

A Project Report

(Review III report)

On

MOVIE RECOMMENDER SYSTEM

Submitted by

Kunal Kalra – 20BCE2035

Swati Rai – 20BCE0996

Arun Ajay – 20BCE0488

Shubhlaxh Porwal – 20BKT0056

For

Software Engineering

CSE3001

Slot: B1, J Component

B.Tech. in Computer Science and Engineering



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

November 2022

Table of Contents

Chapter 1 Introduction to Project

- 1.1 Introduction
- 1.2 Problem Definition
- 1.3 Project Scope
- 1.4 Motivation
- 1.5 Background Study/Literature Survey
- 1.6 SDLC approach used to develop project

Chapter 2 Project Planning

- 2.1 Project Schedule
- 2.2 Effort and Resource Estimation

Chapter 3 Requirement Gathering and Analysis

- 3.1 SRS
- 3.2 Data Modelling
- 3.3 Structural Analysis

Chapter 4 Designs

- 4.1 UML Designs
- 4.2 Algorithms and Flow Charts

Chapter 5 Development

- 5.1 Tools Description and Development approach used
- 5.2 Pseudocodes of Important Modules.

Chapter 6 Testing

- 6.1 Test cases for Important Modules

Chapter 7 Screenshots of Developed Product

Annexures

- 1. Review I Presentation
- 2. Review II Presentation
- 3. Review III Presentation

Chapter 1 - Introduction to Project

1.1 Introduction

The enormous development in the volume of digital information available and the number of Internet users has created a possible challenge of information overload, which makes it difficult to have timely access to items of interest on the Internet. Information retrieval systems like Google, Devil Finder, and Altavista have partially solved this challenge, but prioritization and personalization of information (where a system maps accessible content to a user's interests and preferences) were missing. The need for recommender systems has risen to unprecedented levels as a result of this. Recommender systems are information filtering systems that address the problem of information overload by extracting critical information fragments from a huge amount of dynamically generated data based on the user's choices, interests, or observed behavior about the item. Based on the user's profile, a recommender system can anticipate whether a particular user will like an item. Both service providers and users benefit from recommender systems. They lower the expenses of searching for and selecting things in an online buying environment. Recommendation systems have also been shown to increase the quality and speed of decision-making. Recommender systems increase revenue in an e-commerce context since they are an efficient way of selling more things. Recommender systems in scientific libraries assist users by allowing them to go beyond catalogue searches. As a result, the importance of employing efficient and accurate recommendation algorithms inside a system that provides consumers with relevant and dependable recommendations cannot be overstated.

1.2 Problem Definition

Identify movies that customers want to watch as per their likes and dislikes and providing them a solution give and view rating and reviews for a movie. In recent years, internet has become a huge pool of choices for almost everything. It is difficult for people to customize this ever-growing pool of data. Difficulty in accessing items of interest has become difficult that means time taken to search content has increased significantly.

1.3 Project Scope

In recent years, the internet has become a huge pool of choices for almost everything. It is difficult for people to customize this ever-growing pool of data. Hence a filter to effectively sort more relevant options becomes a necessity. Our tool solves this problem by searching through a large volume of information regarding different movies to provide users with personalized content and services. The tool will recommend movies to users based on their searches and will provide them with various additional information regarding description, genres, ratings etc.

1.4 Motivation

The existing models are based on clustering of data and don't work well with large datasets moreover most of the existing models are too US centric and don't offer much content for Indian audiences. This is where our model comes in , it will recommend movies to user based on various information like genres,ratings etc. It will provide a veru user friendly interface to browse through movies and get details about them

1.5 Background Study/Literature/ Survey

In [1-2], Recommendation system has an ability to provide personalized information on the internet. In this era of internet, lots of RSs have been developed that are based on Content based filtering, Collaborative filtering and Hybrid system and helps to reduce the problem of information overload.

[3] In this study, authors takes about both Collaborative filtering and Hybrid system and how they help to reduce the problem of information overload, they also concluded that collaborative filtering recommendation system provide better recommendation but still facing problem of scalability and sparsity which can be reduced by using the fuzzy clustering and the optimization technique.

In [4] authors came up with a hybrid model based on collaborative filtering to generate movie recommendations which combine dimensional reduction technique

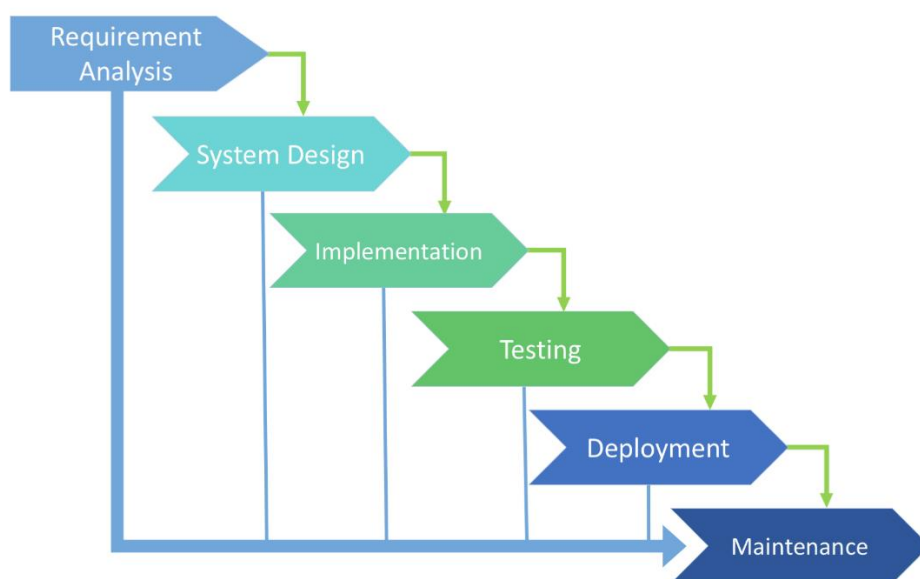
with the clustering algorithm. They focused on improving the approach to deal with higher dimensionality and sparsity issues in practical environment

[5] This paper discussed the two traditional recommendation techniques and highlighted their strengths and challenges with diverse kind of hybridization strategies used to improve their performances. Various learning algorithms used in generating recommendation models and evaluation metrics used in measuring the quality and performance of recommendation algorithms were discussed.

1.6 SDLC Approach used to develop the project

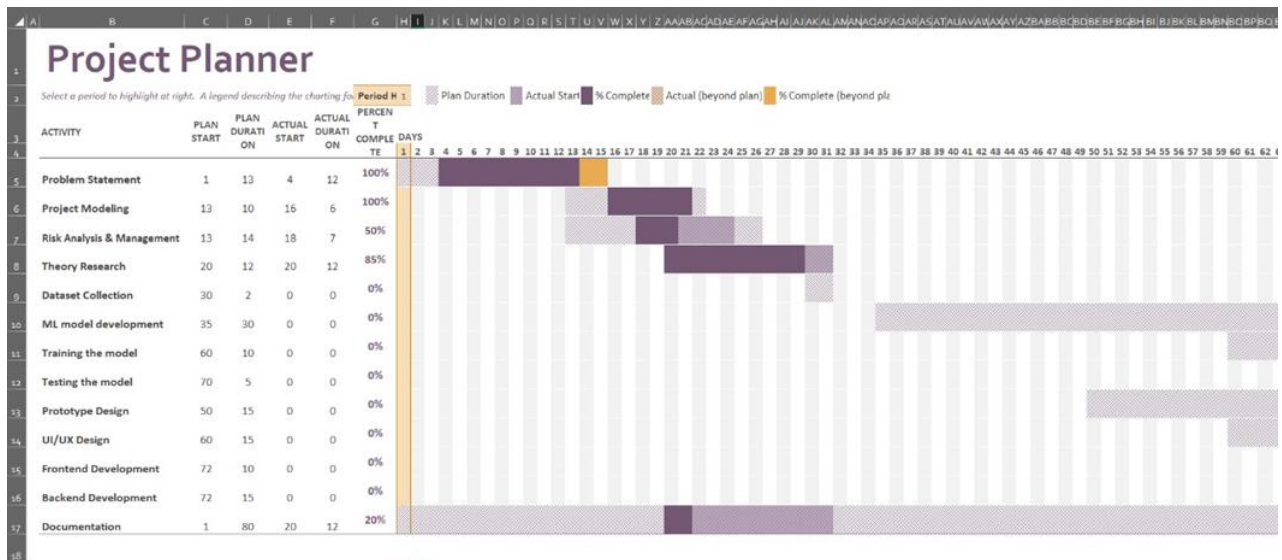
For our project we are using incremental model.

In increment model, we divide the work into different small modules. Each module goes through design, implementation and testing phases. By dividing it into modules, it enables us to work on different aspects parallelly. We will also be capable of presenting the product to the customer at different parts of the development stage and can obtain necessary feedback from them to make further modifications. Thus, even if it isn't up to the specification the customer wants, we will be able to work on it accordingly, which is not possible in other models like the waterfall model.



Chapter 2- Project Planning

2.1 Project Schedule



2.2 Efforts and Resource Estimation

The project has been divided into four modules

- Recommendation Module

This module mainly consists of the ML model which is used to implement the two functions one which recommends similar movies based on search by user and other which recommends similar movies based on movies likes by user.

This is the largest module and required most efforts for Implementation.

- User Info Module

This module consists of all the info related to a user like his Username , Email , Liked movies , Mobile number etc.

It required moderate efforts and resources and was implemented without any hassle

- Register/Login Module

This module consists of the User interface which interacts with the user mainly the register and login page

This was the easiest module of all and required least efforts and resources from our end

- **Movie Module**

This module consists of all the details about the movie including its poster, description , rating and reviews . All these details are extracted directly from the data set .

This module required moderate efforts for implementation

Chapter 3- Requirement Gathering and Analysis

3.1 SRS

Overall Description

Product Perspective

There are various recommender systems available but our product is a better replacement as it combines various module like movie recommendation , feedback systems, review and rating systems .

All this features together provide a seamless user experience.

The following diagram shows how the major components of the overall system interact with each other.

INSERT SYSTEM ARCHITECTURE HERE

Product Functions

- Search and recommend movies based on similarities
- Feedback System
- Review and rating System
- Description of movies

User Classes and Characteristics

People who love to watch movies of a particular genre are user classes of the movie recommender system. Anyone who has a basic knowledge about computer and internet is the user class like a student, adult or a senior citizen

Operating Environment

The website can be accessed on web browsers like google chrome, Microsoft Bing, Firefox and safari (13 or later), etc. on operating systems - windows, Linux or iOS. The website will be hosted through python with pipenv as the internal module.

System Features

Movie recommender

3.1.1 It would be used by users to enter a movie name and would display set a of movies which are related. It has a priority as it's the most important part of this system.

3.1.2 The user enters movie name in the search bar.
The system performs gathering operation.
It displays a set of movies related to the given movie.

Feedback system

3.2.1 It would enable the user to enter his/her reviews and rating regarding a specific movie. It has a high priority.

3.2.2 User enters the movie name
They would then be prompted to enter their review and rating

Review lookup system

3.3. It allows the user to enter a particular movie's name and would then show the respective reviews and ratings given to that particular movie. It has a high priority as it would help users pick a good movie.

3.3.2 User enters the movie's name

The system would display the reviews and rating regarding that movie

Search bar

3.4.1 It allows the user to enter a movie and displays a brief description of the plot, cast, etc. it has medium.

3.4.2 User enters movie name

The system displays a brief description and the cast

External Interface Requirements

User Interfaces

The system would have a search bar wherein users can enter the movie name and the system would recommend movies based on the similarities between them.

It would have a Feedback system, wherein the users can give their reviews and rating regarding different movies.

It would have a review system, wherein the users can enter movie names and can view the respective reviews and rating.

Hardware Interfaces

Our system doesn't have any specific hardware requirements, it would run on most of the latest computers and mobile phones.

Software Interfaces

Python Functions: Numpy, Pandas, Ast -> literal_eval: used to convert string to list for easy extraction, Nltk -> PorterStemmer: Natural Language Toolkit function porter stemmer is used to convert words with similar meaning like love, loving, loves into a single word love. This is used to get more accurate results, Sklearn -> CountVectorizer: CountVectorizer converts a collection of text documents to a matrix of token counts. These tokens are the most repeated words in the text, Sklearn -> cosine_similarity: Compute cosine similarity between sample

Nonfunctional Requirements

1. Performance Requirements

Performance of the software is one of the most important quality and can be very important factor to increase the traffic to the website. It defines how fast a system can respond to a user's action. It should not be more than 1-2 seconds.

2. Usability Requirements

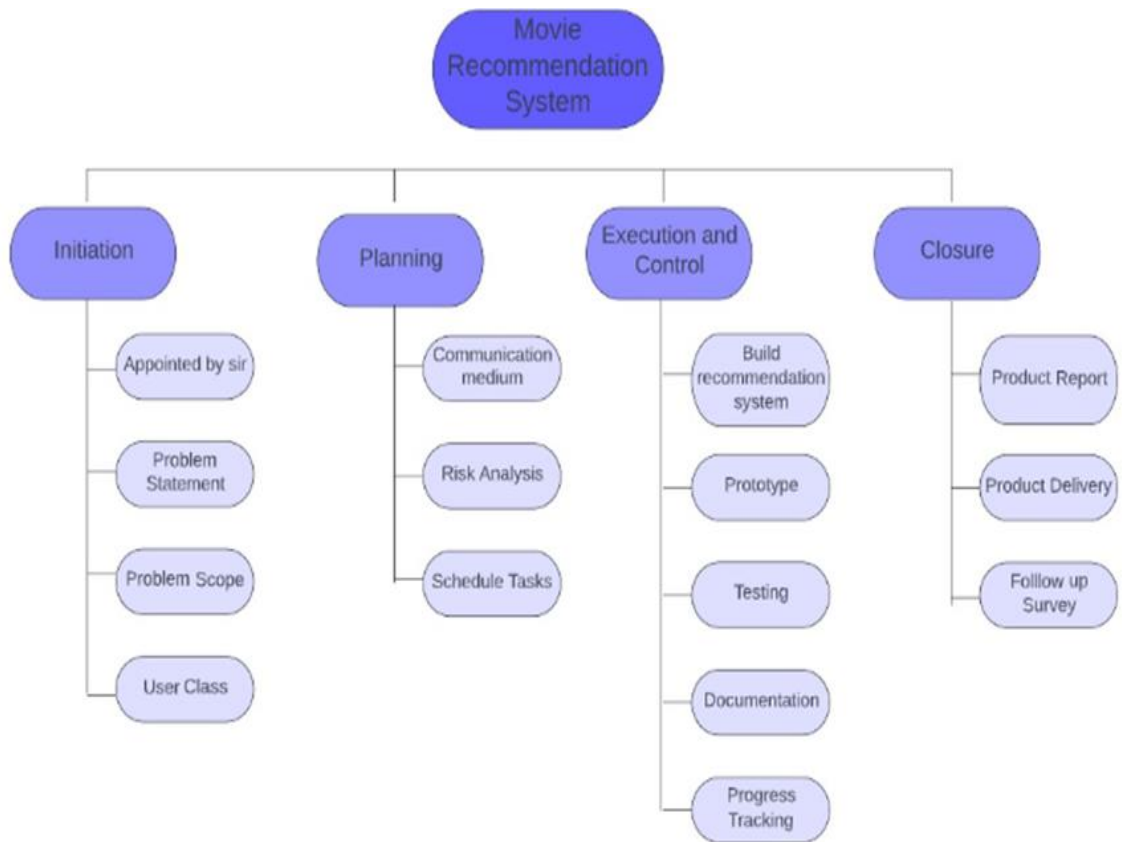
Regardless of the size of your company, even a non-technical user should be able to easily utilise your website. The user take maximum 1 second to decide if he should continue using that website. This can be achieved by focusing on readability, using proper navigation bar and limited Scrolling be it horizontal or vertical.

3. Security Requirements

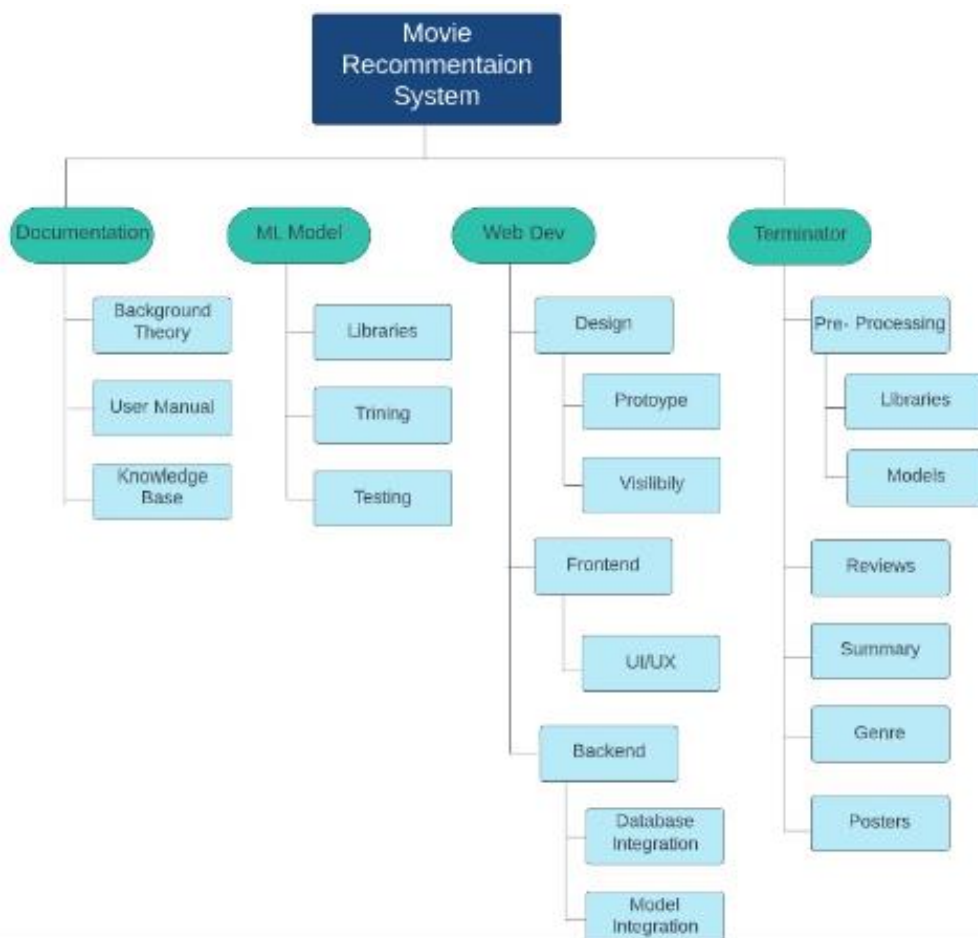
The web application deals with user's personal data like email address, phone number, etc so it is very important to make sure that the data is safe and is not leaked. For example, a cyber attacks may lead to leak of data. This can be achieved by using SSL certificates and some data privacy policies.

3.2 Data Modelling

Phase Based WBS:

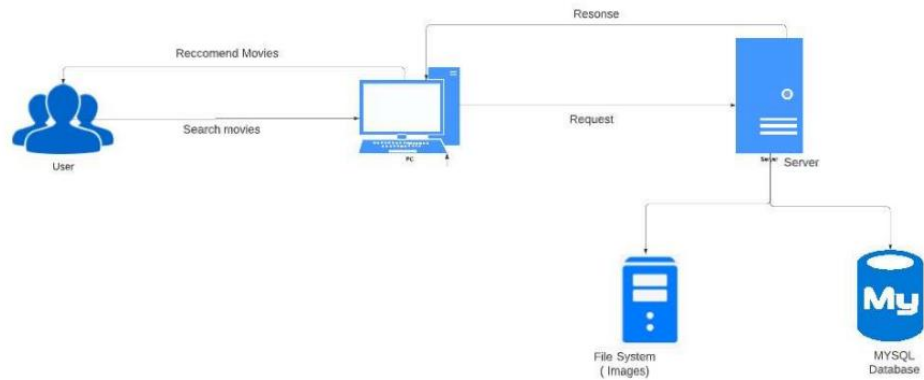


Product Based WBS:



3.3 Structural Analysis

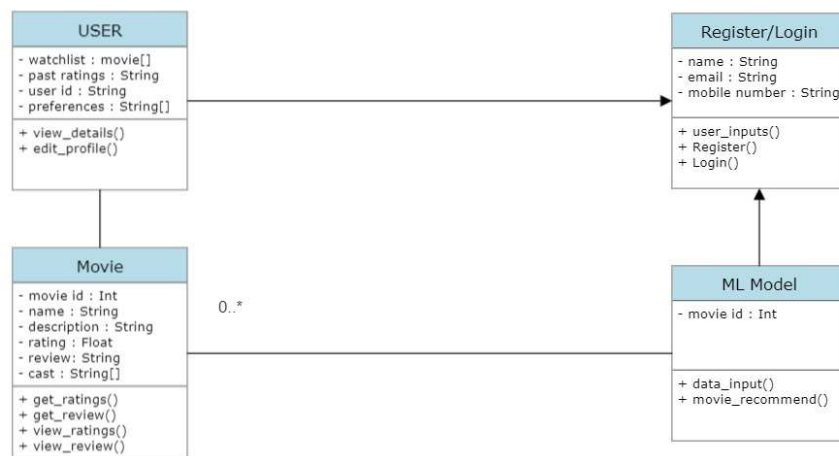
Client-server architecture, also known as the client-server model, is an application network that divides tasks between clients and servers that are either part of the same system or require network communication. The server-client sends the request to another programme, which in turn runs a few programmes that divide the work among the clients and share resources with them, in order to access the service offered by the server. The request-response pattern is represented by a client-server relationship, which should adhere to the common communication protocol that outlines the terms and guidelines for communication. The TCP protocol family is used for client-server communication.



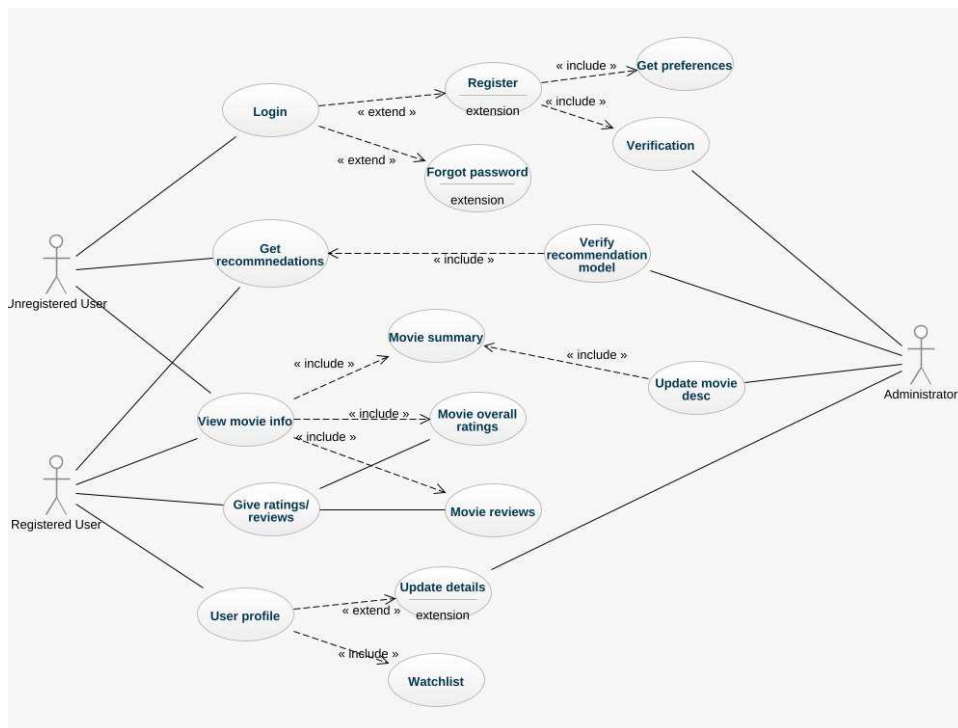
Chapter 4 – Design

4.1 UML

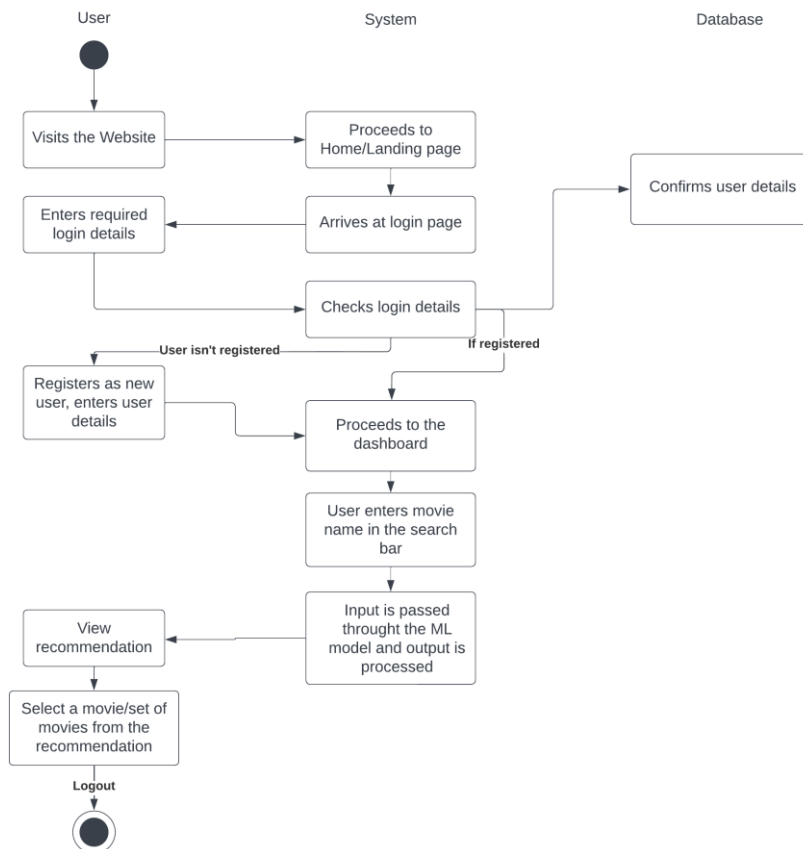
Class diagram:



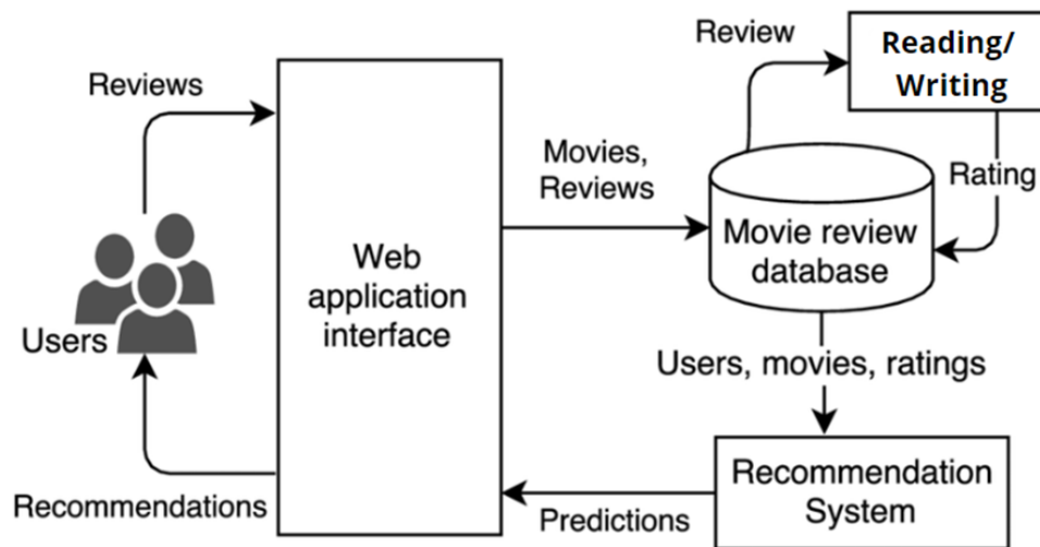
Use case diagram:



Activity diagram:



4.2 Algorithms and flowchart



Chapter 5 – Development

5.1 Tools Description and Development approach used

Software Interfaces used

Python Functions: Numpy, Pandas, Ast -> literal_eval: used to convert string to list for easy extraction, Nltk -> PorterStemmer: Natural Language Toolkit function porter stemmer is used to convert words with similar meaning like love, loving, loves into a single word love. This is used to get more accurate results, Sklearn -> CountVectorizer: CountVectorizer converts a collection of text documents to a matrix of token counts. These tokens are the most repeated words in the text, Sklearn -> cosine_similarity: Compute cosine similarity between samples

Python libraries used

Python Libraries used - Pandas - Numpy - StreamLit - Ast - nltk.stem.porter - sklearn.feature_extraction.text - sklearn.metrics.pairwise

5.2 Pseudocodes of important modules

1 . REGISTER/LOGIN

Log in module will be handling the details of the users who wants to use the movie recommendation System. In this module, new users will be registering on the website giving the necessary details like their username, email id, phone number, password and finally the confirm password for confirmation of the password entered. Then they have to log in with the username and password and can use the movie recommender for watching reviews, rating, etc. Existing users can log in to the home page directly.

Pseudocode:

```
import streamlit as st
import streamlit_authenticator as stauth
from deta import Deta

deta = Deta("d0ufgqex_PrKnXdndhPz5cJf49zgkbuzDMwzASzDN")

user_db = deta.Base("user_db")

def insert_user(username, name, email, password):
    return user_db.insert({"key": username, 'name': name, 'password': password,
'email': email, 'liked':[]})

# Connect to Deta Base with your Project Key

st.header("Create new account")
with st.form("form"):
    name = st.text_input("Name: ")
```



```

# Age = st.text_input("Your Age")
username = st.text_input("Username: ")
email = st.text_input('Email: ')
password = st.text_input("Password", type='password')
conf_password = st.text_input("Confirm Password", type='password')
submit = st.form_submit_button("Register")
if submit:
    if password == conf_password:
        insert_user(username, name, email, password)
        st.success("Registered Successfully")
    else:
        st.error("Password should match")

```

Log in :

```

if authentication_status == True:

```

```

    st.title("User Page")

```

```

def fetch_all():

```

```

    res = user_db.fetch()

```

```

    return res.items

```

```

users = fetch_all()

```

```

usernames = [user["key"] for user in users]

```

```

passwords = [user["password"] for user in users]

```

```

names = [user["name"] for user in users]

```

```

email = [user["email"] for user in users]

```

```

liked = [user['liked'] for user in users]

```

```

hashed_passwords = stauth.Hasher(passwords).generate()

```

```

credential = {'usernames': {}}

```

```

for un, nm, pw, l in zip(usernames, names, hashed_passwords, liked):

```

```

    user_dict = {'name': nm, 'password':pw, 'liked':l}

```

```
credential['usernames'].update({un:user_dict})
```

```
authenticator = stauth.Authenticate(names, usernames, hashed_passwords,  
'cookie_name', 'signature', cookie_expiry_days=30)
```

```
name, authentication_status, username = authenticator.login("Login", "main")  
elif authentication_status == False:  
    st.error("Username/password is incorrect")
```

```
elif authentication_status == None:  
    st.warning("Please enter your username and password")
```

2. USER INFO

This module describes various details about a particular user who is using the system. The different fields included are,

Username

Password

Email address

Age

Also includes functions to edit different fields such as phone number, password, username. It packs all information regarding a particular user into a single module.

3. MOVIE MODULE

Movie module contains all the data about a movie which includes

- Movie_Id
- Movie_Name
- Description
- Rating
- Review

It also includes function to get and view these data from the user.

Overall, it packs all the data and functions regarding a movie into a single module

4. RECOMMENDATION MODULE

This module is used to get the recommendation for movies. Recommendations are given in 2 forms, first the search engine recommendations and second the general user recommendation on the basis of all the movies the user like.

It is split in 3 parts

- Preprocessing phase
- Model building phase
- Prediction/Recommendation phase

1. Preprocessing phase: In this phase we load the movie dataset (movie_csv, credits.csv) and extract useful features that will be used in the recommendations like title, genre, overview, cast, and crew. After extraction process, save the changes to preprocessed_df.csv.

```
import pandas
```

```
import numpy
```

```
read the csv file named movies
```

```
read csv file named credits
```

```
display the movie dtaset
```

```
display the credit dataset
```

```
merge the movie from the movie dataset with the name title
```

```
define variable rating with the title movie_id, vote_average and vote count
```

```
define variable movies with list movies['movie_id', 'title', 'genres', 'overview',  
'keywords', 'cast', 'crew']
```

```
from ast import literal_eval
```

```
define function extract to convert the given string literal to list
```

```
function extract(paramter x)
```

define variable list[]

append the name in the list

return list

apply the function on the movies[genres and movies[keywords]]

drop from the dataset movie

```
movies['title'] = movies['title'].apply(lambda x:x.split())
```

```
movies['overview'] = movies['overview'].apply(lambda x:x.split())
```

```
movies['genres'] = movies['genres'].apply(lambda x:[i.replace(" ", "") for i in x])
```

```
movies['cast'] = movies['cast'].apply(lambda x:[i.replace(" ", "") for i in x])
```

```
movies['crew'] = movies['crew'].apply(lambda x:[i.replace(" ", "") for i in x])
```

```
movies['keywords'] = movies['keywords'].apply(lambda x:[i.replace(" ", "") for i in x])
```

Code and Output:

```
import pandas as pd
import numpy as np

[ ] movies = pd.read_csv('movies.csv')
credits = pd.read_csv('credits.csv')
```

movies.head()

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies
0	237000000	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]]	http://www.avatarmovie.com/	19995	[[{"id": 1463, "name": "culture clash"}, {"id": 1464, "name": "culture clash"}]]	en	Avatar	In the 22nd century, a paraplegic Marine is dispatched to the moon Pandora on a unique mission, but becomes torn between following orders and protecting the world he feels is his home.	150.437577	[[{"name": "Ingenious Film Partners", "id": 289}, {"name": "Twentieth Century Fox", "id": 1}], [{"name": "Twentieth Century Fox", "id": 1}], [{"name": "Twentieth Century Fox", "id": 1}]]
1	300000000	[[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]]	http://disney.go.com/disneypictures/pirates/	285	[[{"id": 270, "name": "ocean"}, {"id": 726, "name": "pirates"}]]	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, has returned. Jack Sparrow, the dread pirate who saved the world, is back.	139.082615	[[{"name": "Walt Disney Pictures", "id": 2}], [{"name": "Walt Disney Pictures", "id": 2}], [{"name": "Walt Disney Pictures", "id": 2}]]
2	245000000	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]]	http://www.sonypictures.com/movies/spectre/	206647	[[{"id": 470, "name": "spy"}, {"id": 618, "name": "thriller"}]]	en	Spectre	A cryptic message from Bond's past sends him on a new mission.	107.376788	[[{"name": "Columbia Pictures", "id": 5}], [{"name": "Columbia Pictures", "id": 5}], [{"name": "Columbia Pictures", "id": 5}]]

credits.head()

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "credit_id": "52fe48009251416c750aca23", "de...}	[{"credit_id": "52fe48009251416c750aca23", "de...}
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Sparrow", "credit_id": "52fe4232c3a36847f800b579", "de...}	[{"credit_id": "52fe4232c3a36847f800b579", "de...}
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "credit_id": "54805967c3a36829b5002c41", "de...}	[{"credit_id": "54805967c3a36829b5002c41", "de...}
3	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Batman", "credit_id": "52fe4781c3a36847f81398c3", "de...}	[{"credit_id": "52fe4781c3a36847f81398c3", "de...}
4	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "credit_id": "52fe479ac3a36847f813eaa3", "de...}	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...}

```
[ ] movies = movies.merge(credits, on='title')

[ ] movies.head()
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies
0	237000000	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]]	http://www.avatarmovie.com/	19995	[[{"id": 1463, "name": "culture clash"}, {"id": 726, "name": "ocean"}]]	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[[{"name": "Ingenious Film Partners", "id": 289...}]]
1	300000000	[[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]]	http://disney.go.com/disneypictures/pirates/	285	[[{"id": 270, "name": "ocean"}, {"id": 726, "name": "ocean"}]]	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[[{"name": "Walt Disney Pictures", "id": 2}, {"id": 289...}]]
2	245000000	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]]	http://www.sonymovies.com/movies/spectre/	206647	[[{"id": 470, "name": "spy"}, {"id": 818, "name": "based on novel"}]]	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[[{"name": "Columbia Pictures", "id": 5}, {"id": 289...}]]

```
[ ] rating = movies[['movie_id', 'vote_average', 'vote_count']]
movies = movies[['movie_id', 'title', 'genres', 'overview', 'keywords', 'cast', 'crew']]

rating.head()
```

	movie_id	vote_average	vote_count
0	19995	7.2	11800
1	285	6.9	4500
2	206647	6.3	4466
3	49026	7.6	9106
4	49529	6.1	2124

```
movies.head()
```

	movie_id	title	genres	overview	keywords	cast	crew
0	19995	Avatar	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]]	In the 22nd century, a paraplegic Marine is di...	[[{"id": 1463, "name": "culture clash"}, {"id": 726, "name": "ocean"}]]	[[{"cast_id": 242, "character": "Jake Sully", "id": 289...}]]	[[{"credit_id": "52fe48009251416c750aca23", "de...}]]
1	285	Pirates of the Caribbean: At World's End	[[{"id": 12, "name": "Adventure"}, {"id": 14, "name": "Fantasy"}]]	Captain Barbossa, long believed to be dead, ha...	[[{"id": 270, "name": "ocean"}, {"id": 726, "name": "ocean"}]]	[[{"cast_id": 4, "character": "Captain Jack Sparrow", "id": 289...}]]	[[{"credit_id": "52fe4232c3a36847600b579", "de...}]]
2	206647	Spectre	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]]	A cryptic message from Bond's past sends him o...	[[{"id": 470, "name": "spy"}, {"id": 818, "name": "based on novel"}]]	[[{"cast_id": 1, "character": "James Bond", "id": 289...}]]	[[{"credit_id": "54805967c3a36829b5002c41", "de...}]]
3	49026	The Dark Knight Rises	[[{"id": 28, "name": "Action"}, {"id": 80, "name": "Crime"}]]	Following the death of District Attorney Harve...	[[{"id": 849, "name": "dc comics"}, {"id": 853, "name": "based on novel"}]]	[[{"cast_id": 2, "character": "Bruce Wayne / Batman", "id": 289...}]]	[[{"credit_id": "52fe4781c3a3684761398c3", "de...}]]
4	49529	John Carter	[[{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}]]	John Carter is a war-weary, former military ca...	[[{"id": 818, "name": "based on novel"}, {"id": 726, "name": "ocean"}]]	[[{"cast_id": 5, "character": "John Carter", "id": 289...}]]	[[{"credit_id": "52fe479ac3a36847613ca33", "de...}]]

```
[ ] from ast import literal_eval
#literal_eval is used to covert given string to list for easy exctation
def extract(x):
    list=[]
    for i in literal_eval(x):
        list.append(i['name'])
    return list

[ ] movies['genres'] = movies['genres'].apply(extract)
movies['keywords'] = movies['keywords'].apply(extract)

movies[['genres', 'keywords']].head()
```

	genres	keywords
0	[Action, Adventure, Fantasy, Science Fiction]	[culture clash, future, space war, space colon...]
1	[Adventure, Fantasy, Action]	[ocean, drug abuse, exotic island, east india ...]
2	[Action, Adventure, Crime]	[spy, based on novel, secret agent, sequel, ml...]
3	[Action, Crime, Drama, Thriller]	[dc comics, crime fighter, terrorist, secret l...]
4	[Action, Adventure, Science Fiction]	[based on novel, mars, medallion, space travel...]

```
def extract_4(x):
    List=[]
    count = 0
    for i in literal_eval(x):
        List.append(i['name'])
        count+=1
        if count >= 4:
            break
    return List

def extract_direct(x):
    List=[]
    for i in literal_eval(x):
        if i['job'] == "Director":
            List.append(i['name'])
            break
    return List

[ ] movies['cast'] = movies['cast'].apply(extract_4)
    movies['crew'] = movies['crew'].apply(extract_direct)

[ ] movies[['cast', 'crew']].head()
```

```
[ ] movies['cast'] = movies['cast'].apply(extract_4)
movies['crew'] = movies['crew'].apply(extract_direct)
```

```
movies[['cast', 'crew']].head()
```

	cast	crew
0	[Sam Worthington, Zoe Saldana, Sigourney Weave...	[James Cameron]
1	[Johnny Depp, Orlando Bloom, Keira Knightley, ...	[Gore Verbinski]
2	[Daniel Craig, Christoph Waltz, Léa Seydoux, R...	[Sam Mendes]
3	[Christian Bale, Michael Caine, Gary Oldman, A...	[Christopher Nolan]
4	[Taylor Kitsch, Lynn Collins, Samantha Morton, ...	[Andrew Stanton]

```
[ ] movies.dropna(inplace=True)
```

```
movies['title'] = movies['title'].apply(lambda x:x.split())
movies['overview'] = movies['overview'].apply(lambda x:x.split())
movies['genres'] = movies['genres'].apply(lambda x:[i.replace(" ",",") for i in x])
movies['cast'] = movies['cast'].apply(lambda x:[i.replace(" ",",") for i in x])
movies['crew'] = movies['crew'].apply(lambda x:[i.replace(" ",",") for i in x])
movies['keywords'] = movies['keywords'].apply(lambda x:[i.replace(" ",",") for i in x])
```

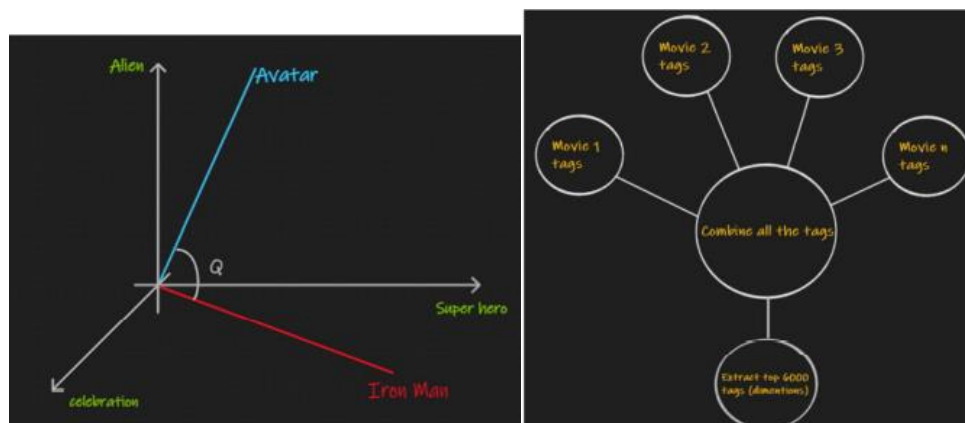
```
[ ] movies.head()
```

	movie_id	title	genres	overview	keywords	cast	crew
0	19995	[Avatar]	[Action, Adventure, Fantasy, ScienceFiction]	[In, the, 22nd, century, a, paraplegic, Marin...	[cultureclash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver, ...	[JamesCameron]
1	285	[Pirates, of, the, Caribbean, At, World's, End]	[Adventure, Fantasy, Action]	[Captain, Barbossa., long, believed, to, be, d...	[ocean, drugabuse, exoticisland, eastindiatrad...	[JohnnyDepp, OrlandoBloom, KeiraKnightley, Ste...	[GoreVerbinski]
2	206647	[Spectre]	[Action, Adventure, Crime]	[A, cryptic, message, from, Bond's, past, send...	[spy, basedonnovel, secretagent, sequel, m6, ...	[DanielCraig, ChristophWaltz, LéaSeydoux, Ralp...	[SamMendes]
3	49026	[The, Dark, Knight, Rises]	[Action, Crime, Drama, Thriller]	[Following, the, death, of, District, Attorney...	[dcomics, crimefighter, terrorist, secretiden...	[ChristianBale, MichaelCaine, GaryOldman, Anne...	[ChristopherNolan]
4	49529	[John, Carter]	[Action, Adventure, ScienceFiction]	[John, Carter, is, a, war-weary, former, mili...	[basedonnovel, mars, medallion, spacetravel, p...	[TaylorKitsch, LynnCollins, SamanthaMorton, Wi...	[AndrewStanton]

```
movie_dataset = movies.merge(rating, on='movie_id')
movie_dataset.head()
```

	movie_id	title	genres	overview	keywords	cast	crew	vote_average	vote_count
0	19995	[Avatar]	[Action, Adventure, Fantasy, ScienceFiction]	[In, the, 22nd, century, a, paraplegic, Marin...	[cultureclash, future, spacewar, spacecolony, ...	[SamWorthington, ZoeSaldana, SigourneyWeaver, ...	[JamesCameron]	7.2	11800
1	285	[Pirates, of, the, Caribbean, At, World's, End]	[Adventure, Fantasy, Action]	[Captain, Barbossa., long, believed, to, be, d...	[ocean, drugabuse, exoticisland, eastindiatrad...	[JohnnyDepp, OrlandoBloom, KeiraKnightley, Ste...	[GoreVerbinski]	6.9	4500
2	206647	[Spectre]	[Action, Adventure, Crime]	[A, cryptic, message, from, Bond's, past, send...	[spy, basedonnovel, secretagent, sequel, m6, ...	[DanielCraig, ChristophWaltz, LéaSeydoux, Ralp...	[SamMendes]	6.3	4466
3	49026	[The, Dark, Knight, Rises]	[Action, Crime, Drama, Thriller]	[Following, the, death, of, District, Attorney...	[dcomics, crimefighter, terrorist, secretiden...	[ChristianBale, MichaelCaine, GaryOldman, Anne...	[ChristopherNolan]	7.6	9106
4	49529	[John, Carter]	[Action, Adventure, ScienceFiction]	[John, Carter, is, a, war-weary, former, mili...	[basedonnovel, mars, medallion, spacetravel, p...	[TaylorKitsch, LynnCollins, SamanthaMorton, Wi...	[AndrewStanton]	6.1	2124

2. Model building: Here we build the model on the basis of which the recommendations will be given. Nltk's PorterStemmer function is used to remove morphological affixes from words, leaving only the word stem from all the dataset. Then all the different attributes of each movie are combined to get tags. Tags contain all information about a movie. From all the available tags in all the movies we find the most recurring 6000 tokens, then find the occurrence of these tokens in tags of each movie using sklearn CountVectorizer function fit_transform. This can be seen as if each movie has 600 dimensions. On the basis of this cosine similarity is calculated between each pair of movie.



3. Predictions/Recommendation: The data obtained from model building is used in giving recommendations. In simple words, a movie 'x' similarity with movie 'y' can be judged using the cosine distance between them.

Recommendations of 2 types are provided to the user, first on the basis of search engine movie; in this case, only movies having the highest cosine similarity will be obtained. Second recommendation function is general user recommendations; user will be recommended on the basis of all the movies he has liked till now. This function simply adds values of each dimension, then predicts on the basis of the new dimensions like in the first function.

Pseudocode:

```
import pandas
import numpy

read csv file movies
```

load similarity

load occurrence

define a function to search movie

function recommend_search

take the movie name as input from the user

distances = similarity[movie_index]

sort the list from the distances

print the movie list after sorting

function recommend_user

split the entered movies

CODE AND OUTPUT:

```
import pandas as pd
import numpy as np

movies = pd.read_csv('movies.csv')
similarity = np.load('similarity.npy')
occurrence = np.load('occurrence.npy')

[ ] def recommend_search():
    movie = input("Enter movie name: ")

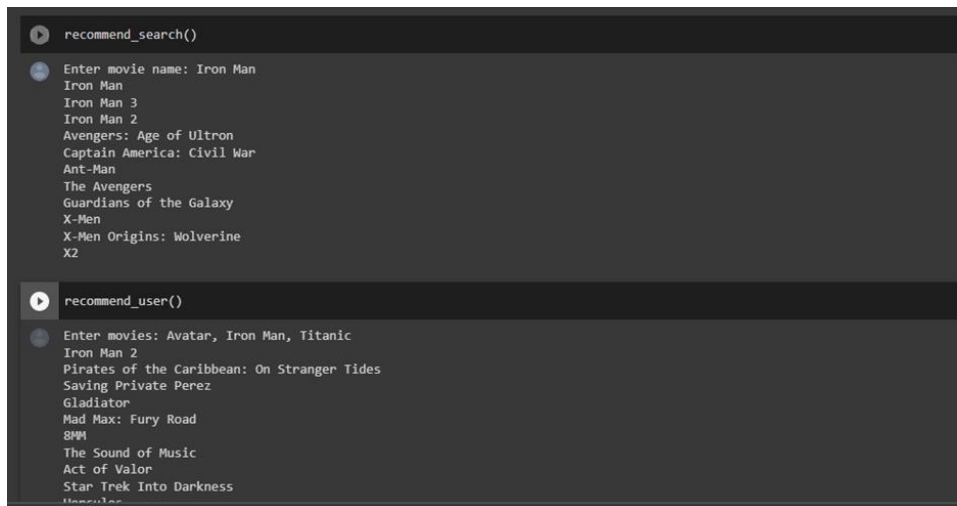
    movie_index = movies[movies['title'] == movie].index[0]
    distances = similarity[movie_index]
    movie_list = sorted(list(enumerate(distances)), reverse=True, key=lambda x: x[1])[0:11]

    for i in movie_list:
        print(movies.iloc[i[0]].title)

def recommend_user():
    num_of_movies = input('Number of movies: ')
    a = [movie for movie in input("Enter movies: ").split(", ")]

    num_of_movies = len(a)
    movie_index_list = [movies[movies['title'] == movie].index[0]
    movie_index_list = [movies[movies['title'] == movie].index[0] for movie in a]
    x = np.linspace(0, 0, 6000)
    for i in movie_index_list:
        x = x + occurrence[i]
    #cumulative vector of all movies
    list=[]
    for i in range(4806):
        t = np.dot(x, occurrence[i])
        list.append(t)
    movie_list = sorted(list(enumerate(list)), reverse=True, key=lambda x: x[1])[num_of_movies:10+num_of_movies]

    for i in movie_list:
        print(movies.iloc[i[0]].title)
```

Chapter 6 – Testing

6.1 Test cases for important modules

Login / Register Module

- Check that password entered follows constraints like alphanumeric and minimum length
- Check that user is given a forgot password option
- Check that all users have a unique username

Recommendation Module

- Test the accuracy of model
- Ensure that the model covers complete data base and result in diversity of items(does not give same result for various inputs)
- Testing the Cold start problem

Dashboard(Basic Search)

- Check error message is displayed in case of movie not found
- Check that user is redirected to movie page on successful search

Movie Page

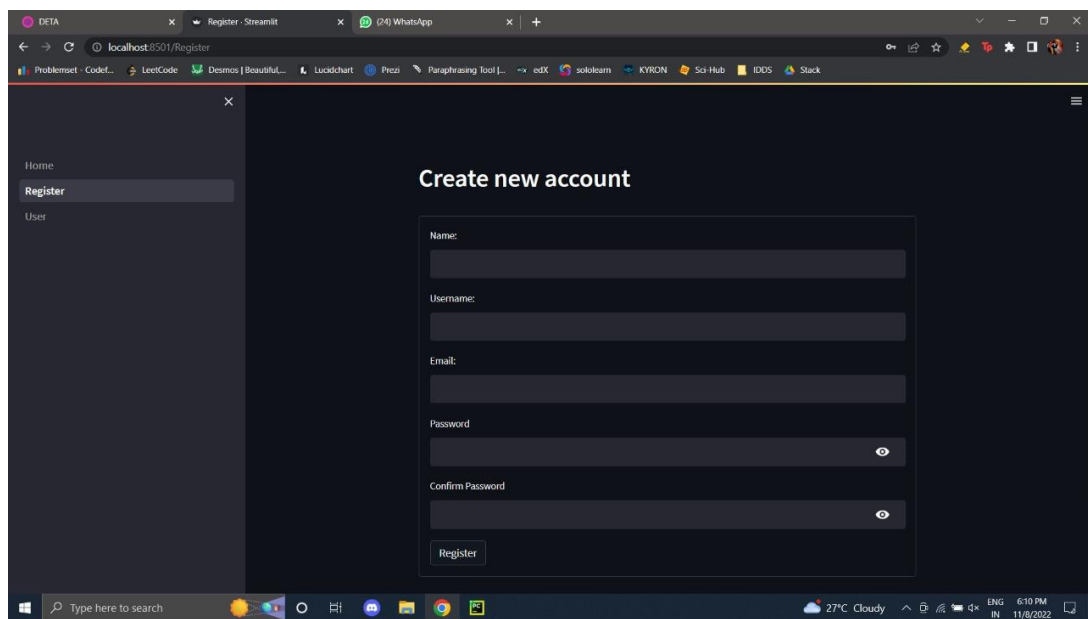
- Test correct formatting of the page
- Checking dataset for any lapses

User Profile

- Check update function works perfectly
- Check recommendation in case of no liked movies
- Check recommendation in the case when liked movies exceed a maximum limit

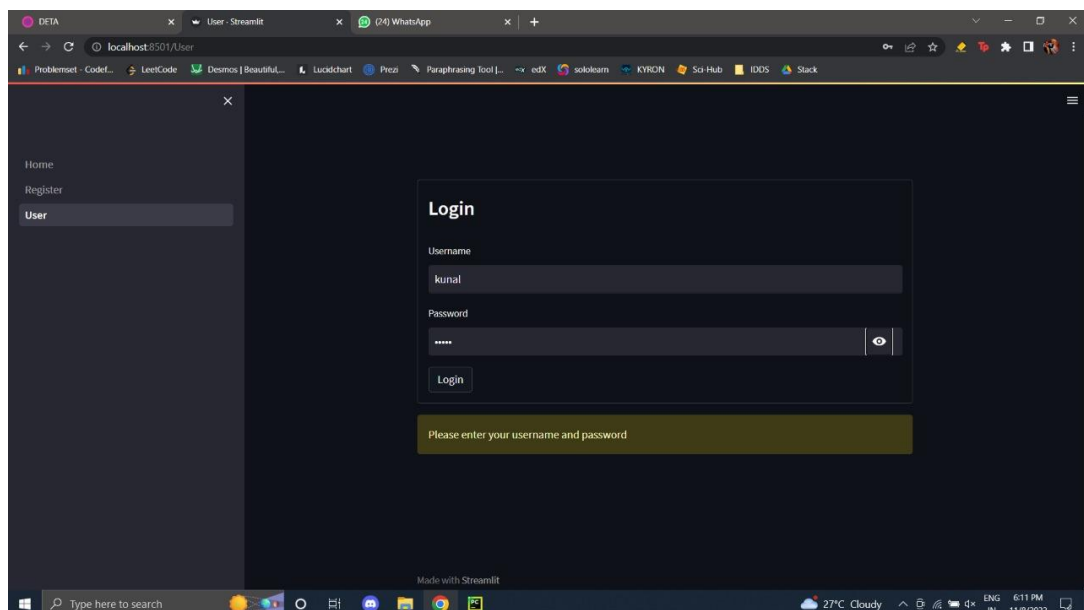
Chapter 7 – Screenshots of developed model

Register Page



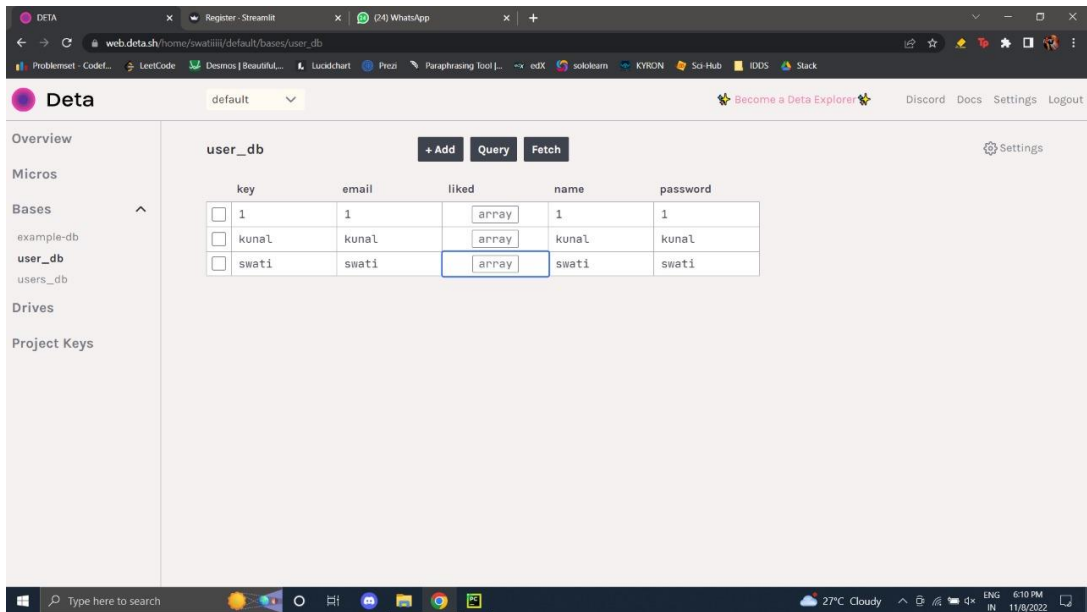
The screenshot shows a web browser window with the URL `localhost:8501/Register`. The page has a dark theme and a sidebar on the left with links for Home, Register, and User. The main content area is titled "Create new account" and contains a registration form with the following fields: Name, Username, Email, Password, and Confirm Password. Each field has a corresponding input box. There are eye icons next to the Password and Confirm Password fields to toggle visibility. A "Register" button is located at the bottom of the form. The browser's taskbar at the bottom shows the system clock as 6:10 PM on 11/8/2022.

Log in Page



The screenshot shows a web browser window with the URL `localhost:8501/User`. The page has a dark theme and a sidebar on the left with links for Home, Register, and User. The main content area is titled "Login" and contains a login form with the following fields: Username and Password. The Username field contains the text "kunal". The Password field contains six asterisks. There is an eye icon next to the Password field to toggle visibility. A "Login" button is located at the bottom of the form. Below the login form, there is a message that says "Please enter your username and password". The browser's taskbar at the bottom shows the system clock as 6:11 PM on 11/8/2022.

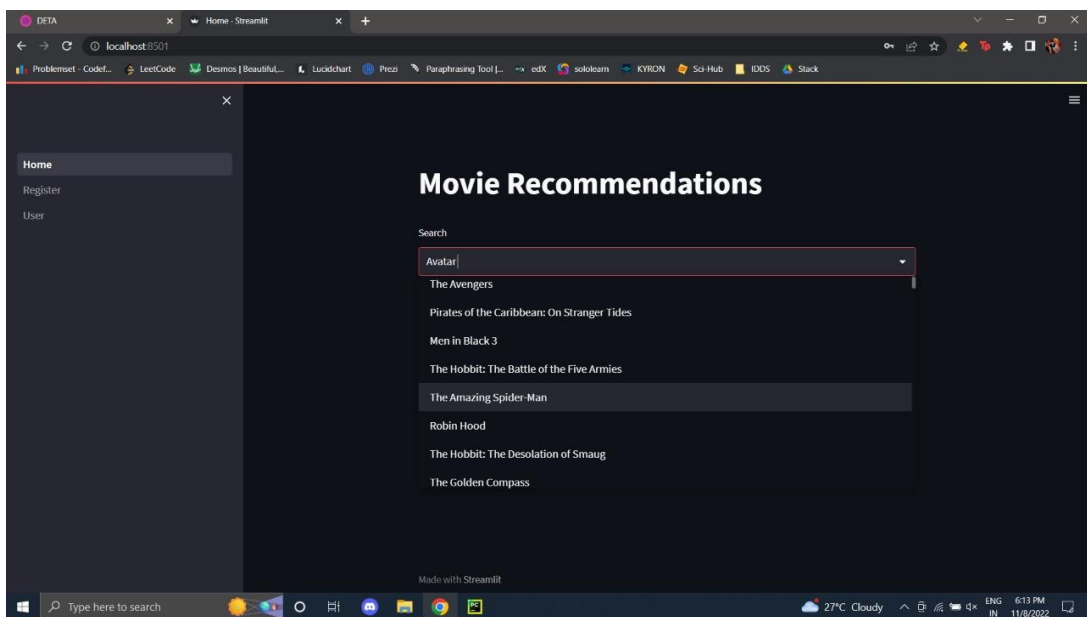
Database



The screenshot shows the DETA database interface. On the left sidebar, there are sections for Overview, Micros, Bases, Drives, and Project Keys. Under Bases, 'user_db' is selected. The main area displays a table with the following data:

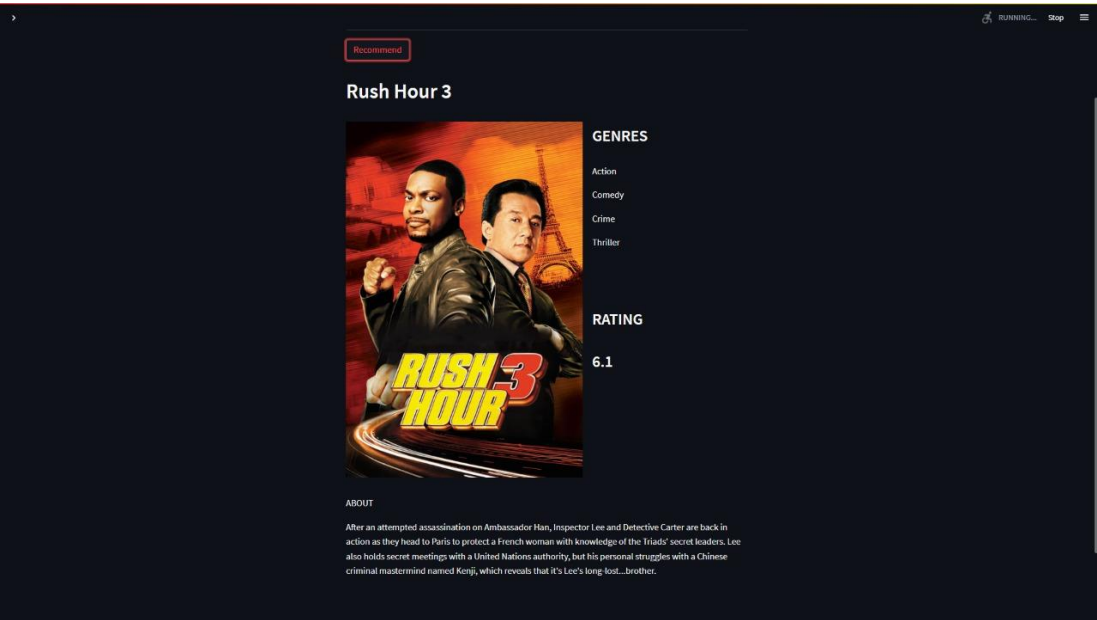
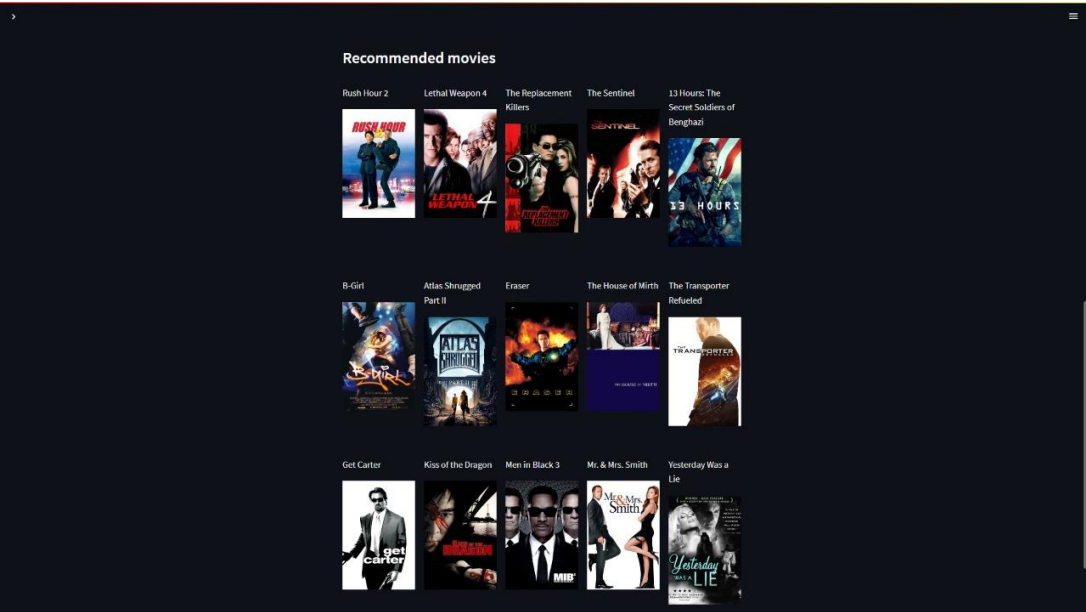
key	email	liked	name	password
1	1	array	1	1
kunal	kunal	array	kunal	kunal
swati	swati	array	swati	swati

Movie Recommendation Page



The screenshot shows a web application for movie recommendations. The sidebar on the left contains links for Home, Register, and User. The main content area is titled 'Movie Recommendations' and includes a search bar. Below the search bar, a dropdown menu displays a list of movie suggestions:

- Avatar
- The Avengers
- Pirates of the Caribbean: On Stranger Tides
- Men in Black 3
- The Hobbit: The Battle of the Five Armies
- The Amazing Spider-Man
- Robin Hood
- The Hobbit: The Desolation of Smaug
- The Golden Compass

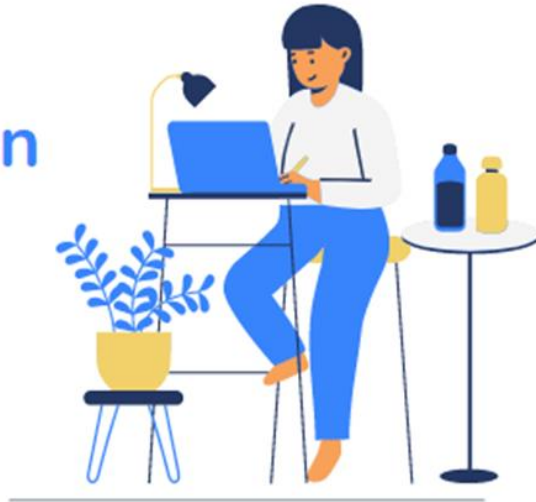


Annexures

1. Review I Presentation

Movie Recommendation System

Review - I



Team - 12

Kunal Kalra 20BCE2035

Arun Ajay 20BCE0488

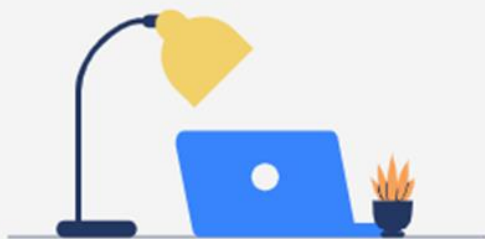
Swati Rai 20BCE0996

Shubhlaxh 20BKT0056



Problem

Identify movies that customers want to watch as per their likes and dislikes and providing them a solution give and view rating and reviews for a movie .



- ✗ Difficulty in accessing items of interest
- ✗ Time taken to search content
- ✗ Cost of searching

Solutions

The recommender system aims to provide user class personalized content and services based on their search history



Accessibility

Anyone with a device and internet can use



Easy to Use

The user friendly Interface makes the website easy to use



Performance

Reduces the cost of searching and produces results in few milliseconds

Scope

- We aim to create a tool that
- Provides personalized content
- Helps to reach the desired content faster
- Lowers the cost of searching
- A platform to get reviews and ratings on movies
- Browse movies
- Helps in Exploration
- Details /Description

Objectives

- Acquire dataset of various movies
- Train a ML model that recommends movies as per user's requests
- Develop a web interface that interacts with user
- Integrate ML model with interface
- Login option for rating and reviews

Requirement Analysis

Stakeholder Requirements

1. Client

1. Proper Marketing
2. Organised Data
3. Browser Compatibility
4. Website Deployment

2. End User

1. Accurate Results
2. Quick-Actions
3. Speed

Functional Requirements



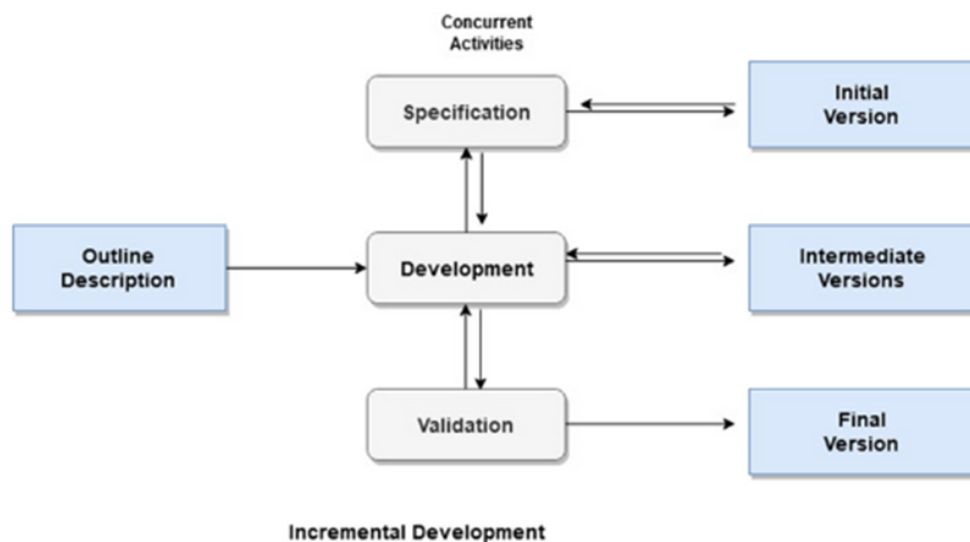
- > Sign Up for New User
- > Log In for Existing Use
- > System Shutdown
- > Responsiveness
- > Easy to use Interface

Non-Functional Requirements

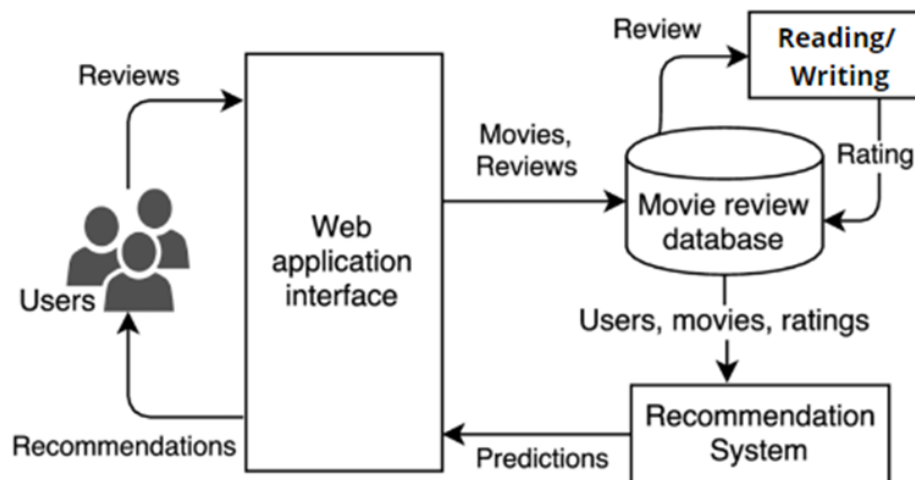


- > Usability
- > Performance
- > Security
- > Reliability
- > Portability

Process Model



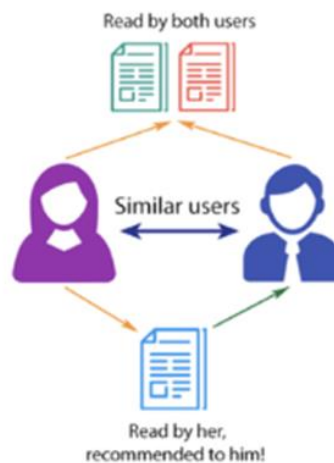
System Architecture



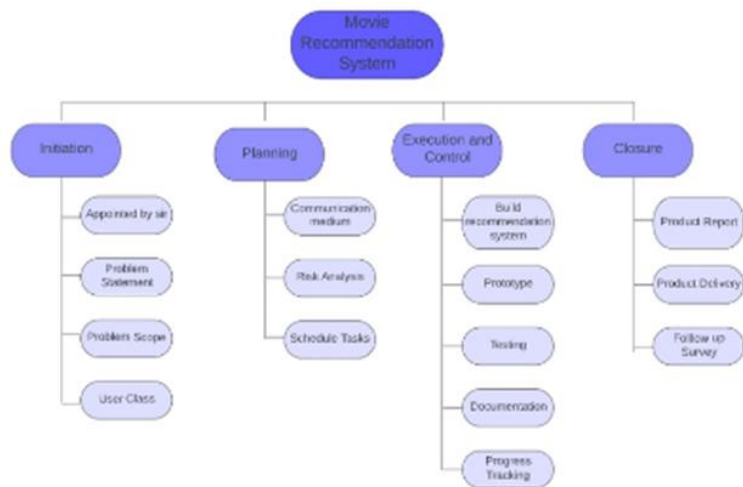
CONTENT-BASED FILTERING



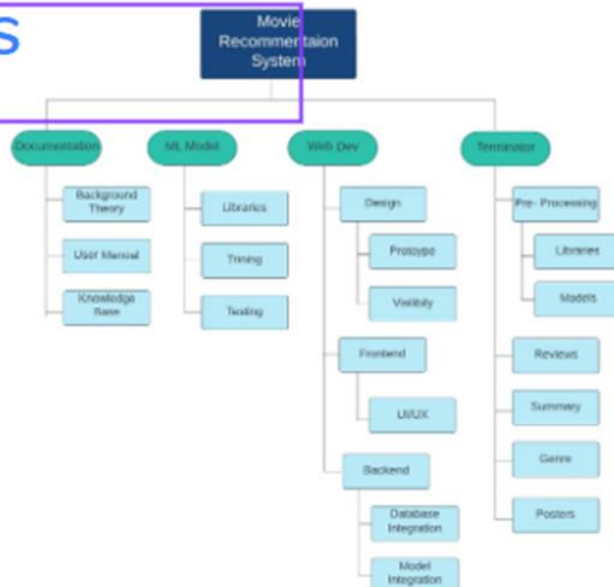
COLLABORATIVE FILTERING



Phase-Based WBS



Product WBS



Scheduling with a Gantt chart



Features:

1. User login page

It would prompt each person entering the website to enter their username and password, which then let them into their profile.

2. Movie recommender

It would be used by users to enter a movie name and would display a set a of movies which are related.

3. Feedback system

It would enable the user to enter his/her reviews and rating regarding a specific movie.

4. Review lookup system

it allows the user to enter a particular movie's name and would then show the respective reviews and ratings given to that particular movie.

5. Search bar

It allows the user to enter a movie and displays a brief description of the plot, cast, etc. It has medium.

Literature Survey

- In [1-2], Recommendation system has an ability to provide personalized information on the internet. In this era of internet, lots of RSs have been developed that are based on Content based filtering, Collaborative filtering and Hybrid system and helps to reduce the problem of information overload.
- [3] In this study, authors takes about both Collaborative filtering and Hybrid system and how they helps to reduce the problem of information overload, they also concluded that collaborative filtering recommendation system provide better recommendation but still facing problem of scalability and sparsity which can be reduced by using the fuzzy clustering and the optimization technique.
- In [4] authors came up with a hybrid model based on collaborative filtering to generate movie recommendations which combine dimensional reduction technique with the clustering algorithm. They focused on improving the approach to deal with higher dimensionality and sparsity issues in practical environment
- [5] This paper discussed the two traditional recommendation techniques and highlighted their strengths and challenges with diverse kind of hybridization strategies used to improve their performances. Various learning algorithms used in generating recommendation models and evaluation metrics used in measuring the quality and performance of recommendation algorithms were discussed.

References

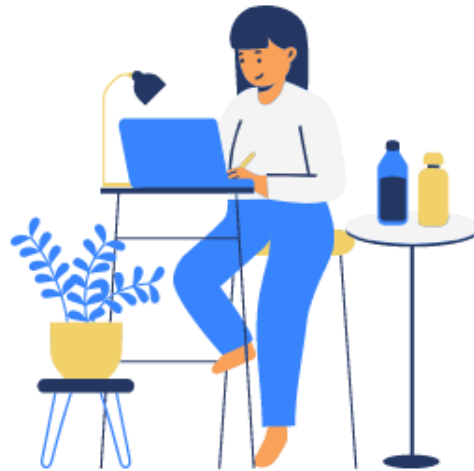
- [1] Nilloofar Zarif, Lucie Polakova, Deepansha Chhabra, What can we learn from user-movie ratings
- [2] Debashis Das, Laxman Sahoo, Sujoy Datta, (2017) A Survey on Recommendation System, International Journal of Computer Applications
- [3] Kumar, Pushpendra; Thakur, Ramjeevan Singh(2018). Recommendation system techniques and related issues: a survey. International Journal of Information Technology, (). -.doi:10.1007/s41870-018-0138-8
- [4] Wang, Zan; Yu, Xue; Feng, Nan; Wang, Zhenhua (2014). An improved collaborative movie recommendation system using computational intelligence. Journal of Visual Languages & Computing, 25(6), 667-675. doi:10.1016/j.jvlc.2014.09.011
- [5] Isinkaye, F.O.; Folajimi, Y.O.; Ojokoh, B.A. (2015). Recommendation systems: Principles, methods and evaluation. Egyptian Informatics Journal, (). S1110866515000341-. doi:10.1016/j.eij.2015.06.005

2. Review II Presentation



Movie Recommendation System

Review - III



Team - 12

Kunal Kalra 20BCE2035
Arun Ajay 20BCE0488
Swati Rai 20BCE0996
Shubhlaxh 20BKT0056



Abstract



- The movie recommender system provides personalized content to the user based on their interests.
- It involves a number of factors to create personalized lists of useful and interesting content specific to each user/individual.
- This is achieved through predictive modelling and heuristics with the data available.

Random Work

Scope



- Our project will help to recommend movies to users based on their searches and will provide them with various additional information regarding description, genres, ratings etc. It will help them to reach the desired content faster. Lowers the cost of searching of movies. A platform to get reviews and ratings on movies. It will provide a very user-friendly interface to browse through the movies and give details about them.

Existing Systems



shutterstock.com - 1708184002

Existing systems are based on clustering of data.

Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups.

Drawbacks of existing systems

Does not work well with large datasets

Does not work well with high dimensions

Too US centric

Cold start problem

Modules

Login register

Log in module will be handling the details of the users who wants to use the movie recommendation System. In this module, new users will be registering on the website giving the necessary details like their username, email id, phone number, password and finally the confirm password for confirmation of the password entered. Then they have to log in with the username and password and can use the movie recommender for watching reviews, rating, etc. Existing users can log in to the home page directly.

User Info

This module describes various details about a particular user who is using the system. The

Username
Password
Email address
Phone number
Gender
Age

Also includes functions to edit different fields such as phone number, password, username. It packs all information regarding a particular user into a single module.

Movie Module



Movie module contains all the data about a movie which includes

- Movie_Id
- Movie_Name
- Description
- Rating
- Review
- Cast

It also includes function to get and view these data from the user.

Overall ,it packs all the data and functions regarding a movie into a single module

Recommendation module

This module is used to get the recommendation for movies. Recommendation are given in 2 forms, first the search engine recommendations and second the general user recommendation on the basis of all the movies the user like.

It is split in 3 parts

- Preprocessing phase
- Model building phase
- Prediction/Recommendation phase

Proposed Systems

Client-server architecture, also known as the client-server model, is an application network that divides tasks between clients and servers that are either part of the same system or require network communication. The server-client sends the request to another programme, which in turn runs a few programmes that divide the work among the clients and share resources with them, in order to access the service offered by the server.

The request-response pattern is represented by a client-server relationship, which should adhere to the common communication protocol that outlines the terms and guidelines for communication. The TCP protocol family is used for client-server communication.

Advantages

Centralized - The primary advantage of a client-server network is the ability for centralized management. You can find all the information in one place. This is very advantageous because the network administrator has total control over management and administration.

Adaptability - Users can quickly increase the number of servers or clients. There are no issues with authorization to network resources expanding because the server is centralized.

Protection - Data is properly protected in client-server networks because of their centralized architecture. Therefore, you can only access the data within login and password as well as two-factor authentication if you are an authorized user.

Operation - Since all files are hosted on a single server, managing them is easy. The required file records can easily be monitored and accessed using a client-server network.

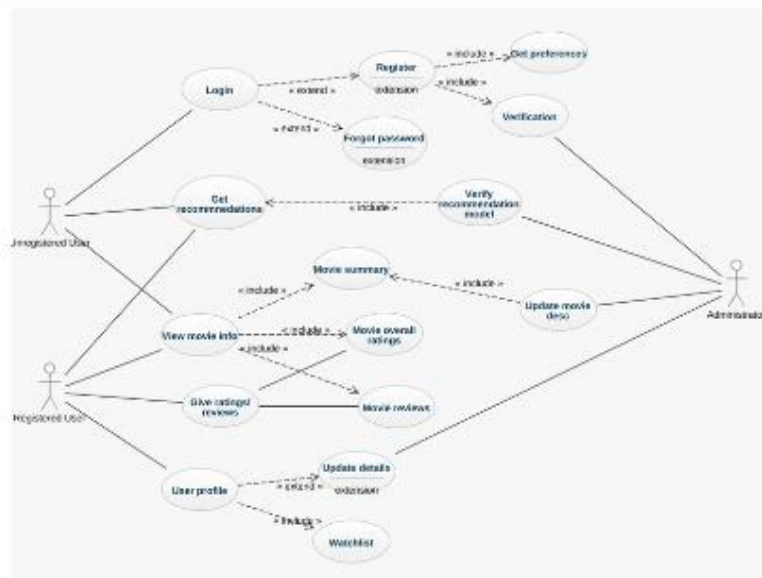
Solving the 'Cold Start' problem - It is difficult to provide a personalized experience for new website visitors for whom the system encounters no browsing history or known preferences because that information is typically used to generate recommendations.

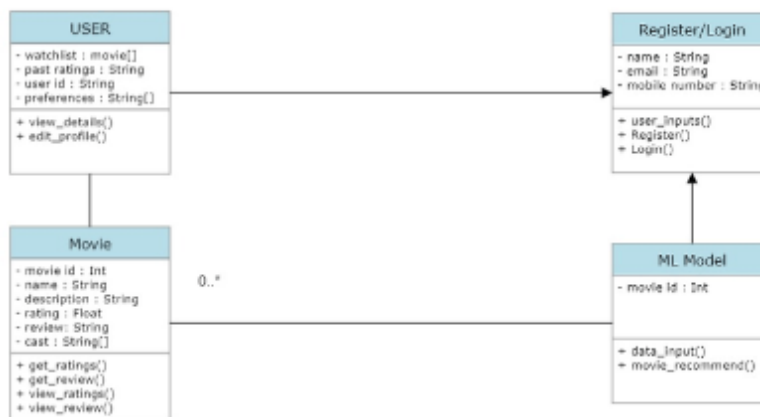
When a brand-new item is added to a web store or a brand-new piece of content is posted to a media platform, initially no one is aware of it. No matter how relevant it might be to some - or many - users, a recommendation with no interactions or ratings is essentially invisible to the system. When a brand-new item is added to a web store or a brand-new piece of content is posted to a media platform, initially no one is aware of it. No matter how relevant it might be to some - or many - users, a recommendation with no interactions or ratings is essentially invisible to the system.

In our system, the new users will be provided with a questionnaire which they'll have to fill in the beginning and provide their preferences. When a new movie is added into the dataset the corresponding details related to it will also be provided.

System Architecture

- Use case diagram
- Class diagram





Software and Hardware Requirements

Arun Ajay

Hardware Requirements:

System: any processor
Hard Disk
Ram

Software Requirements

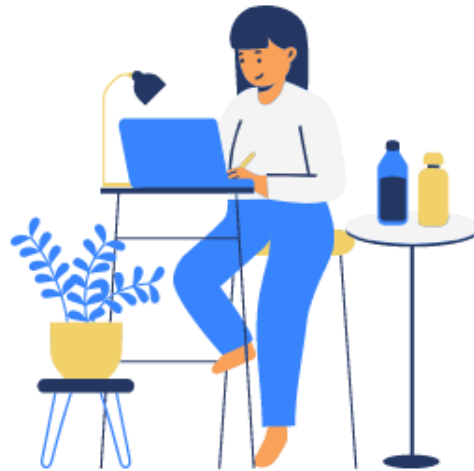
Operating Systems : Windows 7 and above
Front End : HTML,CSS,JS
Back End : PHP,MySQL
Coding language : Python
Software Environment: Jupyter Notebook,Google Collab

3. Review III Presentation



Movie Recommendation System

Review - III



Team - 12

Kunal Kalra 20BCE2035
Arun Ajay 20BCE0488
Swati Rai 20BCE0996
Shubhlaxh 20BKT0056

