**Sagarmatha Colllege of Science and Technology**

**Tribhuvan University**

**Institute of Science and Technology**

****

**A Project Re**p**ort On**

**“Book recommendation using Popularity Based and Collaborative Filtering Algorithms”**

In Partial Fulfilment of Requirements for the Bachelor Degree in Computer Science and Information Technology

**Under the supervision of:**

Mrs. Ganga Subba

**Submitted to:**

Department of Computer Science and Information Technology

Sagarmatha College of Science and Technology  
Sanepa, Lalitpur, Nepal

**Submitted by:**

Paribesh Rimal (20854/075)  
Pranav Sapkota (20856/075)  
Bishal Basnet (20848/075)

**Date**

18/02/2080

**Sagarmatha College of Science and Technology**

**Tribhuvan University**

******Institute of Science and Technology**

**Supervisor’s Recommendation**

I hereby recommend that this project prepared under my supervision by **Paribesh Rimal, Pranav Sapkota and Bishal Basnet** entitled **“Book recommendation using Popularity Based and Collaborative Filtering Algorithms”** in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Information Technology is recommended for the final evaluation.

….………………

**Mrs. Ganga Subba**

Project Supervisor

Sagarmatha college of science and technology

Sanepa, Lalitpur

**Sagarmatha College of Science and Technology**

**Tribhuvan University**

**Institute of Science and Technology**

**CERTIFICATE OF APPROVAL**

The undersigned certify that they have read and recommended to the Department of Computer Science and Information Technology, IOST, Sagarmatha College of Science and Technology, a project report entitled **“Book recommendation using Popularity Based and Collaborative Filtering Algorithms”** submitted by **Paribesh Rimal, Pranav Sapkota and Bishal Basnet**. The Project was carried out under special supervision and within the time frame prescribed by the syllabus. We found the students to be hardworking, skilled and ready to undertake any related work to their field of study and hence we recommend the award of partial fulfilment of Bachelor’s degree of Computer Science and Information Technology.

|  |  |
| --- | --- |
| **Signature of the Supervisor** | **Signature of the HOD/Coordinator** |
| **Signature of the External Examiner** | **Signature of the Internal Examiner** |

# ACKNOWLEDGEMENT

The satisfaction of completion of this project work will be incomplete without mentioning the people responsible for the venture. We are thankful to our prestigious institution Sagarmatha College of Science and Technology which has provided us a congenial environment and surrounding to work in. We are grateful to our Head of Department Mr. Prateek Timalsina, Department of B.Sc.CSIT, for providing us with all elite facilities for academic purpose. We express our sincere thanks to mam, **Mrs. Ganga Subba,** her encouragement, invaluable technical support and assistance throughout the entire project have been of immense help in realizing this Project work. Her guidance gave us the environment to enhance our knowledge, skills and to reach the pinnacle with sheer determination, dedication and hard work. We extend our thanks to the entire faculty of the Department of B.Sc.CSIT, S.C.S.T, Sanepa, who have encouraged me throughout the course of entire presentation. We also extend my thanks to lab instructors, non-teaching staff, librarian, friends and family members for their continous support during the course.

**Project Members:**

Paribesh Rimal (20854/075)  
Pranav Sapkota (20856/075)  
Bishal Basnet (20848/075)

# ABSTRACT

Book Recommendation is a web-based application that provides information on various books. The aim of this project is to offer better suggestions for people interested in learning and reading different books. This project is a web-based project that anyone with internet access can use. It implements machine learning concepts that provide similar scores using machine learning algorithms. Machine learning is a growing scientific field in data science that deals with the ways in which machines learn from experience. It helps people find different books as well as ratings for the books. We have added a book feature where we can upload an image and add book details. Book Recommendation is a recommendation process where users are recommended similar books based on their interests. We can search for various books in the search bar, and similar books will be recommended to the user. In this project, popularity-based and rating-based collaborative filtering algorithms are presented. The main motto of Book Recommendation using Popularity-Based and Collaborative Filtering Algorithms is to guide different book enthusiasts to know about different, similar books. This application is developed using Jupiter, Visual Studio Code as IDE, Python as the main programming language, HTML, CSS, and JavaScript as front-end tools, Bootstrap as the framework, and Flask as the back-end tool.

**Keywords**: Book recommendation, bootstrap, collaborative filtering, flask, visual studio code

Table of Contents

[ACKNOWLEDGEMENT i](#_Toc137985029)

[ABSTRACT ii](#_Toc137985030)

[List of Figures v](#_Toc137985031)

[List of Abbreviation vi](#_Toc137985032)

[CHAPTER-1 1](#_Toc137985033)

[Introduction 1](#_Toc137985034)

[1.1 Introduction 1](#_Toc137985035)

[1.2 Problem Statement 1](#_Toc137985036)

[1.3 Objectives 2](#_Toc137985037)

[1.4 Scope 2](#_Toc137985038)

[1.5 Limitations 2](#_Toc137985039)

[1.6 Development Methodology 2](#_Toc137985040)

[1.7 Report Organization 3](#_Toc137985041)

[CHAPTER-2 5](#_Toc137985042)

[Background Study and Literature Review 5](#_Toc137985043)

[2.1 Background Study 5](#_Toc137985044)

[2.2 Literature Review 5](#_Toc137985045)

[CHAPTER-3 7](#_Toc137985046)

[System Analysis 7](#_Toc137985047)

[3.1 Requirement collection and Analysis 7](#_Toc137985048)

[3.1.1Functional Requirements 7](#_Toc137985049)

[3.1.2Non-Functional Requirements 8](#_Toc137985050)

[3.2 Feasibility Analysis 8](#_Toc137985051)

[3.2.1 Technical 8](#_Toc137985052)

[3.2.2 Operational 8](#_Toc137985053)

[3.2.3 Schedule 9](#_Toc137985054)

[3.3 Process Modelling 9](#_Toc137985055)

[CHAPTER-4 13](#_Toc137985056)

[System Design 13](#_Toc137985057)

[4.1.1 Data set collection 13](#_Toc137985058)

[4.1.2 Data summary 13](#_Toc137985059)

[4.1.3 Data tables 13](#_Toc137985060)

[4.2 Sequence Diagram 15](#_Toc137985061)

[4.3 Algorithms Used 15](#_Toc137985062)

[4.2.1 Popularity based algorithm 15](#_Toc137985063)

[4.2.2 Collaborative Filtering 19](#_Toc137985064)

[CHAPTER-5 24](#_Toc137985065)

[Implementation and Testing 24](#_Toc137985066)

[5.1 Implementation 24](#_Toc137985067)

[5.1.1 Tools Used 24](#_Toc137985068)

[5.1.2 Implementation Details 24](#_Toc137985069)

[Fetching data in project 26](#_Toc137985070)

[5.2 Testing 28](#_Toc137985071)

[5.2.1 Types of Testing 28](#_Toc137985072)

[5.2.1.1 Test Cases for Unit Testing 28](#_Toc137985073)

[5.2.1.2 Test Cases for System Testing 29](#_Toc137985074)

[CHAPTER-6 32](#_Toc137985075)

[Conclusion 32](#_Toc137985076)

[6.1 Conclusion 32](#_Toc137985077)

[6.2 Future Recommendations 32](#_Toc137985078)

[References 33](#_Toc137985079)

[APPENDIX 34](#_Toc137985080)

# List of Figures

Figure 1: Agile Methodology………………………………………………………………3

[Figure 1: Use Case Diagram](#_Toc133383738) 7

[Figure 2: Gantt Chart](file:///C:\Users\User\Downloads\Book%20Recommendation%20System2(2).docx#_Toc133383739) 9

[Figure 4: ER Diagram](#_Toc133383740) 10

[Figure 5: Context Diagram](#_Toc133383741) 10

[Figure 6: Level 1 DFD](#_Toc133383742) 11

Figure 7: Level 2 DFD……………………………………………………………………11

Figure 8: Sequence Diagram...……………………………………………………………14

[Figure 9: Example Table 1](#_Toc133383743)9

[Figure 10: Collaborative Filtering](#_Toc133383744) 20

[Figure 11: Conceptual Framework for filtering](#_Toc133383753) 26

# List of Tables

Table 1: Unit testing for searching books……………………………………………… 27

Table 2: Unit testing for Collaborative filtering recommendation…………………….. 27

Table 3: Unit testing for popularity based recommendation……………………………28

# List of Abbreviation

**Abbreviation Definition**

RS Recommended System

CF Collaborative Filtering

ISBN  International Standard Book Number

# CHAPTER-1

# Introduction

## 1.1 Introduction

A book recommendation system is a computer-based tool that suggests books to users based on their interests and preferences. To propose the user, it employs a variety of algorithms and strategies such as collaborative filtering and popularity-based algorithms that the user has already read or may enjoy. Because of the rise of e-books and online bookstores, book recommendation systems have grown in popularity in recent years. Online bookshops, libraries, and reading programs all use book recommendation systems. They have altered the way people discover and consume books, making it easier than ever to find and enjoy reading material. Data collection and filtering are the two essential components of a book recommendation system. The data gathering component gathers data about the user's tastes and activities. This information is used to make recommendations. The filtering component filters the data and provides customized recommendations using machine learning methods.

Book recommendation system is developed for book readers so that they could obtain various details about different books. It includes all the detailed information about different in this web application. People may come to our site and see various books and receive details they need. People can search for different books and read them. The system will provide top 20 books which has the highest ratings. On picking any books they can rate and find the similar books.

## 1.2 Problem Statement

The recommendation system faces the challenge of predicting how users will react to different content and suggesting the most suitable items for each individual.

Some other problem statements are:

* Can it recommend the books to the users on the basis of their historical behavior?
* Can it fix the problems of data sparsity, scalability and gray sheep?

Data sparsity refers to a situation where the data is thinly distributed, contains null and missing values. Scalability implies that making predictions becomes challenging when dealing with large volumes of rating items. Gray sheep pertains to issues related to time and memory requirements.

## 1.3 Objectives

This project helps to analyze the user dataset, books dataset and ratings data set. The objective of the project are:-

* To implement a machine learning algorithm that can suggest appropriate books to users by considering their interests and the popularity of the books.
* To help users find relevant and interesting content using collaborative filtering algorithm on category basis.

## 1.4 Scope

The recommendation techniques have been widely used in different areas and industries including social and online platforms. The book recommendation system provides engaging and relevant books to the users which helps the users to meet their interests.

Some of the scope of book recommendation system using popularity and collaborative filtering algorithm are:

* It can be used to provide the recommendation of new books according to the category.
* It can be used in recommendation of books according to their interests and preferences in the online or social media platform.

## 1.5 Limitations

The following are some of the drawbacks of the Book Recommendation System:-

* As the book’s detail grows in size, the load on processing and data maintenance activities grows as well.
* People who don’t have knowledge of internet can’t use this system as it requires internet to use this system.
* The system must retrain on every computation on rating on any books.
* The missing values such as rating, ISBN number will change the course of the system.

## 1.6 Development Methodology

Agile methodology is an approach to software development that emphasizes flexibility, collaboration, and customer satisfaction. In the context of developing a book recommendation system, an agile approach would involve breaking down the development process into small, manageable sprints where the team would work closely with stakeholders to gather requirements, prioritize features, and seek feedback. The iterative process allows the team to quickly respond to changing requirements and user needs, leading to a more successful project outcome. The team would focus on delivering functional product increments at the end of each sprint, allowing stakeholders to see progress and make changes as needed.

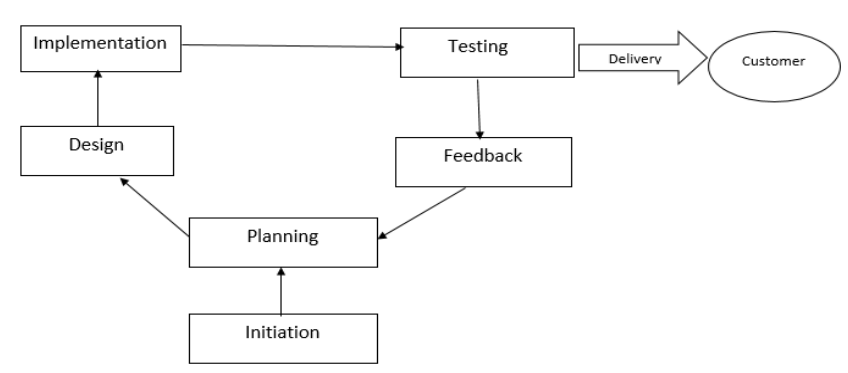
As shown in the picture, our adaptation followed the same flow. We divided our projects into smaller parts and requirements and worked on to it throughout the sprint. Our sprint lasted 3-4 weeks in average and 45 days at most.

Figure 1 : Agile methodology

## 1.7 Report Organization

The report has been prepared following the guidelines provided by Tribhuvan University. The report is separated into different chapters. Each chapter consists of various sub chapters with its content. The preliminary section of the report consists of Title Page, Acknowledgement, Abstract, Table of Contents, List of Figures, and List of Tables. The main report is divided into 6 chapters, which include:

1. Chapter 1: Introduction it includes the general overview of the system and the project as a whole. It includes the Problem Statement, Objectives, Scope/Limitations and the Development Methodology for the project and the system being developed.

2. Chapter 2: Background Study and Literature Review It includes the study of the current scenario/environment the system will be deployed into. It includes the study of the current trends, preferences of people, the existing systems, and areas of improvement among others.

3. Chapter 3: System Analysis It includes the requirement and feasibility analysis of the system that can be generated through the studies presented in the previous two chapters. It will also include the Flowchart, ER and DFD for the system which specifies the workflow, entities, attributes and their relationships.

4. Chapter 4: System Design It includes the design of the database, forms and interface of the system. It also includes the implementation details of the selected methodology and the details of the algorithm used.

5. Chapter 5: Implementation and Testing It includes the details of the different design and development tools used and the implementation details of the modules presented in the form of code snippets of functions, classes. It also includes the testing of the system with different test cases as per the requirement.

6. Chapter 6: Conclusion and Future Recommendations It includes the summary of the system and the project as a whole. It also includes the possibilities/aspects which the system can implement in the future.

The final part of the report consists of References and Appendices. The references are listed in accordance to the IEEE referencing standards and the Appendices includes the screenshots of the system and the major source code snippets.

# CHAPTER-2

# Background Study and Literature Review

## 2.1 Background Study

Book recommendation systems that use both popularity and collaborative filtering algorithms have become increasingly popular in recent years.

Popularity-based recommendation systems recommend items that are popular among users or have high ratings. Collaborative filtering, on the other hand, relies on the user's behavior and preferences to make recommendations. It suggests items that similar users have liked in the past.

To implement a book recommendation system using popularity and collaborative algorithms, we need to start with data preprocessing. This involves collecting and cleaning data about users and books, including user ratings, book metadata, and user profiles.

## 2.2 Literature Review

[1]Okon et.al. (2018) suggested an approach for providing customers with recommendations using an improved Collaborative Filtering algorithm, a quick sort algorithm, and Object-Oriented Analysis. The evaluation metrics were effectively carried out.

[2]Yongen et.al. (2018) put forward a customized technology that uses collaborative filtering for a university library's book recommendation system. The aim was to offer registered users a recommendation service.

[3]Ahmed et.al. (2018) suggested a Hybrid Recommendation system that helps users decide which book to purchase. The system computes the statistical correlation between the profiles of internet users by utilizing the Pearson Correlation Factor, taking into account the number of visits to various websites for each user. This aids in determining the type and degree of correlation among users.

[4]Jian et.al. (2020) proposed a Collaborative Filtering- based Recommendation System For Big Data. The Collaborative Filtering method was used in this system. This System was generated to supplement of the more data to study. Cold start problem and Data Sparsity were the most important problems in Collaborative Filtering based Recommendation System.

A review of the literature has indicated that a significant number of online marketers are utilizing recommendation systems to enhance their sales by presenting customers with products that align with their preferences.

However, these Recommendation Systems are beset by several issues such as data sparsity, cold start, trust, scalability, and privacy. Thus, there is a need for enhanced recommendation systems that address these concerns.

# CHAPTER-3

# System Analysis

## 3.1 Requirement collection and Analysis

The requirement collection and analysis is the important step in the development of the recommendation system. The result of the project is directly related to the collected requirement in the early days. Requirement collection involves the gathering of useful data and needs which helps in the project and requirement analysis involves the understanding of those collected data and check whether they meet the user expectations or not.

### 3.1.1Functional Requirements

The major functional requirements of this system are as follows:

* The top 20 books should be displayed on the home page on the basis of rating.
* The user could search the books according to their interest and needs.
* The user could add new books
* Books should be rated from 1-10.

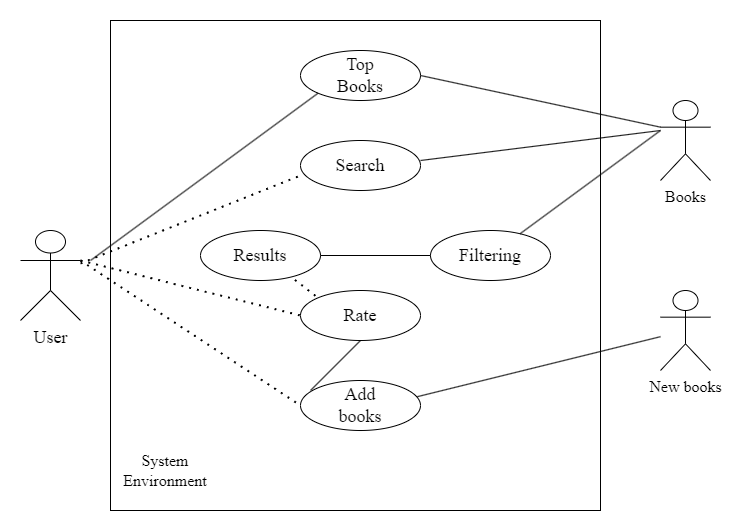


Figure 2: Use Case Diagram

### 3.1.2Non-Functional Requirements

Non-functional requirements of Book Recommendation System using popularity and collaborative filtering algorithm refers to the features and qualities that define the overall performance of the system.

Some Non-Functional Requirements of the project are listed below:-

* Performance: The recommendation in the system are provided fastly with minimum response time. The Recommendation System is able to handle maximum number of books and users.
* Accuracy: The Recommendation system recommend accurate books to the users according to their interests, needs and behavior. The accuracy of the system is optimized by using different techniques.
* Maintainability: The Recommendation System is easy to maintain as it is easy to use. It is easy to update, rate and add books in the system.

## 3.2 Feasibility Analysis

The Book recommendation system that uses popularity based algorithm and collaborative filtering algorithm can be feasible system to provide the conventional books to the users. The success of any recommendation system relies in the availability of efficient and effective data. The feasibility analysis can be done using various factors.

### 3.2.1 Technical

The system is technically feasible due to the data availability, scalability and algorithm suitability. Huge amount of data sets are available to perform the computation of the algorithms to provide meaningful recommendations. The suitable algorithms are selected.

### 3.2.2 Operational

The operational feasibility study focus on the recommendation system that if they are opearationally feasible or not. The system depend on minimizing the operational uneffectiveness and different challenges. The computation process is easier for the algorithm because it uses easier mathematical computation such as average and distance between two books.

### 3.2.3 Schedule

**Gantt chart**

**Book Recommendation System**

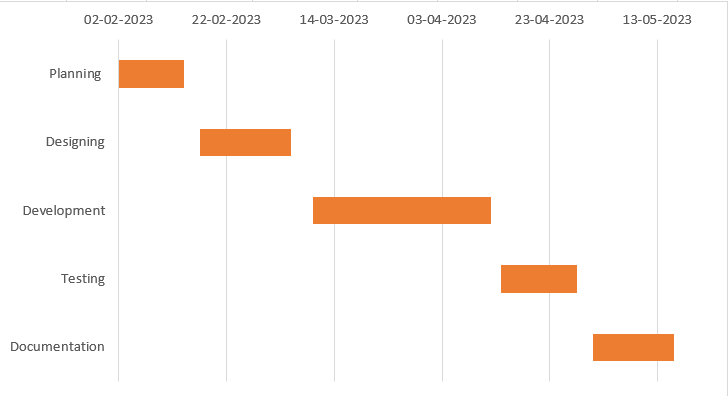


Figure 3: Gantt chart

## 3.3 Process Modelling

Process Modeling refers to the process that illustrates the operations of the system.

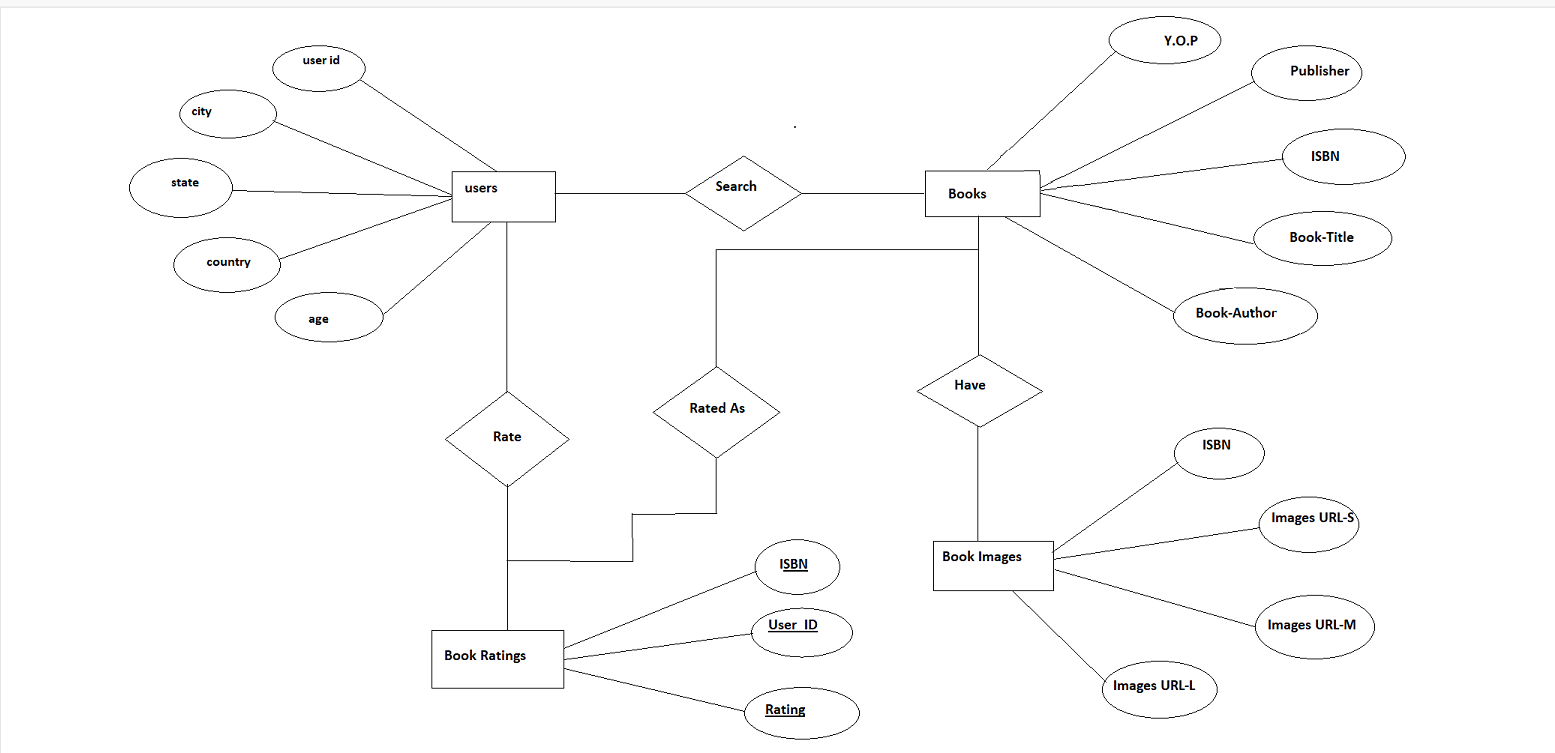
**ER Diagram**

Figure 4: ER Diagram

Figure 4: ER Diagram

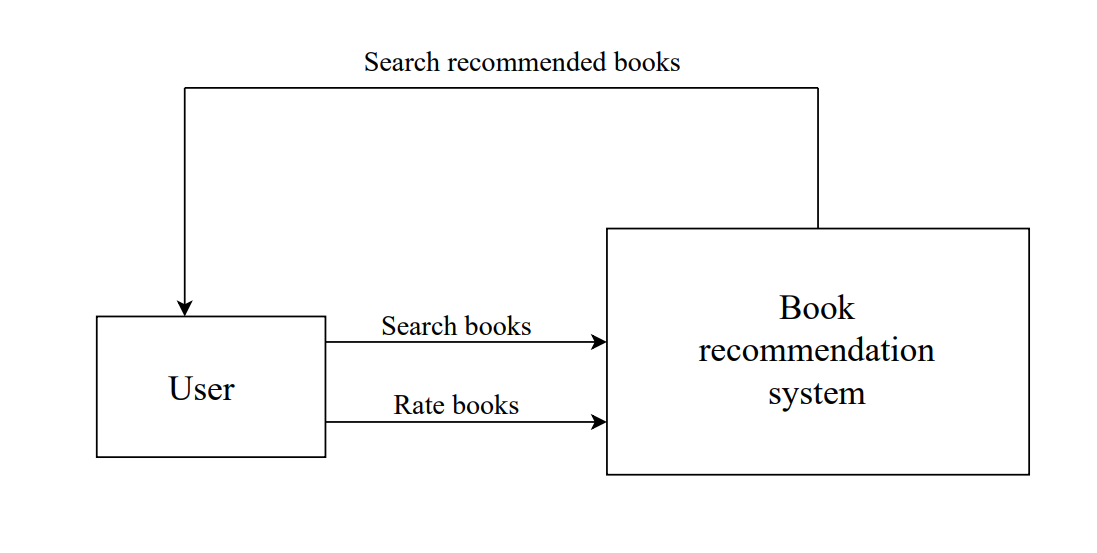
**DFD Diagram**

Figure 5: Context diagram

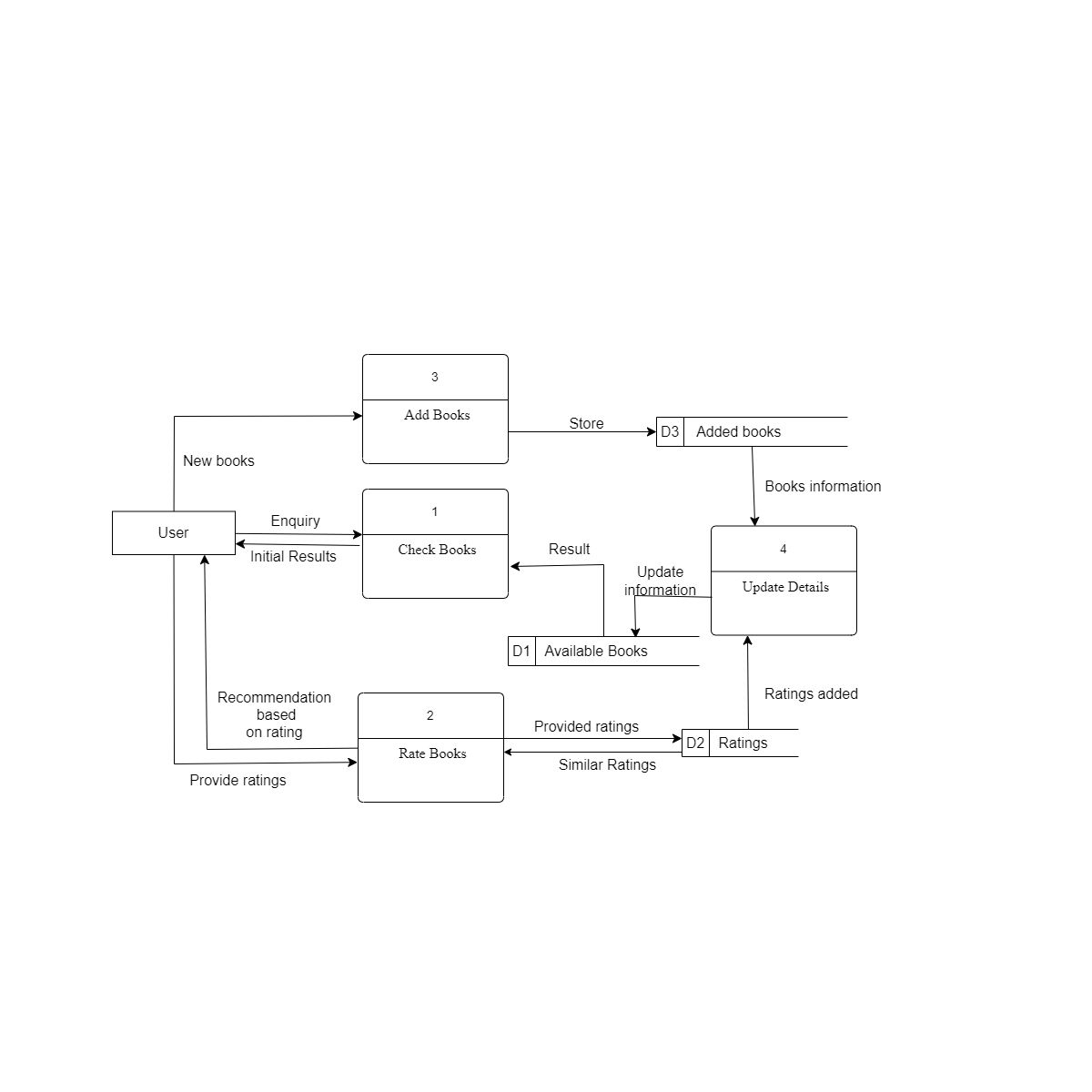


Figure 6: Level 1 DFD

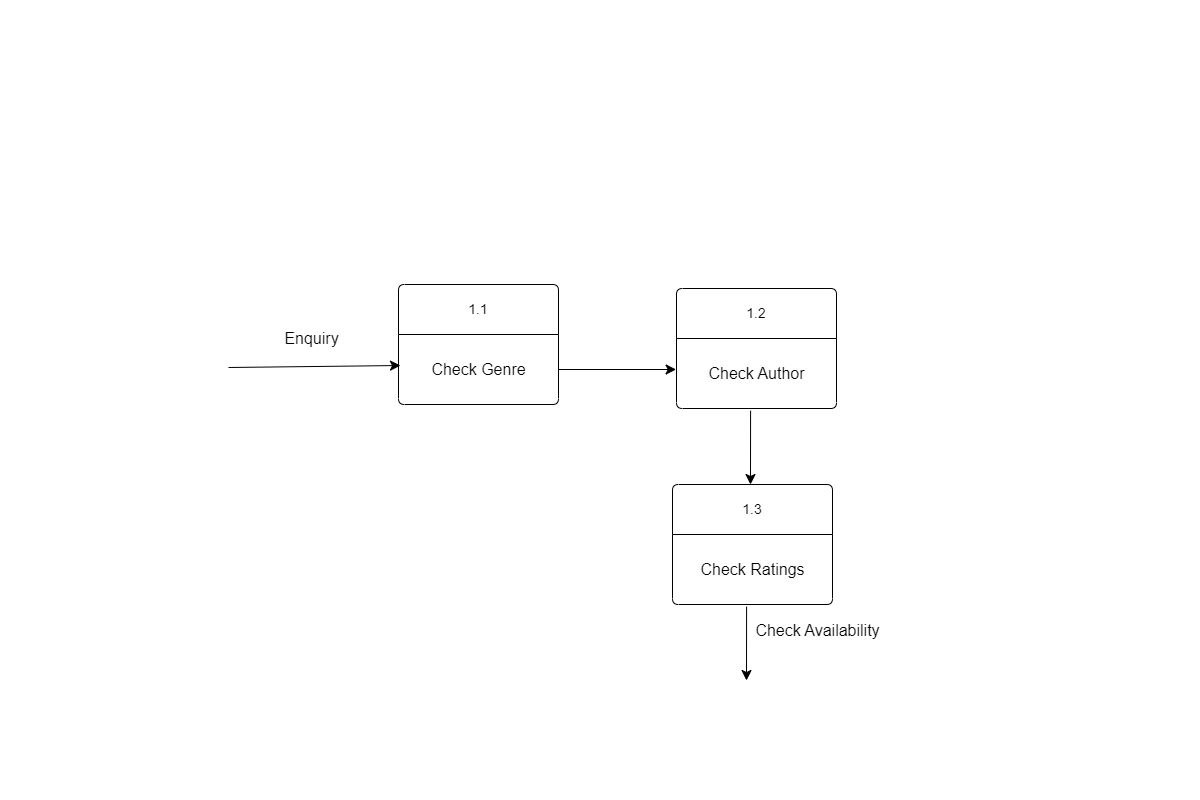


Figure 7: Level 2 DFD

# CHAPTER-4

# System Design

**4.1 Design**

## 4.1.1 Data set collection

Prior to creating a machine learning model, it is crucial to comprehend both the data and the objective. Exploring the data can uncover concealed patterns and valuable information, whereas preprocessing the data readies it for use by machine learning algorithms. The three dataset is used for our project named as “books.csv”, “Ratings.csv”, “user.csv”. The “Books.csv” contain more than 200000 unique values. There are more than 340000 unique values expressed on scale from 1-10(higher values denoting higher appreciation), or implicit, expressed by 0 “Rating.csv”. The “User.csv” contain more than 57000 unique value that has user id (User-ID). Demographic data is provided (Location, Age) if available. Otherwise, these Fields contain null values. The intended dataset, is a secondary dataset that is accessed via Kaggle. Based on the above data sets we recommend the different books on the basis of rating provided by the user.

## 4.1.2 Data summary

* **Users:** This csv file contains the user’s details such as location, age, null values.
* **Books:** Books are identified by their ISBN numbers, and erroneous ISBNs have been eliminated from the dataset. Additionally, certain content-based data, such as Book-Title, Book-Author, Year-Of-Publication, and Publisher, were acquired via Amazon Web Services. It should be noted that only the first author is included in cases where there are multiple authors. Moreover, URLs for cover images in three sizes (Image-URL-S, Image-URL-M, and Image-URL-L) - small, medium, and large - are also provided. These links direct to the Amazon website.
* **Ratings:** In this section, details pertaining to the book's rating are presented. Ratings (Book-Rating) can either be explicit or range from 1-10 (with higher scores indicating a greater level of approval), or implicit, which is represented as 0.

## 4.1.3 Data tables

1. Books.csv

|  |  |
| --- | --- |
| Field | Field type |
| ISBN | INT[15] |
| Book-Title | VARCHAR[50] |
| Book-author | VARCHAR[50] |
| Year of publish | INT[15] |
| Publisher | VARCHAR[50] |
| Book-Image | VARCHAR[50] |

1. User.csv

|  |  |
| --- | --- |
| Field | Field type |
| User\_id | INT[15] |
| Location | VARCHAR[50] |
| Age | INT[15] |

1. Rating.csv

|  |  |
| --- | --- |
| Field | Field type |
| User-ID | INT[15] |
| ISBN | INT[15] |
| Book-Rating | INT[15] |

1. New-book.csv

|  |  |
| --- | --- |
| Field | Field type |
| ISBN | INT[15] |
| Book-Title | VARCHAR[50] |
| Book-author | VARCHAR[50] |
| Book-Rating | INT[15] |
| Book-Image | VARCHAR[50] |

## 4.2 Sequence Diagram

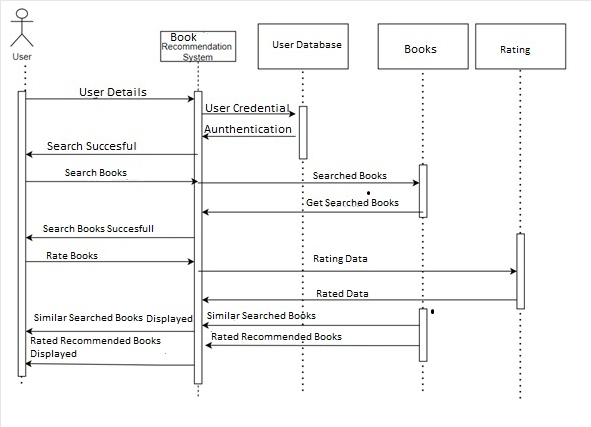


Figure 8: Sequence Diagram

## 4.3 Algorithms Used

1. Popularity based algorithm
2. Collaborative based filtering algorithms

### Popularity based algorithm

1. Read Data: The function reads two CSV files, "books.csv" and "ratings.csv", using pandas. These files contain information about books and user ratings, respectively.  
  
2. Merge Data: The ratings data is merged with the books data using the "ISBN" column as the common identifier. This combines the ratings information with corresponding book details.  
3. Calculate Number of Ratings: The function groups the merged data by book title and counts the number of ratings for each book. The result is stored in a new DataFrame called `num\_rating\_df`.  
  
4. Rename Columns: The column name "Book-Rating" in `num\_rating\_df` is renamed to "num\_ratings" to better represent the data.  
  
5. Calculate Average Rating: Similarly, the function calculates the average rating for each book by grouping the merged data by book title. The result is stored in a new DataFrame called `avg\_rating\_df`.  
  
6. Merge Number of Ratings and Average Rating: The `num\_rating\_df` and `avg\_rating\_df` DataFrames are merged based on the "Book-Title" column. This combines the number of ratings and average rating information into a single DataFrame called `popular\_df`.  
  
7. Filter Popular Books: The function filters the `popular\_df` DataFrame to include only books with at least 250 ratings. Then, it sorts the DataFrame in descending order based on the average rating and selects the top 20 books.  
  
8. Merge with Books Data: The `popular\_df` is merged with the original books data to retrieve additional book details like author, ISBN, and image URL. Duplicate entries based on book title are dropped to ensure uniqueness.  
  
9. Select Relevant Columns: From the merged DataFrame, only specific columns including ISBN, book title, author, image URL, number of ratings, and average rating are selected.  
  
10. Calculate Average Score: The function calculates the average of the "avg\_rating" column in the `popular\_df` DataFrame, representing the average rating score among the top 20 popular books.  
  
11. Print Average Score: The average rating score is printed to the console.  
  
12. Return DataFrame: The function returns the `popular\_df` DataFrame, which contains the information about the top 20 popular books.  
  
This function essentially trains a popularity-based model for book recommendations by considering the number of ratings and average rating scores of books. It retrieves the most popular books based on these criteria and provides relevant details for further analysis or display.

What is the significance of this model?

The Cold-Start Problem is a challenge that arises in computer-based information systems that use automated data modeling. This problem occurs when the system lacks sufficient information to make recommendations for users or items. It is a well-known issue in recommender systems, which are used to filter and present items (such as products, movies, music, books, and news) that are likely to be of interest to the user. These systems compare the user's profile to reference characteristics, such as the content of the items (content-based filtering) or the user's past behavior and social environment (collaborative filtering). Users can interact with the system in various ways, such as by providing ratings, bookmarks, purchases, likes, or the number of page visits, depending on the system.

The Cold-Start Problem can be categorized into three different situations:

•         The first situation is when a recommender system is new, and there are very few users who have provided their interaction data or preferences. In this case, the system struggles to give reliable recommendations because of the lack of data.

•         The second situation occurs when a new item is added to the system, and there is no interaction data available for it. Although the system may have some content information about the item, it becomes challenging for the system to offer accurate recommendations for that particular item.

•         The third situation arises when a new user registers, and they have not yet provided any interaction data. As a result, the system cannot provide personalized recommendations tailored to their preferences until they provide sufficient interaction data.

To mitigate the cold start problem in a book recommendation system, you can implement various strategies to provide accurate and personalized recommendations, even for new users and items. Here are some effective approaches:  
  
1. Content-based recommendations: Utilize the content of books, such as genre, author, publication year, synopsis, or other descriptive attributes, to make recommendations. By analyzing the content and matching it with users' preferences, you can suggest books that are likely to align with their interests.  
  
2. Knowledge-based recommendations: Ask users to provide explicit information about their reading preferences during the onboarding process. This can include favorite genres, authors, or specific books they enjoy. Leverage this information to make initial recommendations for the user.  
  
3. Demographic-based recommendations: Consider using demographic information, such as age, gender, location, or language preferences, to make relevant recommendations. Analyze patterns and preferences among users with similar demographic profiles and suggest books that are popular within those groups.  
  
4. Hybrid approaches: Combine multiple recommendation techniques to mitigate the cold start problem. Utilize a mix of content-based, collaborative filtering, demographic-based, or knowledge-based methods to provide recommendations. As the system gathers more user data, gradually transition to more personalized approaches.  
  
5. Popularity-based recommendations: When dealing with new items or users, prioritize recommending popular books that have received positive reviews and ratings. This approach ensures that users are exposed to widely accepted books while collecting data on their preferences for future recommendations.

### 4.2.2 Collaborative Filtering

1. Read Data: The function reads three CSV files, "users.csv", "books.csv", and "ratings.csv", using pandas. These files contain information about users, books, and user ratings, respectively.  
  
2. Merge Data: The ratings data is merged with the books data using the "ISBN" column as the common identifier. This combines the ratings information with corresponding book details.  
  
3. Filter Experienced Users: The function filters out users who have rated more than 200 books. This is determined by grouping the merged data by "User-ID" and counting the number of ratings for each user. The resulting `experiencedUsers` contains the indices of experienced users.  
  
4. Filter Ratings: The function selects only the ratings from the filtered experienced users by checking if the "User-ID" is in the `experiencedUsers` list. This reduces the dataset to include ratings from experienced users only.  
  
5. Filter Famous Books: The function filters out books that have received at least 50 ratings. This is determined by grouping the filtered ratings data by "Book-Title" and counting the number of ratings for each book. The resulting `famous\_books` contains the indices of famous books.  
  
6. Filter Final Ratings: The function selects only the ratings from the filtered famous books by checking if the "Book-Title" is in the `famous\_books` list. This reduces the dataset to include ratings for famous books from experienced users.  
  
7. Create Pivot Table: The function creates a pivot table called `pt` using the final ratings data. The pivot table has "Book-Title" as the index, "User-ID" as the columns, and "Book-Rating" as the values. This table provides a matrix-like representation of user ratings for each book.  
  
8. Fill NaN Values: Any missing values in the pivot table, representing books that certain users haven't rated, are filled with 0 using the `fillna()` method.  
  
9. Calculate Similarity Scores: The function calculates similarity scores between books using the cosine similarity measure. The `cosine\_similarity\_matrix()` function (not shown in the code) is used to generate a matrix of similarity scores based on the pivot table.  
  
10. Return Results: The function returns the similarity scores, the original books DataFrame, the pivot table (`pt`), and the ratings DataFrame.  
The `trainCollaborativeFiltering()` function prepares the data for collaborative filtering by filtering out less experienced users and less popular books. It then generates a pivot table and calculates similarity scores between books using cosine similarity. The resulting data can be used for collaborative filtering-based book recommendations.A collaborative filtering matrix is a matrix of users and items, where the cells of the matrix contain ratings that users have given to items. The matrix is often sparse, meaning that most cells are empty, since most users have not rated most items. For example, a simple matrix might look like this:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Book 1** | **Book 2** | **Book 3** | **Book 4** |
| User 1 | 3 | 5 |  |  |
| User 2 |  | 2 | 4 |  |
| User 3 | 5 |  |  | 3 |
| User 4 |  | 1 | 2 | 4 |

Figure 9: Example Table

In this matrix, User 1 has rated Book 1 and 2, User 2 has rated Book 2 and 3, User 3 has rated Book 1 and 4, and User 4 has rated Book 2, 3 and 4. The empty cells represent book that have not been rated by a user.

We can use this matrix as a starting point to calculate similarities between users or books and make recommendations based on those similarities.

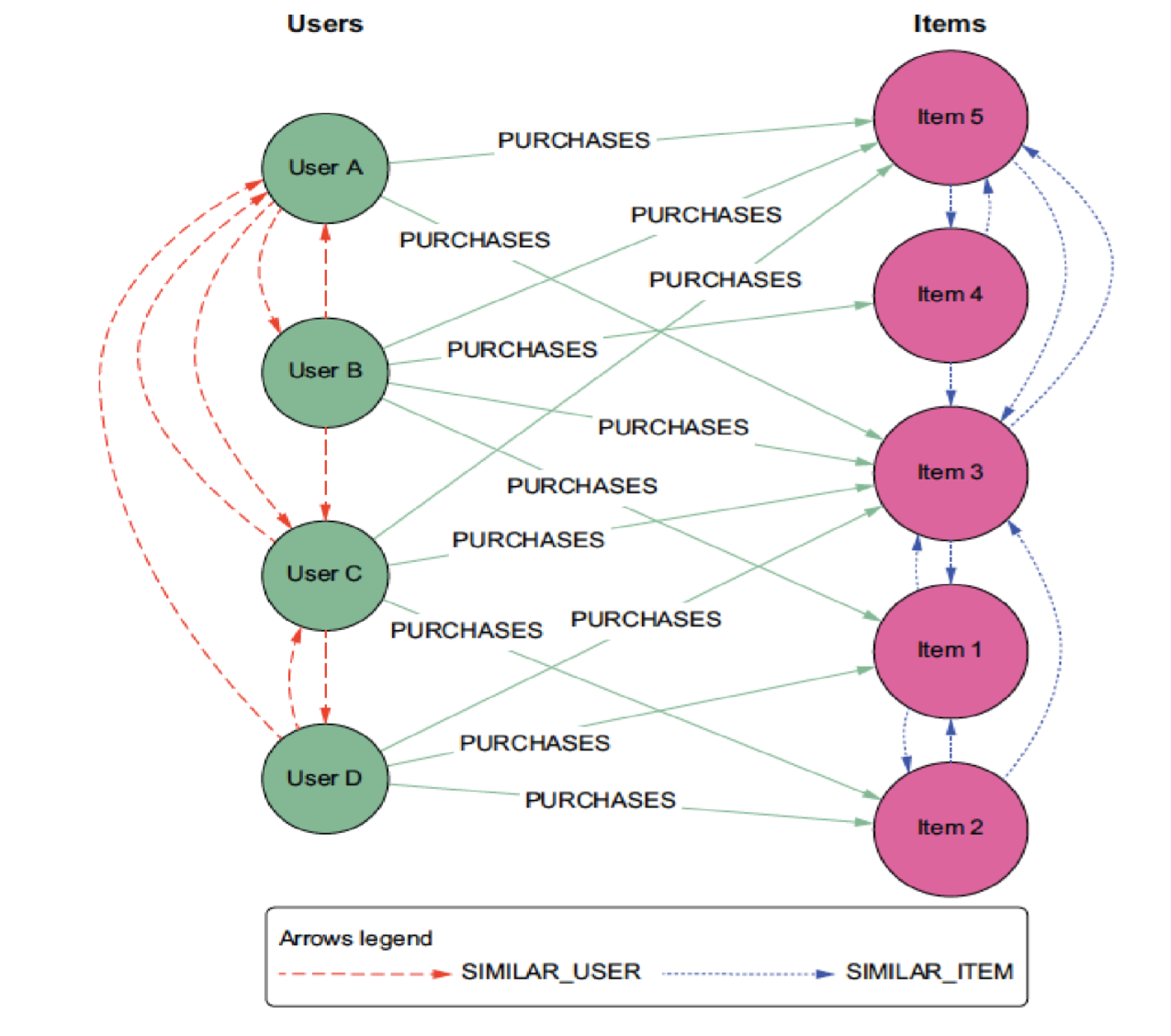


Figure 10: Collaborative Filtering

**4.2.3 Working Principle**

**Cosine similarity** is a measure of similarity between two non-zero vectors, which measures the cosine of the angle between them. It is commonly used in information retrieval and text mining to calculate the similarity between two documents or to find similar documents.

The cosine similarity between two vectors is computed as the dot product of the vectors divided by the product of their magnitudes or Euclidean norms. Mathematically, the cosine similarity between vectors x and y is defined as:

Cos(x, y) = x. y / ||x|| \* ||y||  
where,

* **x. y** = product (dot) of the vectors ‘x’ and ‘y’.
* **||x||**and**||y||** = length of the two vectors ‘x’ and ‘y’.
* **||x|| \* ||y||** = cross product of the two vectors ‘x’ and ‘y’.

A **pivot table** is a data structure that organizes user ratings for different books. The pivot table has books as rows and users as columns, allowing for a compact and structured representation of the user-book interactions. The working principle of a pivot table in a book recommendation system where rows represent books and columns represent users can be explained as follows:  
  
1. Data Organization: The pivot table takes the user ratings data and arranges it in a tabular format, with each row representing a book and each column representing a user. This organization provides a convenient way to summarize and analyze the user ratings for different books.  
  
2. Aggregation of Ratings: The pivot table aggregates the user ratings, placing each rating in the corresponding intersection of the book and user. This allows for easy retrieval and analysis of the ratings for individual books by different users.  
  
3. Handling Missing Values: In the pivot table, if a user has not rated a particular book, the corresponding cell in the table will typically contain a missing value or NaN (Not a Number). This representation handles situations where users have not provided ratings for certain books, ensuring that the table is sparse and can accommodate varying levels of user engagement.  
  
4. Efficient Retrieval: The pivot table enables efficient retrieval of ratings for specific books or users. Given a book, it is easy to retrieve the ratings provided by different users by accessing the corresponding row in the pivot table. Similarly, given a user, it is straightforward to retrieve the ratings they have provided for different books by accessing the relevant column in the table.  
  
5. Analysis and Recommendation: The pivot table serves as a foundation for various analysis and recommendation techniques. For example, collaborative filtering algorithms can use the pivot table to identify similar users or similar books based on their rating patterns. It can also be used to calculate similarity scores between books, generate personalized recommendations, or identify popular books based on user ratings.  
  
Overall, the pivot table in a book recommendation system provides a structured representation of user-book interactions, facilitating efficient analysis, and enabling various recommendation techniques to be applied based on user preferences and behavior.

# CHAPTER-5

# Implementation and Testing

## 5.1 Implementation

The actual coding of the project starts from this stage. Division of works are done and scheduled. The implementation phase is performed according to the collected requirements and need of the system. This phase is regularly monitored for the better user satisfaction. The data are understood thoroughly along with the algorithms used in the system.

## 5.1.1 Tools Used

For the development of this system various software tools were used but no any additional hardware tools were used in the developing process. Software tools were used in both front-end and back-end:

Flask: It was used to develop web applications.

Python: It was used to code the recommendation logic using python.

Draw.io: It was used to develop DFDs, Sequence diagram, Use case diagram.

Pandas: It was used to provide the data structure for data set

CSV: It was used to store data into file and retrieve those data.

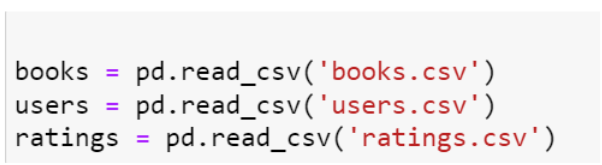
Numpy: Numpy is used to manipulate the data of array.

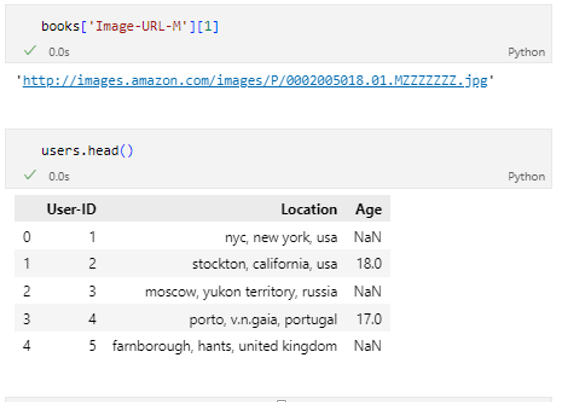
## 5.1.2 Implementation Details

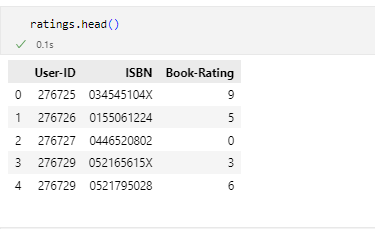
 The implementation steps for a book recommendation system using the cosine similarity algorithm and a pivot table:  
  
1. Load Data: Load the book ratings data into a DataFrame. The data should include information such as user IDs, book titles, and corresponding ratings. This data will be used to create a pivot table.  
  
2. Create Pivot Table: Create a pivot table from the ratings data. Set the book titles as the rows and the user IDs as the columns. The pivot table will serve as a matrix-like representation of user ratings for each book.  
  
3. Calculate Cosine Similarity: Use the cosine similarity algorithm to calculate the similarity scores between books. This involves comparing the rating vectors of each pair of books using cosine similarity, which measures the similarity between two vectors.  
  
4. Recommend Books for a User: To recommend books for a specific user, select the ratings of that user from the pivot table. Calculate the cosine similarity between the user's ratings and all other books in the pivot table. This will result in a similarity score for each book.  
  
5. Sort and Filter Recommendations: Sort the books based on their similarity scores in descending order. Exclude books that the user has already rated to avoid recommending books they have already read. Optionally, you can also consider additional filters, such as excluding books with low ratings or a small number of ratings.  
  
6. Return Top-N Recommendations: Select the top-N books with the highest similarity scores as the recommended books. The value of N can be determined based on the desired number of recommendations to provide to the user.  
  
7. Present Recommendations: Present the recommended books to the user through the application interface, website, or any other suitable medium. Include relevant information such as book titles, authors, cover images, and any additional details that can help the user make a decision.  
  
By following these steps, the book recommendation system utilizes the cosine similarity algorithm and a pivot table to calculate similarity scores between books and provide personalized recommendations to users based on their ratings.

## Fetching data in project

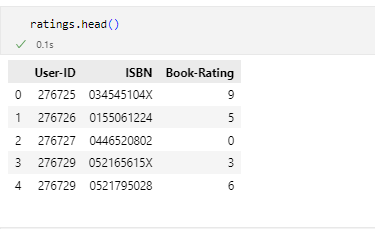
Constructing the csv file into a data frame.

**Loading users.csv**



**Loading books.csv**

**Loading rating.csv**



## 

**Conceptual Framework for filtering:**



Figure 11: Conceptual Framework for filtering

This is a sophisticated type of recommender system that utilizes the similarity between different users and items to make recommendations. It is widely used in e-commerce and online movie platforms, where it analyzes the preferences of similar users to create personalized suggestions.

The system's definition of similarity extends beyond just user preferences and incorporates the similarity between different items as well. When there is a large volume of data available on both users and items, the system can generate more precise recommendations.

## 5.2 Testing

Testing is the process of examining a software program or system to find flaws or defects and to ensure that it fulfills the requirements and quality standards. It entails running a software or system in order to detect bugs, faults, or other problems and ensuring that it operates as planned.

Functional testing, non-functional testing, manual testing, and automated testing are all examples of testing. Non-functional testing assesses performance, security, usability, and other quality factors, whereas functional testing evaluates the functionality of the software system. Manual testing requires human interaction to test the system, whereas automated testing uses tools and software to test the system.

Testing is an important aspect of the software development life cycle (SDLC) since it aids in the early detection of flaws and mistakes and guarantees that the program fulfills the user's needs and expectations. Testing improves the quality and dependability of software while lowering the risk of failure and downtime.

## 5.2.1 Types of Testing

### 5.2.1.1 Test Cases for Unit Testing

Table 1: Unit testing for searching books

|  |  |  |  |
| --- | --- | --- | --- |
| S.N | Test Case | Outcome | Remarks |
| 1 | Valid keyword | Display top 50 books | Appendix 1.4 |
| 2 | Invalid keyword | Books are not displayed | Appendix 1.5 |
| 3 | Case sensitive keyword | Displays top 50 books | Appendix 1.4 |

Table 2: Unit testing for Collaborative filtering recommendation

|  |  |  |  |
| --- | --- | --- | --- |
| S.N | Test Case | Outcome | Remarks |
| 1 | Personalized recommendation is requested | Displays top 4 books | Appendix 1.3 |

Table 3: Unit testing for popularity based recommendation

|  |  |  |  |
| --- | --- | --- | --- |
| S.N | Test Case | Outcome | Remarks |
| 1 | If the condition is satisfied | Displays top 20 books | Appendix 1.1 |

## 5.2.1.2 Test Cases for System Testing

Test Scenario: User interacts with the book recommendation system and receives recommendations  
**Test Steps:**  
1. Launch the book recommendation system.  
2. Verify that the homepage is displayed with a list of popular books.  
3. Click on a book from the popular books list.  
4. Verify that the book details page is displayed with accurate information about the selected book.  
5. Scroll down to the recommended books section on the book details page.  
6. Verify that a list of recommended books is displayed based on collaborative filtering algorithm.  
7. Click on one of the recommended books from the list.  
8. Verify that the book details page for the recommended book is displayed.  
9. Return to the homepage.  
10. Enter a search query for a specific book in the search bar.  
11. Click on the "Search" button.  
12. Verify that the search results page is displayed with relevant books matching the search query.  
13. Click on one of the search results from the list.  
14. Verify that the book details page for the selected search result is displayed.  
15. Scroll down to the recommended books section on the book details page.  
16. Verify that a list of recommended books is displayed based on collaborative filtering algorithm.  
17. Click on one of the recommended books from the list.  
18. Verify that the book details page for the recommended book is displayed.  
19. Repeat steps 10-18 with different search queries and recommended books.  
20. Explore other functionalities of the book recommendation system, such as adding a book and rating a book.  
21. Verify that these functionalities work as expected and the system responds correctly.  
  
**Expected Results:**   
- The homepage should display a list of popular books.  
- Clicking on a book should navigate to the corresponding book details page.  
- The book details page should display accurate information about the selected book.  
- The recommended books section on the book details page should present relevant recommendations based on collaborative filtering algorithm.  
- Clicking on a recommended book should navigate to the corresponding book details page.  
- The search results page should display relevant books matching the search query.  
- Clicking on a search result should navigate to the corresponding book details page.  
- The recommended books section on the book details page of search results should present relevant recommendations based on collaborative filtering algorithm.  
- The book recommendation system functionalities, such as adding a book and rating a book, should work correctly and provide the expected responses.

**5.2 Result Analysis**

When analyzing the results of popularity based and collaborative filtering using the cosine algorithm in a book recommendation system, several factors should be considered. Here are some key aspects to examine:  
  
1. Accuracy of Recommendations: Evaluate the accuracy of the recommendations provided by the collaborative filtering algorithm. Measure how well the system predicts user preferences and matches users with books they enjoy. Metrics such as precision, recall, and mean average precision (MAP) can be used to assess the algorithm's performance.  
  
2. Coverage: Assess the coverage of the collaborative filtering algorithm, which refers to the percentage of available books that the algorithm can recommend. A higher coverage indicates a wider range of recommendations and better utilization of the book database.  
  
3. Diversity of Recommendations: Consider the diversity of recommendations produced by the collaborative filtering algorithm. It is essential to ensure that the algorithm suggests a variety of books from different genres, authors, or categories to cater to users' diverse interests.  
  
4. Novelty: Evaluate the novelty of the recommended books. The algorithm should be able to suggest books that users may not have previously encountered, introducing them to new and exciting reading options.  
  
5. Serendipity: Serendipity refers to the ability of the algorithm to recommend unexpected but enjoyable books to users. Assess whether the collaborative filtering algorithm can surprise users with recommendations they wouldn't have discovered on their own.  
  
6. Cold Start Problem: Analyze how well the collaborative filtering algorithm handles the cold start problem, where there is limited or no user data available for new users or newly added books. Determine whether the algorithm can provide relevant recommendations based on available data or alternative approaches, such as content-based recommendations, are necessary for these cases.  
  
7. User Satisfaction: Gather user feedback and conduct surveys or interviews to gauge user satisfaction with the collaborative filtering recommendations. Understand if users find the recommendations helpful, accurate, and aligned with their preferences.  
  
8. Performance: Measure the performance of the collaborative filtering algorithm in terms of computational efficiency and scalability. Consider factors such as response time, memory usage, and the ability to handle large datasets to ensure the algorithm can handle real-time recommendation requests efficiently.  
  
By conducting a comprehensive analysis of these factors, we can assess the effectiveness and performance of the collaborative filtering using cosine algorithm in the book recommendation system.

# CHAPTER-6

# Conclusion

## 6.1 Conclusion

The majority of readers were between the ages of 20 and 35, and the majority of them were from North American and European nations, specifically the United States, Canada, the United Kingdom, Germany, and Spain.

In the rated books, we observed that the majority of the books received positive ratings, with the highest-rated novels receiving an 8. Ratings below 5 were infrequent. However, there were also books in the dataset that were not rated.

Agatha Christie, William Shakespeare, and Stephen King have the most novels published.

The purposed book recommendation system provides personalized book recommendation to user by utilizing a popularity based and collaborative filtering approaches. By using machine learning the system is able to recommend books that are tailored to specific interest and preference of each user. This system is able to provide the better recommendation to the users without any delaying.

## 6.2 Future Recommendations

Different ideas can be added on this recommendation system in the future.

Some of the recommendations for the future are listed below:

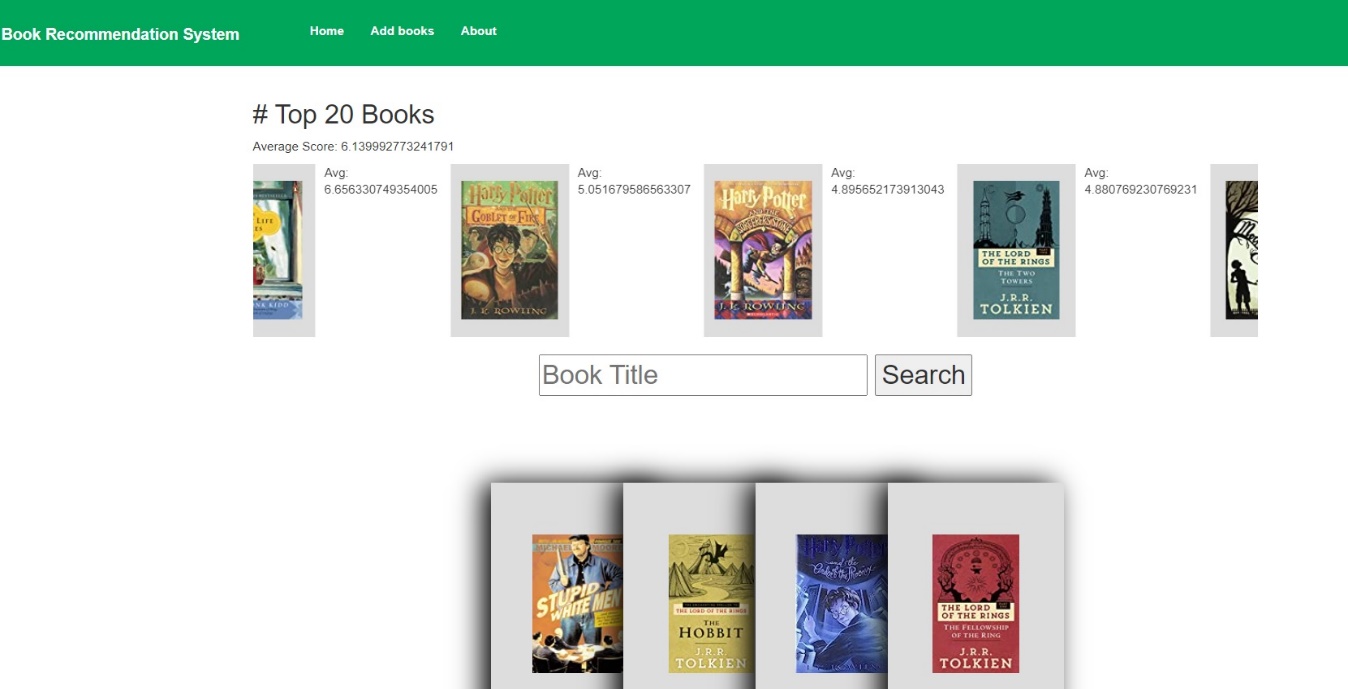
* The database can be integrated.
* The search history can be stored and can be used for the recommendation using different algorithm.
* Hybrid algorithm can be implemented.

# References

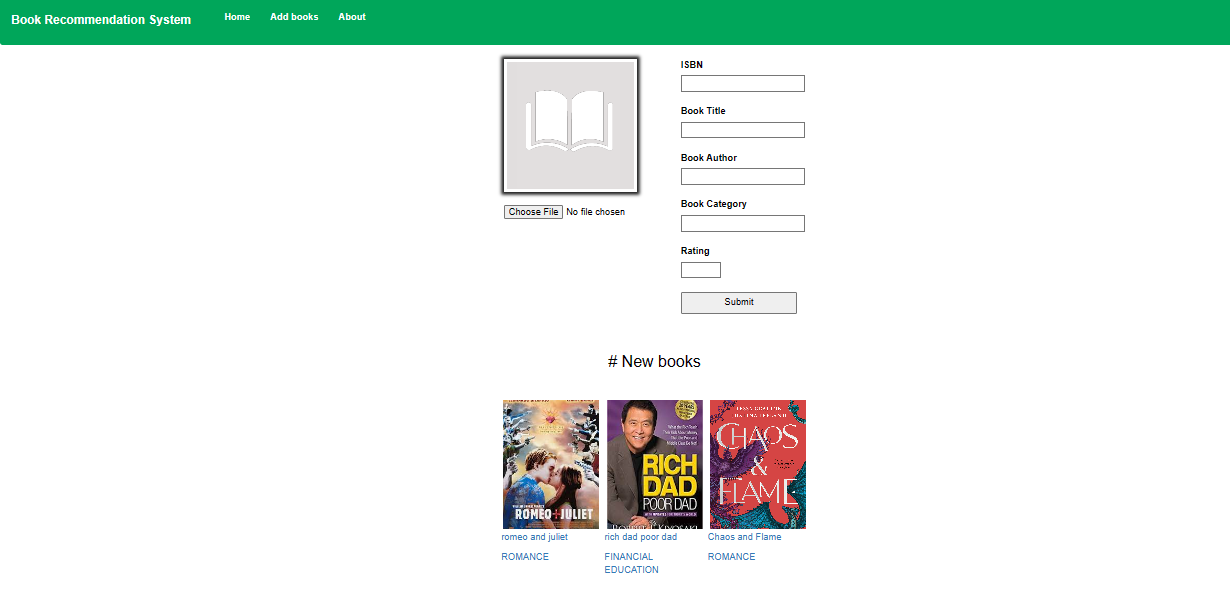
|  |  |
| --- | --- |
| [1] | E. E. B. a. A. P. Okon, "International Journal of Computer Applications(0975-8887)," Vols. Volume 179-No.46, June 2018. |
| [2] | Y. L. A. Shimingwan, "The Design And Implementation Of Books Recommendation System using Collaborative Filtering.," 2018. |
| [3] | Ahmed.M.Omran, "A Novel Recommender System for websites using Hybrid Method," 2018. |
| [4] | T. Z. a. L. C. Jian Shen, "Collaborative filtering-based recommendation system for Big data," 2020. |

# APPENDIX

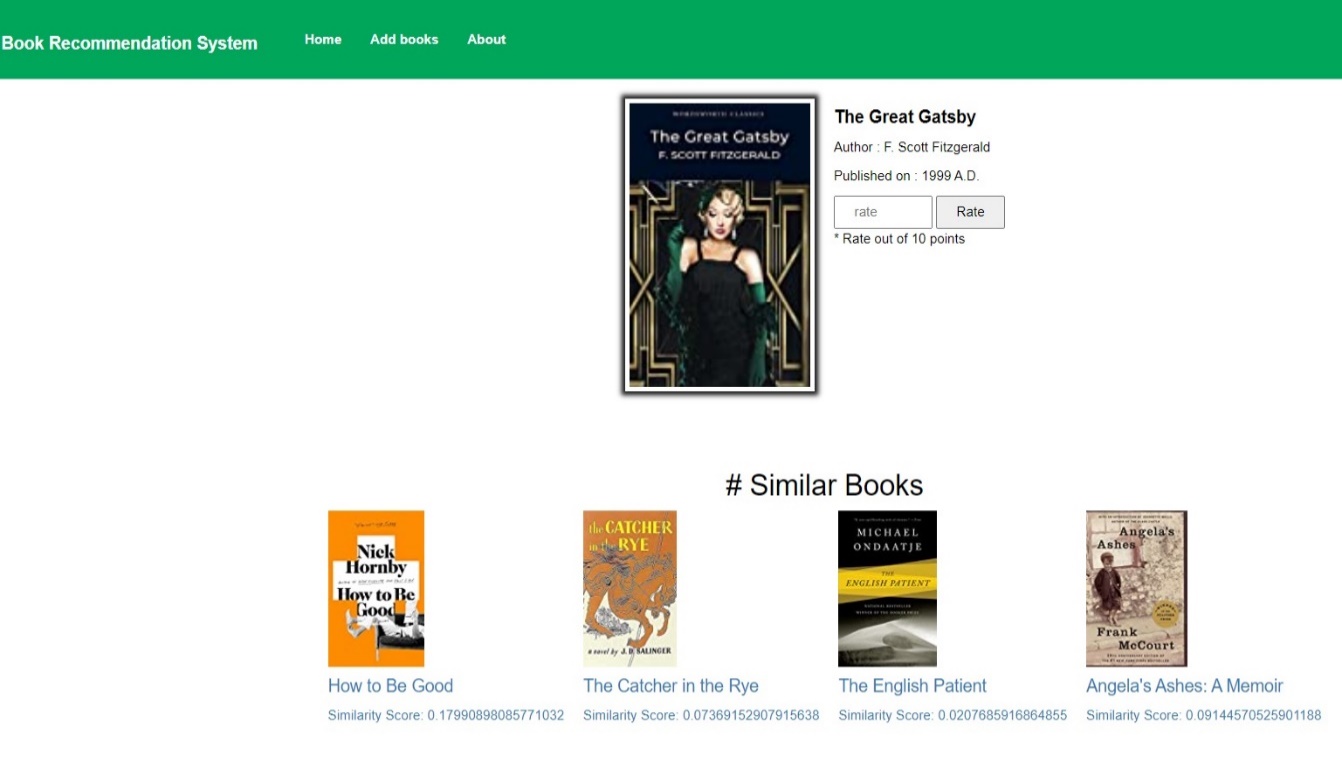
**1.1 Displaying books according to popularity**



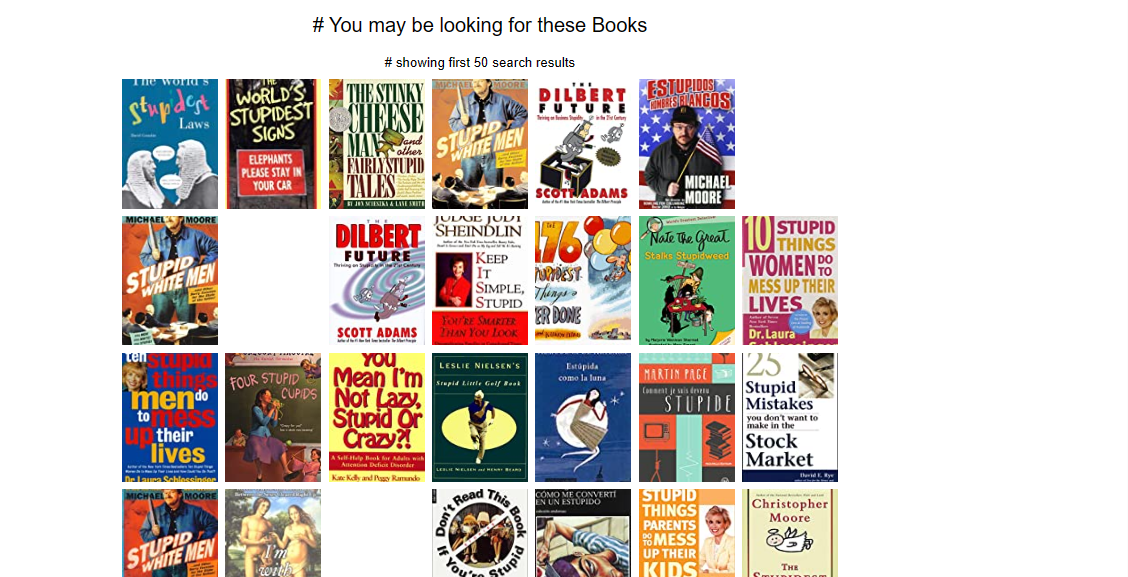
**1.2 Add books section**



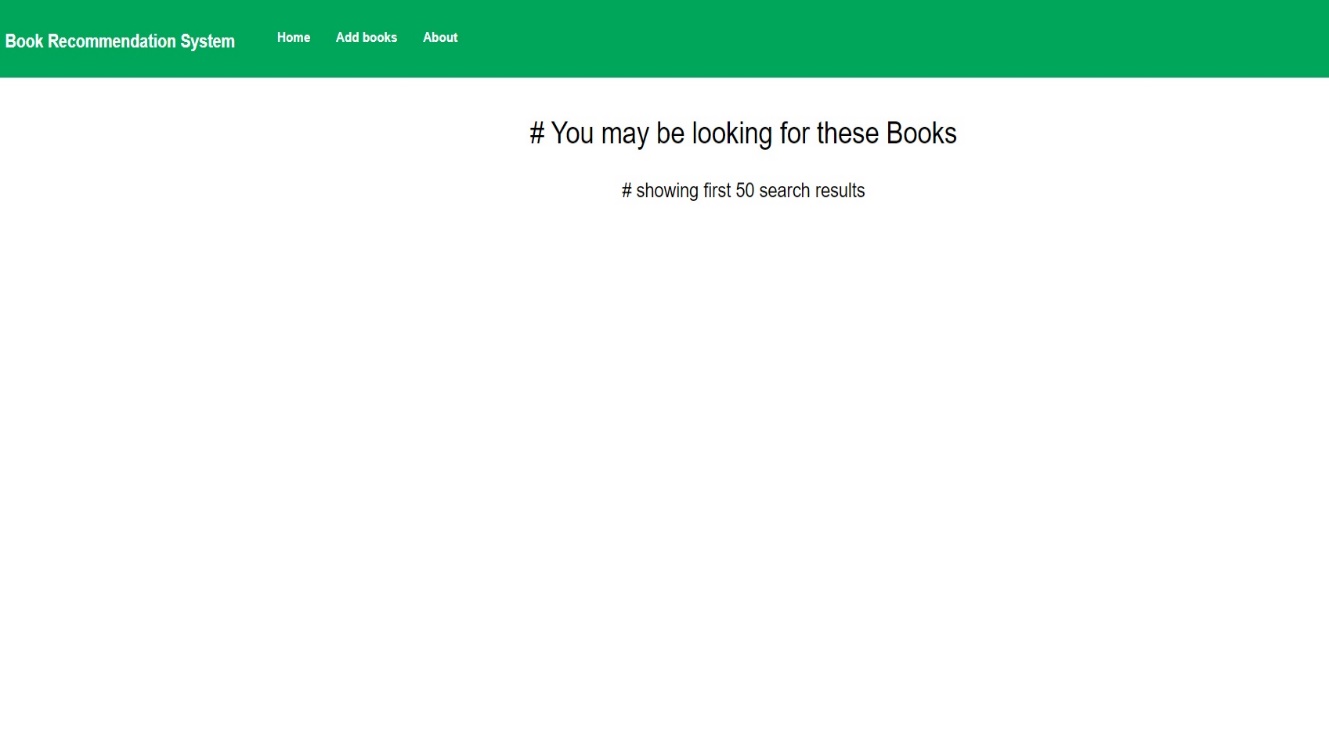
**1.3 Displaying books according to collaborative filtering**



**1.4 Displaying books according to valid keyword**



* 1. **Invalid keyword**



**ABOUT US**

