

Date



## ABSTRACT

## In this hustling world, entertainment is a necessity for each one of us to refresh our mood and energy. Entertainment regains our confidence for work and we can work more enthusiastically. For revitalizing ourselves, we can listen to our preferred music or can watch movies of our choice. For watching favourable movies online we can utilize movie recommendation systems, which are more reliable, since searching of preferred movies will require more and more time which one cannot afford to waste. In this paper, to improve the quality of a movie recommendation system, a Hybrid approach by combining content based filtering and collaborative filtering, using Support Vector Machine as a classifier and genetic algorithm is presented in the proposed methodology and comparative results have been shown which depicts that the proposed approach shows an improvement in the accuracy, quality and scalability of the movie recommendation system than the pure approaches in three different datasets. Hybrid approach helps to get the advantages from both the approaches as well as tries to eliminate the drawbacks of both methods.

## 

## MOVIE RECOMMEND SEARCH SYSTEM

### PROJECT MODEL

# **Iterative Model**



In this Model, you can start with some of the software specifications and develop the first version of the software. After the first version if there is a need to change the software, then a new version of the software is created with a new iteration. Every release of the Iterative Model finishes in an exact and fixed period that is called iteration.

The Iterative Model allows the accessing earlier phases, in which the variations made respectively. The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

### The various phases of Iterative model are as follows:

**1. Requirement gathering & analysis:** In this phase, requirements are gathered from customers and check by an analyst whether requirements will fulfil or not. Analyst checks that need will achieve within budget or not. After all of this, the software team skips to the next phase.

**2. Design:** In the design phase, team design the software by the different diagrams like Data Flow diagram, activity diagram, class diagram, state transition diagram, etc.

**3. Implementation:** In the implementation, requirements are written in the coding language and transformed into computer programmes which are called Software.

**4. Testing:** After completing the coding phase, software testing starts using different test methods. There are many test methods, but the most common are white box, black box, and grey box test methods.

**5. Deployment:** After completing all the phases, software is deployed to its work environment.

**6. Review:** In this phase, after the product deployment, review phase is performed to check the behaviour and validity of the developed product. And if there are any error found then the process starts again from the requirement gathering.

**7. Maintenance:** In the maintenance phase, after deployment of the software in the working environment there may be some bugs, some errors or new updates are required. Maintenance involves debugging and new addition options.

## **Iterative Model - Pros and Cons**

The advantage of this model is that there is a working model of the system at a very early stage of development, which makes it easier to find functional or design flaws. Finding issues at an early stage of development enables to take corrective measures in a limited budget.

The disadvantage with this SDLC model is that it is applicable only to large and bulky software development projects. This is because it is hard to break a small software system into further small serviceable increments/modules.

The advantages of the Iterative and Incremental SDLC Model are as follows −

* Some working functionality can be developed quickly and early in the life cycle.
* Results are obtained early and periodically.
* Parallel development can be planned.
* Progress can be measured.
* Less costly to change the scope/requirements.
* Testing and debugging during smaller iteration is easy.
* Risks are identified and resolved during iteration; and each iteration is an easily managed milestone.
* Easier to manage risk - High risk part is done first.
* With every increment, operational product is delivered.
* Issues, challenges and risks identified from each increment can be utilized/applied to the next increment.
* Risk analysis is better.
* It supports changing requirements.
* Initial Operating time is less.
* Better suited for large and mission-critical projects.
* During the life cycle, software is produced early which facilitates customer evaluation and feedback.

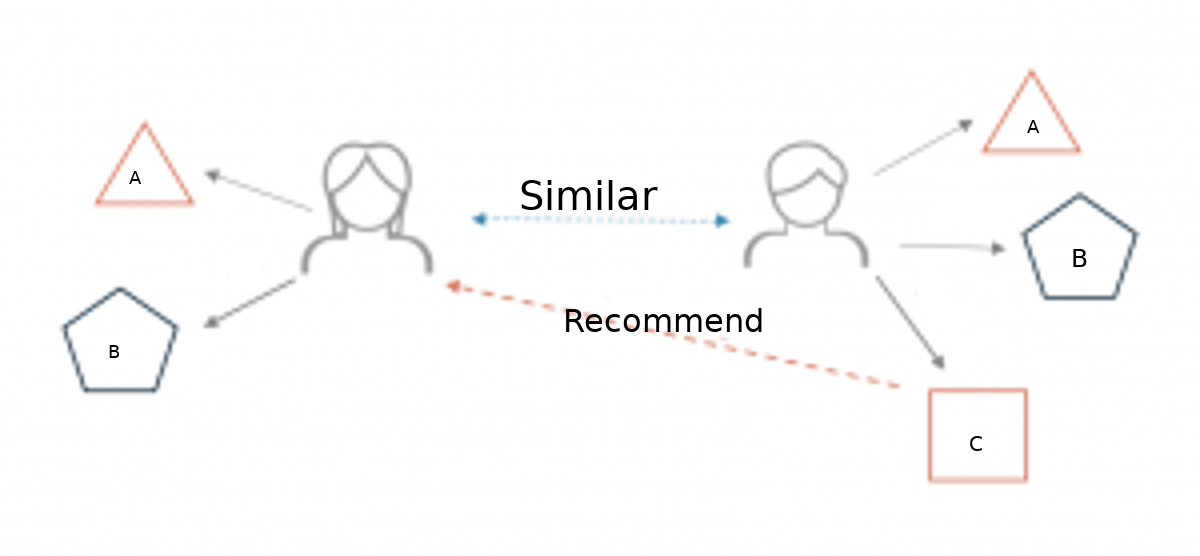
**The disadvantages of the Iterative and Incremental SDLC Model are as follows −**

* More resources may be required.
* Although cost of change is lesser, but it is not very suitable for changing requirements.
* More management attention is required.
* System architecture or design issues may arise because not all requirements are gathered in the beginning of the entire life cycle.
* Defining increments may require definition of the complete system.
* Not suitable for smaller projects.
* Management complexity is more.
* End of project may not be known which is a risk.
* Highly skilled resources are required for risk analysis.
* Projects progress is highly dependent upon the risk analysis phase.

## **INTRODUCTION**

## **1 Relevance of the Project**

A recommendation system or recommendation engine is a model used for information filtering where it tries to predict the preferences of a user and provide suggests based on these preferences. These systems have become increasingly popular nowadays and are widely used today in areas such as movies, music, books, videos, clothing, restaurants, food, places and other utilities. These systems collect information about a user's preferences and behaviour, and then use this information to improve their suggestions in the future. Movies are a part and parcel of life. There are different types of movies like some for entertainment, some for educational purposes, some are animated movies for children, and some are horror movies or action films. Movies can be easily differentiated through their genres like comedy, thriller, animation, action etc. Other way to distinguish among movies can be either by releasing year, language, director etc. Watching movies online, there are a number of movies to search in our most liked movies . Movie Recommendation Systems helps us to search our preferred movies among all of these different types of movies and hence reduce the trouble of spending a lot of time searching our favourable movies. So, it requires that the movie recommendation system should be very reliable and should provide us with the recommendation of movies which are exactly same or most matched with our preferences. A large number of companies are making use of recommendation systems to increase user interaction and enrich a user's shopping experience. Recommendation systems have several benefits, the most important being customer satisfaction and revenue. Movie Recommendation system is very powerful and important system. But, due to the problems associated with pure collaborative approach, movie recommendation systems also suffers with poor recommendation quality and scalability issues.



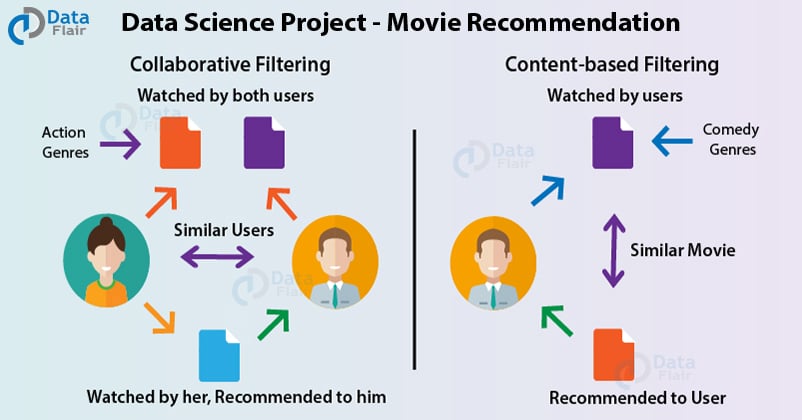
### Different types of recommendation engines

The most common types of recommendation systems are content-based and collaborative filtering recommender systems. In collaborative filtering, the behavior of a group of users is used to make recommendations to other users. The recommendation is based on the preference of other users. A simple example would be recommending a movie to a user based on the fact that their friend liked the movie. There are two types of collaborative models Memory-basedmethods and Model-based methods. The advantage of memory-based techniques is that they are simple to implement and the resulting recommendations are often easy to explain. They are divided into two:

* **User-based collaborative filtering**: In this model, products are recommended to a user based on the fact that the products have been liked by users similar to the user. For example, if Derrick and Dennis like the same movies and a new movie come out that Derick like, then we can recommend that movie to Dennis because Derrick and Dennis seem to like the same movies.
* **Item-based collaborative filtering**: These systems identify similar items based on users’ previous ratings. For example, if users A, B, and C gave a 5-star rating to books X and Y then when a user D buys book Y they also get a recommendation to purchase book X because the system identifies book X and Y as similar based on the ratings of users A, B, and C.

Model-based methods are based on Matrix Factorization and are better at dealing with sparsity. They are developed using data mining, machine learning algorithms to predict users’ rating of unrated items. In this approach techniques such as dimensionality reduction are used to improve accuracy. Examples of such model-based methods include Decision trees, Rule-based Model, Bayesian Model, and latent factor models.

**Content-based systems** use metadata such as genre, producer, actor, musician to recommend items say movies or music. Such a recommendation would be for instance recommending Infinity War that featured Vin Diesel because someone watched and liked The Fate of the Furious. Similarly, you can get music recommendations from certain artists because you liked their music. Content-based systems are based on the idea that if you liked a certain item you are most likely to like something that is similar to it.



**1.2 Problem Statement:**

The goal of the project is to recommend a movie to the user.

Providing related content out of relevant and irrelevant collection of items to users of online service providers.

**1.3 Objective of the Projects**

• Improving the Accuracy of the recommendation system

• Improve the Quality of the movie Recommendation system

• Improving the Scalability.

• Enhancing the user experience

### 1.4 Scope of the Project

The objective of this project is to provide accurate movie recommendations to users. The goal of the project is to improve the quality of movie recommendation system, such as accuracy, quality and scalability of system than the pure approaches. This is done using Hybrid approach by combining content based filtering and collaborative filtering, To eradicate the overload of the data, recommendation system is used as information filtering tool in social networking sites .Hence, there is a huge scope of exploration in this field for improving scalability, accuracy and quality of movie recommendation systems Movie Recommendation system is very powerful and important system. But, due to the problems associated with pure collaborative approach, movie recommendation systems also suffers with poor recommendation quality and scalability issues.

**1.5 Methodology for Movie Recommendation**

### The proposed movie recommendation system gives finer similarity metrics and quality than the existing Movie recommendation system but the computation time which is Taken by the proposed recommendation system is more than the existing recommendation system. This problem can be fixed by taking the clustered data points as an input dataset The proposed approach is for improving the scalability and quality of the movie recommendation system .We use a Hybrid approach , by unifying Content-Based Filtering and Collaborative Filtering, so that the approaches can be profited from each other. For computing similarity between the different movies in the given dataset efficiently and in least time and to reduce computation time of the movie recommender engine we used cosine similarity measure.

### Methodology

* Collecting the data sets : Collecting all the required data set from Kaggle web site.in this project we require movie.csv,ratings.csv,users.csv.
* Data Analysis : make sure that that the collected data sets are correct and analysing the data in the csv files. i.e. checking whether all the column Felds are present in the data sets.
* Algorithms : in our project we have only two algorithms one is cosine similarity and other is single valued decomposition are used to build the machine learning recommendation model.
* Training and Testing the model : once the implementation of algorithm is completed . we have to train the model to get the result. We have tested it several times the model is recommend different set of movies to different users.

### SYSTEM REQUIREMENTS SPECIFICATION

### 

### This involves both the hardware and software requirements needed for the project and detailed explanation of the specifications

* HARDWARE REQUIREMENTS:-

The most common set of requirements defined by any [operating system](https://en.wikipedia.org/wiki/Operating_system) or [software application](https://en.wikipedia.org/wiki/Software_application) is the physical computer resources, also known as [hardware](https://en.wikipedia.org/wiki/Computer_hardware), A hardware requirements list is often accompanied by a [hardware compatibility list](https://en.wikipedia.org/wiki/Hardware_compatibility_list) (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

Hardware Requirements :

• A PC with Windows/Linux OS

• Processor with 1.7-2.4gHz speed

• Minimum of 8gb RAM

• 2gb Graphic card

Software Specification :

• Text Editor (VS-code/WebStorm)

• PyCharm Editor

• Python libraries

**PyCharm Editor**: Python programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management system and deployment..

#### **Visual Studio Code :**

Visual Studio Code, also commonly referred to as VS Code, is a source-code editor made by Microsoft with the Electron Framework, for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality

* SOFTWARE REQUIREMENTS:-

[Software requirements](https://en.wikipedia.org/wiki/Software_requirements) deal with defining software resource requirements and prerequisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or prerequisites are generally not included in the software installation package and need to be installed separately before the software is installed.

**Python libraries:** For the computation and analysis we need certain python libraries which are used to perform analytics. Packages such as SKlearn, Numpy, pandas, Matplotlib are needed.

* **SKlearn:** It features various classification, regression and clustering algorithms including support vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.
* **NumPy :** NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. Pandas: Pandas is one of the most widely used python libraries in data science. It provides high-performance, easy to use structures and data analysis tools. Unlike NumPy library which provides objects for multi-dimensional arrays, Pandas provides in-memory 2d table object called Data frame.
* **Pandas :** Pandas is an**open-source Python Library** providing high-performance data manipulation and analysis tool using its powerful data structures. The name Pandas is derived from the word Panel Data – an Econometrics from Multidimensional data. In 2008, developer Wes McKinney started developing pandas when in need of high performance, flexible tool for analysis of data.
* **Matplotlib** : Matplotlib is a [plotting](https://en.wikipedia.org/wiki/Plotter) [library](https://en.wikipedia.org/wiki/Library_(computer_science)) for the [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) programming language and its numerical mathematics extension [NumPy](https://en.wikipedia.org/wiki/NumPy" \o "NumPy). It provides an [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming) [API](https://en.wikipedia.org/wiki/API) for embedding plots into applications using general-purpose [GUI toolkits](https://en.wikipedia.org/wiki/GUI_toolkit) like [Tkinter](https://en.wikipedia.org/wiki/Tkinter" \o "Tkinter), [wxPython](https://en.wikipedia.org/wiki/WxPython" \o "WxPython), [Qt](https://en.wikipedia.org/wiki/Qt_(software)" \o "Qt (software)), or [GTK](https://en.wikipedia.org/wiki/GTK). There is also a [procedural](https://en.wikipedia.org/wiki/Procedural_programming) "pylab" interface based on a [state machine](https://en.wikipedia.org/wiki/State_machine) (like [OpenGL](https://en.wikipedia.org/wiki/OpenGL)), designed to closely resemble that of [MATLAB](https://en.wikipedia.org/wiki/MATLAB), though its use is discouraged. [SciPy](https://en.wikipedia.org/wiki/SciPy" \o "SciPy) makes use of Matplotlib.

### 

### Feasibility Report

**1. TECHNICAL FEASIBILITY:**

● Machine learning algorithms

● Python

**2. DELIVERABLES:**

A web application that shows different movies and user selection recommends the bestsuited movies for a user from the dataset.

**3. SOFTWARE MODEL:**

For this project, we follow the Iterative Model **.**

**4. FINANCIAL FEASIBILITY**:

All the technologies mentioned above are free to use, and the equipment required is already with us; hence there would be no financial problems for our project.

**5. TECHNICAL FEASIBILITY:**

For the project, we have sufficiently powered laptops with dedicated GPUs that can handle the project’s processing requirements. The time is also enough to implement our application, and the team is educated enough to use all these technologies well**.**

### 6. RESOURCE & PERFORMANCE FEASIBILITY:

Resources that are required for the proper implementation of this project are:

• Programming devices like personal computers, workstations, or laptops.

• Hosting space on the local domain (freely available).

• Programming tools.

• Programming individuals.

The time required to store and process this information is negligible for modern-day processors, as they have high clock speeds. Thus, it’s evident that this system has the required resource and time feasibility.

**7. RISK FEASIBILITY:**

• Size of the application depends upon the dataset into consideration that is dependent on the content customer has, and we are using a free-hosting service it has its limitations

•We have experience creating web apps, but this project is new with the technologies we are using, so we might find it challenging to integrate all these technologies, which might cause some delay in implementation.

**8.SOCIAL/LEGAL FEASIBILITY:**

The project uses freely available open-source development tools and provides the system as an open-source system. Potential customers will only be charged with the maintenance cost. Since it promotes less use of manual work on papers, it will benefit the organization financially and the environment

**9.OUTLINE PLAN :**

#### Front-End

* Back-End(ML Model)
* Data Analyst
* Pre Processing

### USER MODULE DESCRIPTION

The proposed system consists of two modules :-

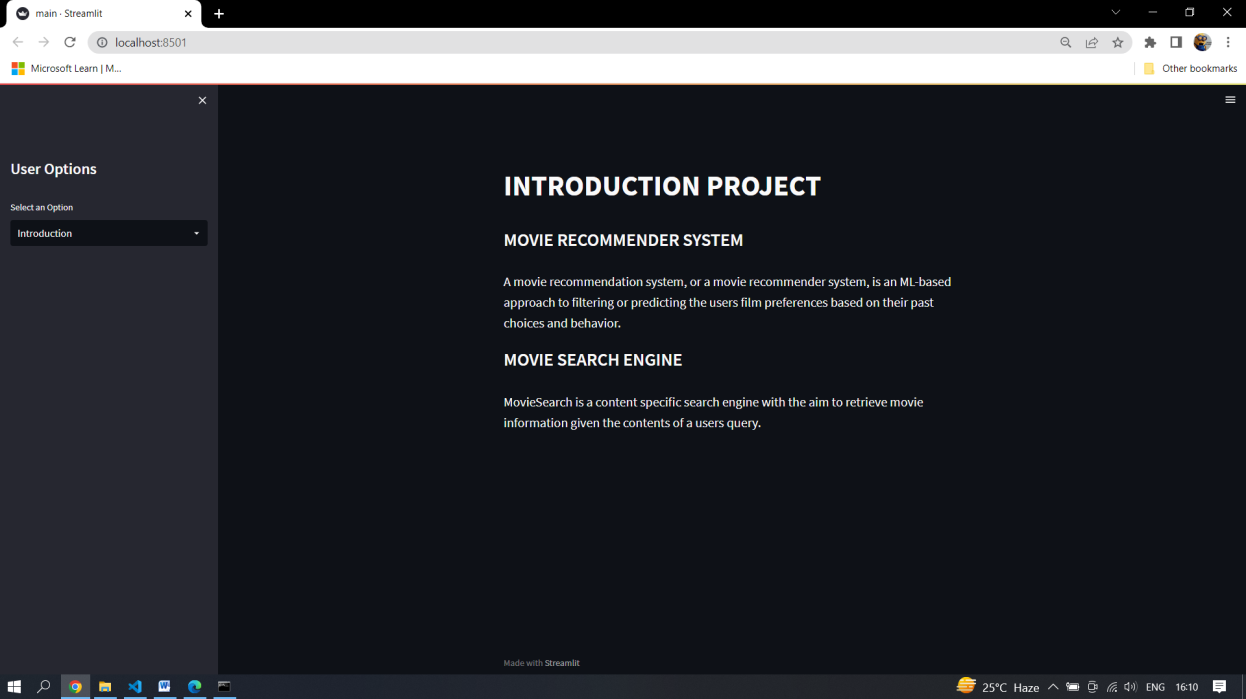
* API MODULE
* SEARCH MODULE
* API MODULE : I made two function , first function fetches poster from TDMN database and second function is fetches video key from TDMN database. Video key is used to you tube url.In the project, YouTube url is used for to recomend which type of movie you want.
* SEARCH MODULE : This module searches the movie which the user types and lists the details about that particular movie like its popularity, rating ,trailer and many more.

### RESULTS AND DISCUSSION

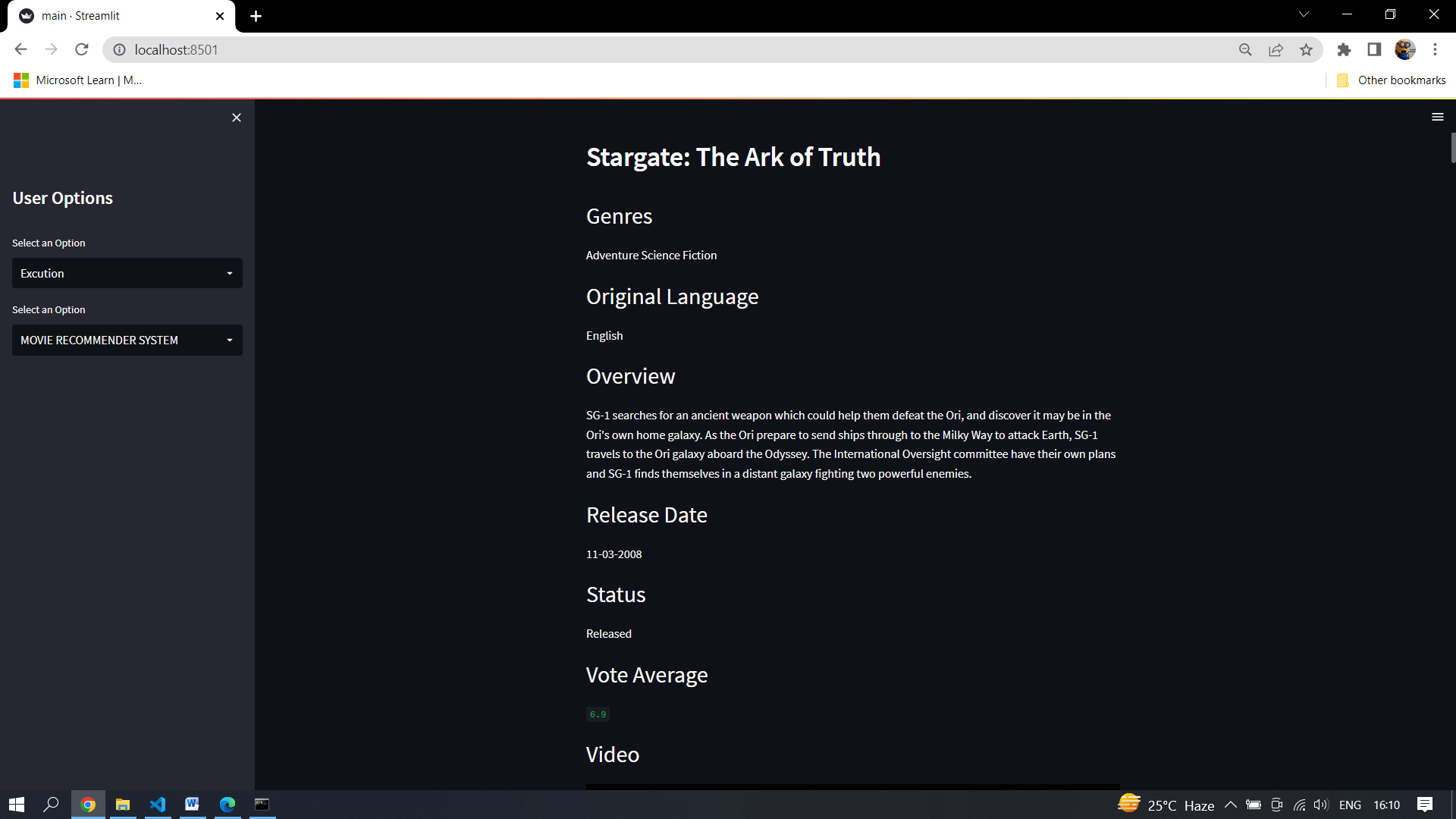
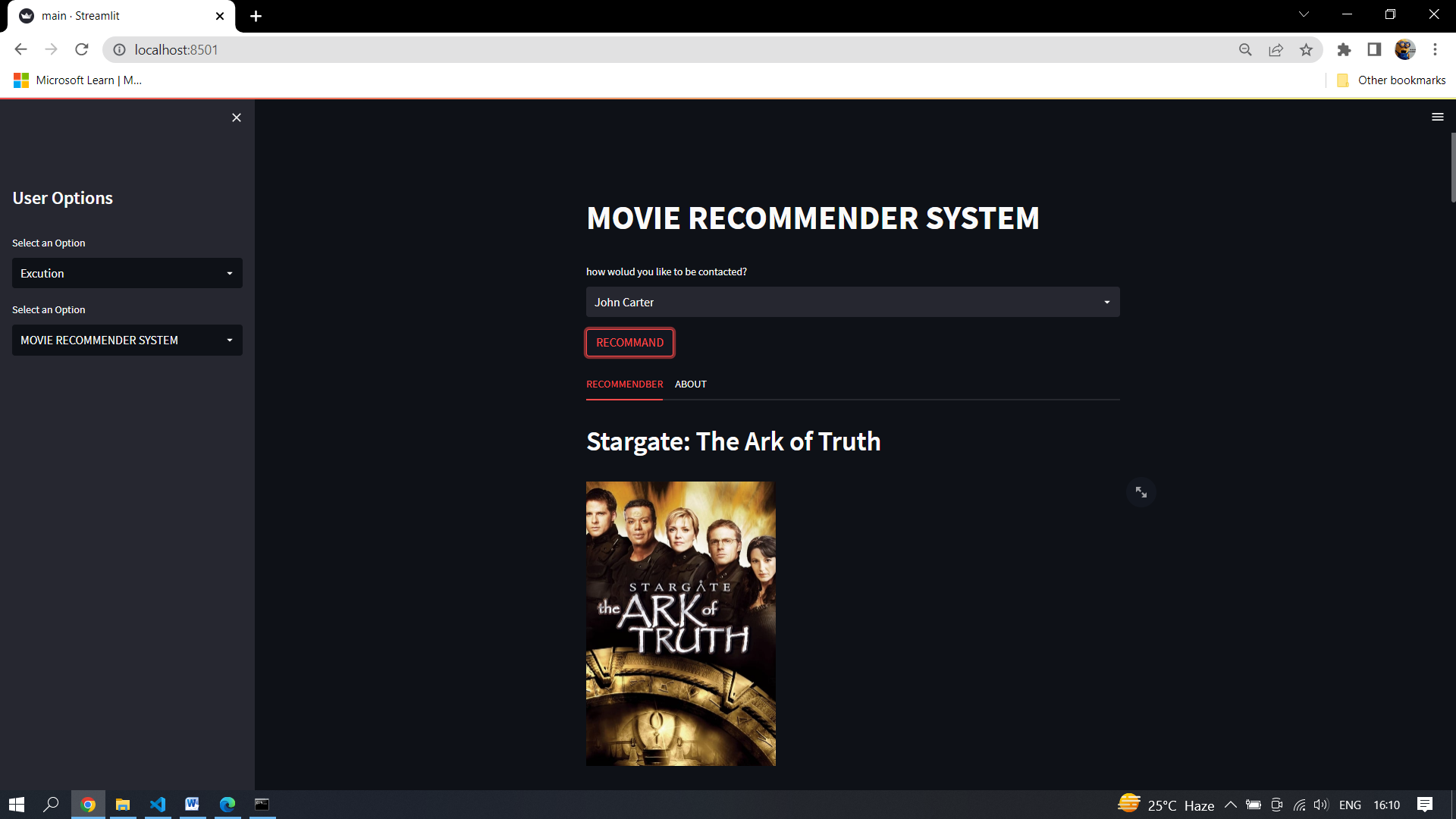
Since our project is movie recommendation system .one can develop a movie recommendation system by using either content based or collaborative filtering or combining both. In our project we have developed a hybrid approach i.e combination of both content and collaborative filtering .Both the approaches have advantages and dis-advantages .in content based filtering the it based on the user ratings or user likes only such kind of movie will recommended to the user.

* **SCEERN SHOT**

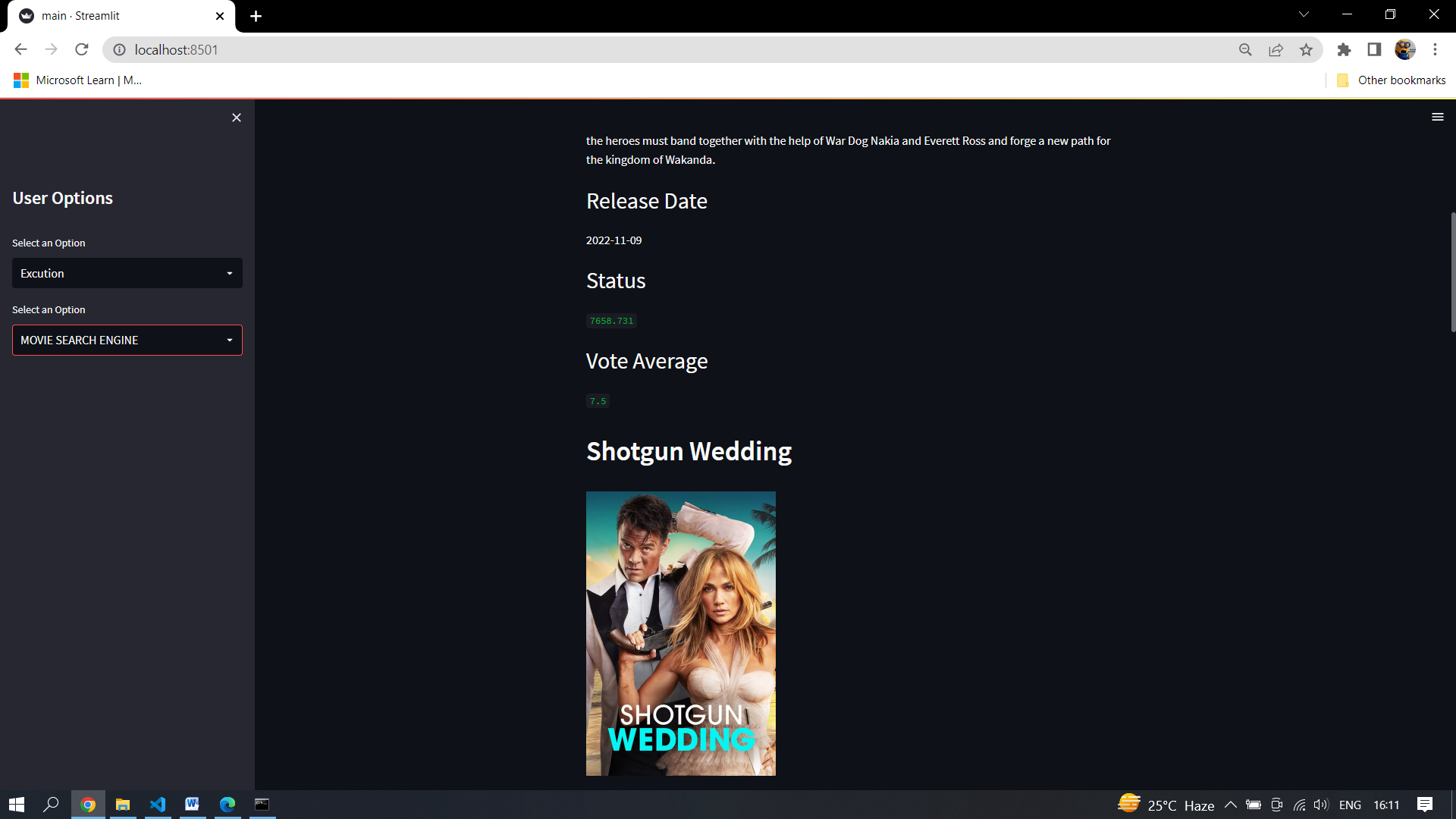
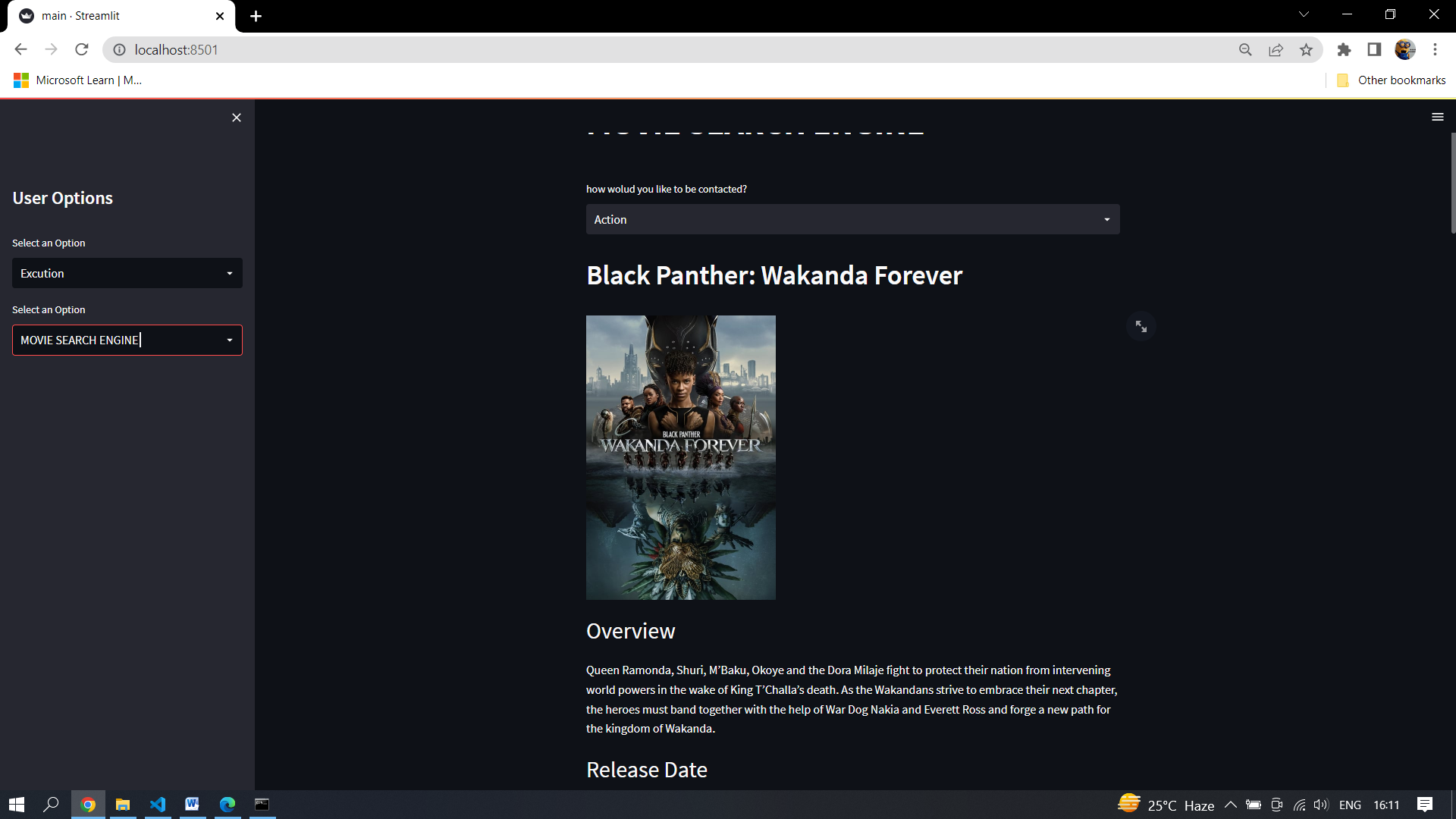
INTRO PAGE



MOVIE RECOMMENDER PAGE



GENRE SEARCH PAGE



DATA PREPROCESSING

* PREPROCESSING INFORMATION

**Dataset Collection**

There are several datasets available to build a movie recommendation system. But for this project, we are going to use a dataset that contains the metadata (cast, crew, budget, etc..) Of the movie.

**Preprocessing**

The dataset contains two CSV files, credits, and movies. The credits file contains all the metadata information about the movie and the movie file contains the information like name and id of the movie, budget, languages in the movie that has been released, etc.

**Content Filtering**

Content-based filtering are also known as cognitive filtering. This filtering recommends item to the user based on his past experience. For example, if a user likes only action movies then the system predicts him only action movies similar to it which he has highly rated. The broader explanation could be suppose the user likes only politics related content so the system suggests the websites, blogs or the news similar to that content.

* PREPROCESSING CODE

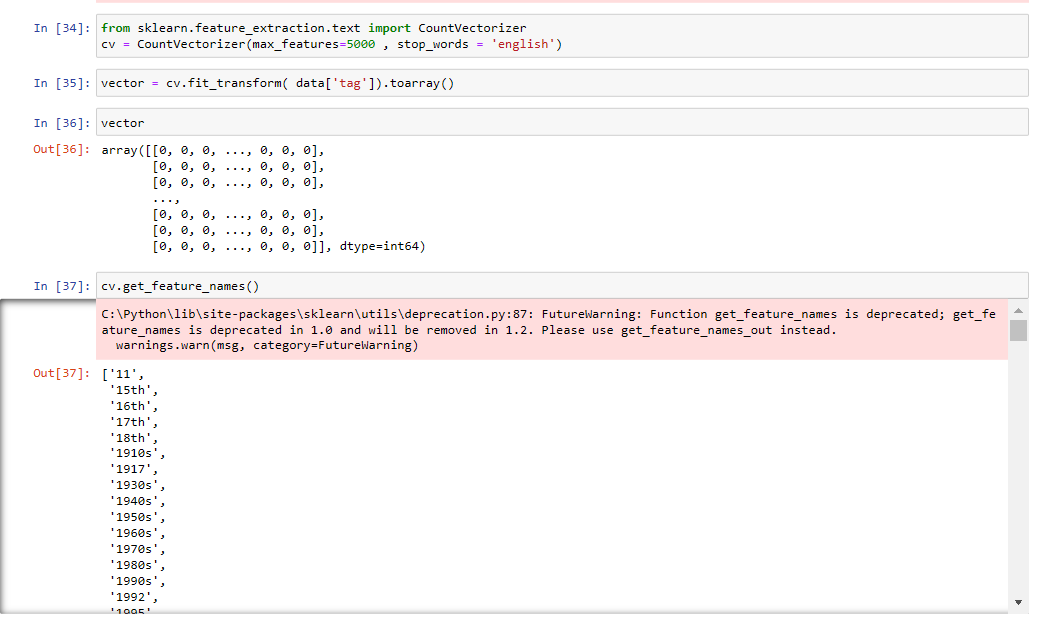
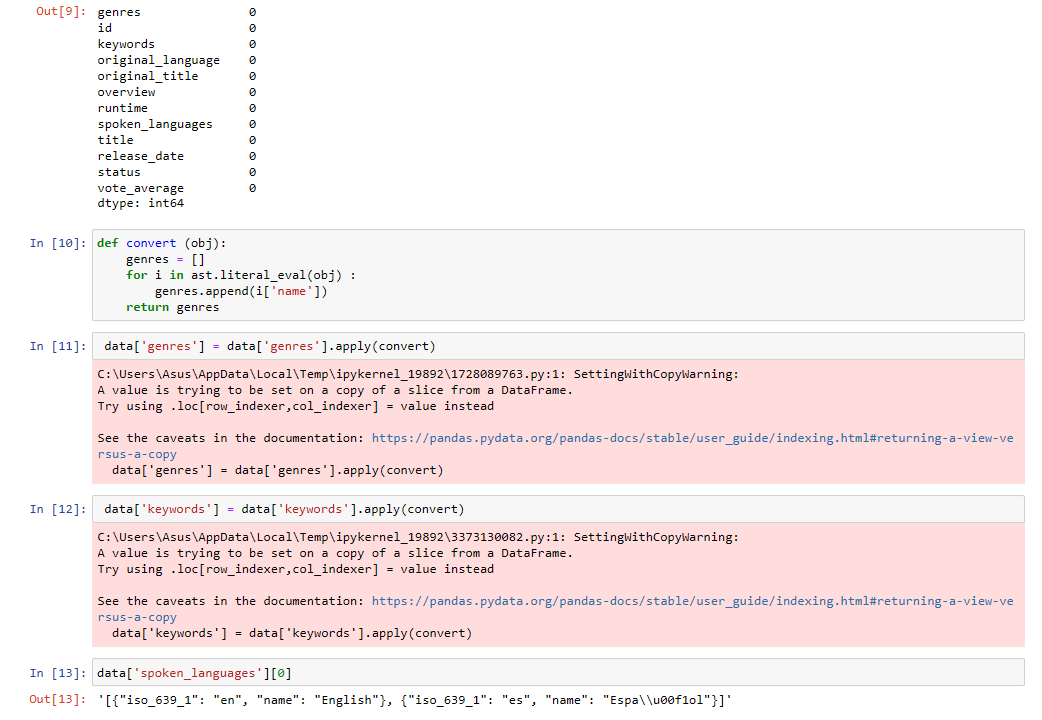
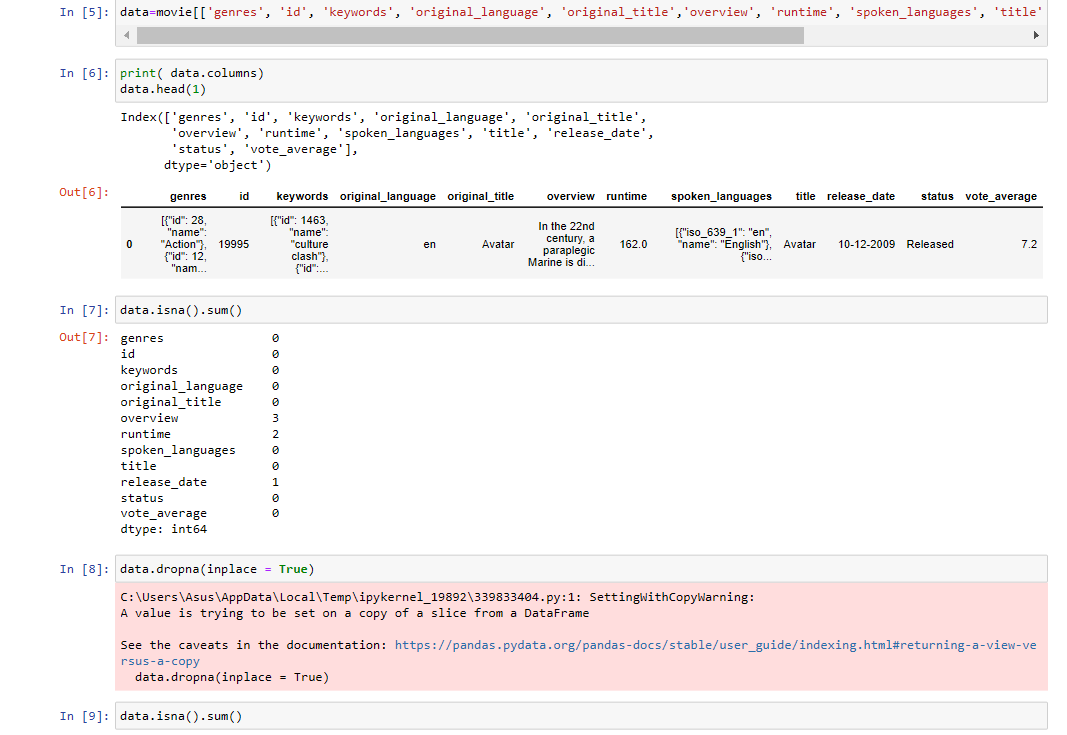
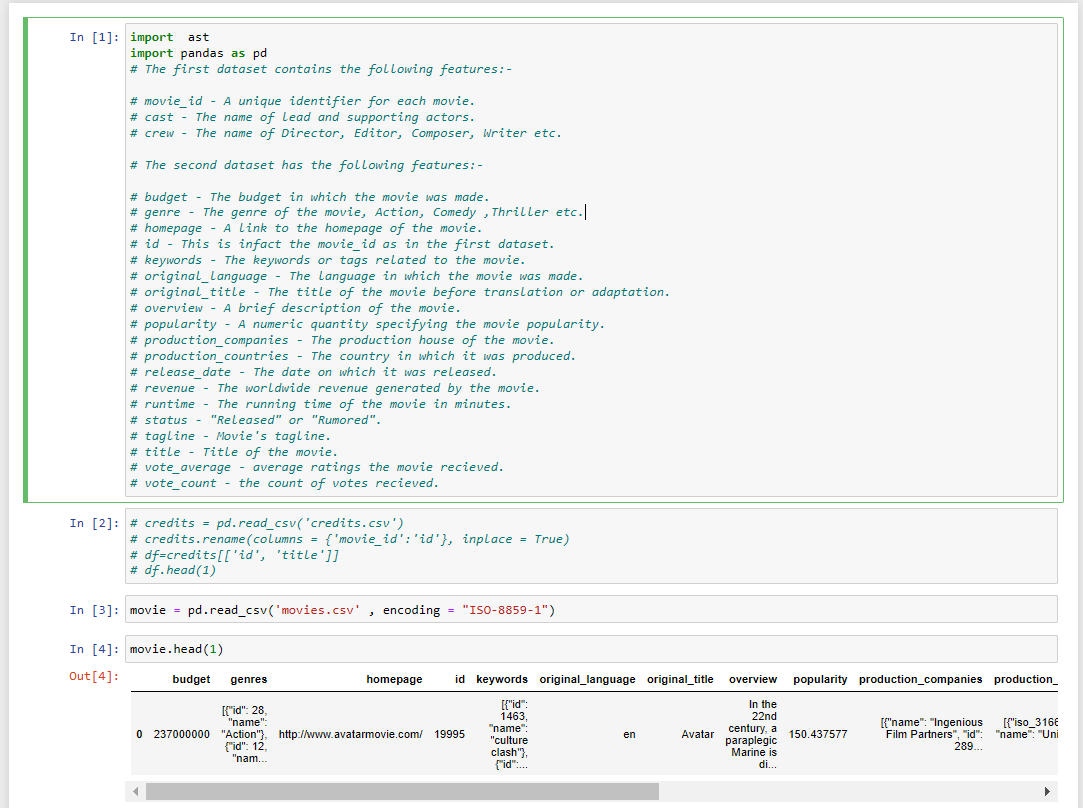
**1.Datasets to use for building recommender systems**

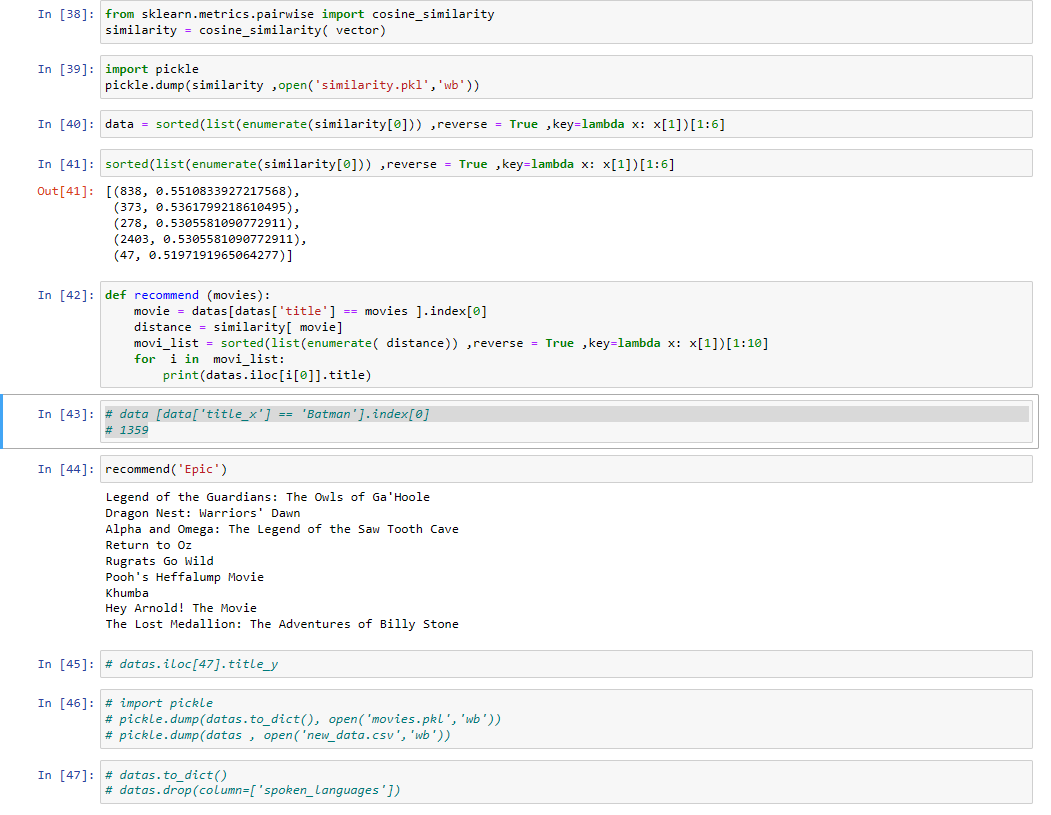
we are going to use the KAGGLE DATABASE . This dataset was put together by the TMDB. It contains 1, 10, and 20 million ratings.  KAGGLE also has a website where you can sign up, contribute reviews and get movie recommendations.

**2.Walkthrough of building a recommender system**

We are going to use the movie lens to build a simple item similarity-based recommender system. The first thing we need to do is to .

.



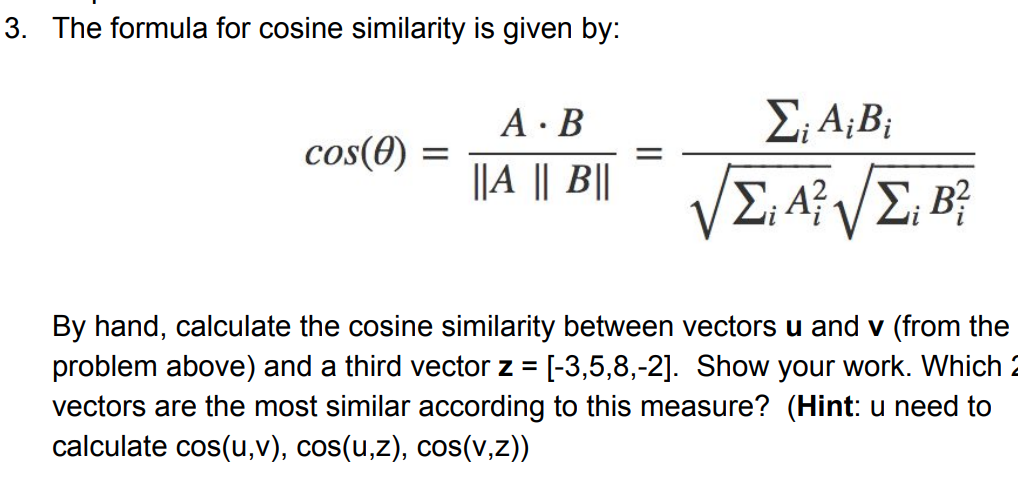
****

**IMPLEMENTATION**

The Proposed System Make Use Different Algorithms and Methods for the implementation of Hybrid Approach

5.1 Cosine Similarity: Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them.

Formula:



**CODE INFORMATION**

1. **Main File**

import streamlit as st

import pandas as pd

from streamlit\_player import st\_player

from api import movie\_list

from search import searching

st.markdown("""

<style>

.id {

font-size:20px;

}

</style>

""", unsafe\_allow\_html=True)

genre\_list = pd.read\_csv('data\list.csv')

datas = pd.read\_csv('data\datafiles.csv' , encoding = "ISO-8859-1")

option=['Introduction' , 'Excution']

def Introduction() :

st.title('INTRODUCTION PROJECT ')

st.subheader('MOVIE RECOMMENDER SYSTEM')

st.markdown('<p class="id">A movie recommendation system, or a movie recommender system, is an ML-based approach to filtering or predicting the users film preferences based on their past choices and behavior.</p>' , unsafe\_allow\_html=True)

st.subheader('MOVIE SEARCH ENGINE')

st.markdown('<p class="id">MovieSearch is a content speciﬁc search engine with the aim to retrieve movie information given the contents of a users query.</p>' , unsafe\_allow\_html=True)

sidebar = st.sidebar

sidebar.title('User Options')

seloption = sidebar.selectbox( 'Select an Option', option)

def Excution():

options=['MOVIE RECOMMENDER SYSTEM' , 'MOVIE SEARCH ENGINE']

def recommended():

st.title('MOVIE RECOMMENDER SYSTEM')

selected\_movie\_name = st.selectbox('how wolud you like to be contacted?' ,(datas['title'].values))

if st.button('RECOMMAND'):

data ,poster , genress , original\_language ,overview , release\_date , status , vote\_average , Video = movie\_list(selected\_movie\_name)

tab1, tab2 = st.tabs(["RECOMMENDBER", "ABOUT"])

st.markdown("""

<style>

.big {

font-size:30px;

}

</style>

""", unsafe\_allow\_html=True)

with tab1 :

for n in range(0,10):

st.header(data[n])

st.image(poster[n] , width=250)

st.write('--------------------------------------------------------')

with tab2:

st.header(data[0])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write( genress [0])

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[0])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[0] )

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[0] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[0] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[0] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[0])

st.write('------------------------------------------------------------------------------------')

st.header(data[1])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write(genress[1] )

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[1])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[1] )

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[1] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[1] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[1] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[1])

st.write('------------------------------------------------------------------------------------')

st.header(data[2])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write( genress[2] )

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[2])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[2] )

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[2] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[2] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[2] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[2])

st.write('------------------------------------------------------------------------------------')

st.header(data[3])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write(genress[3] )

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[3])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[3])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[3] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[3] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[3] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[3])

st.write('------------------------------------------------------------------------------------')

st.header(data[4])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write( genress[4] )

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[4])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[4])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[4] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[4] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[4] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[4])

st.write('------------------------------------------------------------------------------------')

st.header(data[5])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write(genress[5])

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[5])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[5])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[5] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[5] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[5] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[5])

st.write('------------------------------------------------------------------------------------')

st.header(data[6])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write(genress[6])

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[6])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[6])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[6] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[6] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[6] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[6])

st.write('------------------------------------------------------------------------------------')

st.header(data[7])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write('genress -' ,genress[7])

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[7])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write( overview[7])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[7] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[7] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[7] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[7])

st.write('------------------------------------------------------------------------------------')

st.header(data[8])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write(genress[8])

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[8])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[8])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[8] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[8])

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[8] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[8])

st.write('------------------------------------------------------------------------------------')

st.header(data[9])

st.markdown('<p class="big">Genres</p>', unsafe\_allow\_html=True)

st.write(genress[9])

st.markdown('<p class="big">Original Language</p>', unsafe\_allow\_html=True)

st.write(original\_language[9])

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.markdown(overview[9])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[9] )

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[9] )

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[9] )

st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

st\_player(Video[9])

st.write('------------------------------------------------------------------------------------')

seloptions = sidebar.selectbox( 'Select an Option', options)

def search():

st.title('MOVIE SEARCH ENGINE')

st.markdown("""

<style>

.big {

font-size:30px;

}

</style>

""", unsafe\_allow\_html=True)

gener\_id = st.selectbox('how wolud you like to be contacted?' ,(genre\_list['genre\_name'].values))

for index ,items in genre\_list.iterrows():

if items['genre\_name'] == gener\_id:

title ,poster ,overview , release\_date , status , vote\_average = searching(items['genre\_id'])

for i in range(0,5):

st.header(title[i])

st.image(poster[i] , width=250)

st.markdown('<p class="big">Overview</p>', unsafe\_allow\_html=True)

st.write(overview[i])

st.markdown('<p class="big">Release Date</p>', unsafe\_allow\_html=True)

st.write(release\_date[i])

st.markdown('<p class="big">Status</p>', unsafe\_allow\_html=True)

st.write(status[i])

st.markdown('<p class="big">Vote Average</p>', unsafe\_allow\_html=True)

st.write(vote\_average[i])

# st.markdown('<p class="big">Video</p>', unsafe\_allow\_html=True)

# st\_player(video[i])

if seloptions == options[0] :

recommended()

elif seloptions == options[1]:

search()

if seloption == option[0] :

Introduction()

elif seloption == option[1]:

Excution()

**2. Search File**

import requests

# import streamlit as st

# from streamlit\_player import st\_player

#

https://api.themoviedb.org/3/discover/movie?api\_key=88cdc0cdb44a687b97e2d0bb57e5328f&with\_genres=27

def fetch\_poster(search):

return "https://image.tmdb.org/t/p/w500/" + search

def fetch\_video(search): response=requests.get('https://api.themoviedb.org/3/search/movie?api\_key=88cdc0cdb44a687b97e2d0bb57e5328f&query={}&page=1&include\_adult=false'.format(search))

vote\_data=response.json()

response=requests.get('https://api.themoviedb.org/3/movie/{}/videos?api\_key=88cdc0cdb44a687b97e2d0bb57e5328f&language=en-US'.format(vote\_data['results'][0]['id']))

video\_data=response.json()

return 'https://www.youtube.com/watch?v=' +video\_data['results'][0]['key']

def searching (movie\_search):

obj= requests.get('https://api.themoviedb.org/3/discover/movie?api\_key=88cdc0cdb44a687b97e2d0bb57e5328f&with\_genres={}'.format(movie\_search))

title\_list = []

poster\_list = []

overview\_list = []

release\_date\_list = []

popularity\_list = []

vote\_average\_list = []

# video\_list =[]

movie\_list\_id = []

lists = obj.json()

for i in range(0,5):

title\_list.append(lists['results'][i]['title'])

poster\_list.append(fetch\_poster(lists['results'][i]['poster\_path']))

# original\_language\_list.append(lists['results'][i]['original\_language'])

overview\_list.append(lists['results'][i]['overview'])

release\_date\_list.append(lists['results'][i]['release\_date'])

popularity\_list.append(lists['results'][i]['popularity'])

vote\_average\_list.append(lists['results'][i]['vote\_average'])

# elemet = fetch\_video(lists['results'][i]['id'])

# video\_list.append(elemet)

return title\_list,poster\_list,overview\_list, release\_date\_list, popularity\_list, vote\_average\_list

**3.API FILE**

import requests

import pandas as pd

import pickle

def fetch\_poster(movie\_id):

response=requests.get('https://api.themoviedb.org/3/movie/{}?api\_key=88cdc0cdb44a687b97e2d0bb57e5328f&language=en-US'.format(movie\_id))

data=response.json()

return "https://image.tmdb.org/t/p/w500/" + data['poster\_path']

def video(movie\_id): response=requests.get('https://api.themoviedb.org/3/movie/{}/videos?api\_key=88cdc0cdb44a687b97e2d0bb57e5328f&language=en-US'.format(movie\_id))

video\_data=response.json()

return 'https://www.youtube.com/watch?v=' + video\_data['results'][0]['key']

datas = pd.read\_csv('data\datafiles.csv' , encoding = "ISO-8859-1")

similarity = pickle.load(open('data\similarity.pkl','rb'))

'release\_date','status','vote\_average'

def movie\_list( movie):

movie = datas[datas['title'] == movie].index[0]

distance = similarity[movie]

movi\_list = sorted(list(enumerate(distance)) ,reverse = True ,key=lambda x: x[1])[1:11]

data = []

poster =[]

genress = []

original\_language = []

overview = []

release\_date =[]

status = []

vote\_average=[]

video\_path=[]

for i in movi\_list:

movis\_id = datas.iloc[i[0]].id

data.append(datas.iloc[i[0]].title)

# with api

poster.append(fetch\_poster(movis\_id))

genress.append(datas.iloc[i[0]].genres)

original\_language.append(datas.iloc[i[0]].original\_language)

overview.append(datas.iloc[i[0]].overview)

release\_date.append(datas.iloc[i[0]].release\_date)

status.append(datas.iloc[i[0]].status)

vote\_average.append(datas.iloc[i[0]].vote\_average)

# with api

video\_path.append(video(movis\_id))

return data , poster , genress , original\_language ,overview , release\_date , status , vote\_average , video\_path

###### SOFTWARE INTERFACE DESIGN

**“INTRO”** Page is the first pages the user sees when he visit website and this page the user can sidebar on left to one of the pages among various available option , the links to which have been given on the navigate of this page.

**“MOVIES-RECOMMEND**” page allows the user to search given list for the name of the movie and shows the result of the searched and related movies to the searched .

**ATTITUDE**

HEADER

IMAGE

**“ABOUT”** page contains information about the Movie Recommendation System(MRS)

**ATTITUDE**

HEADER

GENRES

ORIGINAL LANGUAGE

OWERVIEW

RELEASE DATE

STATUS

VOTE AVERAGE

VIDEO

**“MOVIES-SEARCH**” page allows the user to enter the name of the movie and shows the result of the searched and related movies to the searched .

**ATTITUDE**

TITLE

POSTER

OVERVIEW

RELEASE DATE

STATUS

VOTE\_AVERAGE

###### SYSTEM ANALYSIS AND DESIGN

System Architecture of Proposed System:

USER

RESULT

GENRE BASED MOVIE SEARCH

MOVIE

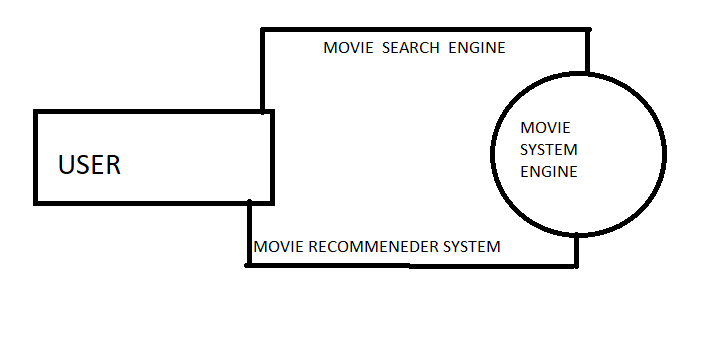
CONTACT BASED MOVIE SEARCH

For each different individual use different list of movies are recommended ,as user login or enters the user id based on two different approaches used in the project each will recommend the set of movies to the particular user by combining the both the set of movie based on the user the hybrid model will recommend the single list of movie to the user.

DFD (DATA FLOW DAIGRAM)

A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement. They are often elements of a formal methodology such as Structured Systems Analysis and Design Method ([SSADM](https://www.techtarget.com/searchsoftwarequality/definition/SSADM)). Superficially, DFDs can resemble flow charts or Unified Modeling Language ([UML](https://www.techtarget.com/searchsoftwarequality/definition/Unified-Modeling-Language)), but they are not meant to represent details of software logic.

* + - * DFD Level 0 is also called a Context Diagram. It’s a basic overview of the whole system or process being analyzed or modeled.



* + - * DFD Level 1 provides a more detailed breakout of pieces of the Context Level Diagram. You will highlight the main functions carried out by the system

USER

###### TESTING

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work to verify that all the system elements have been properly integrated and perform allocated functions. The testing process is actually carried out to make sure that the product exactly does the same thing what is supposed to do. In the testing stage following goals are tried to achieve: -

● To affirm the quality of the project.

● To find and eliminate any residual errors from previous stages.

● To validate the software as a solution to the original problem.

● To provide operational reliability of the system.

7.1 Testing Methodologies

There are many different types of testing methods or techniques used as part of the software testing methodology. Some of the important testing methodologies are:

**Unit Testing :**

Unit testing is the first level of testing and is often performed by the developers themselves. It is the process of ensuring individual components of a piece of software at the code level are functional and work as they were designed to. Developers in a test-driven environment will typically write and run the tests prior to the software or feature being passed over to the test team. Unit testing can be conducted manually, but automating the process will speed up delivery cycles and expand test coverage. Unit testing will also make debugging easier because finding issues earlier means they take less time to fix than if they were discovered later in the testing process. Test Left is a tool that allows advanced testers and developers to shift left with the fastest test automation tool embedded in any IDE.

**Example**

a) In a program we are checking if the loop, method, or

function is working fine

b) Misunderstood or incorrect, arithmetic precedence.

c) Incorrect initialization

**Integration Testing :**

After each unit is thoroughly tested, it is integrated with other units to create modules or components that are designed to perform specific tasks or activities. These are then tested as group through integration testing to ensure whole segments of an application behave as expected (i.e, the interactions between units are seamless). These tests are often framed by user scenarios, such as logging into an application or opening files. Integrated tests can be conducted by either developers or independent testers and are usually comprised of a combination of automated functional and manual tests.

### Integration testing can be performed in different ways, such as:

1. Top-down integration testing: It starts with the highest level modules and integrates them with lower-level modules.
2. Bottom-up integration testing: It starts with the lowest-level modules and integrates them with higher-level modules.
3. Big-Bang integration testing: It combines all the modules and integrates them all at once.
4. Incremental integration testing: It integrates the modules in small groups, testing each group as it is added.

**Example**

(a) Black Box testing:- It is used for validation.

In this, we ignore internal working mechanisms and

focus on **what is the output?**.

(b) White box testing:- It is used for verification.

In this, we focus on internal mechanisms i.e.

**how the output is achieved?**

**System Testing :**

System testing is a black box testing method used to evaluate the completed and integrated system, as a whole, to ensure it meets specified requirements. The functionality of the software is tested from end-to-end and is typically conducted by a separate testing team than the development team before the product is pushed into production.

EXAMPLE

This includes functional as well as nonfunctional

testing

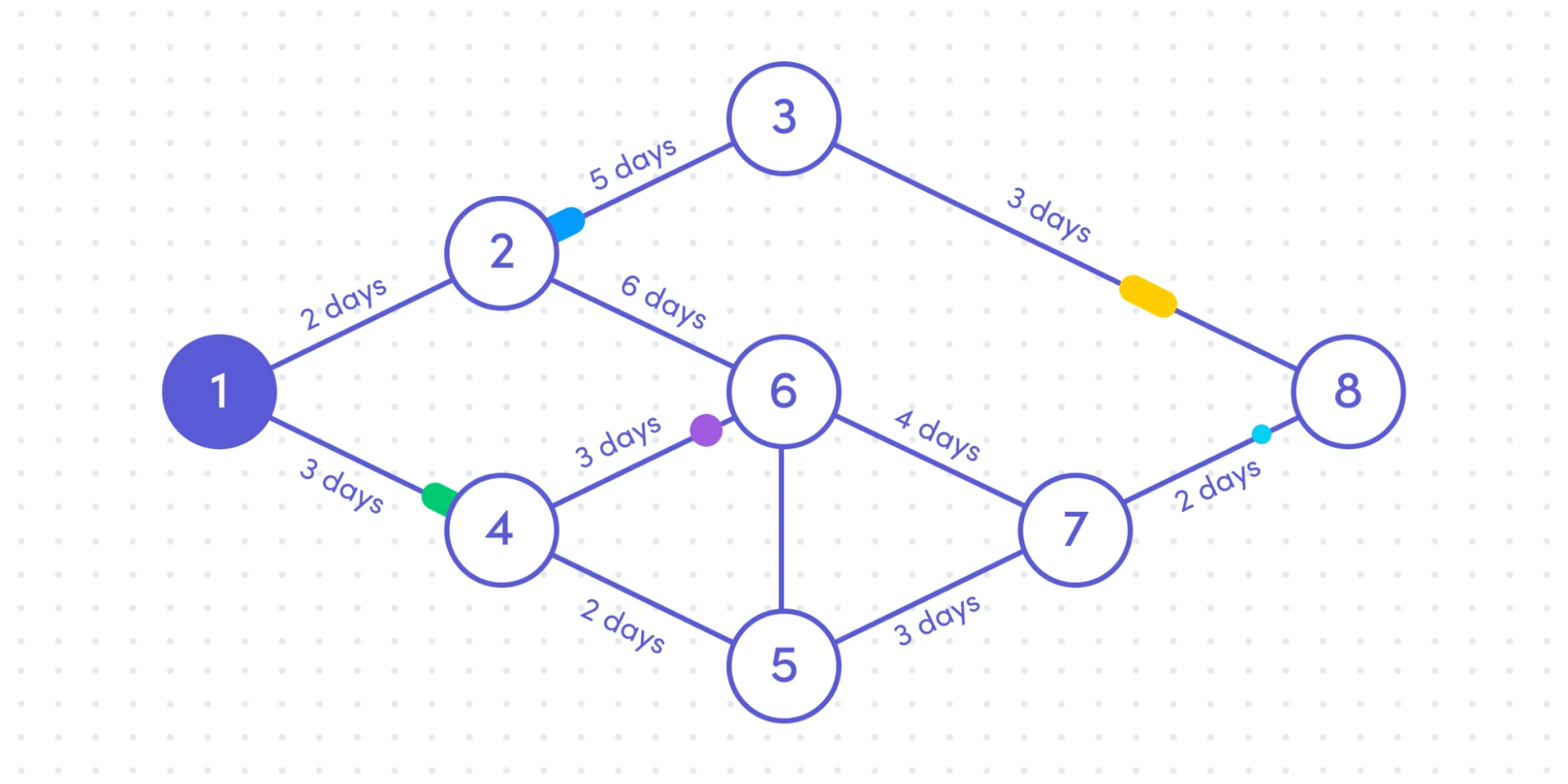
### PLANNING AND SCHEDULING

The objective of software project planning is to provide a framework that enables the manger to Make reasonable estimates of resources , cost and schedules . these estimate are made within a Limited time frame at the beginning of a software project and should be update regulary as the Project progresses . in addition , estimates should attempt tp define best case and worst caseScenarios sp thst project outcomes can be bounded.

* Pert chart

A PERT chart, also known as a PERT diagram, is a tool used to schedule, organize, and map out tasks within a project.

PERT stands for program evaluation and review technique. It provides a visual representation of a project's timeline and breaks down individual tasks.



* GANTT CHART

A Gantt chart is a project management tool that illustrates work completed over a period of time in relation to the time planned for the work. A Gantt chart can include the start and end dates of tasks, milestones, dependencies between tasks, assignees, and more.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TASK | 16-18 oct | 18-21 oct | 21-26 oct | 26-30 oct | 30-1 oct -nov |
| DEVELP PROJECT | 2 days |  |  |  |  |
| Analysis |  | 2 days |  |  |  |
| Designing |  | 2 days |  |  |  |
| Coding |  |  | 5 days |  |  |
| Unit Testing |  |  |  | 1 days |  |
| Beta Testing |  |  |  |  | 2 days |

###### CONCLUSION AND FUTRURE SCOPE

### Future Scope

In the proposed approach, It has considered Genres of movies but, in future we can also consider age of user as according to the age movie preferences also changes, like for example, during our childhood we like animated movies more as compared to other movies. There is a need to work on the memory requirements of the proposed approach in the future. The proposed approach has been implemented here on different movie datasets only. It can also be implemented on the Film Affinity and Netflix datasets and the performance can be computed in the future.

### 8.2 Conclusion

Cosine Similarity is presented in the proposed methodology. Existing pure approaches and proposed hybrid approach is implemented on three different Movie datasets and the results are compared among them. Comparative results depicts that the proposed approach shows an improvement in the accuracy, quality and scalability of the movie recommendation system than the pure approaches. Also, computing time of the proposed approach is lesser than the other two pure approaches.

In ALS , as expected the time taken increase with the size of the data and the RMSE decreases with increase in the size of the data.

In all the methods studied , time taken on cluster is considerably less than the time taken on the local machine .

### 

###### REFERENCES

### 1.[An Introduction to Data Science](https://docs.google.com/file/d/0B6iefdnF22XQeVZDSkxjZ0Z5VUE/edit?pli=1" \t "_blank) (September 2017)

2. Data Analytics Made Accessible by Dr.Anil Maheshwar

## **3.Machine Learning for Absolute Beginners: A Plain English Introduction**

4. Pyhton Data Science Handbook

WEB SITE

[www.Google.com](http://www.Google.com)

[www.Medium.com](http://www.Medium.com)

[www.kaggle.com](http://www.kaggle.com)

[www.TheMovieDatabase.com](http://www.TheMovieDatabase.com)

[www.Youtube.com](http://www.Youtube.com)

www.chatgtp.com