Movie Recommendation and Review System

CSE3001 – Software Engineering

PROJECT BASED COMPONENT REPORT

By

ANUKRITI BHATNAGAR (19BDS0068)

MEHAK BHAMBHANI (19BDS0154)

ATISHAY JAIN (19BDS0033)

AYUSH TRIPATHI (19BDS0038)

School of Computer Science and Engineering

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

May 2021

DECLARATION

I hereby declare that the report entitled "Movie Recommendation and Review System" submitted by me, for the CSE3001 Software Engineering (EPJ) to VIT is a record of bonafide work carried out by me under the supervision of Prof. Vengadeswaran S.

I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for any other courses in this institute or any other institute or university.

Place: Vellore

Date: 30th May 2021

Anukriti
Mehak
Atishay
Ayush
Signature of the Candidate

CONTENTS

1. ABSTRACT

We have made a website which has features of rating sites such as IMDb and rotten tomatoes along with its own recommendation feature and Movie analysis page. Our website is equipped with a login page where a user has to enter their username and password or log in through their other social media accounts like twitter, google or Facebook. If a new user encounters our website, then we have a page that directs the user to create a new account. After entering our website, we offer functionalities like rating a movie, reading movie summary, graphs to analyze movies, search engine for movies and also a recommendation engine that recommends movies to watch based on user's choice.

Recommendation systems are widely used in various applications like YouTube, Spotify, Netflix, Instagram, etc. Many experts believe that the phenomenal success of Netflix could be attributes to its extremely accurate recommendation system. Recommendation systems act as a friend who knows user's preferences and makes considerate suggestion. Thus, this would not only save user's time by preventing them from watching movies that may not match their taste but also enhance user's streaming experience by suggesting the movies best matching their taste.

Showing matching results for a searched keyword and then providing their summary helps the user get all the information they require before watching a movie. This makes sure that user gets the best possible use of his/her time as they can discard movies whose direction doesn't appeal them.

Rating a movie not only gives user a sense of power of being able to influence the image of a movie but also review their choices. This enhances a user's movie streaming journey and pushes them to think about their choices.

We also show graphs for our movie geek users. We show top movies and a graph of rating of various movies.

Hence, we aim to provide all these functionalities to our users at one place to enhance their movie streaming experience and to help get them find the best movie for them. All this at one place. So, a user wouldn't have to go about searching for movies along with their summaries and asking friends for recommendations.

2. INTRODUCTION TO THE PROJECT

- OBJECTIVE:

The main objectives of the movie recommendation and review system are:

1. Giving the user's an option to rate movies.
2. Showing them personalized recommended movies based on their previous choices.
3. Showing a visualization page that has graphs for rating of different movies and a graph showing the top movies based on different metrics.
4. A search bar where users can easily search for movies based on keywords present in its title.
5. Summary of any movie containing an overview of the movie's direction, genre, storyline.
6. Platform which is easily to use, gives authenticating ratings, recommendations and a list of top-rated movies.
7. Enhancing a user's movie streaming journey by providing them all the above functionalities.
8. Prevent the user from watching a movie that don't match their taste.

- MOTIVATION:

Often, we do not find a place where we can find all the necessary information about movies along with some suggestions on which movie to watch. Because of this, we end up watching movies that we rather not thus ruining our leisure time. Thus, we plan to integrate all the above-mentioned functionalities to give the best possible experience to user by helping them find the perfect movie.

- PROJECT SCOPE:

We believe that this project would be helpful to all people ranging from small kids to senior citizens looking to relax by watching a movie. As explained above we aim to provide a unique set of functionalities that any other prominent site fails to provide.

What's in:

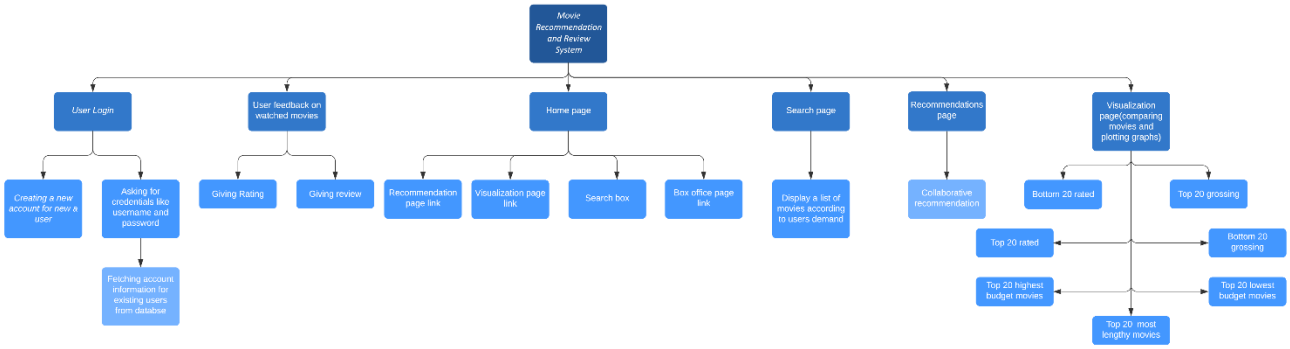
→ Search engine for movies
→ Personalized recommendation system
→ Ratign system
→ Movie summary
→ Graphs for analysis and better depiction of data

What's out:

→ Trailers of movies
→ Information of sites where they could be found
→ Latest film industry related gossip and news
→ A list of upcoming movies

3. PROJECT DESCRIPTION AND GOALS

- WORK BREAKDOWN STRUCTURE:



- PROCESS MODEL:

The movie recommendation and review system is a system that is not very complex. Its problems are well understood, changes involved in this system will be fairly limited, there aren't any high risks involved in this system, it is simple and easy to execute.

Thus, the Waterfall Process Model can be used to develop this system as it is suitable to use this process model for the above specified characteristics of the system.

Prototypes are not required hence the Prototype Model, Spiral Model and RAD Model should not be used. Also, since this system doesn't involve any high risk or a difficult deadline, the Iterative Model should not be used either.

- SOFTWARE REQUIREMENT SPECIFICATION:

- Functional Requirements:

REQ-1: Recommend movies to the users.

User's preferences of movies, user's watching history, user's searching history, other user's preferences of movies are given as input. A collaborative filtering algorithm will be trained on the input data and predict most likely movies. The output will be a list of top 10 preferences of movies. The anticipated errors here could be related to the algorithm used and the product should respond by displaying an error message to the user and restarting the algorithm.

REQ-2: Predict whether movie is hit or flop and show a complete analysis of reviews.

All the reviews made about a particular movie, comments about a movie are given as input. An NLP and deep learning model will process the reviews and returns probabilistic results. The output will be the analysis of all the reviews and prediction of the movie (hit/flop). The anticipated errors here could be related to the processing of reviews and probability calculations which could lead to the inaccurate results of interpreting whether a movie is a hit or a flop. The product should respond by rechecking calculations and processing of reviews.

REQ-3: The purpose is to serve the audience with complete details about a particular movie.

The input will be a movie ID or movie name or user response on movie selection. The movie data is fetched from the internet and saved in the database. The output is the complete detail about the movie, for example: plot, cast, trailer etc. The anticipated errors here could be related to the invalid input of movie by user, or fetching of wrong movie details. The product should respond by displaying a message asking the user to input valid movie details in the first case. In the second case, product should respond by correctly fetching movie details from the database when rerun.

REQ-4: To visualize and make a user interactive playground for users to know about their favourite movies.

The input will be all types of data related to the movies like actors, actresses, box office collection, movie budget etc. Processing involves data wrangling, managing tables, querying tables, data processing. Different types of charts like bar plot, pie charts, line charts etc., will be the output. The anticipated errors here could be in any of the processes mentioned above. The product should respond by displaying an error message, asking the user to wait and then restarting the process.

REQ-5: To let users register/sign in to the website.

The input will be Usernames, Emails, Passwords, other details. The input data is validated by the format expected, authentication, securing the data. The output will be a confirmation message/authentication message. The anticipated errors in this case can only be the invalid input given by the user. If credentials not found in the database, the system asks the user to register.

REQ-6: To get feedback/suggestion/review about the website/content.

A feedback message or email will be the input. The message will be stored in the database. The output will be the confirmation message. The only anticipated error here is the inability to store the message in the database and the product should respond by trying again.

- Non-Functional Requirements:

→ **Performance Requirements**

1) Movie data fetching:

This includes retrieving data from a huge database of movies. Fetching differs a lot in the type of data to be fetched from relational tables. The metric used is response time of the website.

2) Data storage capacity:

The overall capacity that the user can store without putting load to the server. The metric used is load time of the website.

3) Recommendation authenticity:

The recommendation must be relevant and be aligned with user requirements. Inappropriate and irrelevant data must be avoided. The metric used is recommendation algorithm accuracy.

4) Recommendation diversity:

The recommendation must work for all type of user's and all type of genres. Redundancy and repetition should be minimized. The metric used is similarity index of user's activity.

5) Critique correctness:

The prediction about the movie should not be misleading. It should respect user opinions and reviews. Two metrics are used here. The first one is critique algorithm accuracy and the second one is percentage of originality in the review.

→ **Safety Requirements**

The data stored in the database in the form of user credentials or reviews/recommendations given by the various users should not be lost in case of any damage to the system or system failure. In order to prevent this loss of data, the system and the database should be regularly monitored and they should be inspected and tested at regular intervals by the authorities concerned. Also, the database should be backed up after a regular interval of time (every 1-2 weeks).

→ **Security Requirements**

The data provided by users when they register into the website must not be shared with any other users or any external source. The recommendation/review of a movie

provided by a user must be editable by that particular user itself and no other users or external sources, i.e., no one else should be able to change someone else's review/recommendation on a movie. In addition to this, a user can recommend or give their review on a movie only if he/she registers into the website. When the user logs in, the credentials provided by him/her must be authenticated and verified by checking the database to ensure that they have, in fact, registered and are eligible to recommend or give a review on a movie.

→ **Reliability Requirements**

The system must be reliable in order to provide users with an idea of how a particular movie is, if they are planning on watching it. Recommendations should be accurate in such a way that a movie should not be recommended if that user will not like it. The users should be able to completely trust the recommendations provided by the website, which means that the processing and calculations performed must be accurate.

- System Requirements:

The system running our project should be able to connect to high-speed internet, process multiple queries at once, secure access to confidential data (user details). Expect 24 x 7 availability on any kind of Browser or any kind of device. Our system does not offer a movie trailer on the icon tap; users must Google it separately.

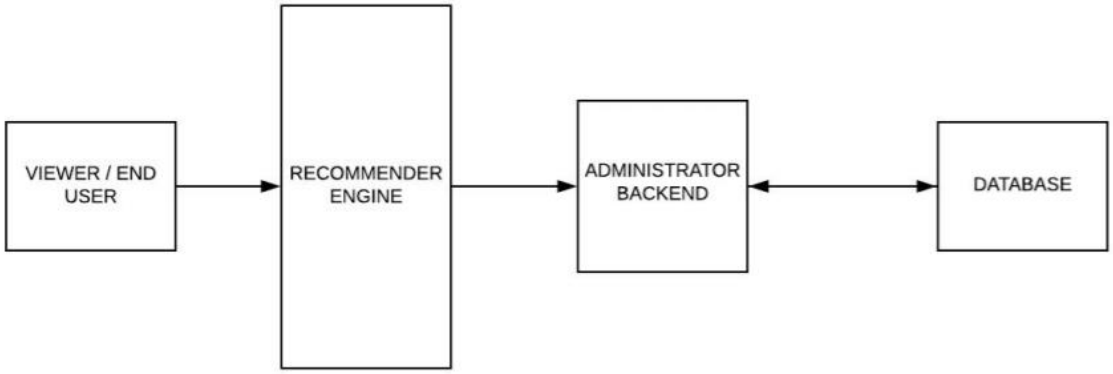
- Domain Requirements:

Domain requirements are the requirements which are characteristic of a particular category or domain of projects. The basic functions that a system of a specific domain must necessarily exhibit come under this category. For our case,

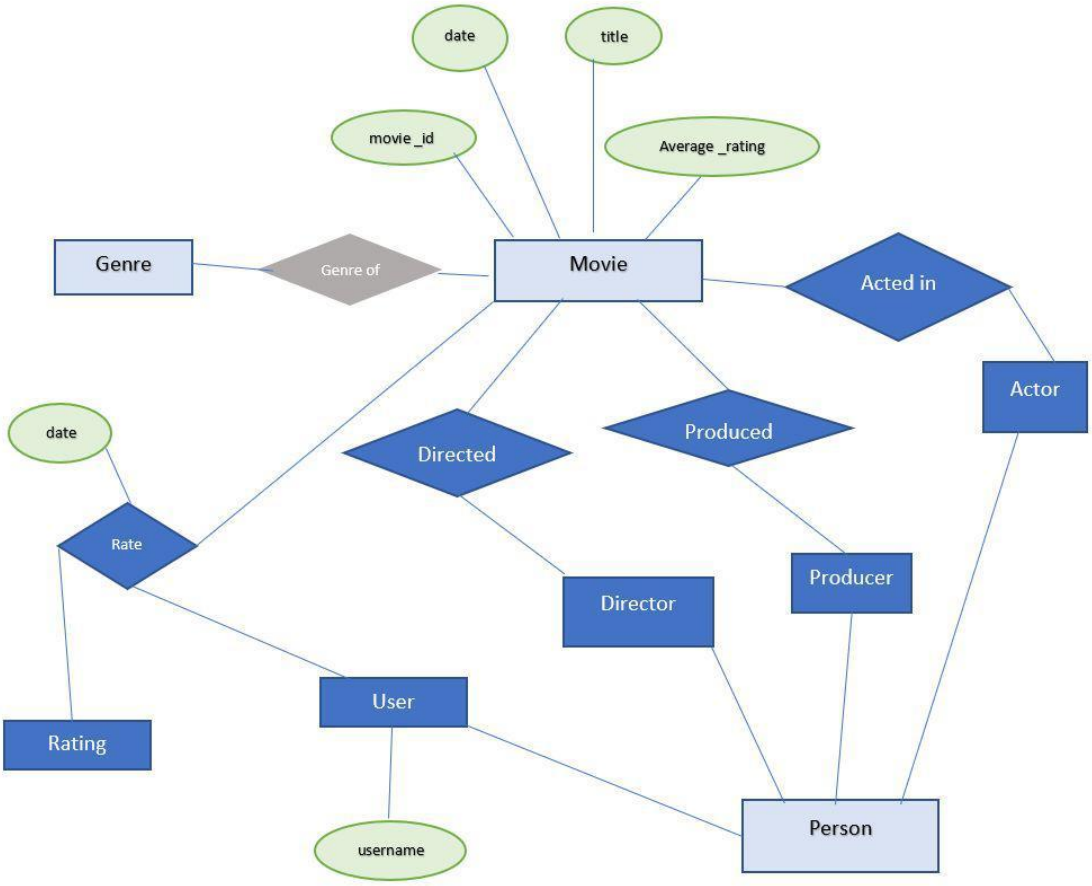
in a Movie Recommendation and Review System, the functionality of being able to visualize various graphs on the screen and being able to access the preferences as well review and ratings of the authorized is a must for our recommendation and prediction features to work.

- SOFTWARE DESIGN DOCUMENT:
  - System Design and Detailed Design:

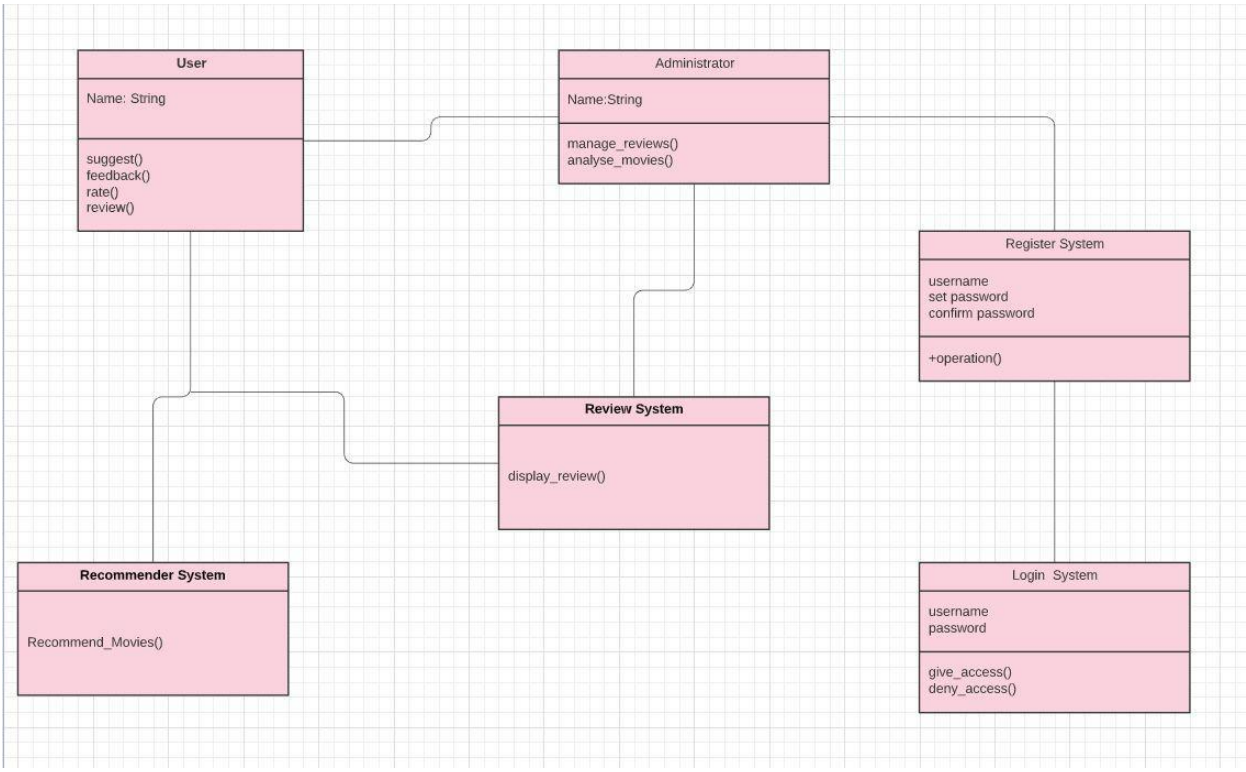
This is how the system works. The user is provided access to the recommender engine where they can give recommendations/reviews on movies or take recommendations from it. The user can also search for a movie. The recommender engine passes this information to the administrator who then inserts the recommendation/review into the database or takes the recommendations required from the database. If the user is searching for a movie, then the movie details of that particular movie are retrieved from the database. The database thus contains the movies database, recommendations database and reviews database.



→ ER Diagram:

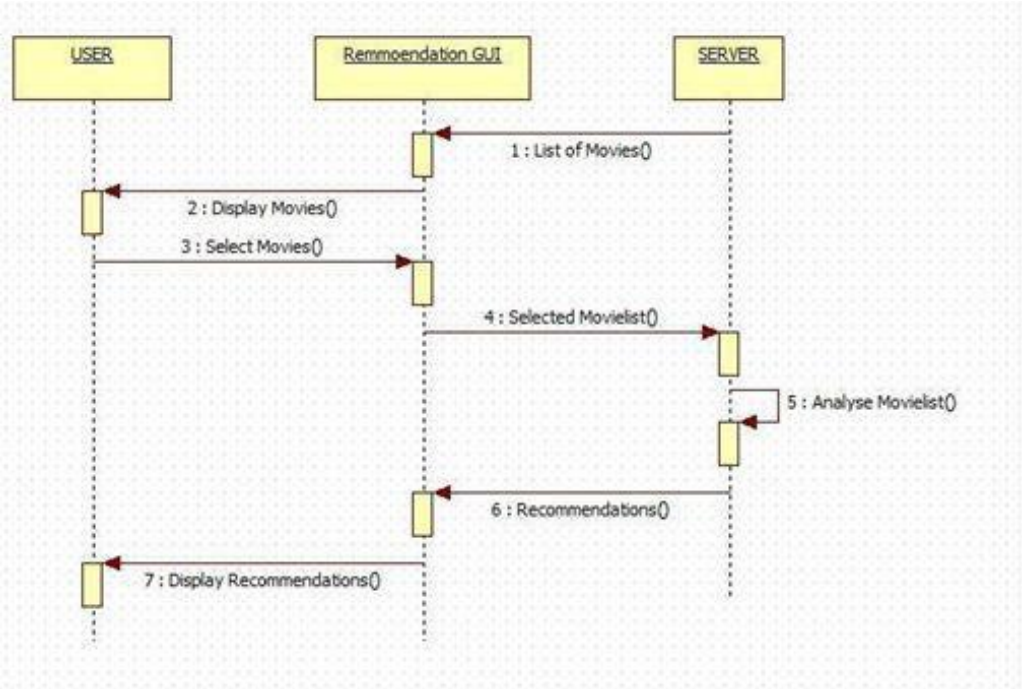


→ Class Diagram:

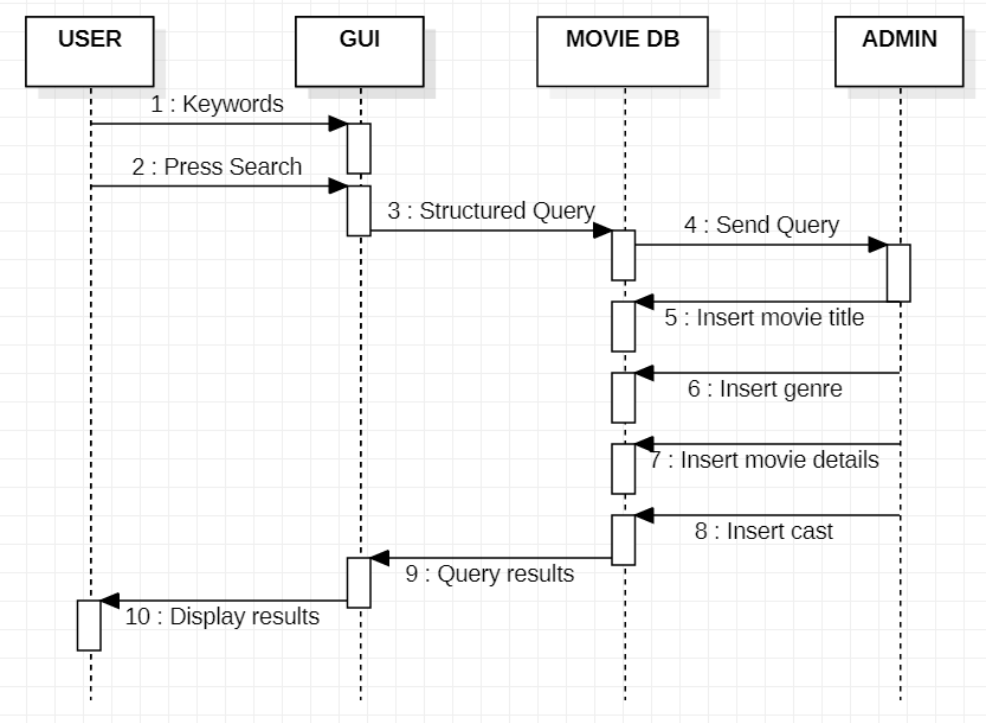


→ Sequence Diagrams:

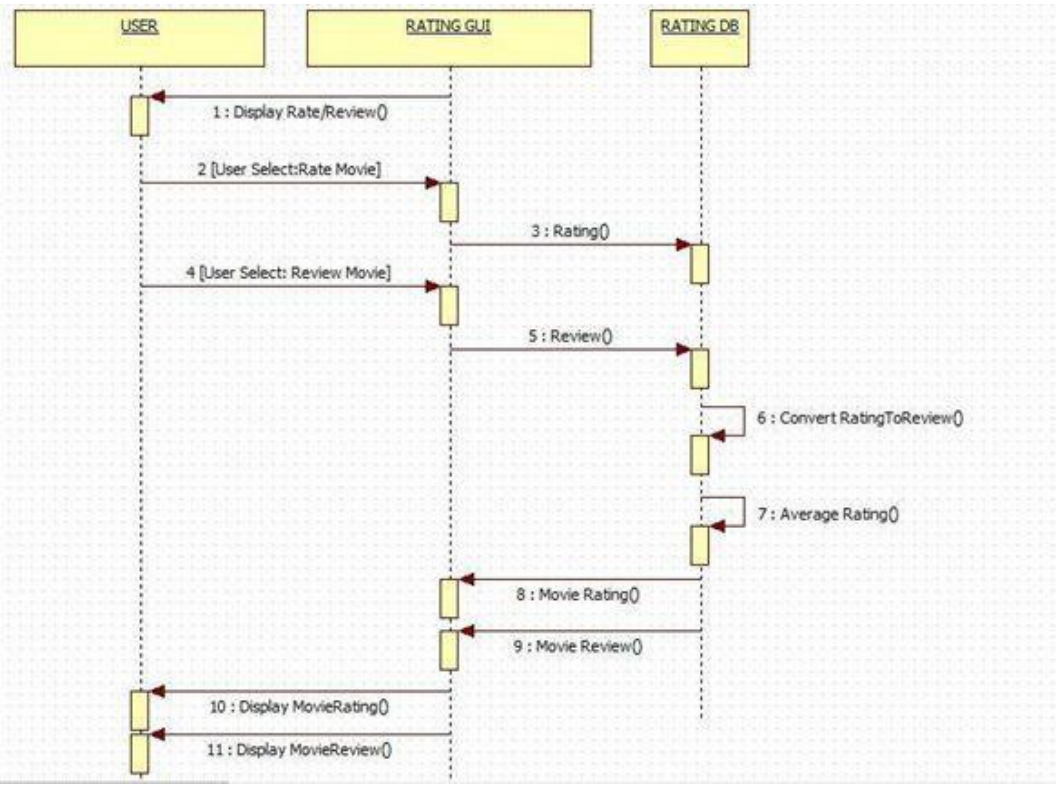
Recommendation of movies:



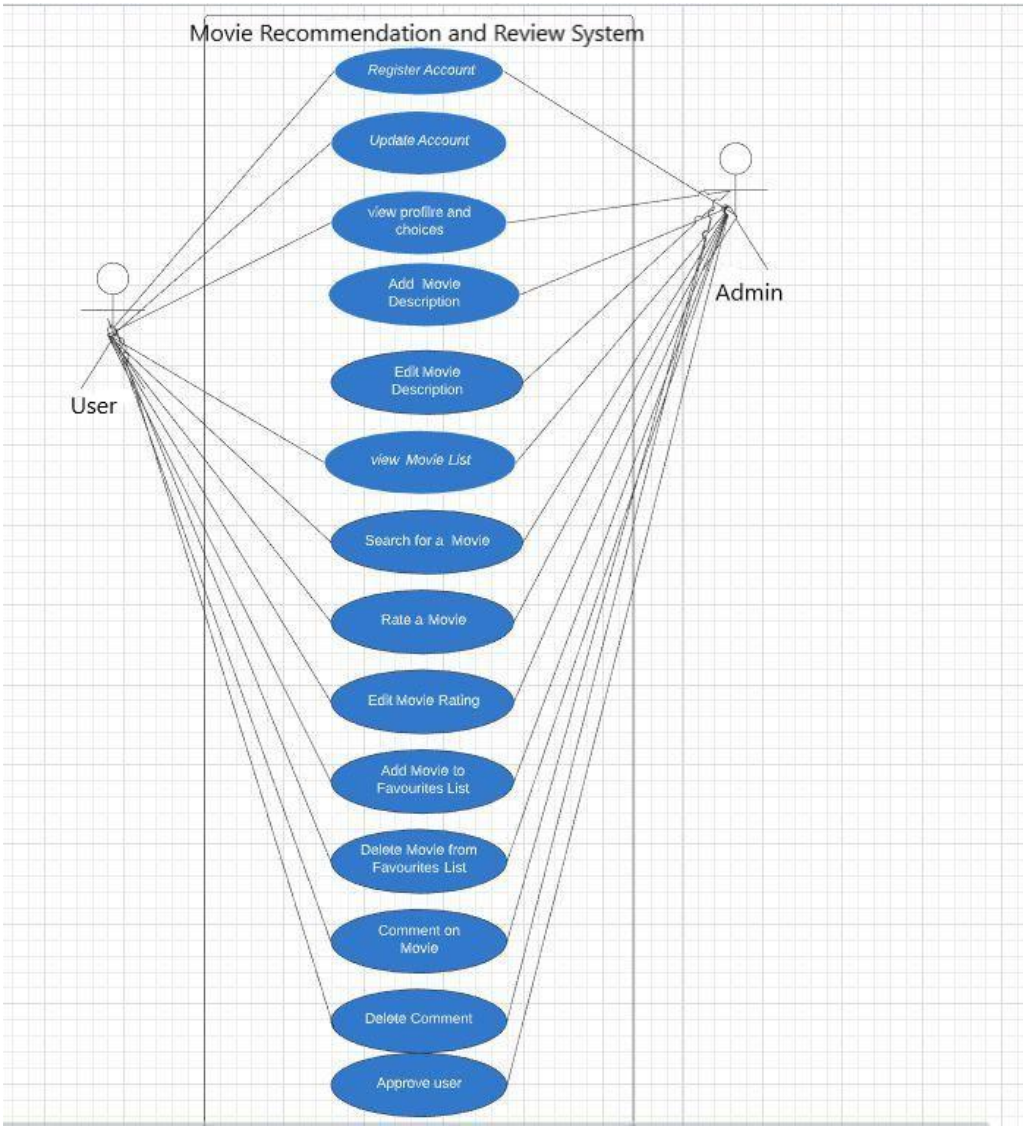
Searching for movies:



Giving rating/review on movies:



→ Use Case Diagram:



The software has been developed using Python Flask, HTML, CSS, Bootstrap, jQuery and JavaScript.

Standards followed:

- Global variables have a limited use.
- Standard headers for different modules have been mentioned.

- All naming conventions for local, global variables have been followed.
- Functions, dependencies and Libraries have been clearly stated and named.
- Proper Indentation has been followed throughout the codes.
- Exception and error handling measures have been taken for both Backend and Frontend codes.
- No identifier has a multiple usage.
- GOTO statements are not used.
- Codes are well documented.
- High Cohesion, low coupling used.
- Modularity maintained by reusing functions to create multiple cards and info pages.

4. MODULE DESCRIPTIONS

→ Sign Up Page:

The Sign Up Page has been designed so that new users who do not have a registered account with this website can create one by providing their necessary details (Email ID, Username and Password) and thus use this website to give or get recommendations/reviews on movies or simply get information about a particular movie.

→ Login Page:

The Login Page has been designed so that the users who have a registered account with this website can login to the website by entering the credentials that they used at the time of signing up (Email ID and Password). After logging into the website, they can then perform various activities such as giving or receiving recommendations/reviews or simply getting information about a particular movie.

→ Home Page:

The Home Page allows users to search for a particular movie that they wish to get information about. This page consists of a navigator at the top with links to the visualization page, recommendation page and a link to logout of the website.

→ Search Results Page:

The Search Page shows the results of the search query made by the user. It returns those movies which have the Search keyword(s) in their title.

→ Movie Information/Review Page:

The Rating Page shows the movie poster, the description about the movie, its rating, the name of the director, the names of the writers and the names of the actors. The users can even submit their own ratings in this page.

→ Recommendation Page:

The Recommendation Page shows the various movies that the recommender engine recommends for the user to watch based on the movies that have been given a good rating by the user previously. The recommender engine uses the Collaborative Filtering algorithm to recommend movies to the users.

→ Visualization Page:

The Visualization Page shows visualization of movie data based on the user's choice of axes. The users can also view the Top 20 Rated movies, Bottom 20 Rated movies, Top 20 Grossing movies, Bottom 20 Grossing movies, 20 Highest Budget movies, 20 Lowest Budget movies and the Top 20 Most Lengthy movies.

→ Logout Page:

The Logout Page thanks the user for visiting our page when the user decides to logout from their session after performing all the activities that the user intended to perform while logged in.

5. USER INTERFACE DESIGN

The User Interface for the website has been designed using Web Development tools like HTML, CSS, Bootstrap and JavaScript. The website has been designed such that it consists of seven main webpages which are the Sign Up page, Login page, Home page, Search Results Page, Movie Review page, Visualization page, Recommended Movies page and the Logout page. The other pages include a Thank You page, graphs page and an error page.

Sign Up Page:

This page asks the user to enter valid details consisting of username, email and password in a form format. The form also contains a Sign Up button for submitting the form filled by users and creating an account. For users with an existing account, the page offers a link to navigate to the Login page.

Login Page:

It provides a form for users to enter the registered username and password and a Login button for submitting. The interface is designed such that the users even have the option to login using their social media handles like Facebook and Twitter or navigate to the Sign Up page to create an account in the conventional way.

Home Page:

The Home Page displays the software logo and title alongside a navigation bar for users to navigate to other pages of the website. The users have space to enter a movie name in the text bar and search for it.

Search Result Page:

The movies are displayed as thumbnails showing the movie poster, title and a button to give/read reviews.

Review Page:

The Review page displays the details of the selected movie which include the movie title, movie poster, a brief overview, user ratings, list of actors and directors. It also allows the user to rate the movies from 1 to 10 and a submit button to submit.

Thank You Page:

Thank You page is displayed when the user submits a review and provides the user with a button to return to the Home Page.

Movie Not Found Page:

In case the movie entered by the user in the Home Page bar is not present in the database or the user has typed it incorrectly, a Movie Not Found page is displayed which gives user suggestions for movie titles based on what the user has entered in the Home Page.

Recommended Movies Page:

The Recommended Movies page displays all the top 10 movies recommended to the user based on previous browsing history. The movies are displayed as thumbnails showing movie poster, movie title, year of release and a button to read further details of the movie.

Visualization Page:

The Visualization page displays a list of buttons containing some predefined visualization conditions at the top. Clicking these buttons displays a labelled graph based on the button. The page also provides the users with an option to select their own criteria and view the result as graphs.

Logout Page:

The Logout Page displays a thank you message to the user.

6. TEST CASES

Module Name	Condition	Test Case	Expected Result	Actual Result	Test Case Pass/Fail
Sign Up Page	User enters valid details	Username: Anukriti Bhatnagar Email: anukritib123@gmail.com Password: abcd1234 Confirm Password: abcd1234	An account should be created and the Login page should be displayed	An account is created and the Login page is displayed	Pass
Sign Up Page	User enters invalid details	Username: Anukriti Bhatnagar Email: anukritib123@.com Password: abcd1234 Confirm Password: abcd1234	The page should not accept the details and refresh keeping username and password intact and email field blank	The page does not accept the details and refreshes keeping username and password intact and email field blank	Pass
Sign Up Page	User enters invalid details	Username: Anukriti Bhatnagar Email: anukritib123@gmail.com	The page should not accept the details and	The page does not accept the details and	Pass

		Password: abcd1234 Confirm Password: abcd	refresh keeping username and email intact and password field blank	refreshes keeping username and email intact and password field blank	
Login Page	User enters valid credentials	Username: Anukriti Bhatnagar Password: abcd1234	The Home Page should be displayed	The Home Page is displayed	Pass
Login Page	User enters invalid credentials	Username: Anukriti Bhatnagar Password: abcd	The page should not accept the details and refresh keeping username intact and password field blank	The page does not accept the details and refreshes keeping username intact and password field blank	Pass
Home Page	Enter keywords to search for a movie	Harry Potter	A list of all Harry Potter movies should be displayed on the page	A list of all Harry Potter movies is displayed on the page	Pass
Home Page	User types a movie which is not present in the database or misspells the movie	'Froz' is typed	The 'Movie Not Found' page should be displayed and should give user suggestions of movie titles	The 'Movie Not Found' page is displayed and user is given suggestions of movie titles	Pass
Search Results Page	Info page should be opened when Movie thumbnail is clicked	'Give Review' button of 'Harry Potter and the Goblet of Fire' is clicked	The details of 'Harry Potter and the Goblet of Fire'	The details of 'Harry Potter and the Goblet of Fire' are displayed	Pass

			should be displayed		
About Page	User enters a numerical value in submit rating option and press Submit	7.0 is entered in the review space for 'Harry Potter and the Goblet of Fire' and submit button is pressed	'Thank You' page should be displayed	'Thank You' page is displayed	Pass
Thank You Page	User clicks on 'Return to Home Page' button	'Return to Home Page' button is clicked.	The Home Page should be displayed	The Home Page is displayed	Pass
Recommended Movies Page	User clicks on the 'Recommended Movies' on the navigation bar	'Recommended Movies' on the navigation bar is clicked	Movies similar to user's browsing history should be displayed	Movies similar to user's browsing history are displayed	Pass
Visualization Page	The user selects conditions for x and y axes	'IMDb ratings' is selected for y axis and 'movie title' is selected for x-axis	The graph of movie titles versus IMDb ratings should be displayed	The graph of movie titles versus IMDb ratings is displayed	Pass
Visualization Page	The user selects conditions for x and y axes	'Genres' is selected for y axis and 'movie title' is selected for x-axis	Message should be displayed saying that numerical values should be selected for y-axis	Message is displayed saying that numerical values should be selected for y-axis	Pass
Visualization Page	The user clicks on one of the buttons for pre-defined visualization conditions	The button for 'Top 20 Movies' is clicked	The graph for 'Top 20 Movies' should be displayed	The graph for 'Top 20 Movies' is displayed	Pass
Logout Page	User clicks the Logout button in	The Logout button is clicked	The user should be	The user is logged out	Pass

	the navigation bar		logged out of the website	of the website	

7. SOFTWARE METRICS

Cost Analysis (Development Phase) based on Size Oriented Metrics–

COCOMO (Constructive Cost Estimation Model) is one of the world's most widely used software estimating models. It estimates the time and effort required to complete a software product based on its size.

Organic: We may classify our development project as organic because it involves the creation of a well-understood application program, the development team is relatively small, and the team members have previous expertise with similar project approaches.

The equation used to calculate the required effort E to build a software product, represented in person months and the expected time T to create the software in months is provided below.

$E = a (KLOC)^{b}$ PM

$T = c (Effort)^{d}$ Months

The value of the constants a, b, c and d is determined by the type of project. Since it is an organic type of project we take the values of a, b, c, d to be equal to 2.4, 1.05, 2.5 and 0.38 respectively. The expected size of the software package is expressed in Kilo Lines of Code (KLOC).

Following is an approximation of the Lines of Code for the front-end and back-end of each module. LOC is affected by the expertise in using a particular tool so as to reduce the number of lines. It is also dependent on

the complexity of the task to be performed.

Modules	Front – end LOC	Back – end LOC
Sign Up page	60	31
Login Page	95	11
Home Page	49	18
Search Results page	45	7
Review Page	49	4
Thank You Page	26	4
Movie Not Found Page	281	4
Visualization Page	138	32
Recommended Movies page	53	88
Logout Page	19	4

Total LOC for front – end = 60 + 95 + 49 + 45 + 49 + 26 + 281 + 138 + 53 + 19 = 815

Total LOC for back – end = 31 + 11 + 18 + 7 + 4 + 4 + 4 + 32 + 88 + 4 = 203

Total LOC for the project = 815 + 203 = 1018

Thus,

Effort = 2.4 (KLOC) $^{1.05}$ PM = 2.4 (1018 / 1000) $^{1.05}$ = 2.4 (1.018) $^{1.05}$ = 2.4 (1.018908) = 2.445380 Person – months.

T = 2.5(Effort) $^{0.38}$ Months = 2.5 (2.445380) $^{0.38}$ = 2.5 (1.40466129) = 3.611653 Months

Persons required = Effort / Time = 1.18344 / 2.665 = 0.6777 ~ 1 person

Thus, cost require for development phase = Average salary * Number of persons = Average salary * 1 = Average salary.

Cost Analysis (Deployment Phase) –

Since this project is a web server hosted program, the only expenses are the cost of web hosting. This can be collected by clicking on advertisements on the website.

The client/end user will not be charged for this.

Furthermore, revenue can be made over a long period of time.

8. CONCLUSION

A movie critique system has been successfully developed, allowing users to check in and choose if a film is a hit or a flop, as well as receive recommendations based on their tastes, budget, gross income, rating, and gamification.

Movies are divided into categories based on their titles, languages, and actors.

After gathering (explicitly or implicitly), processing, and evaluating user or object data, recommendations are made. However, because there is such a wide range of data to handle, building RS is a manual and time-consuming operation for programmers.

One method to facilitate procurement of needs, and the resulting RS development, is the definition of a general user and item model that is

compatible with any domain.

The purpose of this study is to describe a systematic review that aims to identify user and item information used in advanced and implemented RS. The low number of RSs with a collaborative filter suggests that user profiles were not compared or that the user's historical information was not thoroughly researched.

This finding could be explained by privacy concerns over user data.

Because it has access to great databases like Movielens and IMDB, the ease of access to real data is a key aspect in this result. We can advise them on how to make the most of their visualization tools.

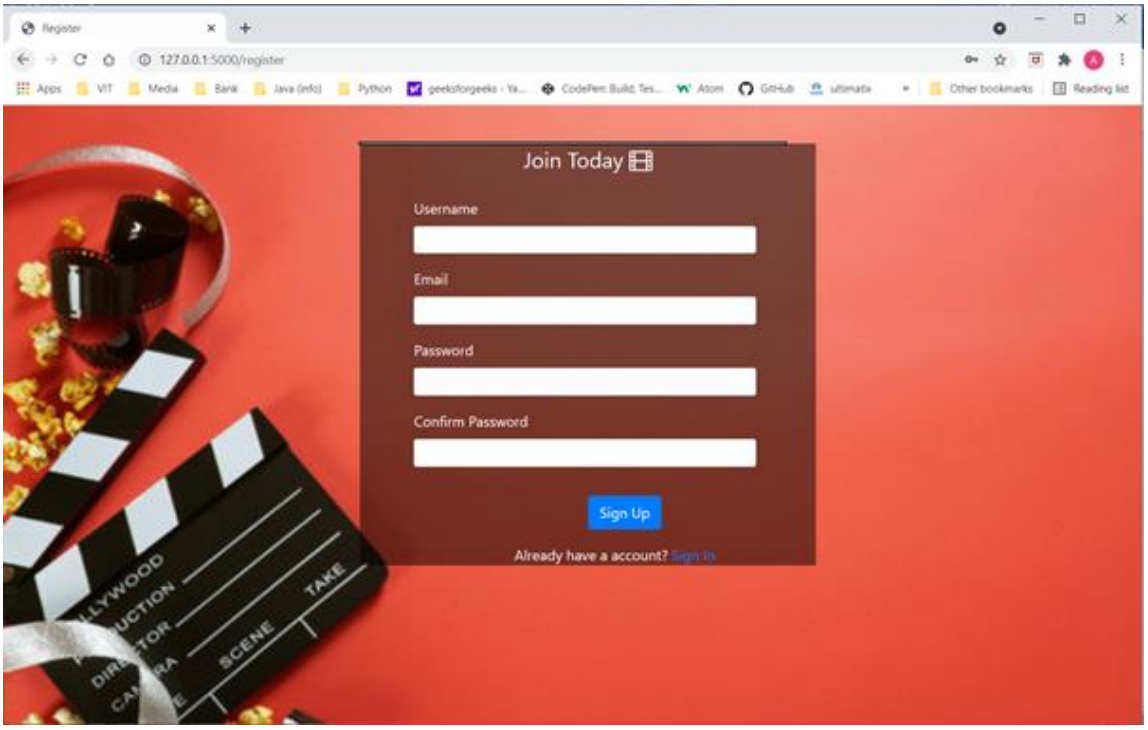
The findings of this study can be expanded to include more information on fresh case studies using other methodologies.

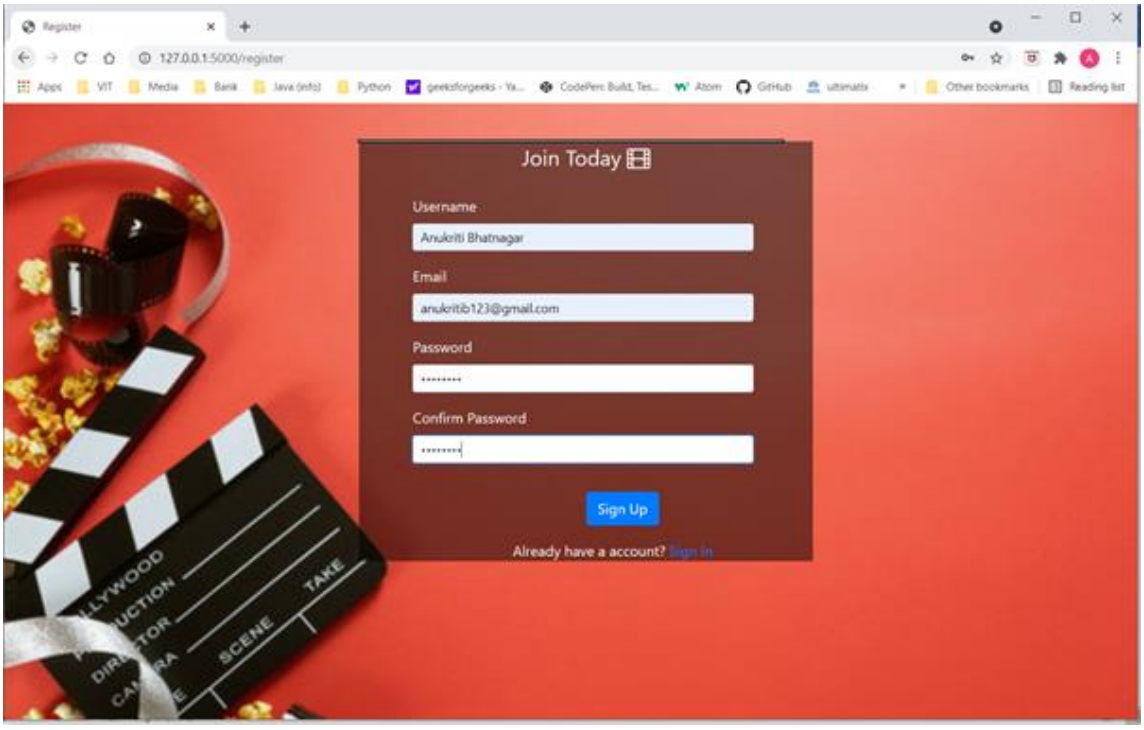
The rise of Big Data, which is made up of very huge datasets with multiple formats in their data and is continually changing, is one big change in computer science in this situation. This has the potential to alter how user and item data is modelled, as well as how information is received from users and objects in the context of requirements engineering.

9. APPENDIX

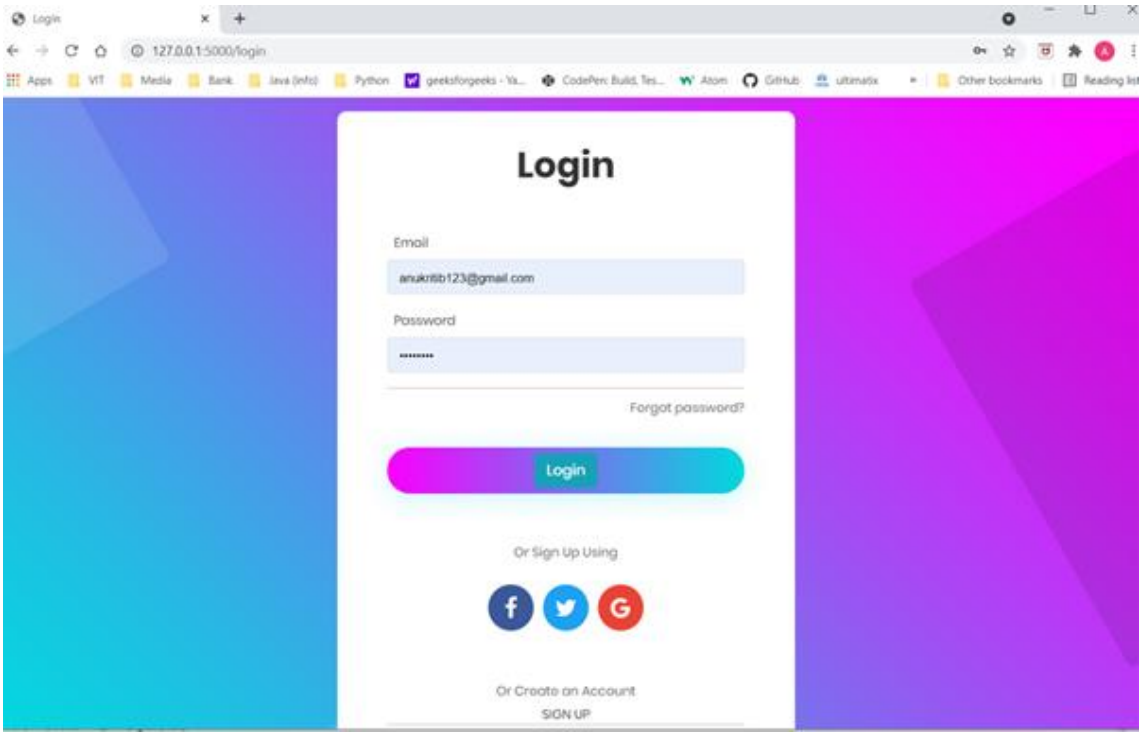
- SCREENSHOTS:

Sign Up Page:

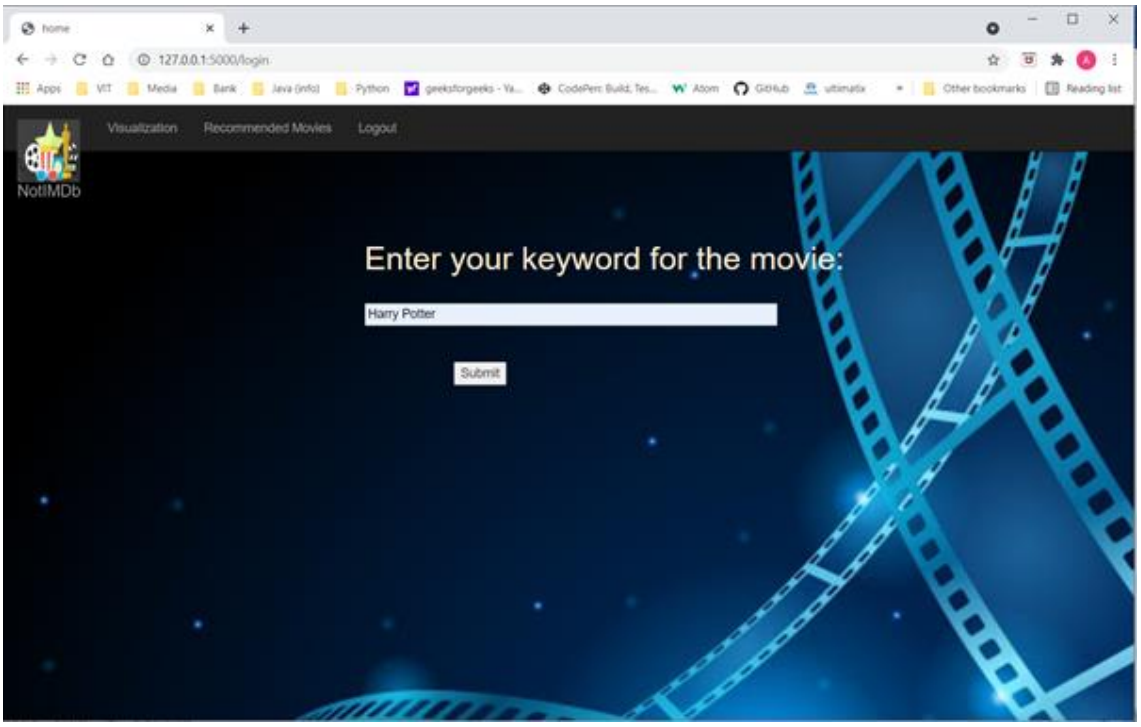

Register
127.0.0.1:5000/register
Join Today
Username
Email
Password
Confirm Password
Sign Up
Already have a account? Sign In


Register
127.0.0.1:5000/register
Join Today
Username
Anukriti Bhatnagar
Email
anukritib123@gmail.com
Password
Confirm Password
Sign Up
Already have a account? Sign In

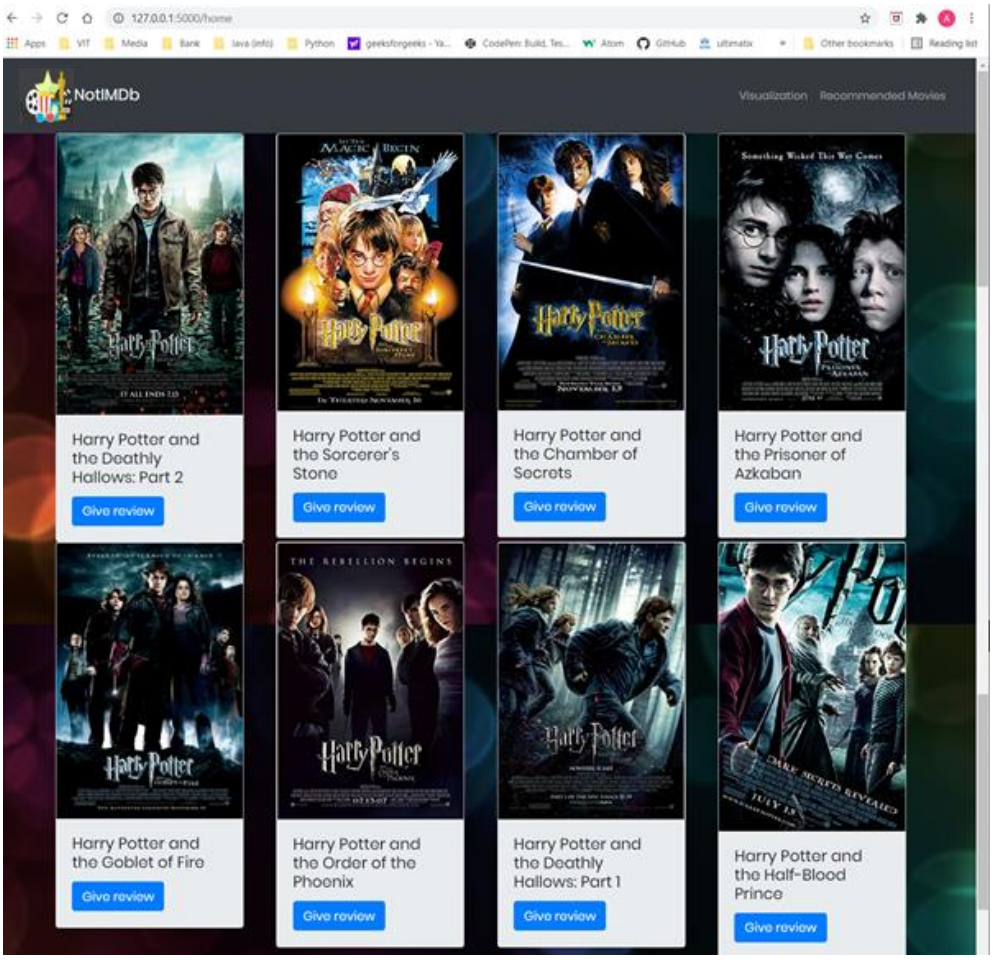
Login Page:



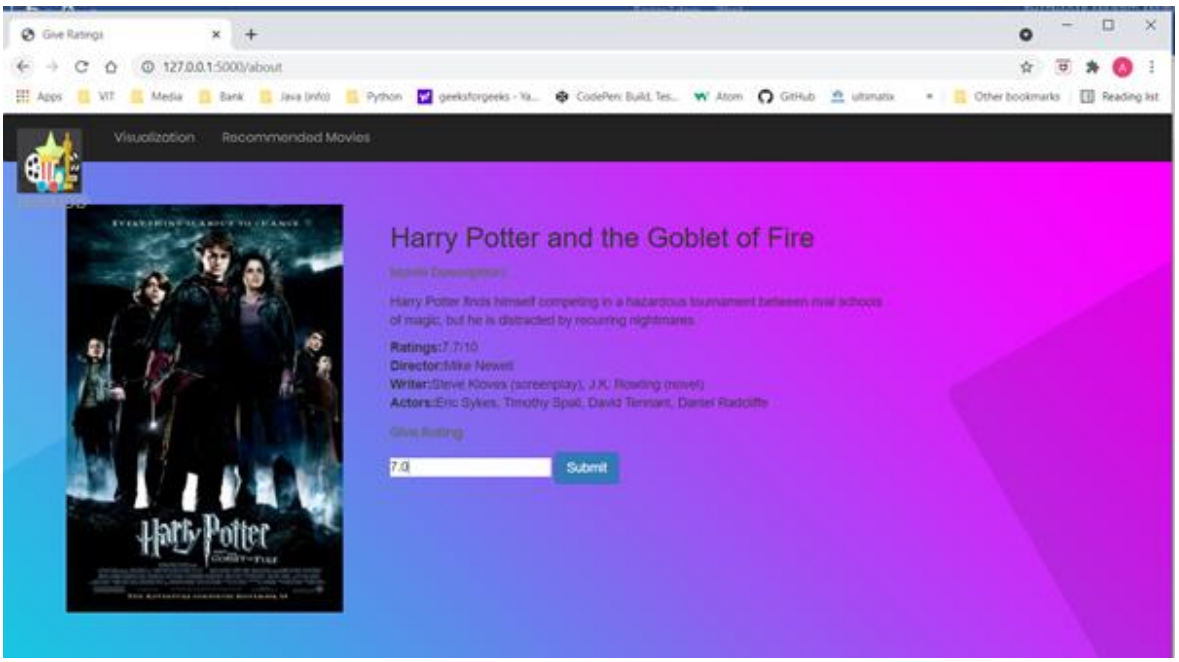
Home Page:



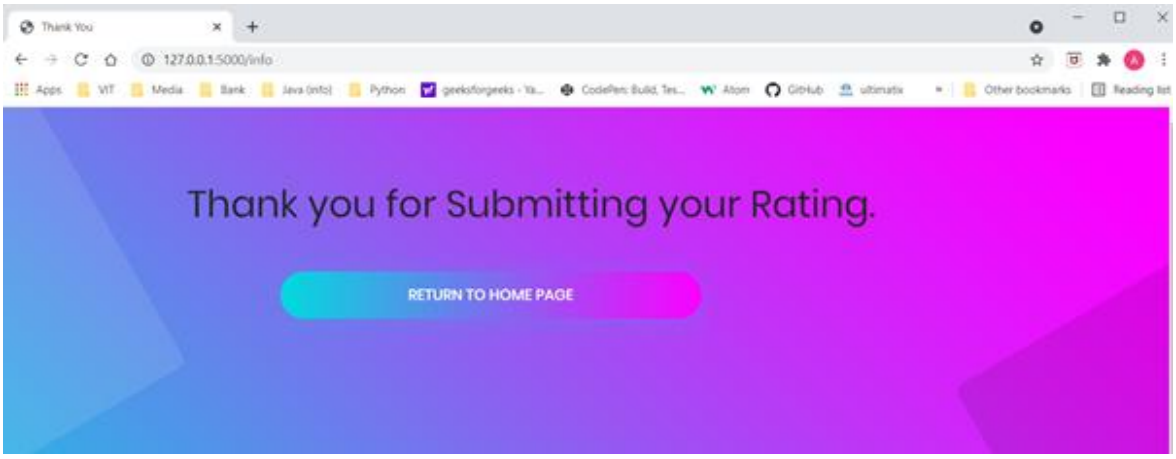
Search Results Page:



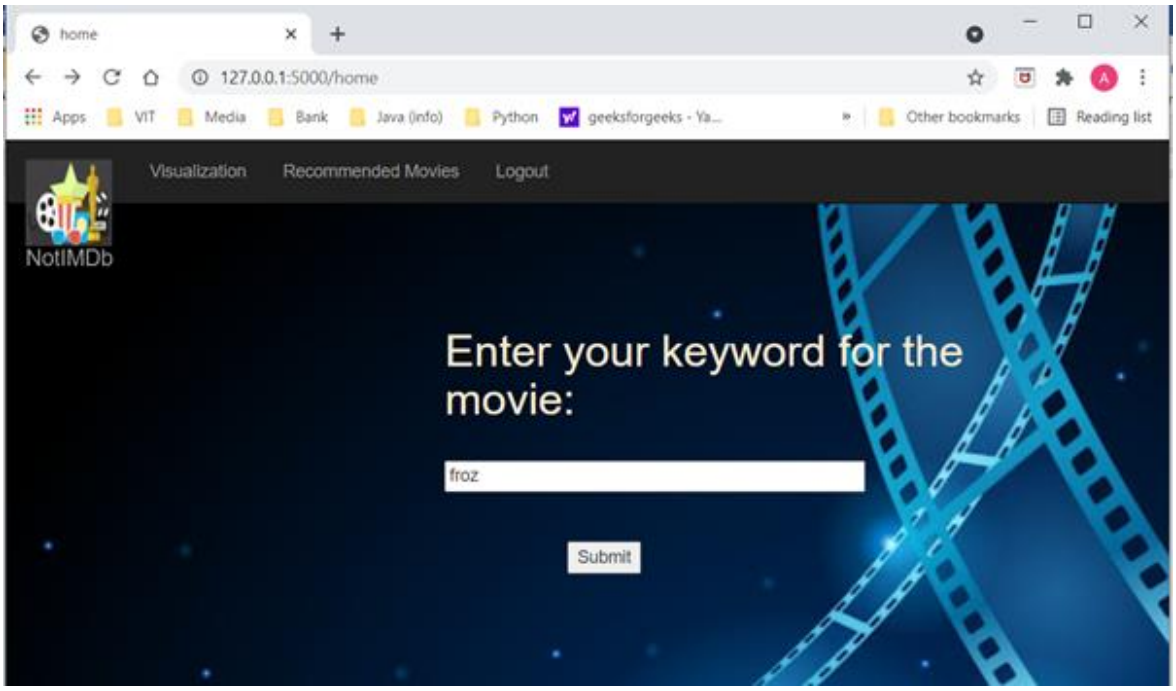
Movie Information/Review Page:

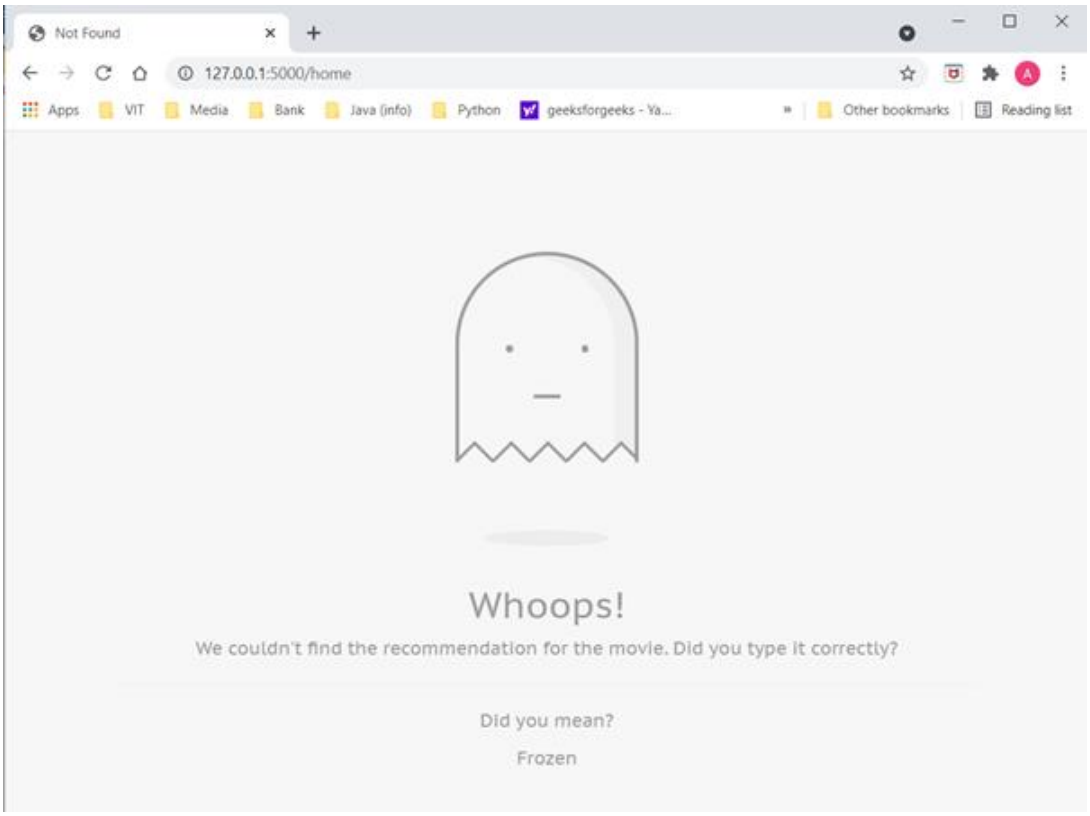


Thank You Page:

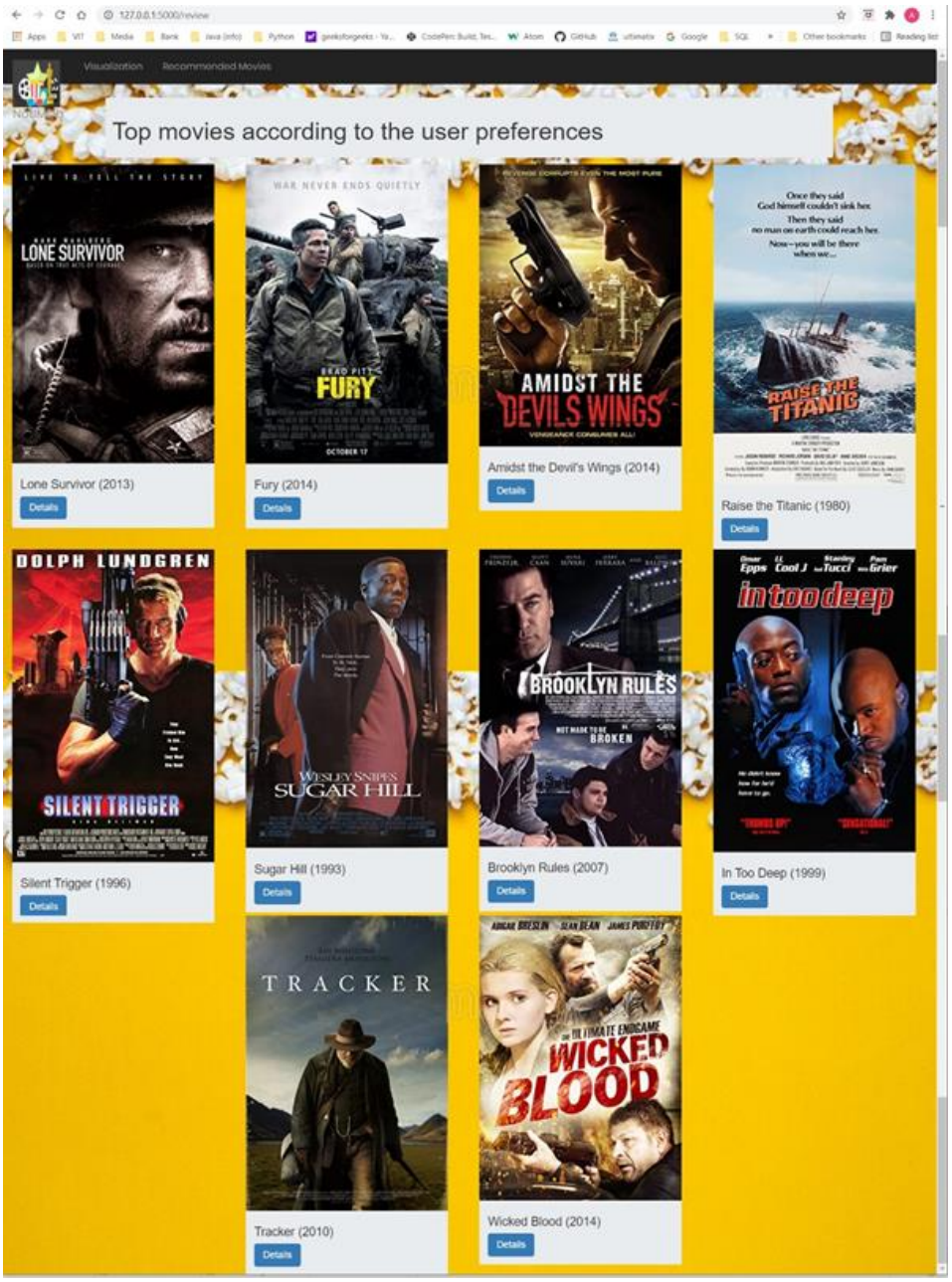


Movie Not Found Page:

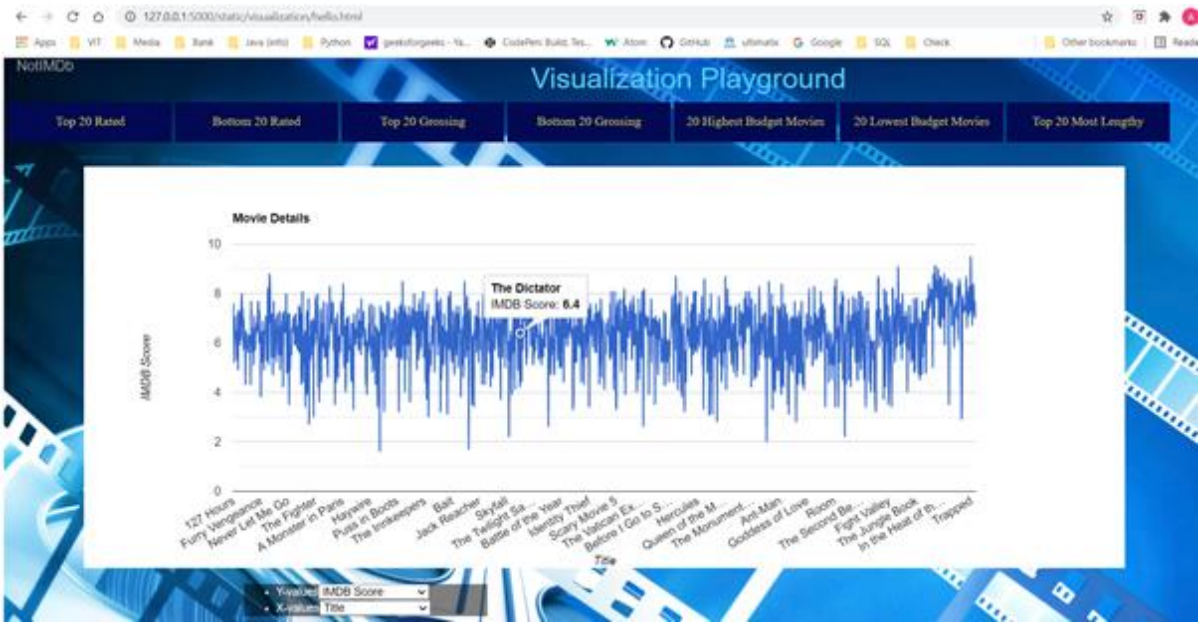




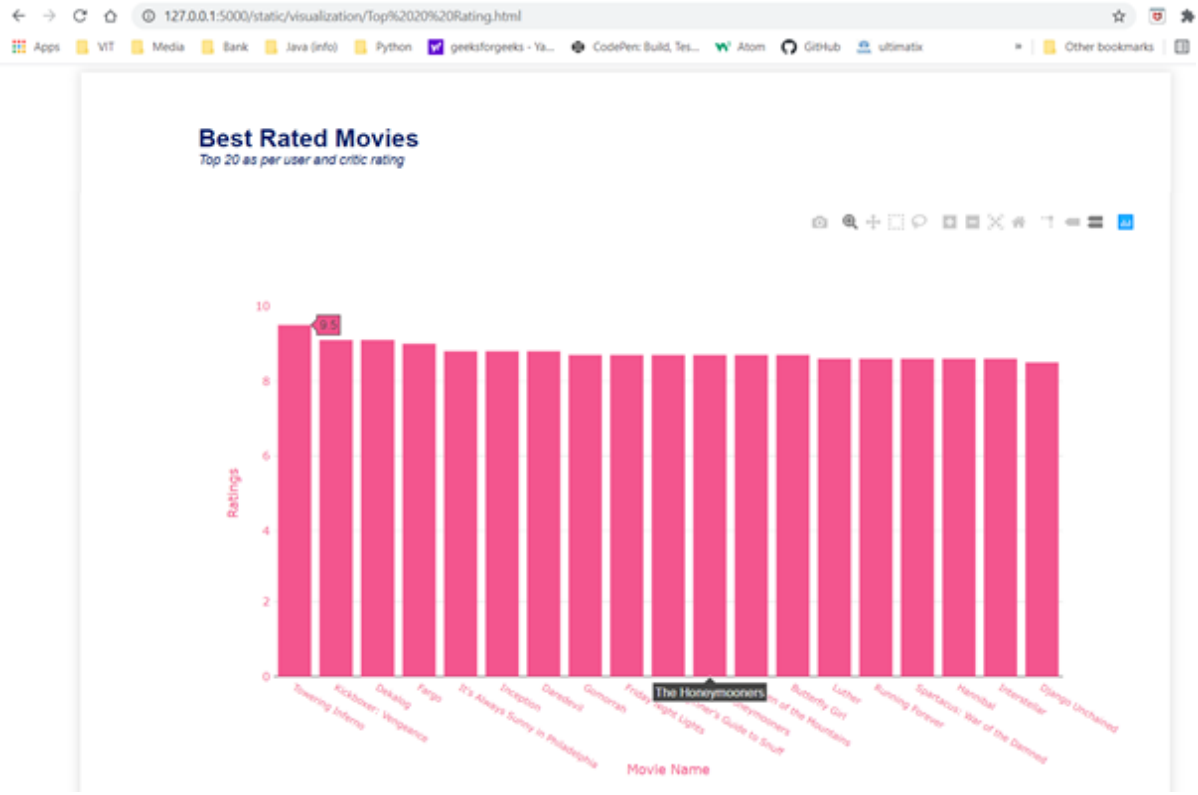
Recommendation Page:



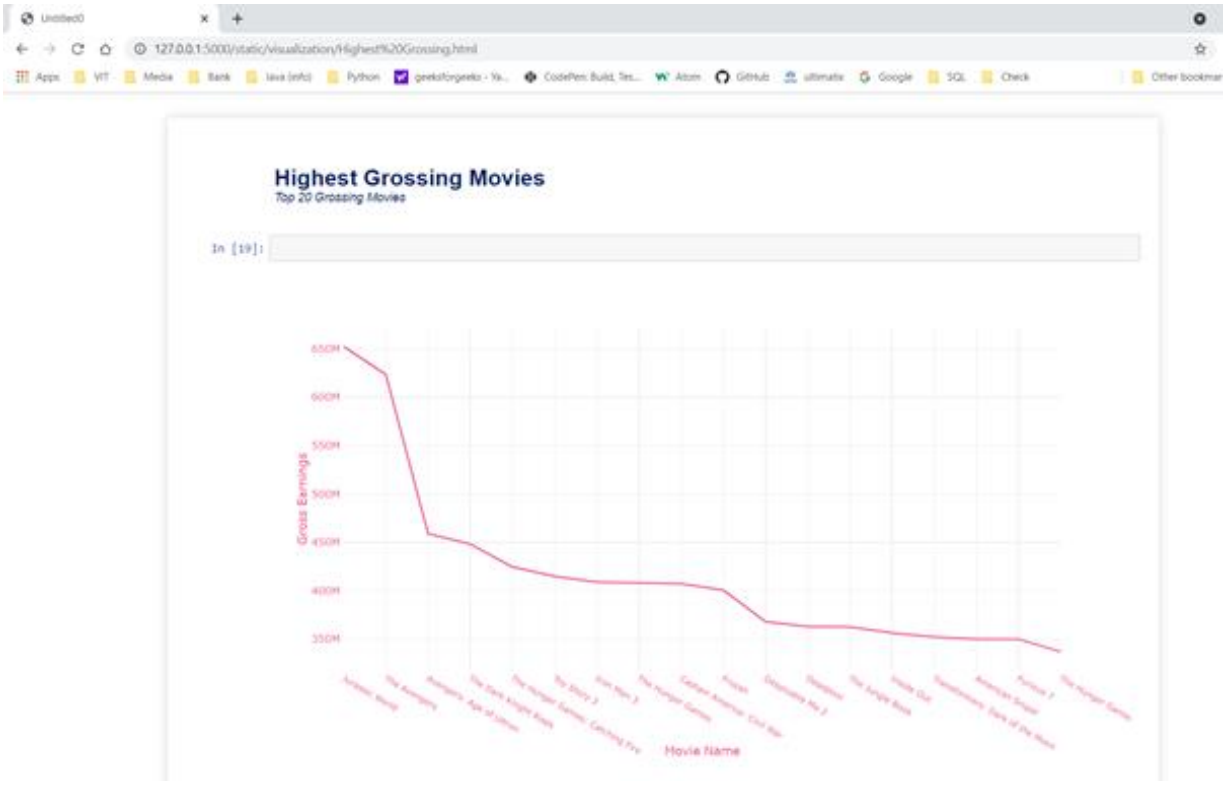
Visualization Page:



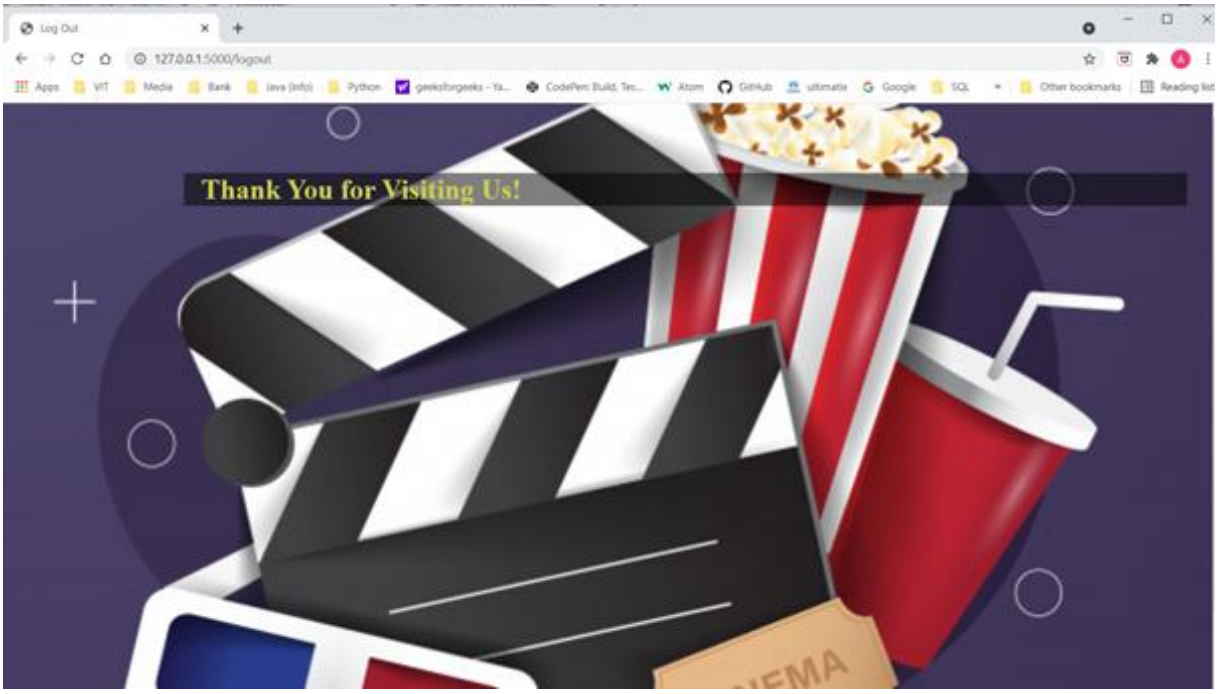
Top 20 Rated Movies:



Highest Grossing Movies:



Logout Page:



- SAMPLE CODING:

FRONT-END CODES:

Sign Up Page:

```
<!DOCTYPE html>

<html>

<head>

    <title>Register</title>

    <meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

<link href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-
awesome.min.css" rel="stylesheet" integrity="sha384-
wvfXpqpZZVQGK6TAh5PVlGOfQNHSoD2xbE+QkPxCAFlNEevoEH3Sl0sibVcOQVnN"
crossorigin="anonymous">

<style>

    .container {

      position: relative;

    }

    .text-block {

      position: absolute;

      padding-left: 20px;

      padding-right: 20px;

      background: black;

      background: rgba(0, 0, 0, 0.5); /* Black background with 0.5 opacity */

      color: #f1f1f1;

      width: 100%;

    }

    </style>

</head>
```

```
<body style="background-image: url('static/images/signup_page1.jpg');background-
repeat:no-repeat; background-size: cover;">

    <div class="container" style="margin-top: 40px;width: 40%;">

        <form method="POST" action="" style="border: 2px solid; background-color
: white";>

        {{form.hidden_tag()}}

    <div class = "text-block">

        <fieldset class="form-group">

            <legend style="text-align: center;">Join Today <i class="fa fa-
film"></i></legend>

            <dir class="form-group" style="width: 90%;">

                {{form.username.label(class="form-control-label")}}

                {{form.username(class="form-control form-control-sm")}}

            </dir>

            <dir class="form-group" style="width: 90%;">

                {{form.email.label(class="form-control-label")}}

                {{form.email(class="form-control form-control-sm")}}

            </dir>

            <dir class="form-group" style="width: 90%;">

                {{form.password.label(class="form-control-label")}}

                {{form.password(class="form-control form-control-sm")}}

            </dir>

            <dir class="form-group" style="width: 90%;">

                {{form.confirm_password.label(class="form-control-label")}}

                {{form.confirm_password(class="form-control form-control-sm")}}

            </dir>

        </fieldset>

        <div class="form-group" style="margin-left: 50%;">

            {{form.submit(class="btn btn-primary")}}

        </div>

    <div style="margin-left: 60px; text-align: center;">

        Already have a account? <a href="{{url_for('login')}}">Sign In</a>
```

```
    </div></div>

    </div></form>

    </body>

        </body>

    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
crossorigin="anonymous"></script>

<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>

</body>

</html>
```

Login Page:

```
<!DOCTYPE html>

<html>

<head>

    <title>LogIn</title>

    <meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

<link rel="stylesheet" type="text/css"
href="static/vendor/bootstrap/css/bootstrap.min.css">

<link rel="stylesheet" type="text/css" href="static/fonts/font-awesome-
4.7.0/css/font-awesome.min.css">
```

```
<link rel="stylesheet" type="text/css" href="static/fonts/iconic/css/material-
design-iconic-font.min.css">

<link rel="stylesheet" type="text/css" href="static/vendor/animate/animate.css">

<link rel="stylesheet" type="text/css" href="static/vendor/css-
hamburgers/hamburgers.min.css">

<link rel="stylesheet" type="text/css"
href="static/vendor/animsition/css/animsition.min.css">

<link rel="stylesheet" type="text/css"
href="static/vendor/select2/select2.min.css">

<link rel="stylesheet" type="text/css"
href="static/vendor/daterangepicker/daterangepicker.css">

<link rel="stylesheet" type="text/css" href="static/css/util.css">

<link rel="stylesheet" type="text/css" href="static/css/main.css">

</head>

<body>

    <div class="limiter">

    <div class="container-login100" style="background-image:
url('static/images/bg-01.jpg');">

        <div class="wrap-login100 p-l-55 p-r-55 p-t-35 p-b-54">

            <form class="login100-form validate-form" method="POST" action="">

                {{form.hidden_tag()}}

                <fieldset class="form-group">

                <span class="login100-form-title p-b-49">

                    Login

                </span>

                <div class="wrap-input100 validate-input m-b-23" data-validate =
"Email is required is required">

                    <dir class="form-group">

                    <span class="label-input100">

                        {{form.email.label(class="form-control-label")}}

                        {{form.email(class="form-control form-control-md")}}

                    </span>

                    </dir>
```

```
                    <div class="wrap-input100 validate-input" data-
validate="Password is required">
                    <dir class="form-group"><span class="label-input100">
                        {{form.password.label(class="form-control-label")}}
                        {{form.password(class="form-control form-control-md")}}
                    </span>
                </dir>
                </div>
                <div class="text-right p-t-8 p-b-31">
                    <a href="#">Forgot password?</a>
                </div>

                <div class="container-login100-form-btn">
                        <div class="wrap-login100-form-btn" style =
"color:white;">
                            <div class="login100-form-bgbtn"></div>
                            <button class="login100-form-btn" style =
"color:white;">
                                {{form.submit(class="btn btn-outline-info")}}
                            </button>
                        </div>
                    </div>

                <div class="txt1 text-center p-t-54 p-b-20">
                    <span>
                        Or Sign Up Using
                    </span>
                </div>
                <div class="flex-c-m">
                    <a href="{{url_for('home')}}" class="login100-social-item
bg1">
                        <i class="fa fa-facebook"></i>
```

```
                    </a>
                    <a href= "{{url_for('home')}}" class="login100-social-item
bg2">
                        <i class="fa fa-twitter"></i>
                    </a>
                    <a href="{{url_for('home')}}" class="login100-social-item
bg3">
                        <i class="fa fa-google"></i>
                    </a>
                </div>
                <div class="flex-col-c p-t-50">
                    <span class="txt1 p-b-5">
                        Or Create an Account
                    </span>
                    <a href="{{url_for('register')}}" class="txt2">
                        Sign Up
                    </a>
                </div>
            </form>
        </div>
    </div>
</div>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
crossorigin="anonymous"></script>

<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>
```

```
<script src="vendor/animsition/js/animsition.min.js"></script>
<script src="vendor/bootstrap/js/popper.js"></script>
<script src="vendor/bootstrap/js/bootstrap.min.js"></script>
<script src="vendor/select2/select2.min.js"></script>
<script src="vendor/daterangepicker/moment.min.js"></script>
<script src="vendor/daterangepicker/daterangepicker.js"></script>
<script src="vendor/countdowntime/countdowntime.js"></script>
<script src="js/main.js"></script>
</body>
</html>
```

Home Page:

```
<!DOCTYPE html>
<html>
<head>
    <title>home</title>
    <meta name="viewport" content="width=device-width, initial-scale=1">
</head>
<link rel="stylesheet" type="text/css" href="main.css">

<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></scrip
t>

<body style="background-image: url('static/images/unnamed (1).jpg'); background-
repeat: no-repeat; background-size: cover;">
    <nav class="navbar navbar-inverse">
```

```
    <div class="container-fluid">
    <div class="navbar-header">
        <a class="navbar-brand" href="{{url_for('home')}}"><img src =
'static/images/movie-review-website.png' height= 70px; width = 70px;>NotIMDb</a>
    </div>
    <ul class="nav navbar-nav">
      <li><a href="{{url_for('hello')}}">Visualization</a></li>
      <li><a href="{{url_for('review')}}">Recommended Movies</a></li>
      <li><a href="{{url_for('logout')}}">Logout</a></li>
    </ul>
    </div>
    </nav>

    <div class="container" style="width: 70%;">

    <h1 style="margin-left: 200px; margin-top: 80px; color:blanchedalmond">Enter
your keyword for the movie:</h1>
    <form method="POST" action="/home" style="margin-left: 200px;width: 90%;">
        <br/>
        <div class="label-input100" >
            <input type="text" name="mk" style="width: 60%;">
        </div>
        <br/><br/>
            <div class="wrap-login100-form-btn" style="margin-left: 100px; width:
40%;">
                <div class="login100-form-bgbtn"></div>
                <button class="login100-form-btn">
                    Submit
                </button>
            </div>
    </form>
```

```
    </div>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
crossorigin="anonymous"></script>

<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>

</body>

</html>
```

Search Results Page:

```
<!DOCTYPE html>

<html>

<head>

    <title>about</title>

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">

    <link rel="stylesheet" type="text/css" href="static/css/main.css">

</head>

<body style="background-image: url('static/images/unnamed.jpg'); background-
repeat: repeat-y; background-size: cover;">

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">

        <div class="container-fluid">

            <div class="navbar-header">
```

```
          <a class="navbar-brand" href="{{url_for('home')}}"><img src =
'static/images/movie-review-website.png' height= 70px; width = 70px;>NotIMDb</a>


         </div>

            <div class="collapse navbar-collapse"
id="navbarSupportedContent"></div>

            <ul class="navbar-nav mr-auto">

              <li class="nav-item"><a class="nav-link"
href="{{url_for('static',filename='visualization/hello.html')
}}">Visualization</a></li>

              <li class="nav-item"><a class="nav-link"
href="{{url_for('review')}}">Recommended Movies</a></li>

            </ul>

            </div>

        </div>

            </nav>

    <div class="container" style="margin-left: auto;margin-right: auto; width:
90%;">

    <div class="row">

    {%for i in data%}

        <div class="col-lg-3 col-sm-6">

            <div class="card" style="width: 15rem;">

                <img class="card-img-top" src={{i.Poster}} alt="Card image cap">

                <div class="card-body">

                    <h5 class="card-title">{{i.Title}}</h5>

                    <form method="POST" action="/about">

                        <button type="submit" class="btn btn-primary"
name="imdbid" value="{{i.imdbID}}">Give review</button>

                    </form>

                </div>

            </div>

        </div>

    {%endfor%}
```

```
    </div>

    </div>

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1"
crossorigin="anonymous"></script>

<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>

</body>

</html>
```

Review Page:

```
<!DOCTYPE html>

<html>

<head>

    <title>Give Ratings</title>

    <meta name="viewport" content="width=device-width, initial-scale=1">

</head>

  <link rel="stylesheet" type="text/css" href="static/css/main.css">

  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/cs
s/bootstrap.min.css">

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js">
</script>

  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.j
s"></script>


<body style="background-image: url('static/images/bg-01.jpg');">
```

```
  <nav class="navbar navbar-inverse">
  <div class="container-fluid">
    <div class="navbar-header">
      <a class="navbar-brand" href="#"><img src = 'static/images/movie-review-
website.png' height= 70px; width = 70px;>NotIMDb</a>
    </div>
    <ul class="nav navbar-nav">
      <li><a href="{{ url_for('static',filename='visualization/hello.html') }}">V
isualization</a></li>
      <li><a href="{{url_for('review')}}">Recommended Movies</a></li>
    </ul>
  </div>
</nav>
<div class="container">
  <div class="row">
    <div class="column-1">
        <span><img src="{{data.Poster}}"></span>
    </div>
    <div class="column-2">
        <h2>{{data.Title}}</h2>
        <p>
            <b>Movie Description:</b>
            <div>{{data.Plot}}</div>
        </p>
        <p>
            <div><b>Ratings:</b>{{data.Ratings[0].Value}}</div>
            <div><b>Director:</b>{{data.Director}}</div>
            <div><b>Writer:</b>{{data.Writer}}</div>
            <div><b>Actors:</b>{{data.Actors}}</div>
        </p>
        <p><b>Give Rating</b></p>
```

```
        <form method="POST" action="/info" >
            <input type="text" name="rating">
            <button type="submit" class="btn btn-primary">Submit</button>
        </form>
    </div>
  </div>
</div>


<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
</body>
</html>
```

Thank You Page:

```
<!DOCTYPE html>
<html>
<head>
    <title>Thank You</title>
</head>
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
<link rel="stylesheet" type="text/css" href="static/css/main.css">
```

```
<body style="background-image: url('static/images/bg-01.jpg');">


    <h1 style="margin-left: 200px; margin-top: 80px;">Thank you for Submitting
your Rating.</h1><br/><br/>


    <form method="POST" action="/ThankYou" style="margin-left: 200px;width:
90%;">


        <div class="wrap-login100-form-btn" style="margin-left: 100px; width:
40%;">

            <div class="login100-form-bgbtn"></div>

            <button class="login100-form-btn">

                Return to Home Page

            </button>

        </div>

    </form>


    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></scriptr

    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
crossorigin="anonymous"></script>

    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>

</body>

</html>
```

Movie Not Found Page:

```
<!DOCTYPE html>

<html lang="en" >

<head>

  <meta charset="UTF-8">

  <title>Not Found</title>

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <link rel='stylesheet'
href='https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css'>

</head>

<body>

<script src="https://kit.fontawesome.com/a076d05399.js"></script>

<link
href='https://fonts.googleapis.com/css?family=Anton|Passion+One|PT+Sans+Caption'
rel='stylesheet' type='text/css'>


<div class="container">

<div class="boo-wrapper">

<div class="boo">

<div class="face"></div>

</div>

<div class="shadow"></div>


<h1>Whoops!</h1>

<p>

We couldn't find the recommendation for the movie. Did you type it correctly?

<br />


        <input type="hidden" value="{{ name }}" id="movie_name"
oninput="checkSimilarity()" readonly/>

                <hr>

                <p>Did you mean?</p>

                <span id="suggestions"></span>
```

```
            </p>
            </div>
            </div>


</body>
<script>
function goBack() {
  window.history.back();
}

function checkSimilarity(){
    filtered_names = []
    var str1 = document.getElementById("movie_name").value;
    for(var i=0;i<all_titles.length;i++){
        var str2 = all_titles[i];
        var score = similarity(str1, str2);
        if(score>0.50){
            filtered_names.push(str2+'<br />');
        }
    }
    if(filtered_names.length<1){
        document.getElementById("suggestions").innerHTML = "Please try again.
Either input movie name is misspelled badly or not present in database.";
    }else{
        document.getElementById("suggestions").innerHTML = filtered_names;
    }
}

function similarity(s1, s2) {
      var longer = s1;
```

```
      var shorter = s2;
      if (s1.length < s2.length) {
        longer = s2;
        shorter = s1;
      }
      var longerLength = longer.length;
      if (longerLength == 0) {
        return 1.0;
      }
      return (longerLength - editDistance(longer, shorter)) /
parseFloat(longerLength);
    }

    function editDistance(s1, s2) {
      s1 = s1.toLowerCase();
      s2 = s2.toLowerCase();

      var costs = new Array();
      for (var i = 0; i <= s1.length; i++) {
        var lastValue = i;
        for (var j = 0; j <= s2.length; j++) {
          if (i == 0)
            costs[j] = j;
          else {
            if (j > 0) {
              var newValue = costs[j - 1];
              if (s1.charAt(i - 1) != s2.charAt(j - 1))
                newValue = Math.min(Math.min(newValue, lastValue),
                  costs[j]) + 1;
              costs[j - 1] = lastValue;
              lastValue = newValue;
```

```
            }
          }
        }
        if (i > 0)
          costs[s2.length] = lastValue;
      }
      return costs[s2.length];
    }


    checkSimilarity()


function randomNum()
        {
            "use strict";
            return Math.floor(Math.random() * 9)+1;
        }
            var loop1,loop2,loop3,time=30, i=0, number, selector3 =
document.querySelector('.thirdDigit'), selector2 =
document.querySelector('.secondDigit'),
                selector1 = document.querySelector('.firstDigit');
            loop3 = setInterval(function()
            {
              "use strict";
                if(i > 40)
                {
                    clearInterval(loop3);
                    selector3.textContent = 4;
                }else
                {
                    selector3.textContent = randomNum();
                    i++;
```

```
        }
    }, time);
    loop2 = setInterval(function()
    {
      "use strict";
        if(i > 80)
        {
            clearInterval(loop2);
            selector2.textContent = 0;
        }else
        {
            selector2.textContent = randomNum();
            i++;
        }
    }, time);
    loop1 = setInterval(function()
    {
      "use strict";
        if(i > 100)
        {
            clearInterval(loop1);
            selector1.textContent = 4;
        }else
        {
            selector1.textContent = randomNum();
            i++;
        }
    }, time);
</script>
```

```
</body>

</html>
```

Recommendation Page:

```
<!DOCTYPE html>

<html>

<head>

    <title>Your Recommendations</title>

    <meta name="viewport" content="width=device-width, initial-scale=1">

</head>

<link rel="stylesheet" type="text/css" href="../static/css/main.css">


<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">

<body style="background-image: url('static/images/popcorn.jpg'); background-
repeat: repeat-y; background-size: cover;">

    <nav class="navbar navbar-inverse">

        <div class="container-fluid">

        <div class="navbar-header">

            <a class="navbar-brand" href="{{url_for('home')}}"><img src =
'static/images/movie-review-website.png' height= 70px; width = 70px;>NotIMDb</a>

        </div>

        <ul class="nav navbar-nav">

          <li class="nav-item"><a class="nav-link" href="{{url_for('hello')
}}">Visualization</a></li>

          <li><a href="{{url_for('review')}}">Recommended Movies</a></li>

        </ul>

        </div>

        </nav>

    <div class="container" style="width: 80%">

        <div class="jumbotron-1">
```

```
            <h1>Top movies according to the user preferences</h1>
        </div>
    </div>
<!--    <div class="container" style="margin-left: auto;margin-right: auto;
width: 70%;">
    <div class="row">
-->
    {%for i in details%}
        <div class="col-lg-3 col-sm-4 ">
            <div class="card" style="width: 30rem;">
                <img class="card-img-top" src={{i.Poster}} alt="Card image cap">
                <div class="card-body">
                    <h4 class="card-title">{{i.Title}} ({{i.Year}})</h4>
                    <form method="POST" action="/about">
                        <button type="submit" class="btn btn-primary"
name="imdbid" value="{{i.imdbID}}">Details</button>
                    </form>
                </div>
            </div>
        </div>
<!--    </div>
    </div>
-->
    {%endfor%}


    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-
q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo"
crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-
ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/l8WvCWPIPm49"
crossorigin="anonymous"></script>
```

```
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-
ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>

</body>

</html>
```

Visualization Page:

```
<!DOCTYPE html>

<html>

<head>

    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">


    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>

    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></scrip
t>


    <style type="text/css">

        .button {

          background-color: #040953;

          border: none;

          color: rgb(205, 207, 77);

           text-align: center;

          text-decoration: none;

          display: inline-block;

          font-size: 16px;

          font-family: Poppins-Regular;

          padding: 12px;

          width: 12%;
```

```
          height: 50px;
        }

        .container-review1{width:1700px;
        padding-right:1px;
        padding-left:1px;
        margin-right:1px;margin-left:5px}
    </style>

    <title>Movie Visualization</title>
</head>
<body style="background-image: url('static/images/Film-Reel-Wallpaper.jpg');
background-repeat: no-repeat; background-size: cover;">
    <nav class="navbar navbar-inverse">
        <div class="container-fluid">
        <div class="navbar-header">
            <a class="navbar-brand" href="{{url_for('home')}}"><img src =
'static/images/movie-review-website.png' height= 70px; width = 70px;>NotIMDb</a>
        </div>
        <ul class="nav navbar-nav">
            <li><a href="{{url_for('hello')}}">Visualization</a></li>
            <li><a href="{{url_for('review')}}">Recommended Movies</a></li>
        </ul>
        </div>
    </nav>
    <div class="container-review1">
        <h1 style="color: #55cef3;" align="center">Visualization Playground</h1>
        <a href="{{url_for('Top_20_Rating')}}" class="button">Top 20 Rated</a>
        <a href="{{url_for('Bottom_20_Rating')}}" class="button">Bottom 20
Rated</a>
        <a href="{{url_for('Highest_Grossing')}}" class="button">Top 20
Grossing</a>
```

```
        <a href="{{url_for('Least_Grossing')}}" class="button">Bottom 20
Grossing</a>

        <a href="{{url_for('Highest_Budget')}}" class="button">20 Highest Budget
Movies</a>

        <a href="{{url_for('Lowest_Budget')}}" class="button">20 Lowest Budget
Movies</a>

        <a href="{{url_for('Most_lengthy_movies')}}" class="button">Top 20 Most
Lengthy</a>

    </div>

    <div class="container-login100">

        <div style="margin-left: 100px; margin-top: 30px;"><div id="chart"
style="width:1300px; height:500px;"></div></div>

        <br/>


        <ul style= "background: black; background: rgba(0, 0, 0, 0.5); color:
#f1f1f1; margin-left: 300px; width:20%">

            <li>Y-values<select class="chart" id="xrange" style = "color:
black;"></select></li>

            <li>X-values<select class="chart" id="yrange" style = "color:
black;"></select></li>

        </ul>


    </div>


<script src="https://www.google.com/jsapi"></script>

<script src="http://code.jquery.com/jquery-1.10.1.min.js"></script>

<!--<script src="./jquery.csv-0.71.js"></script> -->

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery-csv/0.71/jquery.csv-
0.71.min.js"></script>

<script>


    // load the visualization library from Google and set a listener

    google.load("templates", "1", {packages:["corechart"]});

    google.setOnLoadCallback(drawChart);
```

```
    function drawChart() {

        // grab the CSV
        thecsvfile = "MI.csv"

        $.get(thecsvfile, function(csvString) {

            // transform the CSV string into a 2-dimensional array
            var arrayData = $.csv.toArrays(csvString, {onParseValue:
$.csv.hooks.castToScalar});

            // use arrayData to load the select elements with the appropriate
options
            for (var i = 0; i < arrayData[0].length; i++) {
                //* this adds the given option to both select elements
                 $("select").append("<option value='" + i + "'>" +
arrayData[0][i] + "</option>");
            }

            // set the default selection
            $("#xrange option[value='1']").attr("selected","selected");
            $("#yrange option[value='0']").attr("selected","selected");
            $("domain option[value='3']").attr("selected","selected");

            // this new DataTable object holds all the data
            var data = new google.visualization.arrayToDataTable(arrayData);

            // this view can select a subset of the data at a time
            var view = new google.visualization.DataView(data);
            view.setColumns([0,1]);
```

```
            var options = {
                title: "Movie Details",
                hAxis: {title: data.getColumnLabel(0), minValue:
data.getColumnRange(0).min, maxValue: data.getColumnRange(0).max},
                vAxis: {title: data.getColumnLabel(1), minValue:
data.getColumnRange(1).min, maxValue: data.getColumnRange(1).max},
                legend: 'none'
            };

            var chart = new
google.visualization.LineChart(document.getElementById('chart'));
            chart.draw(view, options);

            // set listener for the update button
            $("select").click(function(){

               // determine selected domain and range
                var xrange = +$("#xrange option:selected").val();
                var yrange = +$("#yrange option:selected").val();
                var domain = +$("#domain option:selected").val();

                // update the view
                view.setColumns([yrange,xrange]);

                // update the options
                options.hAxis.title = data.getColumnLabel(yrange);
                options.hAxis.minValue = data.getColumnRange(xrange).min;
                options.hAxis.maxValue = data.getColumnRange(xrange).max;
                options.vAxis.title = data.getColumnLabel(xrange);
                options.vAxis.minValue = data.getColumnRange(yrange).min;
                options.vAxis.maxValue = data.getColumnRange(yrange).max;
```

```
                chart.draw(view, options);
            });
        });
    }
</script>
</body>
</html>
```

Logout Page:

```
<html>
<head>
        <title>Log Out</title>
        <style>
            .text-block {
      position: absolute;
      padding-left: 20px;
      padding-right: 20px;
      background: black;
      background: rgba(0, 0, 0, 0.5); /* Black background with 0.5 opacity */
      color:rgb(205, 207, 77);;
      width: 80%;
    }
        </style>
</head>
<body style="background-image: url('static/images/movie-element-premium.jpg');
background-repeat: no-repeat; background-size:cover;background-position:
center;height:100%;width:100%">
    <h1 style="margin-left: 200px; margin-top: 80px; color: rgb(205, 207,
77);"><div class="text-block">Thank You for Visiting Us!</h1><br/><br/>
    </div></body>
</html>
```

BACK-END CODE:

```
from re import X
from flask import Flask, render_template, request, url_for,flash, redirect
from flask.wrappers import Response
from scipy.sparse import data
from forms import RegistrationForm, LoginForm
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
from datetime import datetime
from flask_login import LoginManager, UserMixin, login_user
import csv
import random
import difflib
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
pd.options.display.max_columns = None

from scipy import stats
from ast import literal_eval
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.metrics.pairwise import linear_kernel, cosine_similarity
from nltk.stem.snowball import SnowballStemmer
from nltk.stem.wordnet import WordNetLemmatizer
from nltk.corpus import wordnet

import urllib.request
import json
```

```
app = Flask(__name__)
app.config['SECRET_KEY']='a8ca03f6bb27fb5d2e9543b0a5c0ded3'
app.config['SQLALCHEMY_DATABASE_URI']='sqlite:///site.db'
db=SQLAlchemy(app)
bcrypt=Bcrypt(app)
login_manager=LoginManager(app)

## New Code (Start)
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity

df2 = pd.read_csv('tmdb.csv')
count = CountVectorizer(stop_words='english')
count_matrix = count.fit_transform(df2['soup'])

cosine_sim2 = cosine_similarity(count_matrix, count_matrix)

df2 = df2.reset_index()
indices = pd.Series(df2.index, index=df2['title'])
all_titles = [df2['title'][i] for i in range(len(df2['title']))]
## New Code (End)

@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))

class User(db.Model, UserMixin):
    id=db.Column(db.Integer,primary_key=True)
    username=db.Column(db.String(20),unique=True,nullable=False)
    email=db.Column(db.String(120),unique=True,nullable=False)
```

```
    password=db.Column(db.String(60),nullable=False)
    posts=db.relationship('Post',backref='author',lazy=True)
    def __repr__(self):
        return f"User('{self.username}','{self.email}')"

class Post(db.Model):
    id=db.Column(db.Integer,primary_key=True)
    title=db.Column(db.String(100),nullable=False)
    date_posted=db.Column(db.DateTime,nullable=False,default=datetime.utcnow)
    content=db.Column(db.Text,nullable=False)
    user_id=db.Column(db.Integer,db.ForeignKey('user.id'),nullable=False)
    def __repr__(self):
        return f"Post('{self.title}','{self.date_posted}')"

@app.route("/login",methods=['GET','POST'])
def login():
    form=LoginForm()
    if form.validate_on_submit():
        user=User.query.filter_by(email=form.email.data).first()
        if user and bcrypt.check_password_hash(user.password,form.password.data):
            login_user(user,remember=form.remember.data)
            return render_template('home.html')
        else:
            flash('Login Unsucessful')

    return render_template('login.html',title='Login',form=form)

@app.route("/register",methods=['GET','POST'])
def register():
    form=RegistrationForm()
```

```
    if form.validate_on_submit():

hashed_password=bcrypt.generate_password_hash(form.password.data).decode('utf-8')
        user=User(username=form.username.data, email=form.email.data,
password=hashed_password)
        db.session.add(user)
        db.session.commit()
        flash('Account has been created for {form.username.data}!','success')
        return redirect(url_for('login'))
    return render_template('register.html',title='Register',form=form)


@app.route("/home",methods=['GET','POST'])
def home():
    if request.method=='POST':
        mk=request.form['mk']
        try:
            url='http://www.omdbapi.com/?apikey=34035a0a&type=movie&s='
            url=url+str(mk)
            url=url.replace(" ","+")
            json_obj=urllib.request.urlopen(url)
            data=json.load(json_obj)
            data=data['Search']

            with open('movieR.csv', 'a',newline='') as csv_file:
                fieldnames = ['Movie']
                writer = csv.DictWriter(csv_file, fieldnames=fieldnames)
                writer.writerow({'Movie': mk})

        except KeyError:
            return(render_template('negative.html',name=mk))
```

```
        return render_template('about.html',data=data)

    return render_template('home.html')

@app.route("/about",methods=['GET','POST'])
def about():
    if request.method=='POST':
        imdb_id=request.form['imdbid']
        url='http://www.omdbapi.com/?apikey=34035a0a&i='+str(imdb_id)
        json_obj=urllib.request.urlopen(url)
        data=json.load(json_obj)
        return render_template('info.html',data=data)

@app.route("/info",methods=['GET','POST'])
def info():
    if request.method=='POST':
        return render_template('ThankYou.html')
    return render_template('home.html')

@app.route("/ThankYou",methods=['GET','POST'])
def ThankYou():
    if request.method=='POST':
        return render_template('home.html')

f_name = ''
@app.route("/review",methods=['GET','POST'])
def review():

    NewMovies=[]
    prediction = []
```

```
with open('movieR.csv','r') as csvfile:
    readCSV = csv.reader(csvfile)
    NewMovies.append(random.choice(list(readCSV)))
    m_name = NewMovies[0][0]
    m_name = m_name.title()

    try:
        result_final = get_recommendations(m_name)
    except KeyError:
        print(" ** Anurag (Error): ", m_name)
        return render_template('review.html')

    f_name=m_name
    print(" ** Anurag (OK): ",f_name, ' ', m_name)
    names = []
    dates = []
    ratings = []
    overview=[]
    types=[]
    mid=[]
    for i in range(len(result_final)):
        names.append(result_final.iloc[i][0])
        dates.append(result_final.iloc[i][1])
        ratings.append(result_final.iloc[i][2])
        overview.append(result_final.iloc[i][3])
        types.append(result_final.iloc[i][4])
        mid.append(result_final.iloc[i][5])
        prediction.append(result_final.iloc[i][0])

suggestions = get_suggestions()
```

```
    details=[]
    url='http://www.omdbapi.com/?apikey=34035a0a&type=movie&t='
    for i in prediction:
        i=i.replace(' ','%20')
        json_obj=urllib.request.urlopen(url+i)
        data=json.load(json_obj)
        details.append(data)

    return render_template('review.html',details=details)

def get_recommendations(title):
    cosine_sim = cosine_similarity(count_matrix, count_matrix)
    idx = indices[title]
    sim_scores = list(enumerate(cosine_sim[idx]))
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    sim_scores = sim_scores[1:11]
    movie_indices = [i[0] for i in sim_scores]
    tit = df2['title'].iloc[movie_indices]
    dat = df2['release_date'].iloc[movie_indices]
    rating = df2['vote_average'].iloc[movie_indices]
    moviedetails=df2['overview'].iloc[movie_indices]
    movietypes=df2['keywords'].iloc[movie_indices]
    movieid=df2['id'].iloc[movie_indices]

    return_df = pd.DataFrame(columns=['Title','Year'])
    return_df['Title'] = tit
    return_df['Year'] = dat
    return_df['Ratings'] = rating
    return_df['Overview']=moviedetails
```

```
    return_df['Types']=movietypes
    return_df['ID']=movieid
    return return_df

def get_suggestions():
    data = pd.read_csv('tmdb.csv')
    return list(data['title'].str.capitalize())

@app.route("/hello",methods=['GET','POST'])
def hello():
    if request.method=='GET':
        return render_template('hello.html')

@app.route("/Top_20_Rating",methods=['GET','POST'])
def Top_20_Rating():
    if request.method=='GET':
        return render_template('Top_20_Rating.html')

@app.route("/Bottom_20_Rating",methods=['GET','POST'])
def Bottom_20_Rating():
    if request.method=='GET':
        return render_template('Bottom_20_Rating.html')

@app.route("/Highest_Grossing",methods=['GET','POST'])
def Highest_Grossing():
    if request.method=='GET':
        return render_template('Highest_Grossing.html')

@app.route("/Least_Grossing",methods=['GET','POST'])
def Least_Grossing():
```

```
    if request.method=='GET':
        return render_template('Least_Grossing.html')

@app.route("/Highest_Budget",methods=['GET','POST'])
def Highest_Budget():
    if request.method=='GET':
        return render_template('Highest_Budget.html')

@app.route("/Lowest_Budget",methods=['GET','POST'])
def Lowest_Budget():
    if request.method=='GET':
        return render_template('Lowest_Budget.html')

@app.route("/Most_lengthy_movies",methods=['GET','POST'])
def Most_lengthy_movies():
    if request.method=='GET':
        return render_template('Most_lengthy_movies.html')

@app.route("/logout",methods=['GET','POST'])
def logout():
    if request.method=='GET':
        return render_template('logout.html')

if __name__=='__main__':
    app.run(debug=True)
```