VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM, KARNATAKA

Minor Project-1

Report on:

Book Recommendation system

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF ENGINEERING

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted by

Sl . No	USN	Name
1	2SD22EC017	Chaitanya Kawale
2	2SD22EC025	Harshvardhan B.M
3	2SD22EC027	Kedar Mujumdar
4	2SD22EC056	Prem Patil

Under the guidance of

Prof. Raghuram K.M

DEPARTMENT OF ELECTRONICS AND COMMUNICATION

S.D.M. COLLEGE OF ENGINEERING & TECHNOLOGY,

DHARWAD-580002

2024-25

S.D.M COLLEGE OF ENGINEERING & TECHNOLOGY, DHARWAD-580002

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the Minor-Project-1 work and presentation entitled **"Book Recommendation system"** *is a bonafide work carried out by* **Chaitanya Kawale(2SD22EC017)**,**Harshvardhan.B.M(2SD22EC025)**,**Kedar.Mujumdar (2SD22EC027)**,and **Prem.Patil(2SD22EC056),** *students* ***of S. D. M. College of Engineering & Technology, Dharwad****, in partial fulfillment for the award of* **Bachelor of Engineering in Elctronics and Communication Engineering** of *Visvesvaraya Technological University, Belgaum, during the year 2024-25. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report deposited in the department library. The Minor-Project- 1 has been approved, as it satisfies the academic requirements in respect of project report prescribed for the said degree.*

Prof. Raghuram K.M
Project Guide

Dr. S A Joshi
HOD-ECE

ABSTRACT

In an era where the volume of published books has grown exponentially, a robust book recommender system plays a pivotal role in aiding readers to discover literature that aligns with their preferences. The importance of such a system is underscored by its ability to personalize reading experiences, improve reader engagement, and foster a deeper connection between readers and literature. This project aims to develop an intelligent book recommender system that not only enhances the user experience but also bridges the gap between readers and the ever-expanding literary world.

The core functionality of this system revolves around analyzing user preferences and suggesting books tailored to their interests. By leveraging data such as user ratings, genres, book descriptions, and metadata, the recommender system ensures relevant and accurate recommendations. The project utilizes advanced methodologies, integrating *content-based filtering* and *collaborative filtering* techniques to create a hybrid model. Content-based filtering focuses on recommending books similar to those a user has previously liked, utilizing features such as genres, authors, and descriptions. Collaborative filtering, on the other hand, identifies patterns among user interactions to suggest books that other similar users have enjoyed. Together, these approaches provide a comprehensive and effective recommendation mechanism.

The project employs machine learning algorithms and similarity measures, such as cosine similarity and matrix factorization, to analyze and predict user preferences. A user-friendly interface further ensures ease of interaction, enabling users to seamlessly explore their recommendations. The implementation also emphasizes scalability and real-time performance, making it suitable for large-scale deployment in online bookstores, digital libraries, or educational platforms.

Table of Contents

Problem Statement:

Creating a web based platform for book recommender system which recommendes books based on title and previous search history of registered user.

Introduction:

In today's digital age, the sheer volume of available books has grown exponentially, making it challenging for readers to discover content that aligns with their tastes and preferences. This challenge has given rise to **book recommendation systems**, which help users navigate vast book collections by suggesting titles that they are likely to enjoy. These systems play a crucial role in enhancing user experience by providing personalized book recommendations, similar to how platforms like Netflix or Spotify recommend movies and music.

With the exponential growth in the volume of available books and user-generated content (reviews, ratings, etc.), a **Book Recommendation System** becomes essential to assist readers in discovering new books aligned with their interests. The system utilizes machine learning algorithms to predict which books a user may enjoy based on their preferences and the behaviors of similar users. This project aims to build a system that leverages **collaborative filtering** to suggest personalized book recommendations, enhancing user satisfaction and engagement.

Literature survey:

- This paper proposes a simple comprehensible system for book recommendations that help readers to recommend the correct book. In recent years, data analysis challenge has been centered on for the administration recommendation system.

- Book recommendation system has been developed rapidly because of the net technology and library modernization, which provide a replacement means for the librarians to amass the readers demands.

- Recommender Systems are around for more than a decade currently. Selecting what book to scan next has always been an issue for several. Even for college kids, deciding which textbook or book of facts to scan on a subject unknown to them could be a massive question. In this paper authors have given a model for a web based customized hybrid book recommender system that exploits varied aspects of giving recommendations except for the regular cooperative and content based filtering approaches.

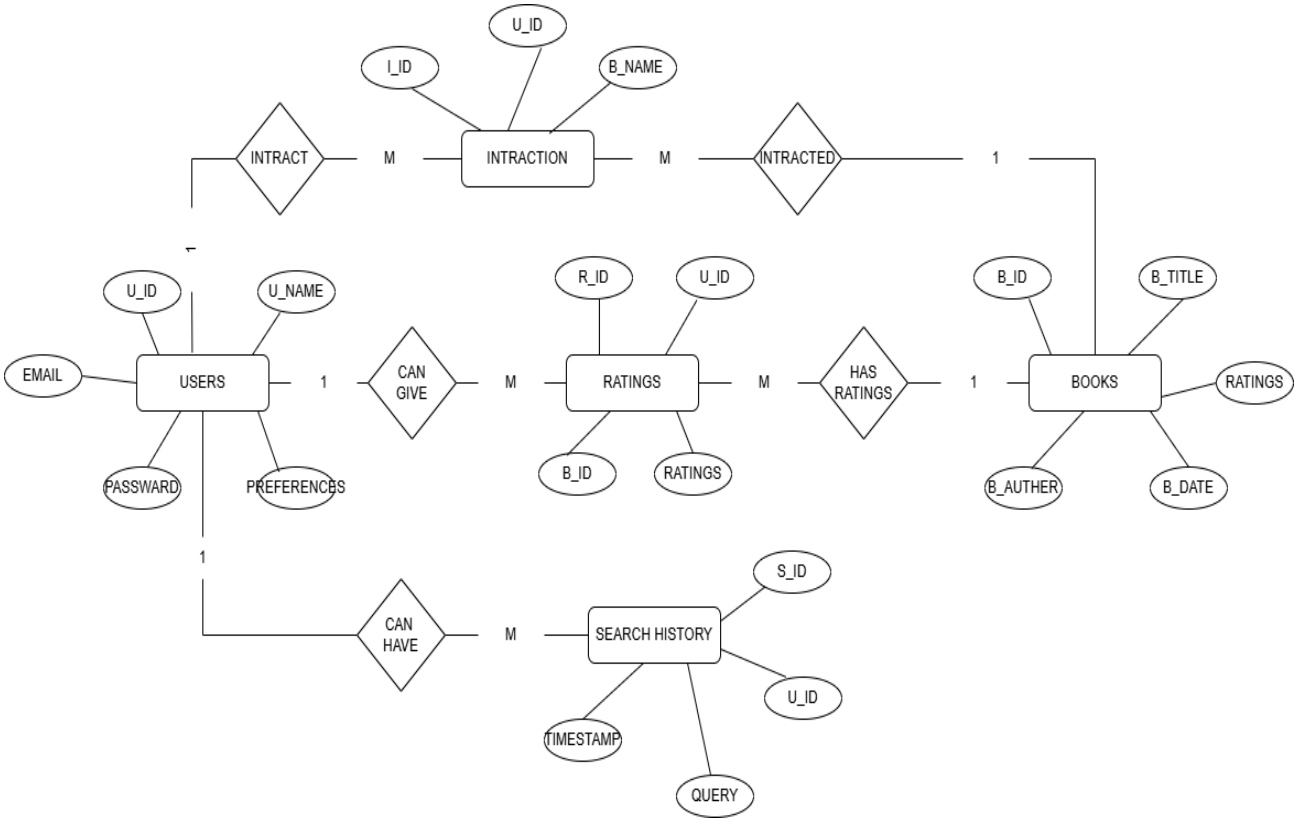
Sr. No.	Paper	Best Technique	Other Techniques Tested	Dataset	Performance Parameter
1.	Machine Learning based Efficient Recommendation System for Book Selection using User based Collaborative Filtering Algorithm [1]	Collaborative filtering using Constrained Pearson Correlation as similarity measure	Collaborative filtering using, 1. Cosine Similarity, 2. Pearson Correlation Coefficient, 3.Constrained Pearson Correlation, 4. JACCARD Similarity Measures on TAG matrix	Kaggle good reads data set	Recall, F1 Score, Mean Absolute Presidion
2.	An Online Book Recommendation System Based on Web Service [2]	Web Service	×	Langlang bookstore, Dangdang bookstore	Pareto curve

3.	Book Recommendation System Based on Combine Features of Content Based Filtering, Collaborative Filtering and Association Rule Mining [3]	Combination of content filtering, collaborative filtering and association rule mining	×	Book transaction database	×
4.	Book Recommendation System Using Opinion Mining Technique [4]	Assigning weights to features based on user opinion	×	Data from different search engines	×
5.	Book Recommendation System for Digital Library Based on User Profiles by Using Association Rule [5]	UCL Model	UL Model	65,521 transactions during January 2012 to February 2014	Precision
6.	Web-based Personalized Hybrid Book Recommendation System [6]	Combination of collaborative, content and demographic filtering	×	Scraped data	×
7.	Online Book Recommendation System [7]	Collaborative filtering method based on Pearson correlation coefficient	×	Data from www.readly.ru	Quality of recommendations, Speed of getting recommendations

It can be concluded that problem of book recommendation can be solved by various techniques such as collaborative filtering, web service, hybrid techniques, Opinion mining & association rules. Furthermore, performance of the system is dependent upon dataset used.

Detailed Design:

ER Diagram:

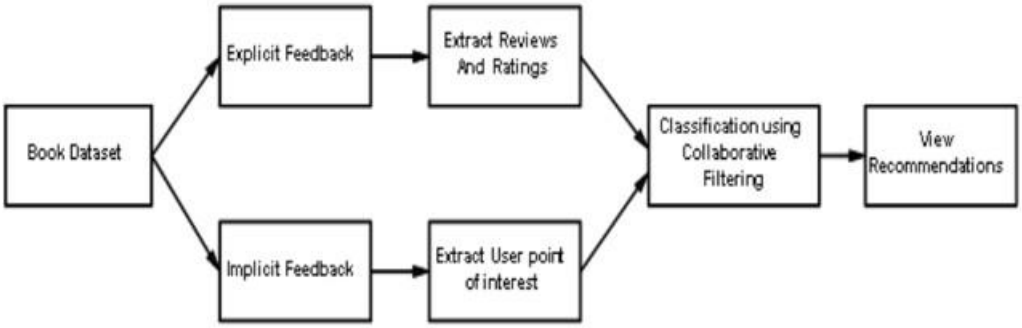


Users:
Primary Key: User_id
Relationships:
One-to-Many with Ratings: A user can rate multiple books.
One-to-Many with Interactions: A user can interact with multiple books.
One-to-Many with Search_History: A user can have multiple search queries.

Books:
Primary Key: Book_id
Relationships:
One-to-Many with Ratings: A book can be rated by multiple users.
One-to-Many with Interactions: A book can be interacted with by multiple users.

Ratings:
Primary Key: Rating_id
Foreign Keys:
User_id (references Users)
Book_id (references Books)
Relationships:
Many-to-One with Users: Each rating is associated with one user.
Many-to-One with Books: Each rating is associated with one book

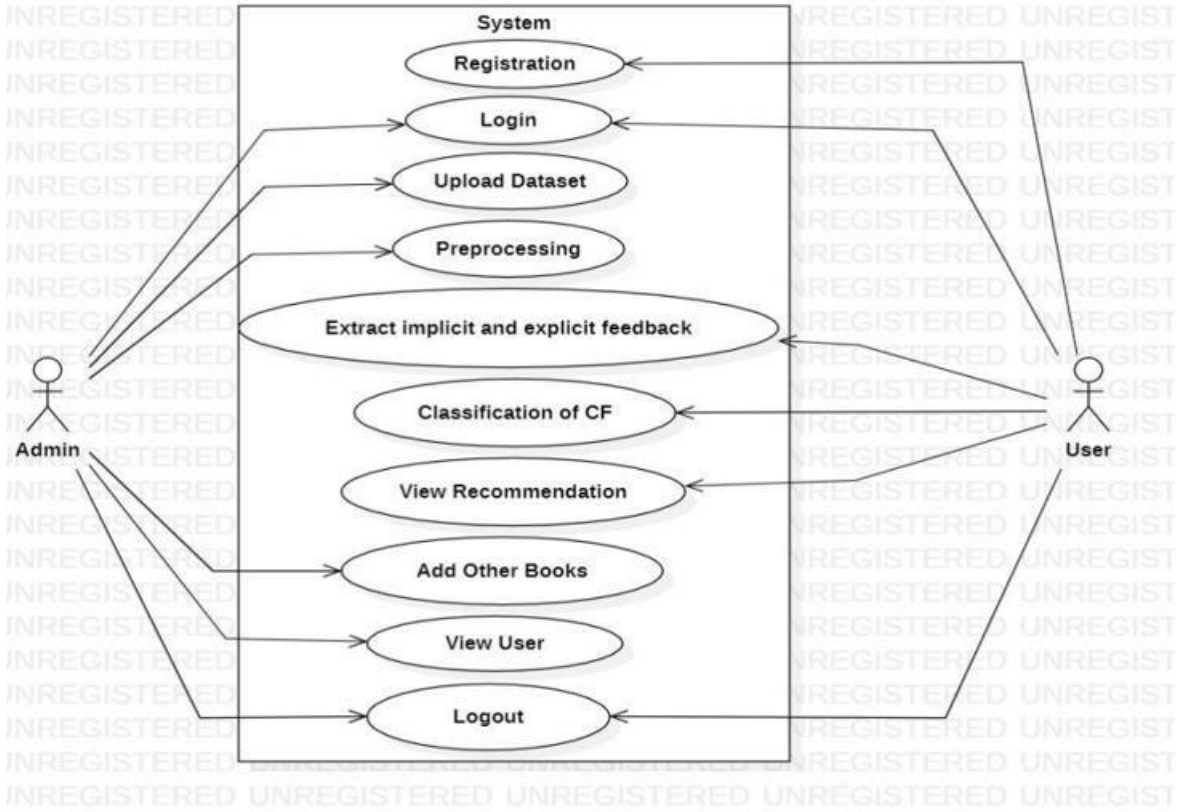
System Architecture:



Feedback Sources:

- Explicit Feedback: This includes data provided directly by users, such as reviews and ratings.
- Implicit Feedback: This refers to user behavior that indicates preferences, such as browsing history, clicks, or time spent on specific book pages.

Use-case Diagram:



Registration

- Both Admin and Users can register into the system to create their accounts.

Login

- Both Admin and Users log into the system using their credentials.

Upload Dataset (Admin)

- Admin can upload a dataset that contains books, user interactions, ratings, or other relevant data.

Preprocessing (Admin)

- The system preprocesses the uploaded dataset to clean and organize it, making it ready for analysis.

Extract Implicit and Explicit Feedback

- Implicit Feedback: Data inferred from user actions (e.g., clicks, time spent on pages).
- Explicit Feedback: Direct input from users, such as ratings and reviews.

- This feedback is extracted and prepared for collaborative filtering (CF).

Classification of CF (Collaborative Filtering)

- The system applies collaborative filtering techniques to classify user preferences and generate personalized recommendations.

View Recommendation

- Users can view the system's recommendations based on their preferences and interactions.

Add Other Books (Admin)

- Admin can add new books to the system to keep the dataset updated.

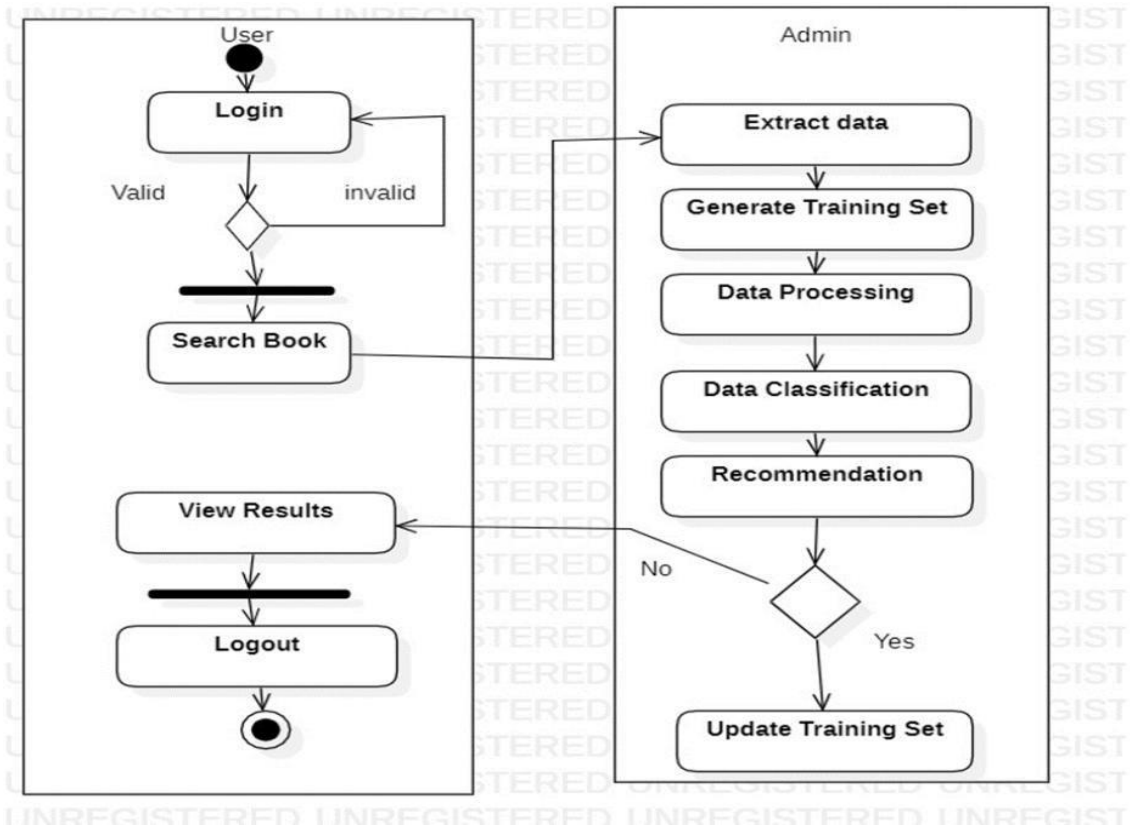
View User (Admin)

- Admin can view user details, interactions, and their preferences.

Logout

- Both Admin and Users can log out of the system when they finish their session.

Activity Diagram:



User Process:
Login: The user logs into the system. Valid credentials grant access; invalid ones prompt retry.
Search Book: The user searches for books using keywords or preferences.
View Results: The system displays recommendations or search results.
Logout: The user ends their session after viewing results.
Admin Process:
Extract Data: Collects user activity or book data.
Generate Training Set: Prepares data for the recommendation engine.
Data Processing: Cleans and organizes data for analysis.
Data Classification: Categorizes data to train the model (e.g., clustering books/users).
Recommendation: Generates book suggestions using the trained model.

Update Training Set: Refines the dataset if recommendations need improvement.

Project Specific Requirement:

Functional Requirements:
User Profiles: Allow users to register, log in, and save preferences.
Recommendations: Suggest books using collaborative (similar user behavior) methods.
Search & Browse: Let users search and explore books by categories like genre or author.
Ratings & Reviews: Enable users to rate, review, and see aggregated ratings.
Admin Tools: Let admins manage book data and moderate user activity.

Non-Functional Requirements
Performance: Fast loading recommendations and search results.
Scalability: Handle many users and large book databases.
Usability: Simple, mobile-friendly, and intuitive interface.
Security: Protect user data and ensure secure logins.

Technical Requirements
Database: Store books and user data.
Frontend: Build a responsive UI using Html and Css in Pycharm.
Algorithms: Use machine learning algorithm for recommendations.
Hosting: Deploy on cloud platforms like AWS or Google Cloud.

Frontend:
Develop the interface using HTML, CSS.

Backend:
Use a backend framework (Flask).

Implementation:

Book recommendation system:

- Take the dataset from kaggle.com
- Creating home screen using html in which top 50 popular books are shown.
- Creating of recommend page where in search bar if you enter a book title then it runs recommend function and then recomendes 4 similar books.
- App.py: This file is used to write Python scripts, functions, classes, and modules.You can use it to create standalone scripts or reusable components for larger projects.

Popularity code:

This function returns us the top popular fifty books.

Code:

```
ratings_with_name = ratings.merge(books,on='ISBN')
num_rating_df=ratings_with_name.groupby('Book-Title').count()['Book
Rating'].reset_index()
num_rating_df.rename(columns={'Book-Rating':'Num-Ratings'}, inplace=True)
ratings_with_name = ratings.merge(books,on='ISBN')
ratings_with_name['Book-Rating'] = pd.to_numeric(ratings_with_name['Book-
Rating'], errors='coerce')
ratings_with_name = ratings_with_name.dropna(subset=['Book-Rating'])
avg_rating_df = ratings_with_name.groupby('Book-Title')['Book-
Rating'].mean().reset_index()
avg_rating_df.rename(columns={'Book-Rating': 'avg_rating'}, inplace=True)
popular_df=num_rating_df.merge(avg_rating_df,on='Book-Title')
popular_df=popular_df[popular_df['NumRatings']>=250].sort_values(by='avg_rati
ng', ascending=False).head(50)
sorted_df=popular_df.merge(books,on='Book-Title').drop_duplicates('Book-
Title')[['Book-Title','Book-Author','Image-URL-M','Num-Ratings','avg_rating']]
```

User Similarity Calculation :

Compute user similarity using the Pearson correlation coefficient:

$$p(x, y) = \sum i(rxi - \bar{rx})(ryi - \bar{ry}) \sqrt{\sum (rxi - \bar{rx})2 \sum (ryi - \bar{ry})2i\ i}$$

Where: rxi : the rating of user x for item i.

$r\bar{x}$: the mean rating of user x.

ryi : the rating of user y for item i.

$r\bar{y}$: the mean rating of user y.

Nearest Neighbour Selection:

Identify k-nearest neighbours of each user based on computed similarity.

Determine the set Nu of k-users most similar to user u.

Code:

```
x = ratings_with_name.groupby('User-ID').count()['Book-Rating'] > 200
padhe_likhe_users = x[x].index
filtered_rating = ratings_with_name[ratings_with_name['User-
ID'].isin(padhe_likhe_users)]
y = filtered_rating.groupby('Book-Title').count()['Book-Rating']>=50
famous_books = y[y].index
final_ratings = filtered_rating[filtered_rating['Book-Title'].isin(famous_books)]
pt = final_ratings.pivot_table(index='Book-Title',columns='User-ID',values='Book-
Rating')
pt.fillna(0,inplace=True)
from sklearn.metrics.pairwise import cosine_similarity
similarity_scores = cosine_similarity(pt)
```

Recommend function:

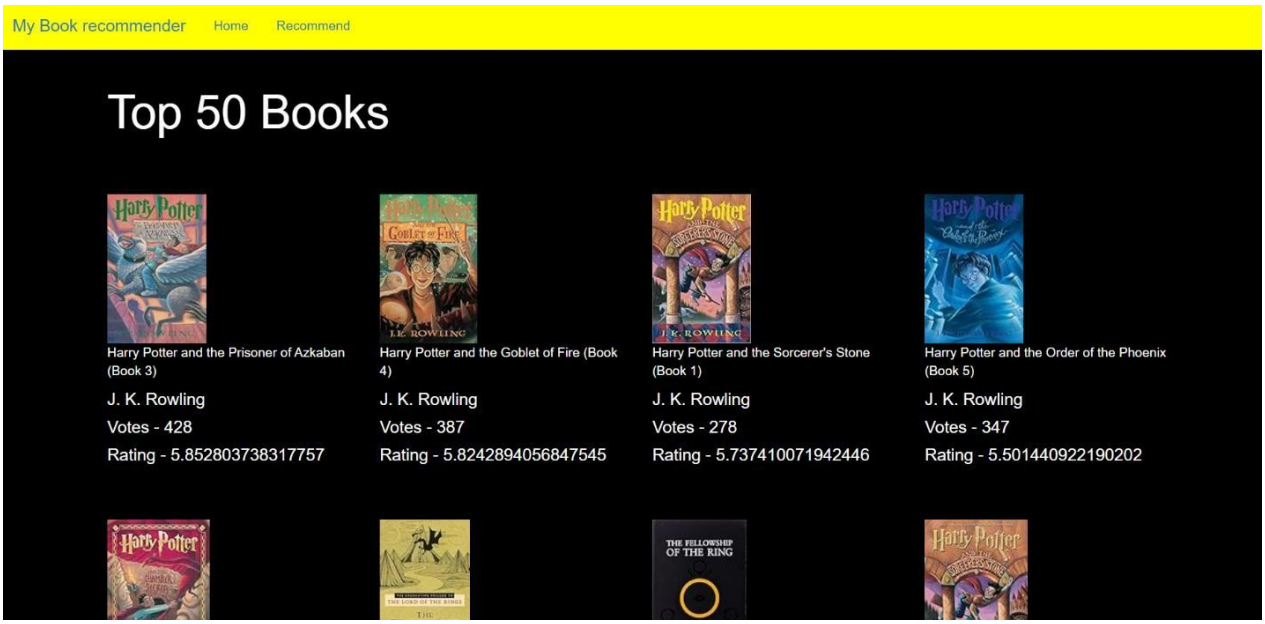
Takes the title of book as input and recommends 4 similar books.

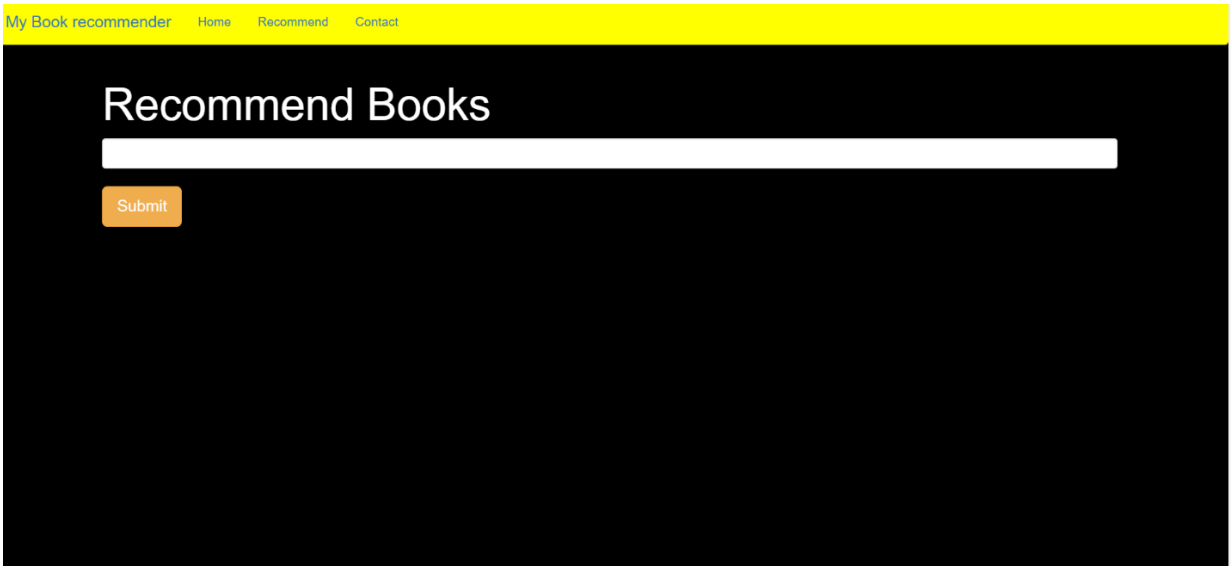
Code:

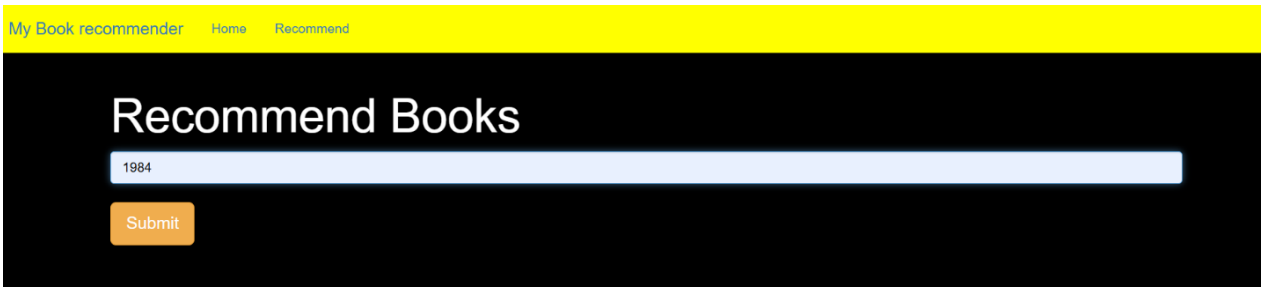
```
def recommend(book_name):
   # index fetch
   index = np.where(pt.index==book_name)[0][0]
   similar_items = sorted(list(enumerate(similarity_scores[index])),key=lambda
x:x[1],reverse=True)[1:5]
   data = []
   for i in similar_items:
```

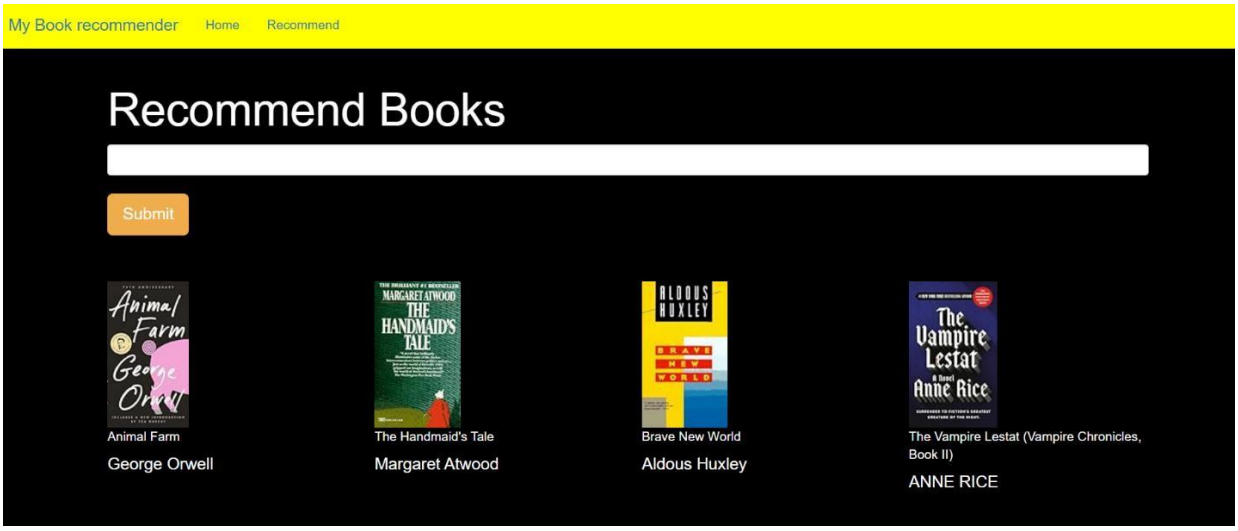
```
    item = []
    temp_df = books[books['Book-Title'] == pt.index[i[0]]]
    item.extend(list(temp_df.drop_duplicates('Book-Title')['Book-Title'].values))
    item.extend(list(temp_df.drop_duplicates('Book-Title')['Book-
    Author'].values))
    item.extend(list(temp_df.drop_duplicates('Book-Title')['Image-URL-
    M'].values))
    data.append(item)
return data
```

Results:


My Book recommender
Home
Recommend
Top 50 Books
Harry Potter and the Prisoner of Azkaban (Book 3)
J. K. Rowling
Votes - 428
Rating - 5.852803738317757
Harry Potter and the Goblet of Fire (Book 4)
J. K. Rowling
Votes - 387
Rating - 5.8242894056847545
Harry Potter and the Sorcerer's Stone (Book 1)
J. K. Rowling
Votes - 278
Rating - 5.737410071942446
Harry Potter and the Order of the Phoenix (Book 5)
J. K. Rowling
Votes - 347
Rating - 5.501440922190202


My Book recommender
Home
Recommend
Contact
Recommend Books
Submit


My Book recommender
Home
Recommend
Recommend Books
1984
Submit


My Book recommender
Home
Recommend
Recommend Books
Submit
Animal Farm
George Orwell
The Handmaid's Tale
Margaret Atwood
Brave New World
Aldous Huxley
The Vampire Lestat (Vampire Chronicles, Book II)
ANNE RICE

Conclusion and Future scope

Conclusion:
The book recommender system successfully fulfills the objective of Popularity filtering and collaborative filtering for book recommendation. It uses KNN (K-nearest-neighbour) Machine Learning Algorithm to recommend books. The project's emphasis on scalability and real-time performance.

Future Scope:
We will add content based filtering and personalized filtering to users who sign-in in the website and record their search history and recommend books to them based on their interaction and search history.

Reference:

- Book Recommendation System Based on Combine Features of Content Based Filtering, Collaborative Filtering and Association Rule Mining [3].
- Book Recommendation System for Digital Library Based on User Profiles by Using Association Rule [5].
- Machine Learning based Efficient Recommendation System for Book Selection using User based Collaborative Filtering Algorithm [1]