A

MINI PROJECT REPORT

ON

NETFLIX MOVIE RECOMMENDATION SYSTEM

Submitted in partial fulfillment of the requirements

For the award of Degree of

BACHELOR OF ENGINEERING

IN

CSE (AI ML)

Submitted By

P. PRADYUMNA **245320748107**

Under the guidance

Of

Mrs. M. SRAVANI

ASSISTANT PROFESSOR

Department of CSE(AIML)

NEIL GOGTE INSTITUTE OF TECHNOLOGY

Kachavanisingaram Village, Hyderabad, Telangana 500058.

JANUARY 2023

NEIL GOGTE INSTITUTE OF TECHNOLOGY

A Unit of Keshav Memorial Technical Education (KMTES)
Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

CERTIFICATE

This is to certify that the Mini project work entitled "**NETFLIX MOVIE RECOMMENDATION SYSTEM"** *is a bonafide work carried out by* **P.PRADYUMNA(245320748107)** of III year V semester **Bachelor of Engineering** *in CSE(***AIML)** *by Osmania University,Hyderabad during the academic year* **2022-2023** *is a record of bonafide work carried out by them. The results embodied in this report have not been submitted to any other University or Institution for the award of any degree*

Internal Guide

Mrs. M. Sravani
Assistant Professor

Head of Department

Dr. K. Madhuri
Associate Professor

External

NEIL GOGTE INSTITUTE OF TECHNOLOGY
A Unit of Keshav Memorial Technical Education (KMTES)
Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

DECLARATION

We hereby declare that the Mini Project Report entitled, "**NETFLIX MOVIE RECOMMENDATION SYSTEM**" submitted for the B.E degree is entirely our work and all ideas and references have been duly acknowledged. It does not contain any work for the award of any otherdegree.

Date:

P. PRADYUMNA **245320748107**

NEIL GOGTE INSTITUTE OF TECHNOLOGY

A Unit of Keshav Memorial Technical Education (KMTES)
Approved by AICTE, New Delhi & Affiliated to Osmania University, Hyderabad

ACKNOWLEDGEMENT

We are happy to express our deep sense of gratitude to the principal of the college **Prof. R. Shyam Sunder, Professor**, Neil Gogte Institute of Technology, for having provided us with adequate facilities to pursue our project.

We would like to thank**, Dr. K. Madhuri**, **Head of the Department,** CSE(AIML), Neil Gogte Institute of Technology, for having provided the freedom to use all the facilities available in the department, especially the laboratories and the library.

We would also like to thank my internal guide **Mrs. M. Sravani, Assistant Professor** for her technical guidance & constant encouragement.

We sincerely thank our seniors and all the teaching and non-teaching staff of the Department of Computer Science & Engineering for their timely suggestions, healthy criticism and motivation during this work.

Finally, we express our immense gratitude with pleasure to the other individuals who have either directly or indirectly contributed to our need at the right time for the development and success of this work.

ABSTRACT

Netflix is all about connecting people to the movies they love. To help customers find those movies, they developed world class movie recommendation system: Cosine similarity. Its job is to predict whether someone will enjoy a movie based on how much they liked or disliked other movies. Netflix use those predictions to make personal movie recommendations based on each customer's unique tastes. And while Cosine similarity is doing well, it can always be made better.

Now there are a lot of interesting alternative approaches to how Cosine similarity works that Netflix haven't tried. Some are described in the literature, some aren't. We're curious whether any of these can beat Cosine similarity by making better predictions. Because, frankly, if there is a much better approach it could make a big difference to our customers and our business.

Hence, initially, recommendations for a movie are obtained from 100% items of the dataset. Then those results are compared with recommendations from top 20% items in the dataset from each genre-pair. Time taken for both cases are also recorded to be compared. Also, the algorithm is applied to find items that have gained popularity among few voters which tells us that these items are new and rising in popularity .

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

TESTCASES

CHAPTER – 1

INTRODUCTION

1.1 :PROBLEM STATEMENT

Netflix provided a lot of anonymous rating data, and a prediction accuracy bar that is 10% better than what cosine similarity can do the same training data set. (Accuracy is a measurement of how closely predicted ratings of movies match subsequent actual ratings.)

1.2 :MOTIVATION

The primary goal of movie recommendation systems is **to filter and predict only those movies that a corresponding user is most likely to want to watch**. The ML algorithms for these recommendation systems use the data about this user from the system's database.

1.3 :SCOPE

Netflix Movie recommendation systems use a set of different filtration strategies and algorithms to help users find the most relevant films. The most popular categories of the ML algorithms used for movie recommendations include content-based filtering and collaborative filtering systems.

1.4 :OUTLINE

We take feedback from every visit to the Netflix service and continually re-train our algorithms with those signals to improve the accuracy of their prediction of what you're most likely to watch. Our data, algorithms, and computation systems continue to feed into each other to produce fresh recommendations to provide you with a product that brings you joy.

CHAPTER – 2

LITERATURE SURVEY

2.1 : EXISTING SYSTEM:

Netflix has 125 million hours of content that can be viewed each day. It has everything from current drama series to lifestyle shows, reality shows, the latest movies, history content, documentaries, design shows, comedy, kids shows and many other genres. But it will recommend Only high rated movies. For some movies there will be no rating and it will not recommend anything. Many of the users just watch the movies and they will not rate anything for that type of movies.

2.2 : PROPOSED SYSTEM:

The basic concept behind a Netflix movie recommendation system is quite simple. In particular, there are two main elements in every recommender system: users and items. The system generates movie predictions for its users, while items are the movies themselves.

The primary goal of Netflix movie recommendation systems is to filter and predict only those movies that a corresponding user is most likely to want to watch. The ML algorithms for these recommendation systems use the data about this user from the system's database. This data is used to predict the future behavior of the user concerned based on the information from the past.

CHAPTER - 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 : Software Requirements:

Operating System	:	Windows 7 (Min)
Front End	:	Html & CSS
Back End	:	Python (Flask using Gunicorn)
Database	:	Microsoft Excel

3.2 : Hardware Requirements:

Processor	:	Intel Pentium® Dual Core Processor (Min)
Speed	:	2.9 GHz (Min)
RAM	:	2 GB (Min)
Hard Disk	:	2 GB (Min)

CHAPTER-4

SYSTEM DESIGN

4.1 : Use case Diagram:

The use cases in the diagram represents the main processes in a Netflix movie recommendation system. Then they will be broken down into more specific use cases depending on the included processes of the main use case. Each of these use cases explains how the system handles the actions or scenarios requested by the user. The UML Use Case Diagram is a design used as one of the Methodology on Netflix movie recommendation system development. It represents the main functions or processes of the system as well as the specific processes included. They were also labelled properly to guide programmers and users about the structure of Online Food Ordering System.

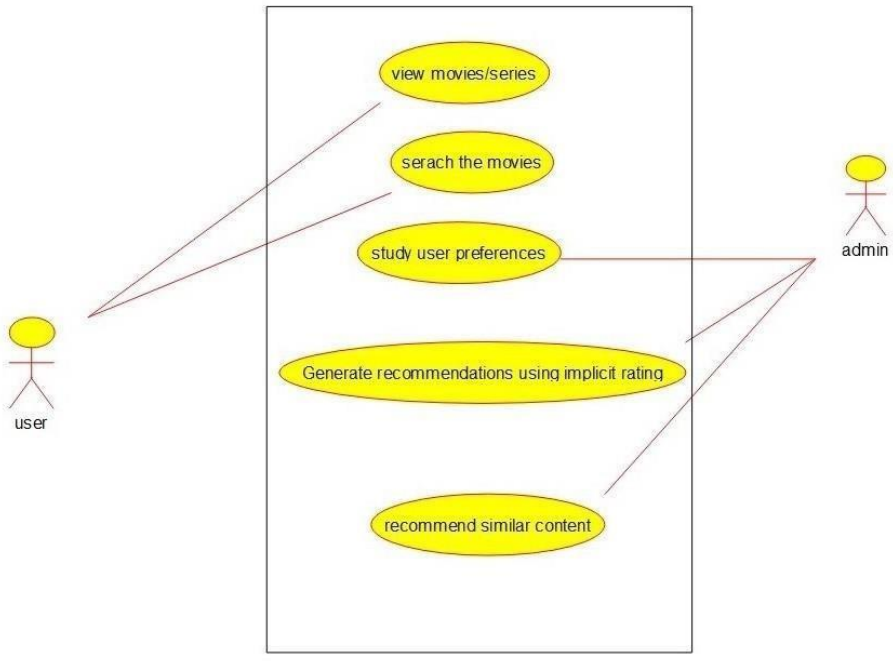


Fig 4.1: Use case diagram for User

4.2 : Class Diagram:

The Class diagram for Netflix movie recommendation system shows the structures of information or data that will be handled in the system. These data or information will be represented by classes. Each of the classes will have their attributes in accord to the methods they will use. So, the UML Class diagram was illustrated by a box with 3 partitions and the upper part was the name of the class, the middles are the attributes, and the bottom is for the methods. The arrows on them represents their relationships in each other. So, the classes that must be made in a server as customer, authorization, data, identification and subscription. The mentioned classes were just general. You must also relate the database on your Class Diagram for your system.

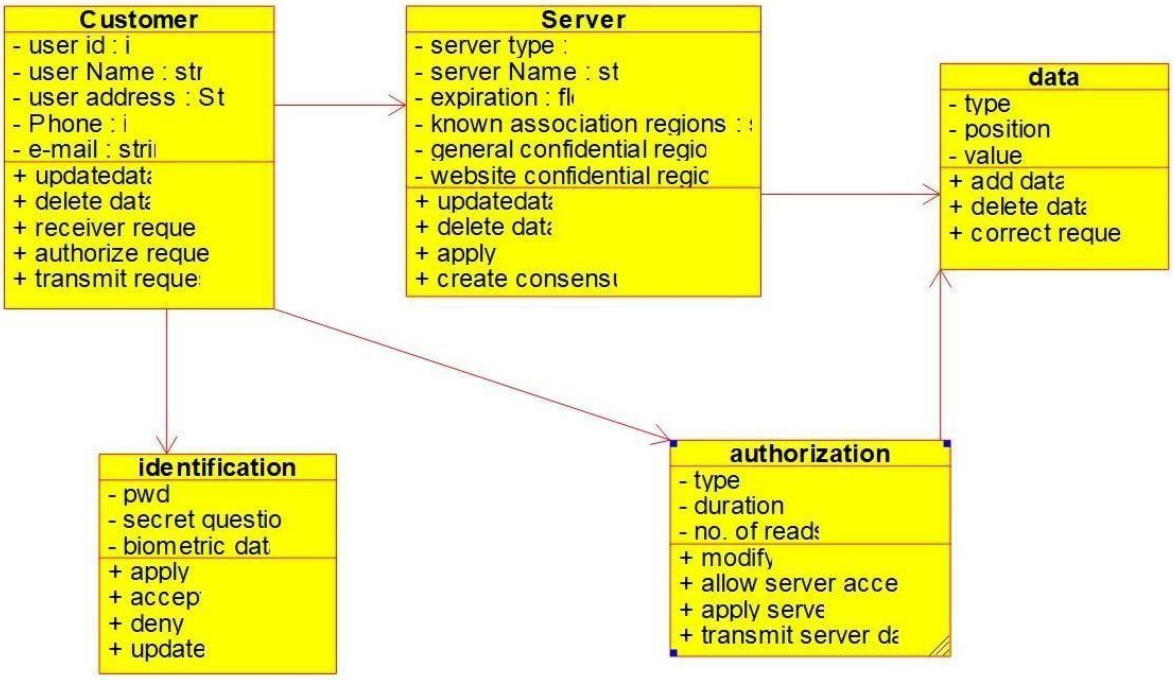


Fig.4.2: Class diagram for Netflix Movie Recommendation.

4.3 : Sequence Diagram:

The designed sequence diagram illustrates the series of events that occurs in Netflix movie recommendation system. In this illustration, the actors are represented by a stick man and the transactions or classes are represented by objects. It will give you clear explanation about the behavior of Netflix movie recommendation system in terms of processing the flow of instructions. This designed sequence diagram is able to show programmers and readers about the sequence of messages between the actor and the objects.

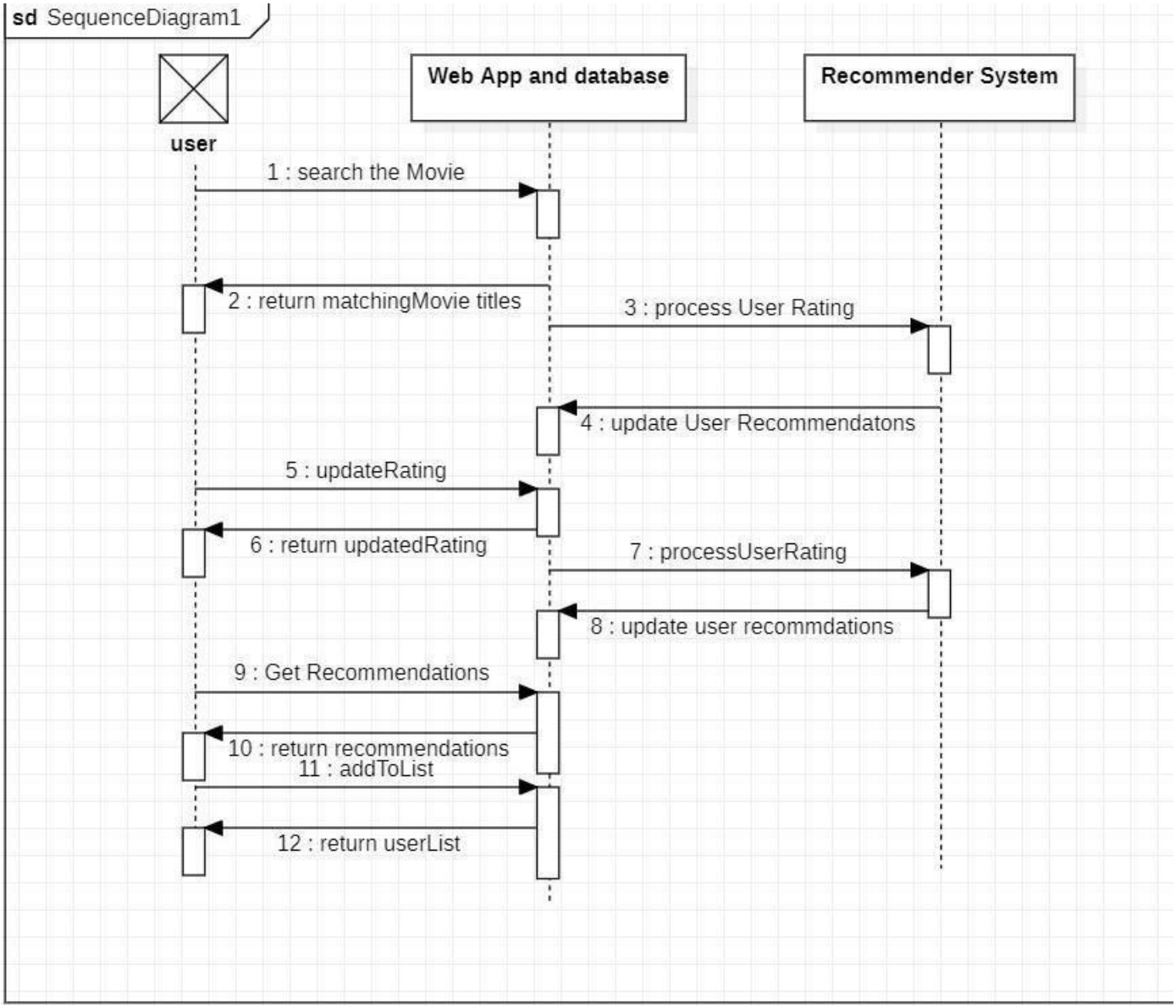


Fig4.3: Sequence Diagam for Netflix Movie Recommendation

4.4 :Architecture Of Netflix Movie Recommendation System:

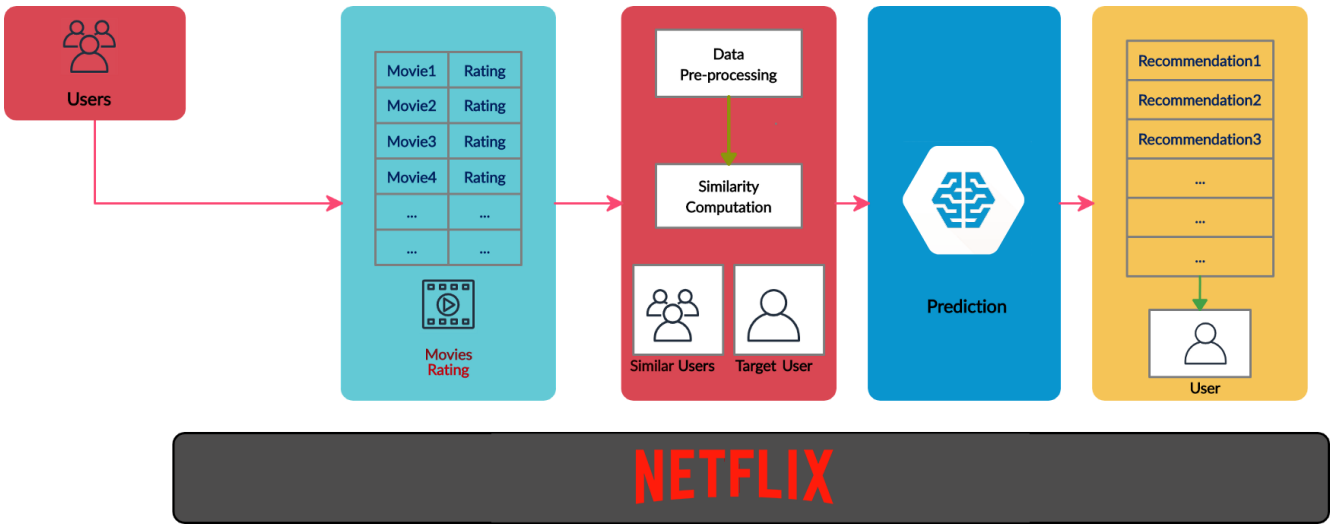
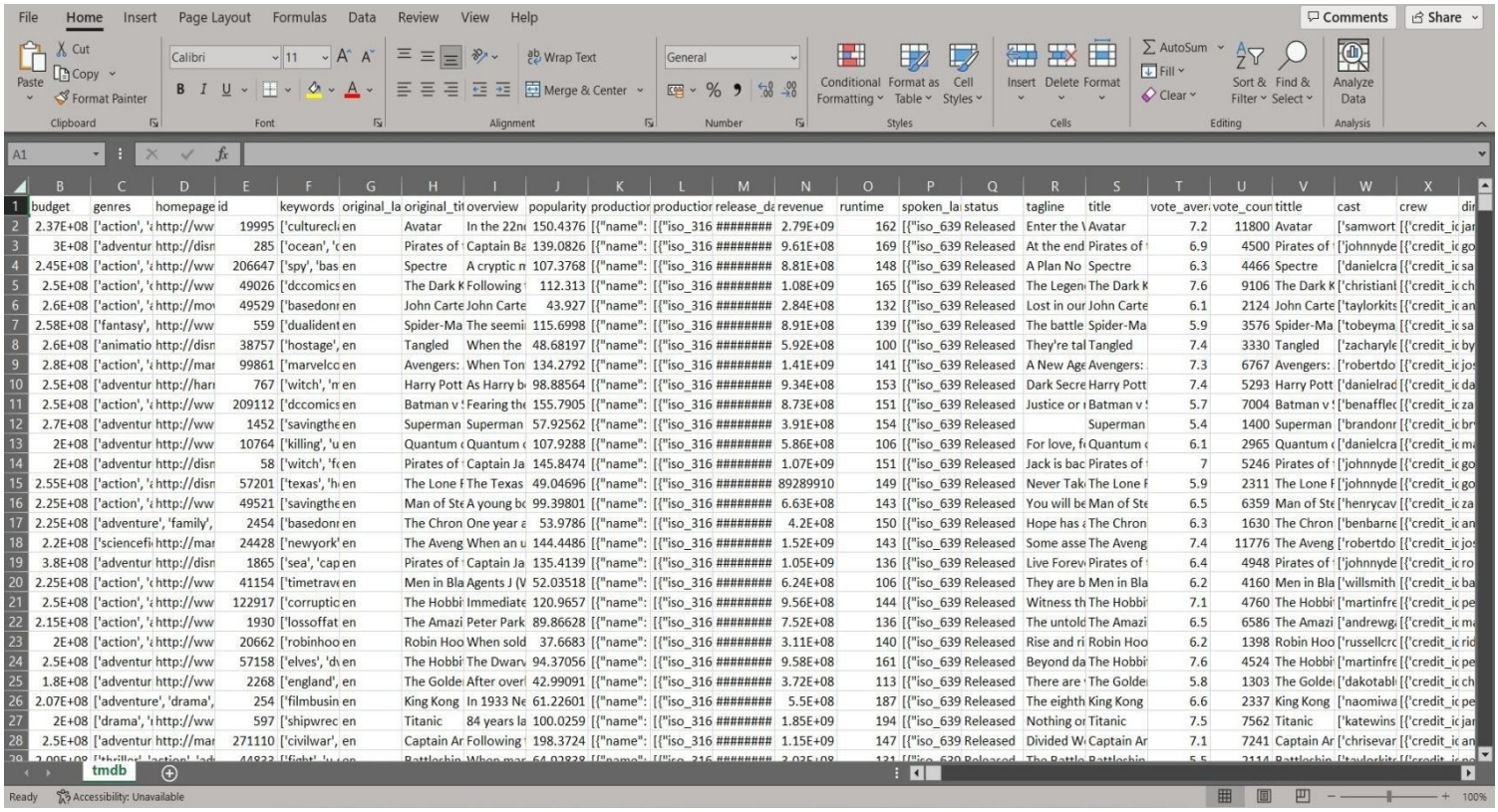


Fig4.4: Architecture of Netflix Movie Recommendation

CHAPTER – 5

IMPLEMENTATION



budget	genres	homepage	id	keywords	original_la	original_tit	overview	popularity	productior	productior	release_da	revenue	runtime	spoken_la	status	tagline	title	vote_aver	vote_coun	tittle	cast	crew
2.37E+08	['action', 'a	http://ww	19995	['culturecl	en	Avatar	In the 22n	150.4376	[{"name":	[{"iso_316	########	2.79E+09	162	[{"iso_639	Released	Enter the \	Avatar	7.2	11800	Avatar	['samwort	[{'credit_ic
3E+08	['adventur	http://disn	285	['ocean', '	en	Pirates of	Captain Ba	139.0826	[{"name":	[{"iso_316	########	9.61E+08	169	[{"iso_639	Released	At the end	Pirates of	6.9	4500	Pirates of	['johnnyde	[{'credit_ic
2.45E+08	['action', 'a	http://ww	206647	['spy', 'bas	en	Spectre	A cryptic n	107.3768	[{"name":	[{"iso_316	########	8.81E+08	148	[{"iso_639	Released	A Plan No	Spectre	6.3	4466	Spectre	['danielcra	[{'credit_ic
2.5E+08	['action', 'a	http://ww	49026	['dccomics	en	The Dark K	Following	112.313	[{"name":	[{"iso_316	########	1.08E+09	165	[{"iso_639	Released	The Legen	The Dark K	7.6	9106	The Dark K	['christianl	[{'credit_ic
2.6E+08	['action', 'a	http://mo	49529	['basedon	en	John Carte	John Carte	43.927	[{"name":	[{"iso_316	########	2.84E+08	132	[{"iso_639	Released	Lost in our	John Carte	6.1	2124	John Carte	['taylorkits	[{'credit_ic
2.58E+08	['fantasy',	http://ww	559	['dualident	en	Spider-Ma	The seemi	115.6998	[{"name":	[{"iso_316	########	8.91E+08	139	[{"iso_639	Released	The battle	Spider-Ma	5.9	3576	Spider-Ma	['tobeyma	[{'credit_ic
2.6E+08	['animatio	http://disn	38757	['hostage',	en	Tangled	When the	48.68197	[{"name":	[{"iso_316	########	5.92E+08	100	[{"iso_639	Released	They're tal	Tangled	7.4	3330	Tangled	['zacharyle	[{'credit_ic
2.8E+08	['action', 'a	http://mar	99861	['marvelco	en	Avengers:	When Ton	134.2792	[{"name":	[{"iso_316	########	1.41E+09	141	[{"iso_639	Released	A New Age	Avengers:	7.3	6767	Avengers:	['robertdo	[{'credit_ic
2.5E+08	['adventur	http://har	767	['witch', 'n	en	Harry Pott	As Harry b	98.88564	[{"name":	[{"iso_316	########	9.34E+08	153	[{"iso_639	Released	Dark Secre	Harry Pott	7.4	5293	Harry Pott	['danielrad	[{'credit_ic
2.5E+08	['action', 'a	http://ww	209112	['dccomics	en	Batman v S	Fearing the	155.7905	[{"name":	[{"iso_316	########	8.73E+08	151	[{"iso_639	Released	Justice or	Batman v S	5.7	7004	Batman v S	['benaffleo	[{'credit_ic
2.7E+08	['adventur	http://ww	1452	['savingthe	en	Superman	Superman	57.92562	[{"name":	[{"iso_316	########	3.91E+08	154	[{"iso_639	Released		Superman	5.4	1400	Superman	['brandonr	[{'credit_ic
2E+08	['adventur	http://ww	10764	['killing', 'u	en	Quantum o	Quantum o	107.9288	[{"name":	[{"iso_316	########	5.86E+08	106	[{"iso_639	Released	For love, fo	Quantum o	6.1	2965	Quantum o	['danielcra	[{'credit_ic
2E+08	['adventur	http://disn	58	['witch', 'fo	en	Pirates of	Captain Ja	145.8474	[{"name":	[{"iso_316	########	1.07E+09	151	[{"iso_639	Released	Jack is bac	Pirates of	7	5246	Pirates of	['johnnyde	[{'credit_ic
2.55E+08	['action', 'a	http://disn	57201	['texas', 'h	en	The Lone F	The Texas	49.04696	[{"name":	[{"iso_316	########	89289910	149	[{"iso_639	Released	Never Tak	The Lone F	5.9	2311	The Lone F	['johnnyde	[{'credit_ic
2.25E+08	['action', 'a	http://ww	49521	['savingthe	en	Man of Ste	A young bo	99.39801	[{"name":	[{"iso_316	########	6.63E+08	143	[{"iso_639	Released	You will be	Man of Ste	6.5	6359	Man of Ste	['henrycav	[{'credit_ic
2.25E+08	['adventure', 'family',		2454	['basedon	en	The Chron	One year a	53.9786	[{"name":	[{"iso_316	########	4.2E+08	150	[{"iso_639	Released	Hope has a	The Chron	6.3	1630	The Chron	['benbarne	[{'credit_ic
2.2E+08	['sciencefi	http://mar	24428	['newyork'	en	The Aveng	When an u	144.4486	[{"name":	[{"iso_316	########	1.52E+09	143	[{"iso_639	Released	Some asse	The Aveng	7.4	11776	The Aveng	['robertdo	[{'credit_ic
3.8E+08	['adventur	http://disn	1865	['sea', 'cap	en	Pirates of	Captain Ja	135.4139	[{"name":	[{"iso_316	########	1.05E+09	136	[{"iso_639	Released	Live Forev	Pirates of	6.4	4948	Pirates of	['johnnyde	[{'credit_ic
2.25E+08	['action', 'a	http://ww	41154	['timetrave	en	Men in Bla	Agents J (V	52.03518	[{"name":	[{"iso_316	########	6.24E+08	106	[{"iso_639	Released	They are b	Men in Bla	6.2	4160	Men in Bla	['willsmith	[{'credit_ic
2.5E+08	['action', 'a	http://ww	122917	['corruptic	en	The Hobbi	Immediate	120.9657	[{"name":	[{"iso_316	########	9.56E+08	144	[{"iso_639	Released	Witness th	The Hobbi	7.1	4760	The Hobbi	['martinfre	[{'credit_ic
2.15E+08	['action', 'a	http://ww	1930	['lossoffat	en	The Amazi	Peter Park	89.86628	[{"name":	[{"iso_316	########	7.52E+08	136	[{"iso_639	Released	The untold	The Amazi	6.5	6586	The Amazi	['andrewg	[{'credit_ic
2E+08	['action', 'a	http://ww	20662	['robinhoo	en	Robin Hoo	When sold	37.6683	[{"name":	[{"iso_316	########	3.11E+08	140	[{"iso_639	Released	Rise and ri	Robin Hoo	6.2	1398	Robin Hoo	['russellcro	[{'credit_ic
2.5E+08	['adventur	http://ww	57158	['elves', 'd	en	The Hobbi	The Dwarv	94.37056	[{"name":	[{"iso_316	########	9.58E+08	161	[{"iso_639	Released	Beyond da	The Hobbi	7.6	4524	The Hobbi	['martinfre	[{'credit_ic
1.8E+08	['adventur	http://ww	2268	['england',	en	The Golde	After over	42.99091	[{"name":	[{"iso_316	########	3.72E+08	113	[{"iso_639	Released	There are	The Golde	5.8	1303	The Golde	['dakotabl	[{'credit_ic
2.07E+08	['adventure', 'drama',		254	['filmbusin	en	King Kong	In 1933 Ne	61.22601	[{"name":	[{"iso_316	########	5.5E+08	187	[{"iso_639	Released	The eighth	King Kong	6.6	2337	King Kong	['naomiwa	[{'credit_ic
2E+08	['drama', 'r	http://ww	597	['shipwrec	en	Titanic	84 years la	100.0259	[{"name":	[{"iso_316	########	1.85E+09	194	[{"iso_639	Released	Nothing or	Titanic	7.5	7562	Titanic	['katewins	[{'credit_ic
2.5E+08	['adventur	http://mar	271110	['civilwar',	en	Captain Ar	Following	198.3724	[{"name":	[{"iso_316	########	1.15E+09	147	[{"iso_639	Released	Divided W	Captain Ar	7.1	7241	Captain Ar	['chrisevar	[{'credit_ic

Figure 5.1: Movies Dataset

5.2 SAMPLE CODE

```
import flask
import pandas as pd
from
sklearn.feature_extractio
n.text import
TfidfVectorizer
from
sklearn.metrics.pairwise
import cosine_similarity

app =
flask.Flask(_name_,
template_folder='templat
es')

df2 =
pd.read_csv('./model/tmd
b.csv')

tfidf =
TfidfVectorizer(stop_wo
rds='english',analyzer='w
ord')

#Construct the required
TF-IDF matrix by fitting
and transforming the
data
tfidf_matrix =
tfidf.fit_transform(df2['s
oup'])
print(tfidf_matrix.shape)
```

```
#construct cosine similarity matrix
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)
print(cosine_sim.shape)

df2 = df2.reset_index()
indices = pd.Series(df2.index, index=df2['title']).drop_duplicates()

# create array with all movie titles
all_titles = [df2['title'][i] for i in range(len(df2['title']))]

def get_recommendations(title):
    # Get the index of the movie that matches the title
    idx = indices[title]
    # Get the pairwise similarity scores of all movies with that movie
    sim_scores = list(enumerate(cosine_sim[idx]))
    # Sort the movies based on the similarity scores
    sim_scores = sorted(sim_scores, key=lambda x: x[1], reverse=True)
    # Get the scores of the 10 most similar movies
    sim_scores = sim_scores[1:11]
    # print similarity scores
    print("\n movieId      score")
    for i in sim_scores:
        print(i)

    # Get the movie indices
    movie_indices = [i[0] for i in sim_scores]

    # return list of similar movies
    return_df = pd.DataFrame(columns=['Title','Homepage'])
    return_df['Title'] = df2['title'].iloc[movie_indices]
    return_df['Homepage'] = df2['homepage'].iloc[movie_indices]
    return_df['ReleaseDate'] = df2['release_date'].iloc[movie_indices]
    return return_df

# Set up the main route
@app.route('/', methods=['GET', 'POST'])

def main():
    if flask.request.method == 'GET':
        return(flask.render_template('index.html'))

    if flask.request.method == 'POST':
        m_name = " ".join(flask.request.form['movie_name'].title().split())
#       check = difflib.get_close_matches(m_name,all_titles,cutout=0.50,n=1)
```

```
if m_name not in all_titles:
        return(flask.render_template('notFound.html',name=m_name))
    else:
        result_final = get_recommendations(m_name)
        names = []
        homepage = []
        releaseDate = []
        for i in range(len(result_final)):
            names.append(result_final.iloc[i][0])
            releaseDate.append(result_final.iloc[i][2])
            if(len(str(result_final.iloc[i][1]))>3):
                homepage.append(result_final.iloc[i][1])
            else:
                homepage.append("#")


        return
flask.render_template('found.html',movie_names=names,movie_homepage=homepage,search_name=m_name,
movie_releaseDate=releaseDate)

if__name__== '__main__':
    app.run(host="127.0.0.1", port=8080, debug=True)
    #app.run()
```

CHAPTER – 6

TESTING

6.1 TEST CASES

Test Case to check whether the required Software is installed on the systems

Test Case ID:	1
Test Case Name:	Required Software Testing
Purpose:	To check whether the required Software is installed on the systems
Input:	Enter python command
Expected Result:	Should Display the version number for the python
Actual Result:	Displays python version
Failure	If the python environment is not installed, then the Deployment fails

Table 6.1 python Installation verification

Test Case to check Program Integration Testing

Test Case ID:	2
Test Case Name:	Installing flask,Numpy,Pandas
Purpose:	To ensure that all the modules work together
Input:	All the modules should be accessed.
Expected Result:	All the modules should be functioning properly.
Actual Result:	All the modules should be functioning properly.
Failure	If any module fails to function properly, the implementation fails.

Table 6.2 python Flask Testing

Test Case to Collect Dataset and Load the Dataset

Test Case ID:	3
Test Case Name:	Collect Dataset and Load the Dataset
Purpose:	Check Dataset is collected, and the data is stored
Input:	Provide Dataset as input
Expected Result:	Dataset is collected and view the Dataset and store the Dataset
Actual Result:	Load the Dataset and view the Dataset and store
Failure	If the dataset is not loaded, it will throw an error.

Table 6.3 Collect Dataset and Load the Dataset

Test Case to check whether the species is recognized

Test Case ID:	4
Test Case Name:	Movie Recommendations
Purpose:	Movie Recommendations in python
Input:	Search a Movie
Expected Result:	Recommendations of a Movie
Actual Result:	We get the recommendations of that particular movie
Failure	If the Movie is not found, it will display an error

Table 6.4 Movies Recommendation

6.5: Testing:

```
C:\Users\prady\OneDrive\Desktop\Pro\pro1\pro2>python app.py
(4803, 11520)
(4803, 4803)
 * Tip: There are .env or .flaskenv files present. Do "pip install python-dotenv" to use them.
 * Serving Flask app 'app'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:8080
Press CTRL+C to quit
 * Restarting with stat
(4803, 11520)
(4803, 4803)
 * Tip: There are .env or .flaskenv files present. Do "pip install python-dotenv" to use them.
 * Debugger is active!
 * Debugger PIN: 107-208-316
127.0.0.1 - - [03/Jan/2023 21:00:47] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [03/Jan/2023 21:00:48] "GET /favicon.ico HTTP/1.1" 404 -

 movieId      score
(7, 0.6295283489537541)
(26, 0.461282761061055)
(85, 0.3601435118703404)
(79, 0.3112075604490576)
(169, 0.2812601083492169)
(174, 0.2625099793089952)
(421, 0.24548205972860185)
(1294, 0.24233908368019086)
(1382, 0.22091681716955158)
(4129, 0.21542796572484335)
127.0.0.1 - - [03/Jan/2023 21:00:51] "POST / HTTP/1.1" 200 -
127.0.0.1 - - [03/Jan/2023 21:00:57] "GET / HTTP/1.1" 200 -

 movieId      score
(7, 0.6295283489537541)
(26, 0.461282761061055)
(85, 0.3601435118703404)
(79, 0.3112075604490576)
(169, 0.2812601083492169)
(174, 0.2625099793089952)
(421, 0.24548205972860185)
```

CHAPTER - 7

SCREENSHOTS

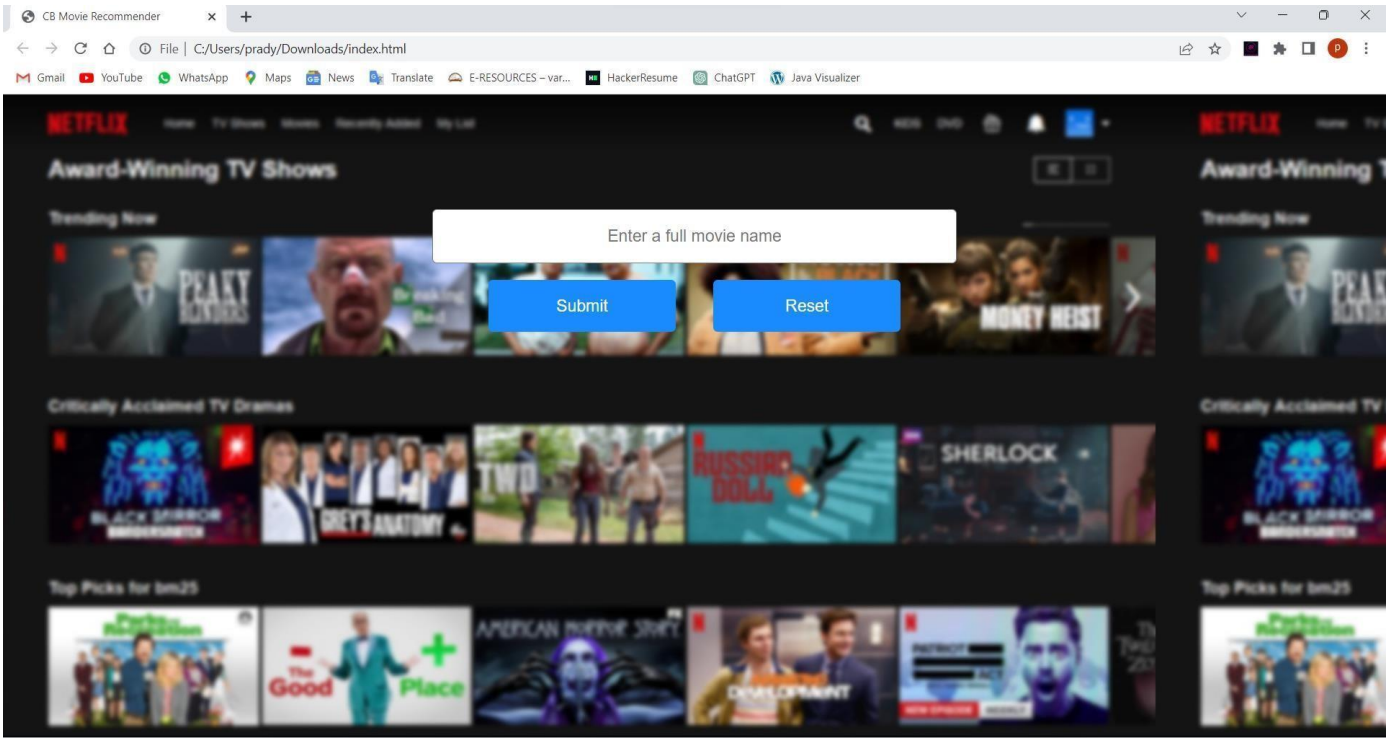


Figure 7.1: HOME PAGE

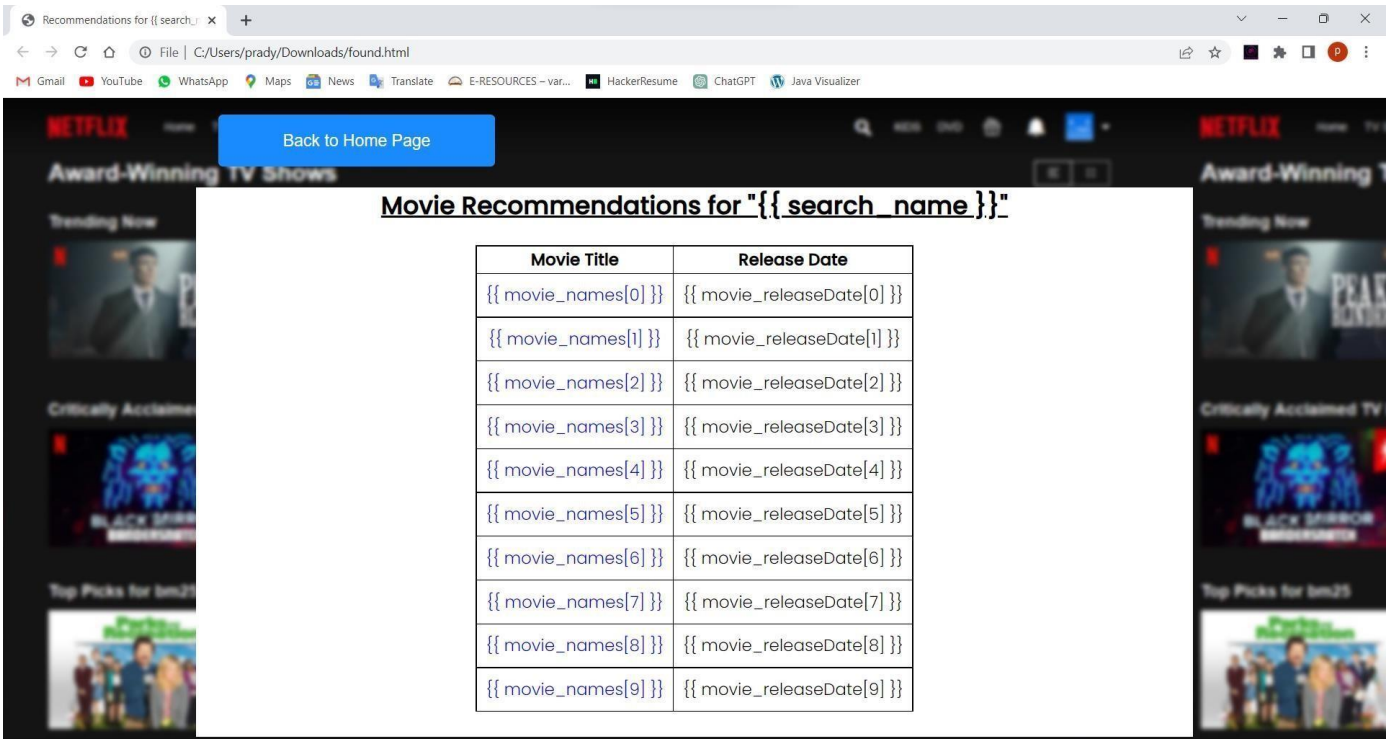


Figure 7.2: MOVIE FOUND

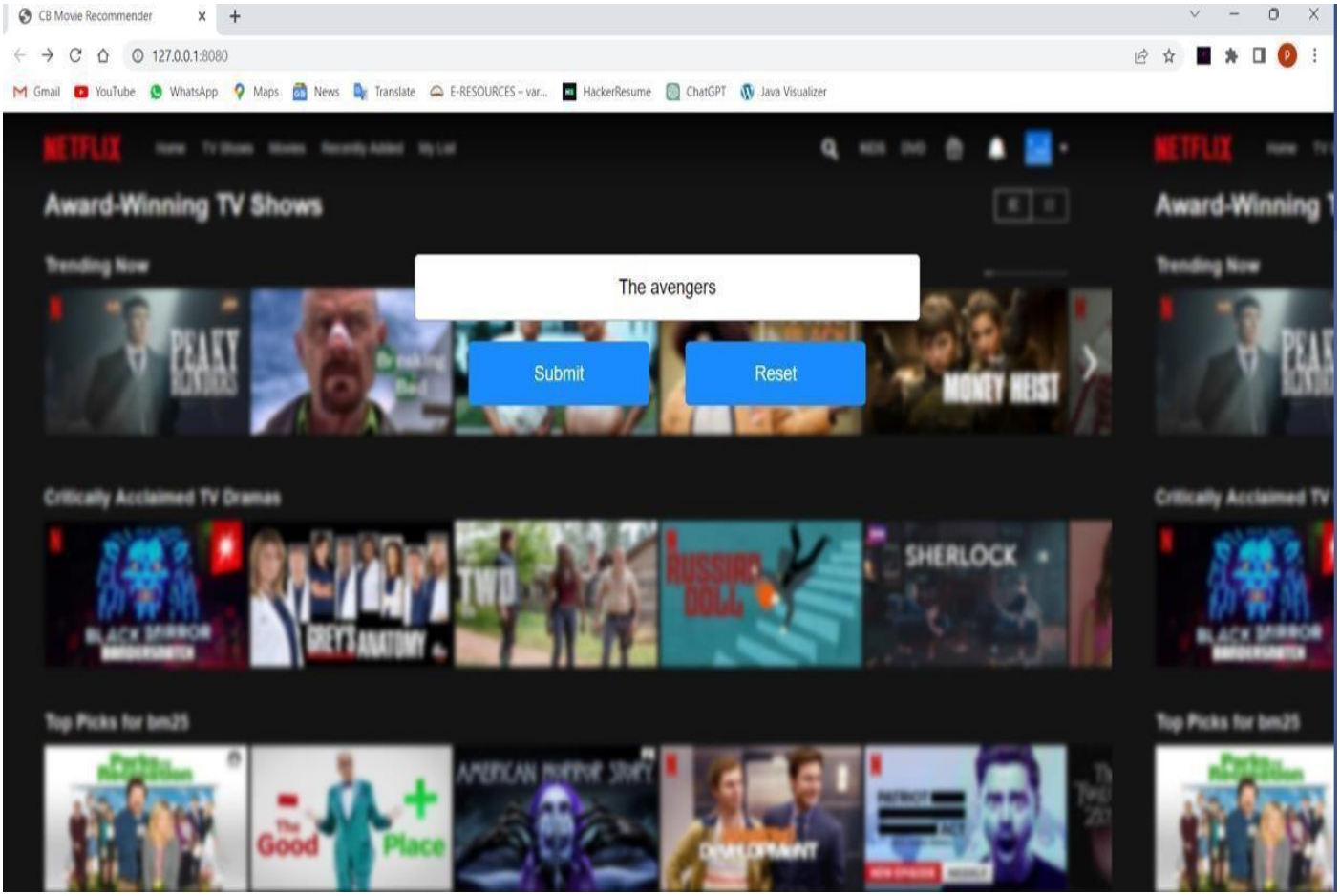


Figure 7.3: Searching for a Movie

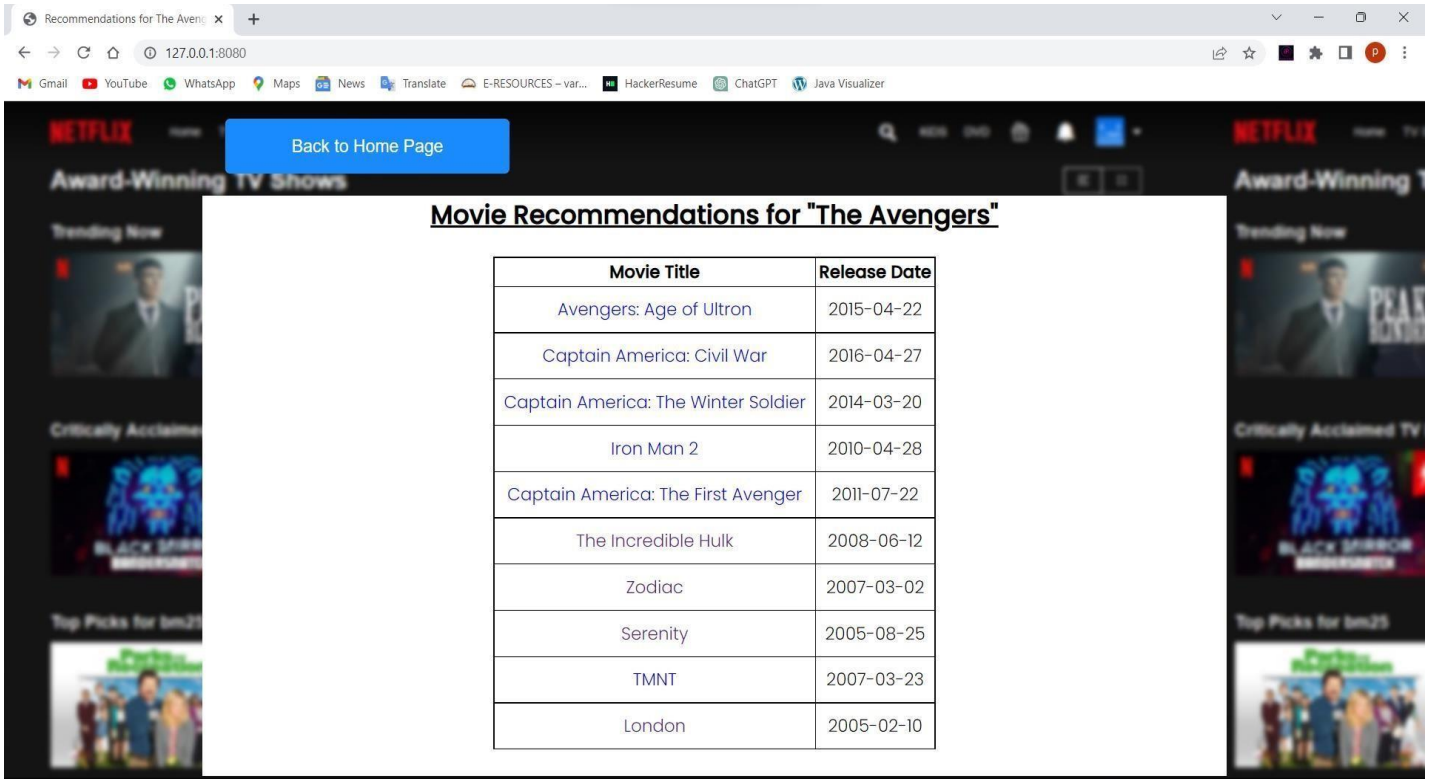


Figure 7.4: MOVIE RECOMMENDATIONS FOUND

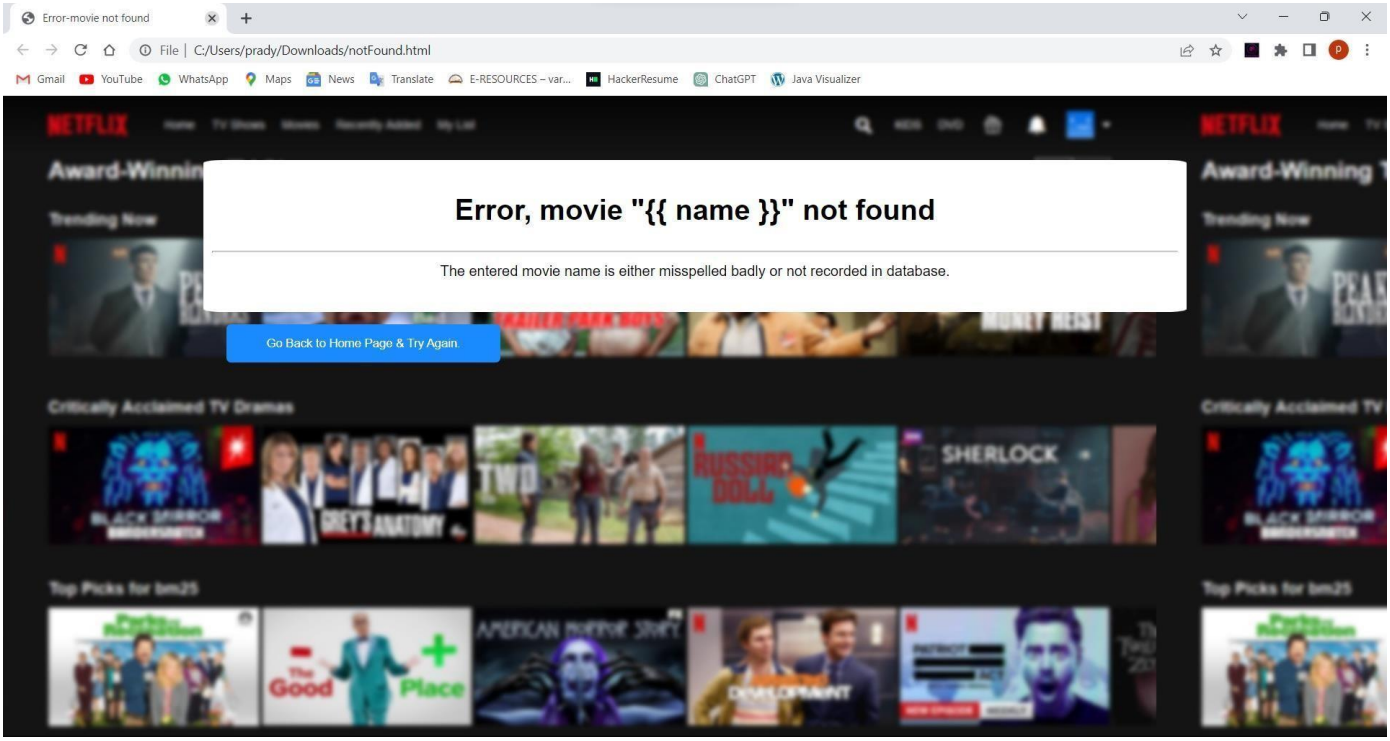


Figure 7.5: MOVIES NOT FOUND

CHAPTER - 8

CONCLUSION AND FUTURE SCOPE

By employing several algorithms to select movies, I was able to meet the project's objectives. I've discussed the several algorithms that makeup Netflix's recommender system, as well as the steps I take to improve it and some of our current difficulties. Humans are faced with an increasing number of choices in every aspect of their lives, including media such as videos, music, and books, as well as other taste-based questions such as vacation rentals, restaurants, and so on, but also in areas such as health insurance plans, treatments, and tests, job searches, education and learning, dating and finding life partners, and many other areas where choice is important. I am convinced that recommender systems will continue to play an important part in using the massive amounts of data now accessible to make these decisions more manageable, successfully guiding people to the truly best few options for them to assess, resulting in more informed judgments. I also believe that recommender systems can democratize access to long-tail products, services, and information since machines can learn from far bigger data sets than experts, allowing them to make effective predictions in areas where human capacity is simply insufficient to generalize usefully.

APPENDIX A: TOOLS AND TECHNOLOGIES

- **PYTHON V3:** The Python language comes with many libraries and frameworks that make coding easy. This also saves a significant amount of time.

- **Flask:** Flask is used for developing web applications using python, implemented on Werkzeug and Jinja2. Advantages of using Flask framework are: There is a built-in development server and a fast debugger provided.

- **Gunicorn:** Gunicorn is a pure-Python HTTP server for WSGI applications. It allows you to run any Python application concurrently by running multiple Python processes within a single dyno. It provides a perfect balance of performance, flexibility, and configuration simplicity
- NUMPY: NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices.

- **WINDOWS 11:** Windows 11 was used as the operating system.

- **Pandas:** Pandas is a Python library. Pandas is used to analyze data

- **Kaggle:** Kaggle is an online community platform for data scientists and machine learning enthusiasts. Kaggle allows users to collaborate with other users, find and publish datasets, use GPU integrated notebooks, and compete with other data scientists to solve data science challenges.