# BOOK RECOMMENDATION SYSTEM

# GROUP MEMBERS

- Vikraj Shambhu
- Shaikh Banisha Begum
- Astika Urmaliya
- Shashank Patil
- Unnati Kalmase

# OBJECTIVE

1. Exploratory Data Analysis
2. Visualization
3. Model Building
4. Deployment

# Objective

A Book Recommendation System aims to suggest books to users based on their preferences, reading history, and other relevant factors. The main objective is to enhance user experience by providing personalized book recommendations that are both relevant and engaging.

# Libraries

## Which we have used in the whole project

```
import pandas as pd
import  numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
from sklearn.feature_extraction.text import TfidfVectorizer
from PIL import Image
import requests
import re
from sklearn.decomposition import TruncatedSVD
from sklearn.metrics import mean_squared_error
from math import sqrt
from scipy.sparse import csr_matrix
from sklearn.metrics.pairwise import cosine_similarity
```

# EXPLORATORY DATA ANALYSIS

```
rating.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1149780 entries, 0 to 1149779
Data columns (total 3 columns):
 #   Column       Non-Null Count    Dtype
---  ------       --------------    -----
 0   User-ID      1149780 non-null  int64
 1   ISBN         1149780 non-null  object
 2   Book-Rating  1149780 non-null  int64
dtypes: int64(2), object(1)
memory usage: 26.3+ MB
```

```
users.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 278858 entries, 0 to 278857
Data columns (total 3 columns):
 #   Column    Non-Null Count   Dtype
---  ------    --------------   -----
 0   User-ID   278858 non-null  int64
 1   Location  278858 non-null  object
 2   Age       168096 non-null  float64
dtypes: float64(1), int64(1), object(1)
memory usage: 6.4+ MB
```

```
books.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271360 entries, 0 to 271359
Data columns (total 8 columns):
 #   Column               Non-Null Count   Dtype
---  ------               --------------   -----
 0   ISBN                 271360 non-null  object
 1   Book-Title           271360 non-null  object
 2   Book-Author          271358 non-null  object
 3   Year-Of-Publication  271360 non-null  object
 4   Publisher            271358 non-null  object
 5   Image-URL-S          271360 non-null  object
 6   Image-URL-M          271360 non-null  object
 7   Image-URL-L          271357 non-null  object
dtypes: object(8)
memory usage: 16.6+ MB
```

```
print("Books Shape: " ,books.shape )
print("Ratings Shape: " ,rating.shape )
print("Users Shape: " ,users.shape )
```

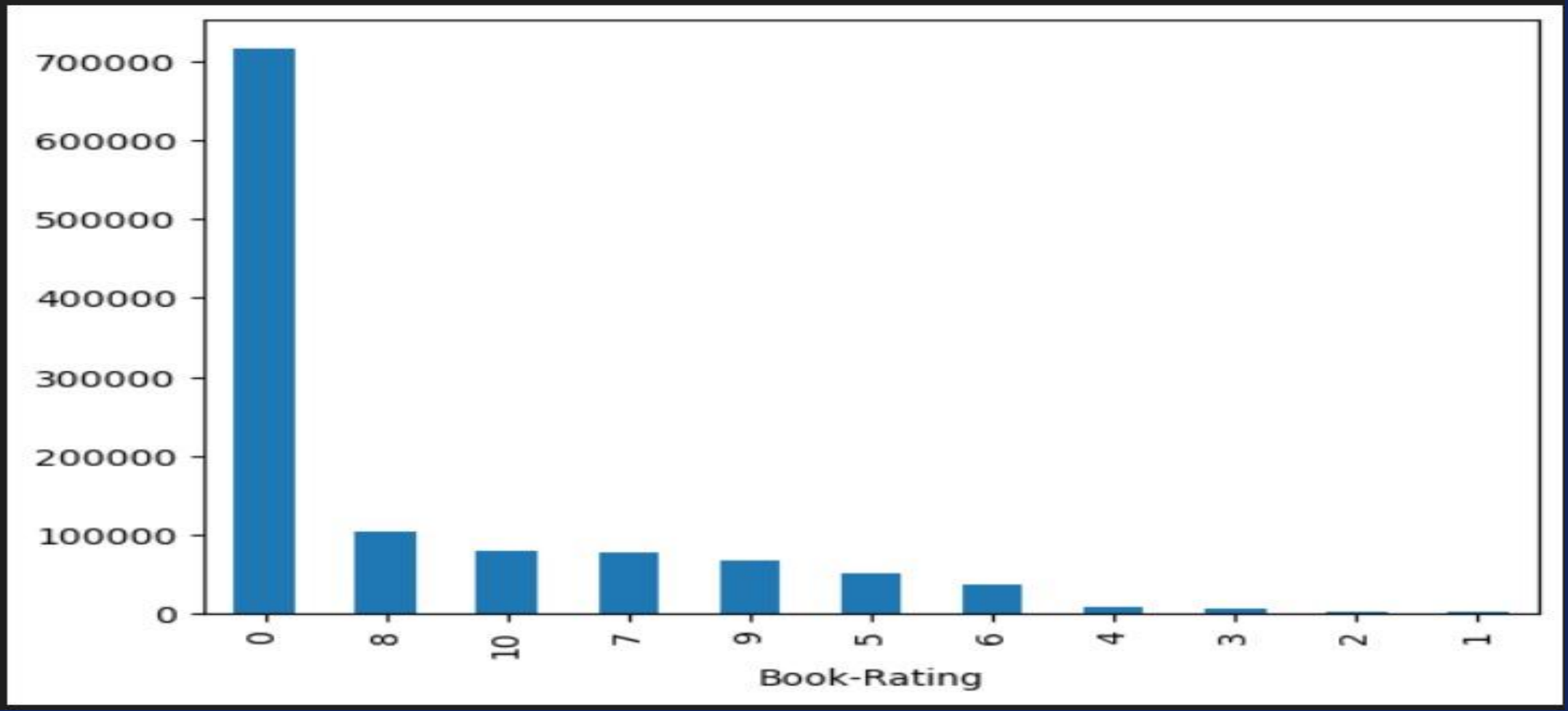
```
Books Shape:  (271360, 8)
Ratings Shape:  (1149780, 3)
Users Shape:  (278858, 3)
```

```
books.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 271360 entries, 0 to 271359
Data columns (total 8 columns):
 #   Column               Non-Null Count   Dtype 
---  ------               --------------   ----- 
 0   ISBN                 271360 non-null  object
 1   Book-Title           271360 non-null  object
 2   Book-Author          271358 non-null  object
 3   Year-Of-Publication  271360 non-null  object
 4   Publisher            271358 non-null  object
 5   Image-URL-S          271360 non-null  object
 6   Image-URL-M          271360 non-null  object
 7   Image-URL-L          271357 non-null  object
dtypes: object(8)
memory usage: 16.6+ MB
```

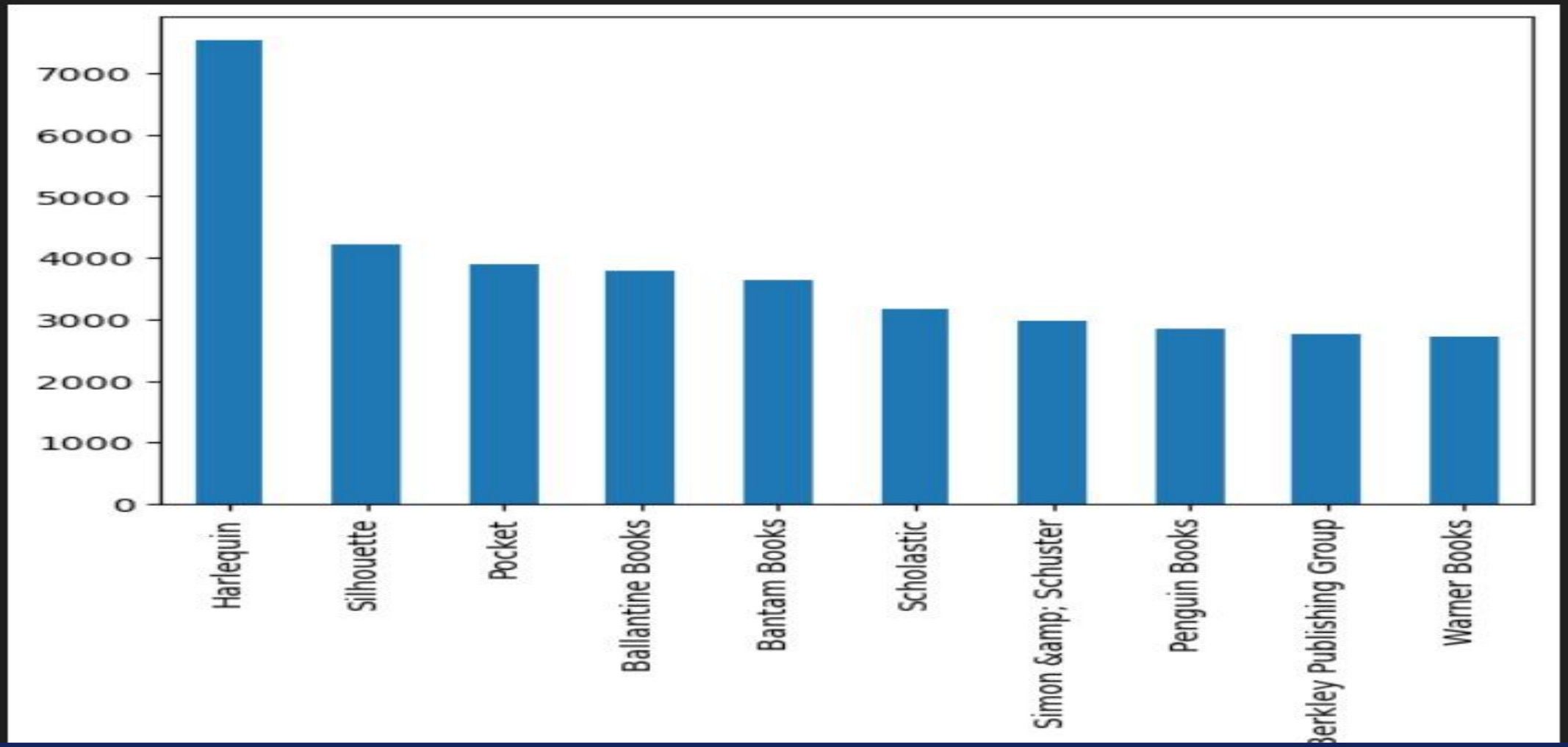
# VISUALIZATION

```
rating['Book-Rating'].value_counts().plot(kind='bar');
```



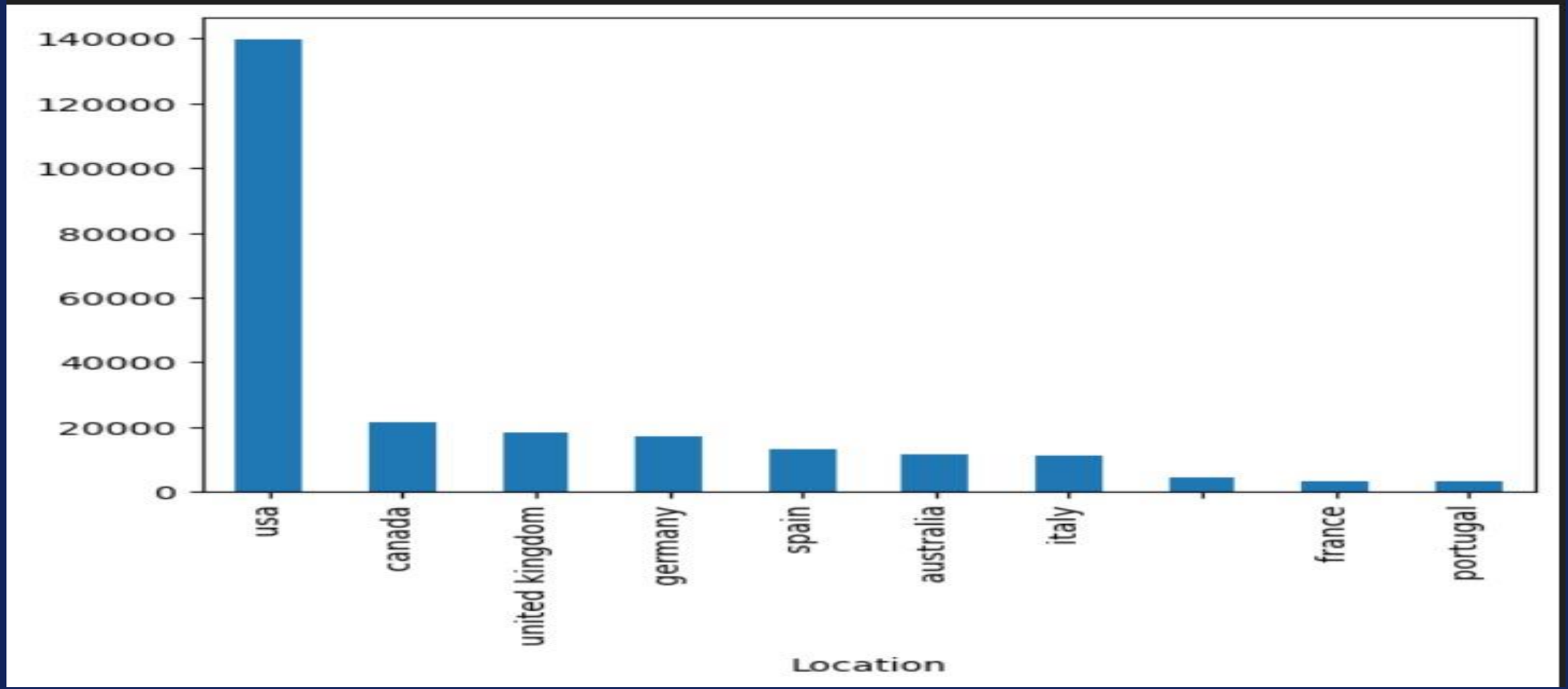
```
# top 10 Publisher
books['Publisher'].value_counts().head(10).plot(kind='bar')
```

```
<Axes: xlabel='Publisher'>
```



```
# top 10 user location
users['Location'].value_counts().head(10).plot(kind='bar')
```

```
<Axes: xlabel='Location'>
```



# POPULARITY BASED RECOMMENDATION SYSTEM

Popularity based recommendation systems are based on the rating of items by all the users.

- Popularity based recommendation systems works with the trend.
- It basically uses the items which are in trend right now.

```
ratings_with_name = rating.merge(books,on='ISBN')
```

```
num_rating = ratings_with_name.groupby('Book-Title').count()['Book-Rating'].reset_index()
num_rating.rename(columns={'Book-Rating':'num_ratings'},inplace=True)
num_rating
```

| | Book-Title | num_ratings |
|---|---|---|
| 0 | A Light in the Storm: The Civil War Diary of ... | 4 |
| 1 | Always Have Popsicles | 1 |
| 2 | Apple Magic (The Collector's series) | 1 |
| 3 | Ask Lily (Young Women of Faith: Lily Series, ... | 1 |
| 4 | Beyond IBM: Leadership Marketing and Finance ... | 1 |

```
avg_rating = ratings_with_name.groupby('Book-Title')['Book-Rating'].mean().reset_index()
avg_rating.rename(columns={'Book-Rating': 'avg_rating'}, inplace=True)
avg_rating
```

| | Book-Title | avg_rating |
|---|---|---|
| 0 | A Light in the Storm: The Civil War Diary of ... | 2.250000 |
| 1 | Always Have Popsicles | 0.000000 |
| 2 | Apple Magic (The Collector's series) | 0.000000 |
| 3 | Ask Lily (Young Women of Faith: Lily Series, ... | 8.000000 |
| 4 | Beyond IBM: Leadership Marketing and Finance ... | 0.000000 |

```
popular_df = num_rating.merge(avg_rating,on='Book-Title')
popular_df
```

| | Book-Title | num_ratings | avg_rating |
|---|---|---|---|
| 0 | A Light in the Storm: The Civil War Diary of ... | 4 | 2.250000 |
| 1 | Always Have Popsicles | 1 | 0.000000 |
| 2 | Apple Magic (The Collector's series) | 1 | 0.000000 |
| 3 | Ask Lily (Young Women of Faith: Lily Series, ... | 1 | 8.000000 |
| 4 | Beyond IBM: Leadership Marketing and Finance ... | 1 | 0.000000 |

```
popular_df = popular_df.merge(books,on='Book-Title').drop_duplicates('Book-Title')[['Book-Title','Book-Author','Image-URL-M','num_ratings','avg_rating']]
```

```
popular_df
```

| | Book-Title | Book-Author | Image-URL-M | num_ratings | avg_rating |
|---|---|---|---|---|---|
| 0 | Harry Potter and the Prisoner of Azkaban (Book 3) | J. K. Rowling | http://images.amazon.com/images/P/0439136350.0... | 428 | 5.852804 |
| 3 | Harry Potter and the Goblet of Fire (Book 4) | J. K. Rowling | http://images.amazon.com/images/P/0439139597.0... | 387 | 5.824289 |
| 5 | Harry Potter and the Sorcerer's Stone (Book 1) | J. K. Rowling | http://images.amazon.com/images/P/0590353403.0... | 278 | 5.737410 |
| 9 | Harry Potter and the Order of the Phoenix (Boo... | J. K. Rowling | http://images.amazon.com/images/P/043935806X.0... | 347 | 5.501441 |
| 13 | Harry Potter and the Chamber of Secrets (Book 2) | J. K. Rowling | http://images.amazon.com/images/P/0439064872.0... | 556 | 5.183453 |
| 16 | The Hobbit : The Enchanting Prelude to The Lor... | J.R.R. TOLKIEN | http://images.amazon.com/images/P/0345339681.0... | 281 | 5.007117 |

```
]: popular_df['Image-URL-M'][0]
```

```
]: 'http://images.amazon.com/images/P/0439136350.01.MZZZZZZZ.jpg'
```

# Collaborative Filtering Based Recommender System

```
x = ratings_with_name.groupby('User-ID').count()['Book-Rating'] > 200
users = x[x].index
users
```

```
Index([   254,    2276,    2766,    2977,    3363,    4017,    4385,    6251,    6323,
          6543,
          ...
          271705, 273979, 274004, 274061, 274301, 274308, 275970, 277427, 277639,
          278418],
        dtype='int64', name='User-ID', length=811)
```

```
# filtered_rating = ratings_with_name[ratings_with_name['User-ID'].isin(padhe_likhe_users)]
filtered_rating = ratings_with_name[ratings_with_name['User-ID'].isin(users)]
filtered_rating
```

| | User-ID | ISBN | Book-Rating | Book-Title | Book-Author | Year-Of-Publication | Publisher | Image-URL-S | |
|---|---|---|---|---|---|---|---|---|---|
| 2 | 6543 | 034545104X | 0 | Flesh Tones: A Novel | M. J. Rose | 2002 | Ballantine Books | http://images.amazon.com/images/P/034545104X.0... | http://images.amazon.com/i |
| 5 | 23768 | 034545104X | 0 | Flesh Tones: A Novel | M. J. Rose | 2002 | Ballantine Books | http://images.amazon.com/images/P/034545104X.0... | http://images.amazon.com/i |
| 7 | 28523 | 034545104X | 0 | Flesh Tones: A Novel | M. J. Rose | 2002 | Ballantine Books | http://images.amazon.com/images/P/034545104X.0... | http://images.amazon.com/i |

```
]: y = filtered_rating.groupby('Book-Title').count()['Book-Rating']>=50
famous_books =y[y].index
```

```
]: final_rating = filtered_rating[filtered_rating['Book-Title'].isin(famous_books)]
```

```
]: pt = final_rating.pivot_table(index='Book-Title',columns='User-ID',values='Book-Rating')
```

```
]: pt
```

]:

| User-ID | 254 | 2276 | 2766 | 2977 | 3363 | 4017 | 4385 | 6251 | 6323 | 6543 | ... | 271705 | 273979 | 274004 | 274061 | 274301 | 274308 | 275970 | 277427 | 277639 | 2784 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Book-Title** | | | | | | | | | | | | | | | | | | | | | |
| **1984** | 9.0 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | ... | 10.0 | NaN | NaN | NaN | NaN | NaN | 0.0 | NaN | NaN | Na |
| **1st to Die: A Novel** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 9.0 | ... | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | Na |

```
pt.fillna(0 , inplace= True)
```

```
from sklearn.metrics.pairwise import cosine_similarity
```

```
similarity_scores = cosine_similarity(pt)
```

```
similarity_scores.shape
```

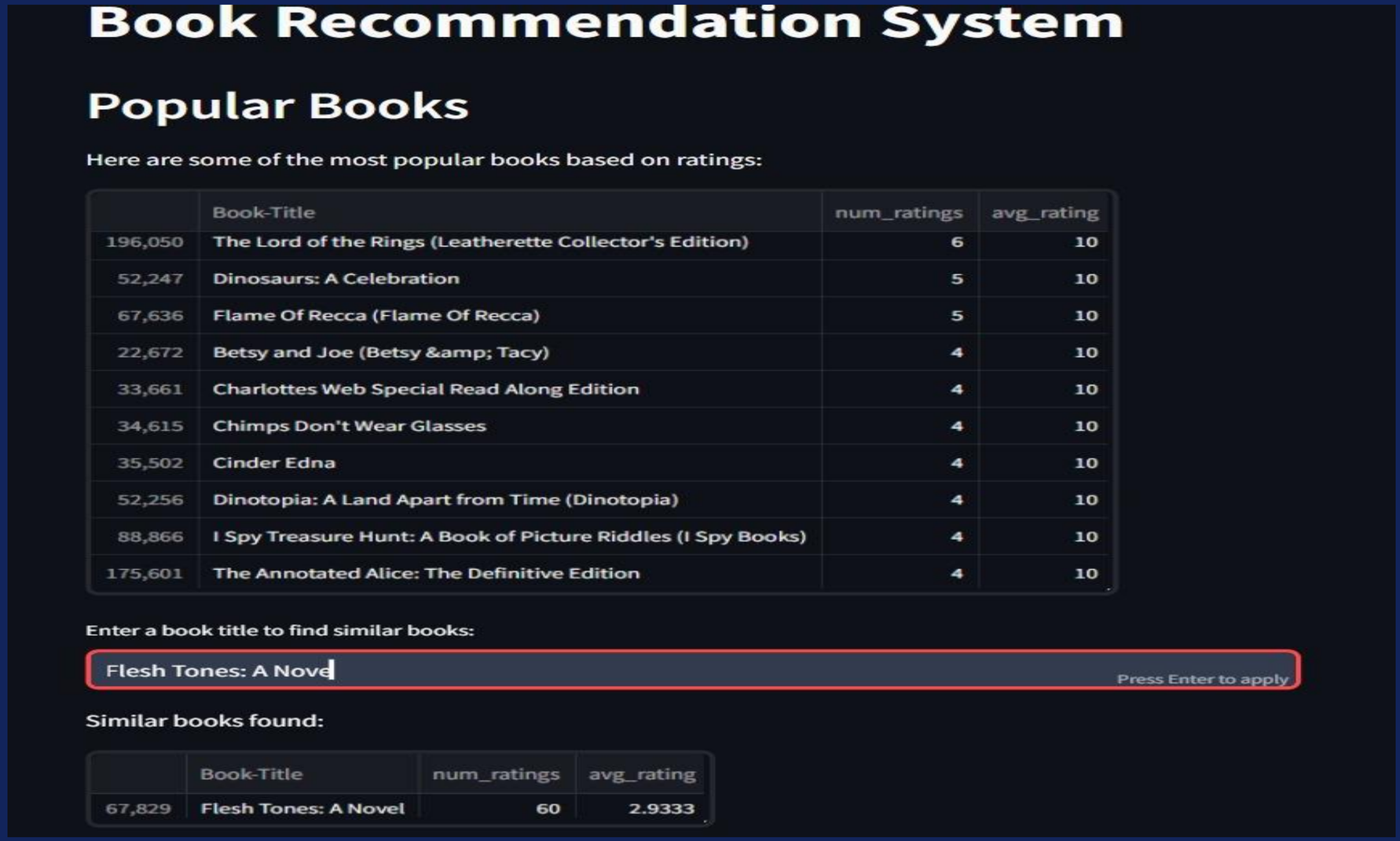
```
(706, 706)
```

```
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],re

    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
```

```
recommend('1984')
```

```
[['Animal Farm',
  'George Orwell',
  'http://images.amazon.com/images/P/0451526341.01.MZZZZZZZ.jpg'],
 ["The Handmaid's Tale",
  'Margaret Atwood',
  'http://images.amazon.com/images/P/0449212602.01.MZZZZZZZ.jpg'],
 ['Brave New World',
  'Aldous Huxley',
  'http://images.amazon.com/images/P/0060809833.01.MZZZZZZZ.jpg'],
 ['The Vampire Lestat (Vampire Chronicles, Book II)',
  'ANNE RICE',
```

# DEPLOYMENT




| | Book-Title | num_ratings | avg_rating |
|---|---|---|---|
| 196,050 | The Lord of the Rings (Leatherette Collector's Edition) | 6 | 10 |
| 52,247 | Dinosaurs: A Celebration | 5 | 10 |
| 67,636 | Flame Of Recca (Flame Of Recca) | 5 | 10 |
| 22,672 | Betsy and Joe (Betsy & Tacy) | 4 | 10 |
| 33,661 | Charlottes Web Special Read Along Edition | 4 | 10 |
| 34,615 | Chimps Don't Wear Glasses | 4 | 10 |
| 35,502 | Cinder Edna | 4 | 10 |
| 52,256 | Dinotopia: A Land Apart from Time (Dinotopia) | 4 | 10 |
| 88,866 | I Spy Treasure Hunt: A Book of Picture Riddles (I Spy Books) | 4 | 10 |
| 175,601 | The Annotated Alice: The Definitive Edition | 4 | 10 |

| | Book-Title | num_ratings | avg_rating |
|---|---|---|---|
| 67,829 | Flesh Tones: A Novel | 60 | 2.9333 |

# Challenges Faced In This Project And How We Overcome Those

- The given data set were in 3 different files, so we merged all the three different files to run the model
- There were also NA in the given data set, so we filled those NA with its mean values

# Conclusion

The development and implementation of a book recommendation system provide significant value to users by simplifying the process of discovering new books tailored to their interests. This system leverages various recommendation techniques, such as collaborative filtering, content-based filtering, and hybrid approaches, to generate personalized suggestions that enhance user engagement and satisfaction.

# THANK YOU