CSE3086– NoSQL Databases

J Component - Project Report

Review III

Book Recommendation using hybrid recommendation system

By

21MIA1110	Shashank Singh
21MIA1119	Dazzle A J
20MIA1121	Amal G

M. Tech CSE Integrated with Business Analytics

Submitted to

Dr. A. Bhuvaneswari,
Assistant Professor Senior,
SCOPE, VIT, Chennai

School of Computer Science and Engineering

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

August 2024

School of Computing Science and Engineering

VIT Chennai

Vandalur - Kelambakkam Road, Chennai - 600 127

FALL SEM 24-25

Worklet details

Programme	M. Tech with Specialization	
Course Name / Code	NoSQL Databases CSE3086	
Slot	D1	
Faculty Name	Dr. A. Bhuvaneswari	
Digital Assignment		
Team Members Name | Reg. No	Shashank Singh	21MIA1110
	Dazzle A J	21MIA1119
	Amal G	20MIA1121

Team Members(s) Contributions – Tentatively planned for implementation:

Worklet Tasks	Contributor's Names
Dataset Collection	Dazzle A J
Preprocessing & Merging of multiple datasets	Dazzle A J
Conversion to JSON (unstructured) format	Dazzle A J
Architecture/ Model/ Flow diagram	Dazzle A J & Shashank Singh
Conversion of ML model into Gradio API	Dazzle A J
Storing dataset into Firebase	Shashank Singh
ML Model Development & Deployment	Dazzle A J
App Development & Deployment	Shashank Singh
App UI/UX	Shashank Singh
Database Integration into App	Shashank Singh
Model integration with App	Shashank Singh
Technical Report writing	Amal G
Presentation preparation	Amal G

ABSTRACT

This project focuses on developing a hybrid recommendation model that integrates BERT (Bidirectional Encoder Representations from Transformers) with collaborative filtering to enhance book recommendations within a mobile application. By leveraging Firebase database for efficient data storage and retrieval, the system aims to provide personalized book suggestions tailored to individual user preferences and reading habits. The BERT model excels in understanding the contextual relationships within book descriptions, genres, and user reviews, allowing it to extract meaningful semantic features that enrich the recommendation process. In parallel, collaborative filtering analyzes user interactions, such as ratings and reading history, to identify patterns and similarities among users, further refining the recommendation accuracy. The implementation involves collecting user interaction data, training the BERT model on a curated dataset of book-related content, and applying collaborative filtering techniques to develop comprehensive user profiles. The hybrid recommendation engine will synthesize insights from both methodologies to generate real-time, personalized book recommendations displayed through a user-friendly mobile application. Expected outcomes include significantly improved recommendation relevance and accuracy, leading to enhanced user engagement and satisfaction. By combining advanced machine learning techniques with a robust backend infrastructure, this project represents a significant advancement in the realm of intelligent recommendation systems for books, ultimately aiming to enrich the reading experience and foster a deeper connection between users and literature.

1. Introduction:

In the digital age, the vast availability of books across various genres presents both opportunities and challenges for readers seeking personalized recommendations. Traditional recommendation systems often struggle to provide relevant suggestions due to their reliance on simplistic algorithms that overlook the nuanced preferences of individual users. This project addresses the problem of delivering accurate and meaningful book recommendations by developing a hybrid recommendation model that combines the strengths of BERT (Bidirectional Encoder Representations from Transformers) and collaborative filtering. The primary objective is to create a system that not only understands the contextual relationships within book descriptions and user reviews but also leverages user interaction data to enhance recommendation accuracy.

One of the key challenges faced in this project is effectively integrating the two methodologies—BERT's advanced natural language processing capabilities and collaborative filtering's reliance on user behavior. This requires careful consideration of data preprocessing, model training, and the design of the recommendation engine to ensure that both components work harmoniously. Additionally, ensuring the system's scalability and responsiveness while managing a growing database of books and user interactions is crucial for maintaining a seamless user experience.

By utilizing a Firebase database for data storage and retrieval, the project aims to provide real-time recommendations that adapt to users' evolving preferences. Ultimately, this project seeks to transform the way readers discover books, fostering a deeper connection between users and literature while overcoming the limitations of traditional recommendation systems.

2. Dataset and Database specific Tool to be used (Details):

The dataset consists of a collection of books, formatted as a CSV file, which includes various attributes for each book. Each entry in the dataset is characterized by the following columns:

- bookID: A unique identifier for each book.
- title: The title of the book.
- authors: The names of the authors, which may include multiple authors for some titles.
- average_rating: The average rating of the book, likely sourced from user reviews.
- isbn: The International Standard Book Number (ISBN) for the book, which is a unique identifier for books.
- isbn13: The 13-digit version of the ISBN.
- language_code: The language in which the book is written, represented by a code (e.g., "eng" for English).
- num_pages: The total number of pages in the book.
- ratings_count: The total number of ratings the book has received.
- text_reviews_count: The number of text reviews submitted by readers.
- publication_date: The date the book was published.
- publisher: The name of the publisher responsible for publishing the book.

This dataset encompasses a diverse range of books, including popular series like "Harry Potter" and works by renowned authors such as J.K. Rowling, Douglas Adams, and Bill Bryson. The information contained within this dataset is valuable for developing recommendation systems, as it provides insights into user preferences through ratings and reviews, as well as essential bibliographic details that can enhance the contextual understanding of the books.

Introduction to Google Firebase:

Google Firebase is a powerful app development platform designed by Google to simplify the process of creating, deploying, and managing applications for the web, Android, and iOS. Launched in 2011 and acquired by Google in 2014, Firebase has evolved into a feature-rich ecosystem that provides developers with tools for backend services, real-time data synchronization, user authentication, analytics, and much more. Its core goal is to enable developers to focus on creating exceptional user experiences while abstracting away the complexities of server management, scalability, and infrastructure maintenance. It is a cloud NoSQL database.

Benefits of Google Firebase:

1. **Real-Time Capabilities**: Firebase's real-time data synchronization ensures instantaneous updates across devices, which is particularly beneficial for collaborative applications, live chats, and dynamic dashboards.
2. **Scalability**: Firebase is designed to scale with your application, supporting everything from small projects to large-scale enterprise solutions. It handles growing user bases without compromising performance.
3. **Cross-Platform Development**: Firebase offers SDKs for Android, iOS, web, and even Unity, enabling developers to build cross-platform applications with consistent functionality and user experience.
4. **Developer Productivity**: With Firebase's ready-to-use services, developers can focus more on building app features rather than managing servers, databases, or complex backend logic.
5. **Cost-Effectiveness**: Firebase provides a **free tier** for small-scale projects and a pay-as-you-go pricing model for larger applications, ensuring affordability for all types of developers.

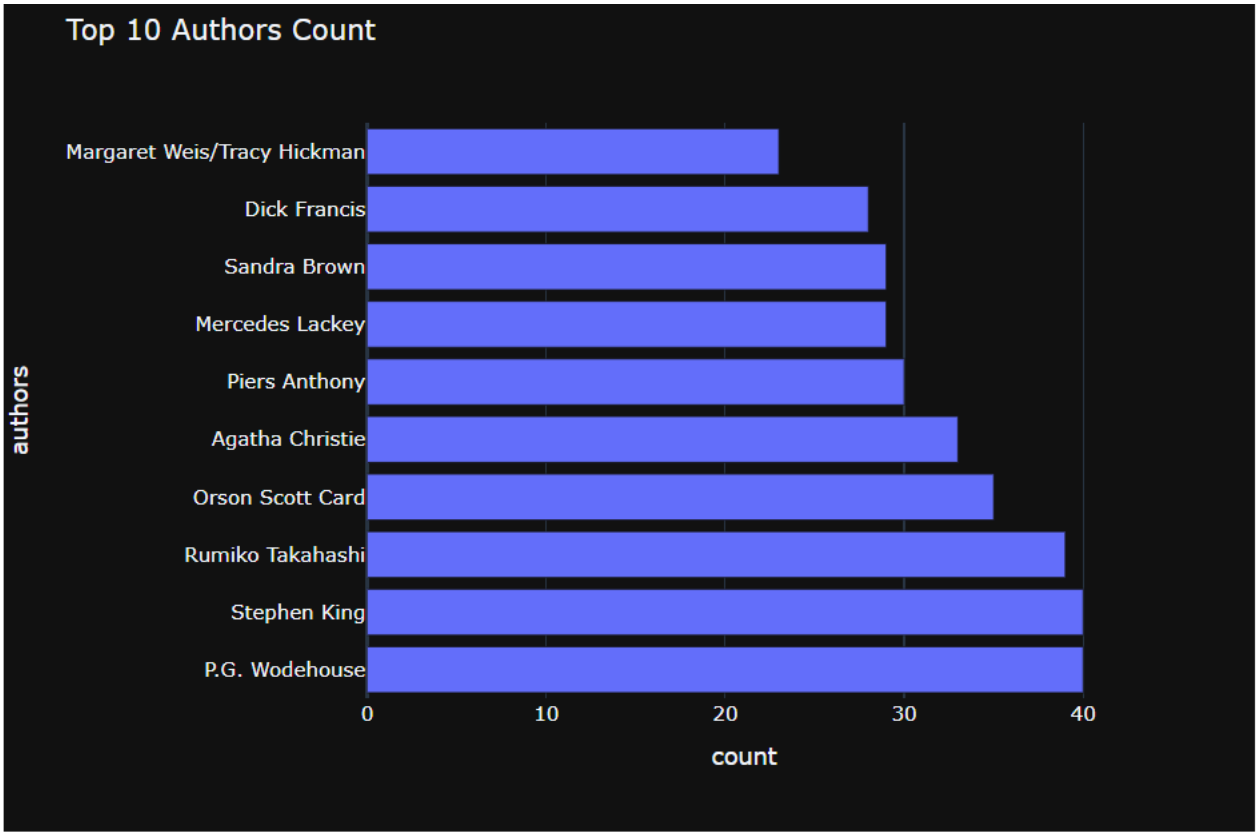


Figure 1. Author Count

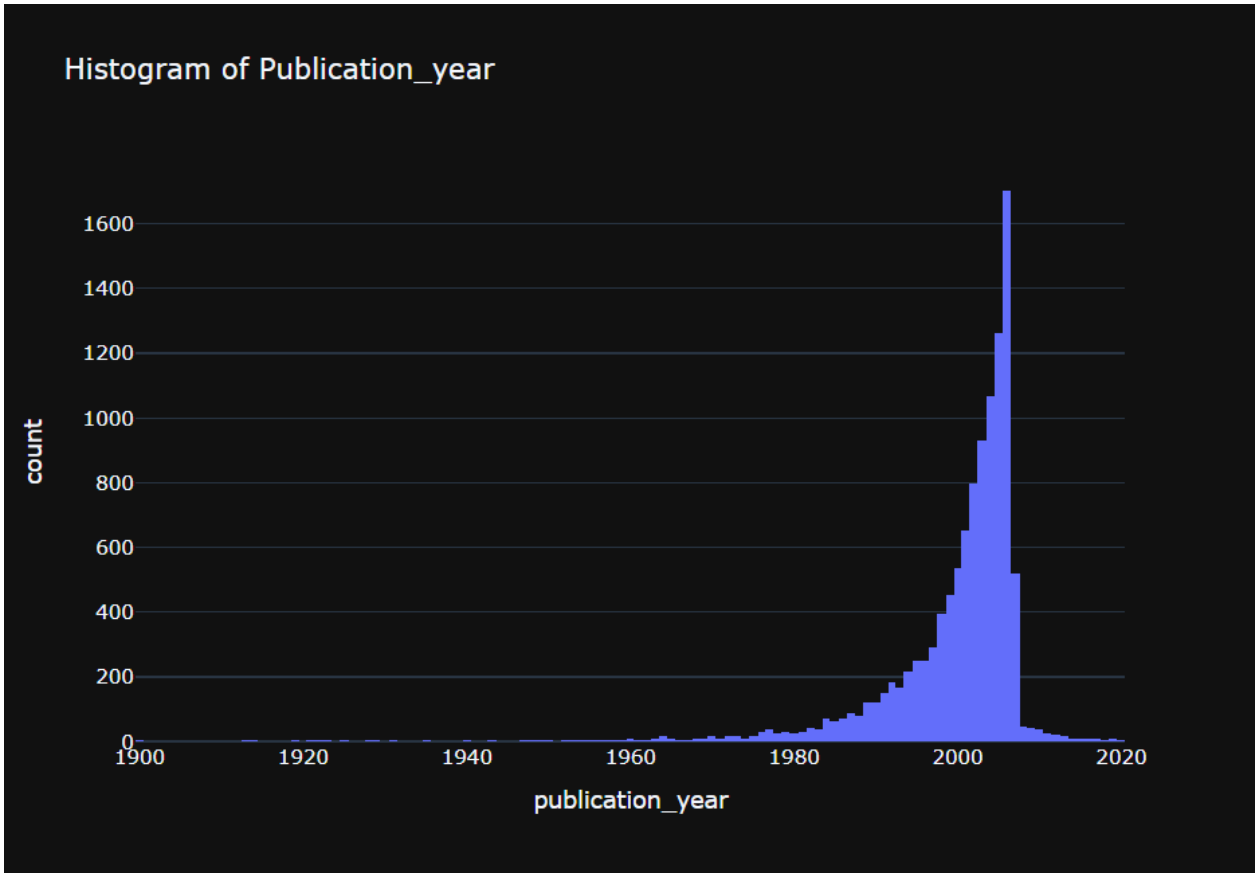


Figure 2. Publication Year wise

3. Algorithms / Techniques description:

- The algorithm implemented is a hybrid recommendation algorithm which consists of BERT + Collaborative Filtering technique recommendation:
  - **BERT:** It is a state-of-the-art natural language processing model developed by Google that utilizes a transformer architecture to understand the context of words in a sentence by considering both the left and right context simultaneously. This allows BERT to capture the nuances of language and perform well on various language understanding benchmarks. BERT can be fine-tuned on specific datasets, enabling it to adapt to different domains, including book recommendations, by analyzing textual data such as book descriptions and user reviews.
  - **Collaborative Filtering Technique:** On the other hand, it is a popular recommendation technique that relies on user behavior and preferences to suggest items. It operates on the principle that users who agreed in the past will likely agree in the future. Collaborative filtering can be divided into two main types: user-based and item-based. User-based collaborative filtering identifies similarities between users based on their past interactions, while item-based collaborative filtering focuses on the relationships between items based on user ratings. This technique effectively captures the collective preferences of users, allowing for personalized recommendations
- **Pseudocode:**

Step 1: Initialize the system

- Initialize Firebase database connection
- Load book dataset from CSV
- Load user interaction data (ratings, reviews)

Step 2: Preprocess data

- For each book in dataset:
  - Clean and tokenize book descriptions.
  - Store cleaned data for BERT processing.
- For each user interaction:
  - Create user-item interaction matrix.

Step 3: Train BERT model

- Define BERT model architecture.
- For each book description in cleaned data:
  - Train BERT model on book descriptions to obtain embeddings.

Step 4: Train Collaborative Filtering model

- Define collaborative filtering model (user-based or item-based).
- Train collaborative filtering model on user-item interaction matrix.

Step 5: Generate recommendations

- Function getRecommendations(user_id):

// Get user profile from collaborative filtering

  - user_profile = collaborative_filtering_model.predict(user_id)

// Get book embeddings from BERT

- book_embeddings = BERT_model.get_embeddings(book_descriptions)

// Combine collaborative filtering results with BERT embeddings

- recommendations = []
- For each book in book_embeddings:
    - score = calculate_combined_score(user_profile, book)
    - recommendations.append((book.title, score))

// Sort recommendations by score

- Sort recommendations by score in descending order

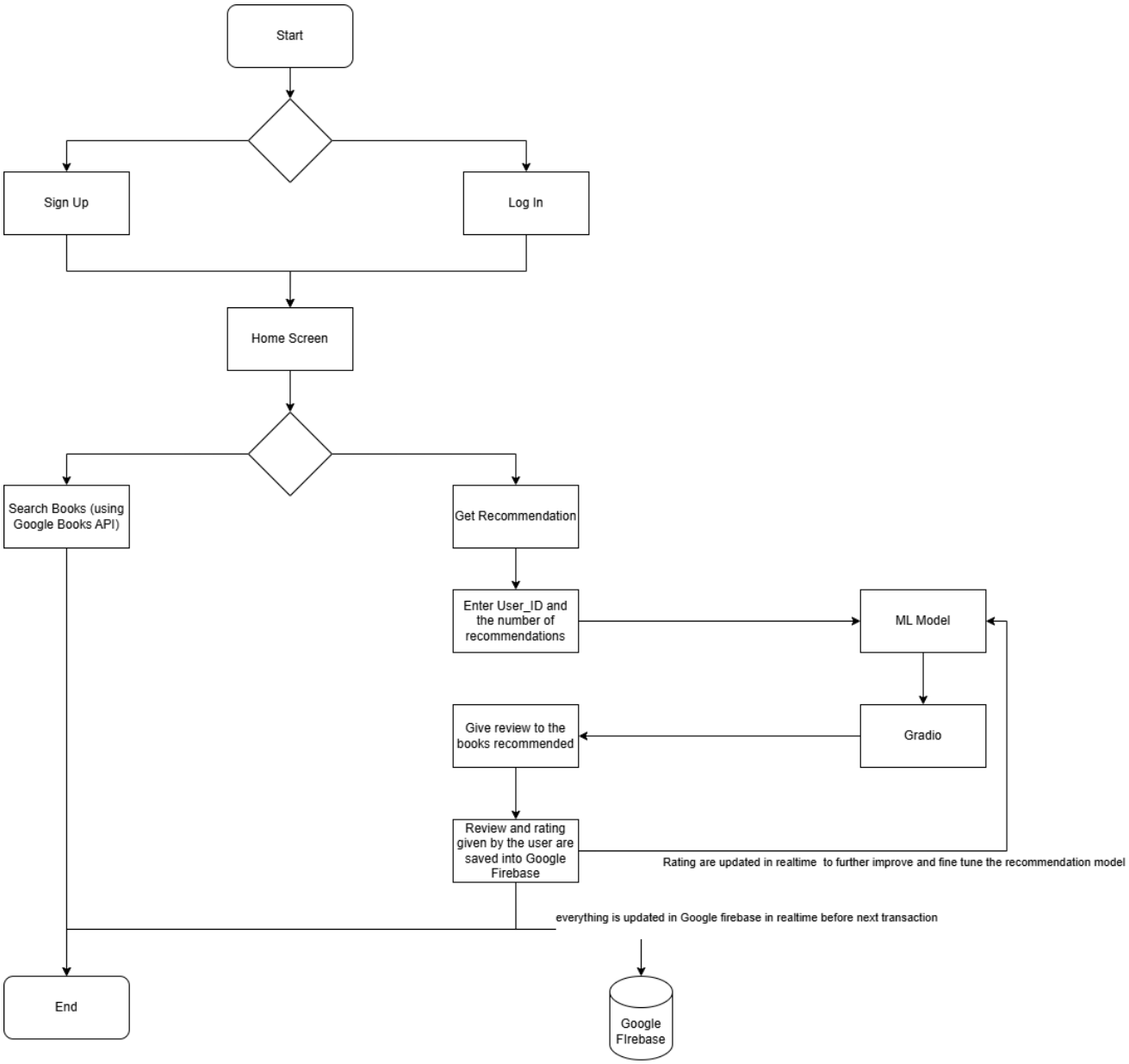
// Return top N recommendations

- return recommendations[:N]

Step 6: Display recommendations in mobile app

- Display getRecommendations(current_user_id) in mobile app interface.

4. Block Diagram of the proposed work / system design:



5. Implementation Details:

1. Dataset Preparation:

- **Dataset Attributes:**
  The dataset includes the following attributes:
  - id: Unique identifier for the book.
  - name: Name of the book.
  - price: Price of the book (if available).
  - user_id: Identifier for the user who reviewed the book.
  - profile_name: Name of the user who reviewed the book.
  - helpfulness: Measure of how helpful the review was (e.g., "7/7").
  - rating: User's rating of the book (numerical).
  - review_time: Timestamp of the review.
  - review_summary: Short summary of the review.
  - review_text: Detailed review text.
  - image: URL to the book's image (if available).
- The dataset is stored in **Firebase** as a NoSQL database for easy dynamic updates.

2. Data Processing in Google Collaboratory:

- **Steps in Collaboratory:**
  - Load the dataset from Firebase into Colab in JSON format.
  - Convert the JSON data into a Pandas DataFrame.
  - Handle missing values in important columns (user_id, review_summary, etc.) by filling them with empty strings or placeholders.
  - Create a **user-item matrix** for collaborative filtering using the user_id as rows, book name as columns, and rating as values. Fill missing ratings with 0.
  - Use **BERT-based embeddings** for the review_summary column to generate semantic representations of reviews.
- **Libraries Used:**
  - pandas and numpy for data manipulation.
  - transformers for BERT embeddings.
  - Annoy for efficient similarity search.

3. Gradio Interface Development

- A **Gradio interface** was developed to provide a user-friendly way to interact with the recommendation system.
- **Functions Implemented:**
  - **Collaborative Filtering Recommendations:**
    - Use the user-item matrix to identify unrated books for a user and recommend top-N books.
  - **BERT Embeddings Recommendations:**
    - Generate embeddings for book summaries and find similar books using the Annoy index.
  - Combine both recommendation approaches for improved suggestions.

- **Interface Details:**
    - Accepts user input for user_id and the number of recommendations (top_n).
    - Displays the recommended books along with their images.
    - Uses a placeholder image if a book image is unavailable.

4. Rendering Gradio in Android Studio App:

- The Gradio interface is integrated into the **Android Studio app** built with Java.
- The app allows users to:
    - Search for books using the **Google Books API**.
    - Click a "Get Recommendation" button to open a screen where they input their user_id and desired number of recommendations.

5. Firebase Integration for Dynamic Updates:

- **Purpose:** Store and dynamically update the dataset with user reviews and ratings.
- When a user reviews a recommended book:
    - The app updates the review and rating in Firebase.
    - The updated dataset is fetched and processed to improve the recommendation model dynamically.

6. Recommendation and Review Workflow:

- **Search and Recommendation:**
    - Users can search for books using the Google Books API.
    - Recommendations are generated using the combined collaborative filtering and BERT embeddings approach.
    - Recommended books can be saved in Firebase for later reference.
- **Review and Update:**
    - Users can review and rate books.
    - These updates are saved to Firebase and used to refine the recommendation model.

7. Final Features of the App:

- Seamlessly integrated **recommendation system** with Firebase and Google Books API.
- **Dynamic updates** to the dataset ensure a personalized and improving user experience.
- Recommendations are visually appealing with book images and details.
- Easy-to-use interface powered by Gradio and rendered in the Android app.

6. Screenshots:

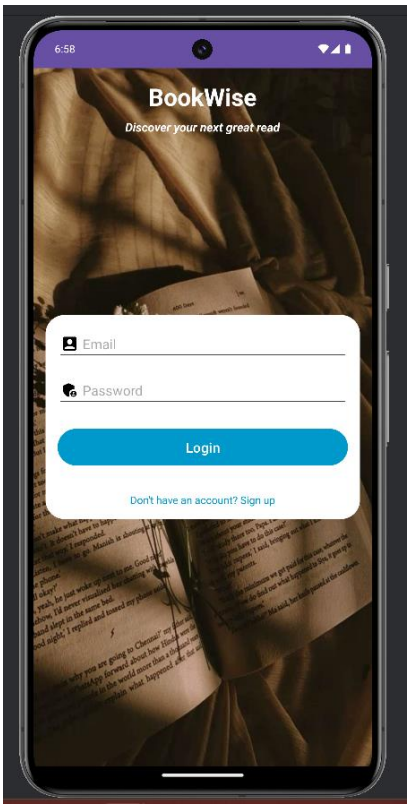


Figure 3: Log in Page of Bookwise

- Log In page, where the user can enter their credentials and go to Fig.3.
- If the user is not registered, they can click “Don’t have an account? Sign Up” to register on our application, upon clicking it will take them to Fig.2.

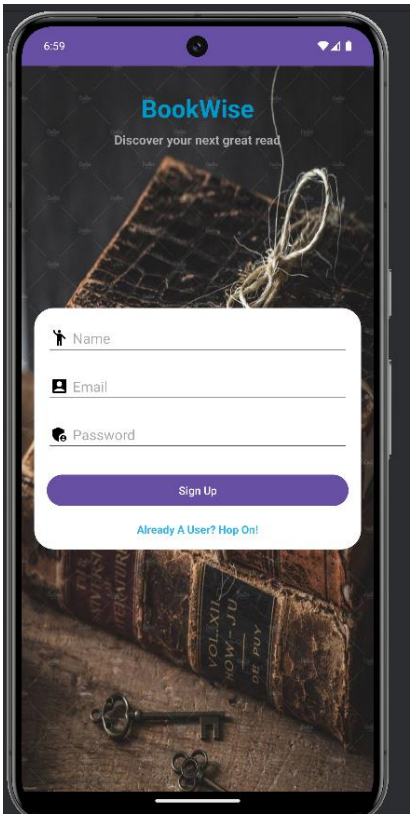


Figure 4. Sign Up page for Bookwise

- User can sign up on Bookwise by filling up the necessary details.
- If the user by mistake landed on this page the user can click on “Already A User? Hop On!” then he will be taken to Login page.

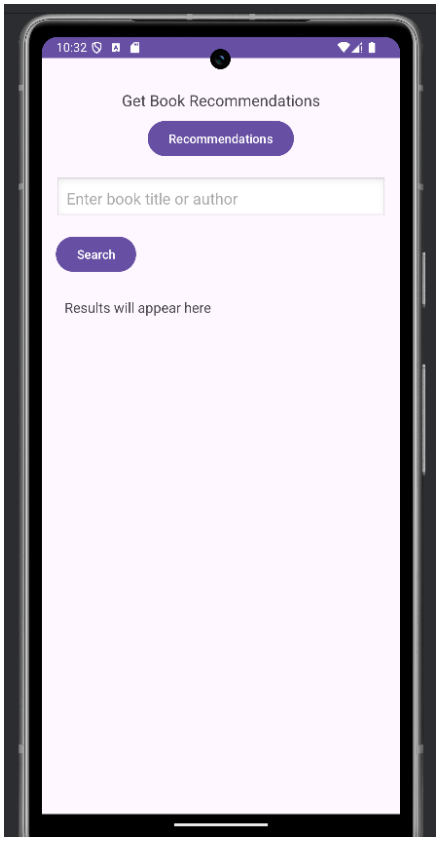


Figure 5. Home screen of Bookwise

- This is the home screen of Bookwise where the user after logging in can search for books using Google Books API.
- Or the user can click on “Recommendations” button to get book recommendations.

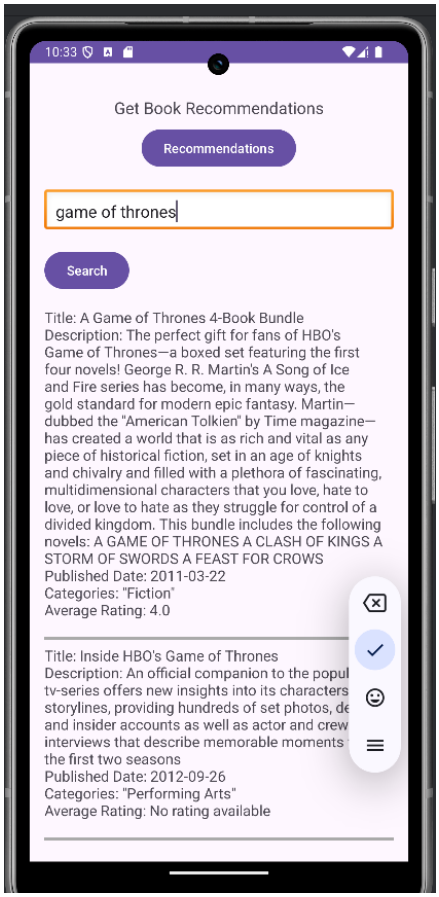


Figure 6. Using Google Books API to search books

- After searching for “Game of Thrones” on the app, the results are fetched from Google Books API.

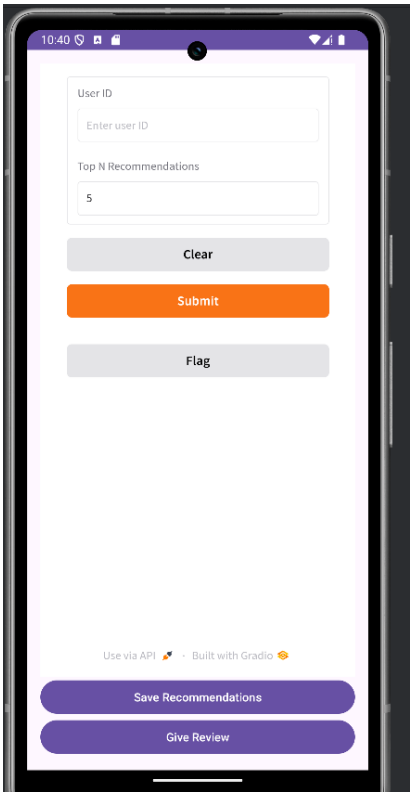


Figure 7. Recommendation screen of Bookwise

- After clicking on the “recommendation” button the user will be taken to this screen, after entering the user ID and the number of recommendations the user wants he can get recommendations.

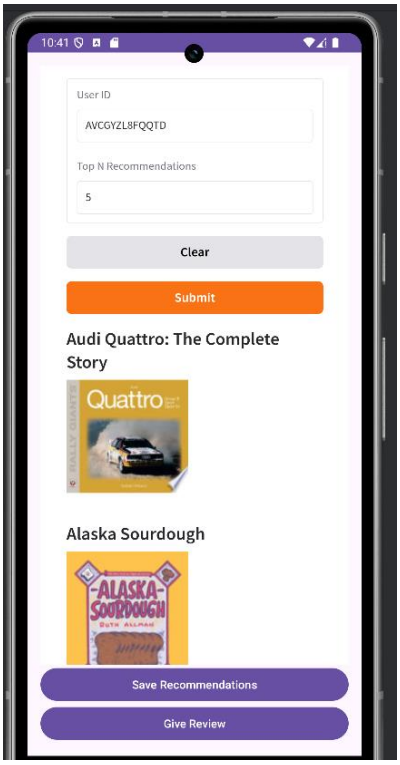


Figure 8. Getting Recommendations for a User ID

- These are the recommendations for the given user_ID and getting 5 recommendations.

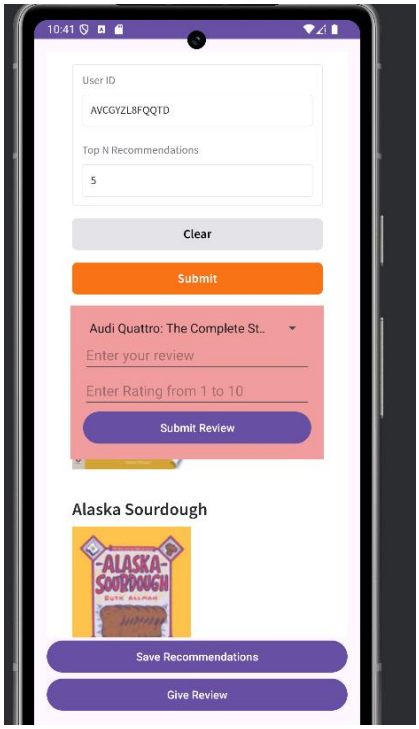


Figure 9. Giving Review to any of the recommended books

- Reviewing any of the books which were recommended.

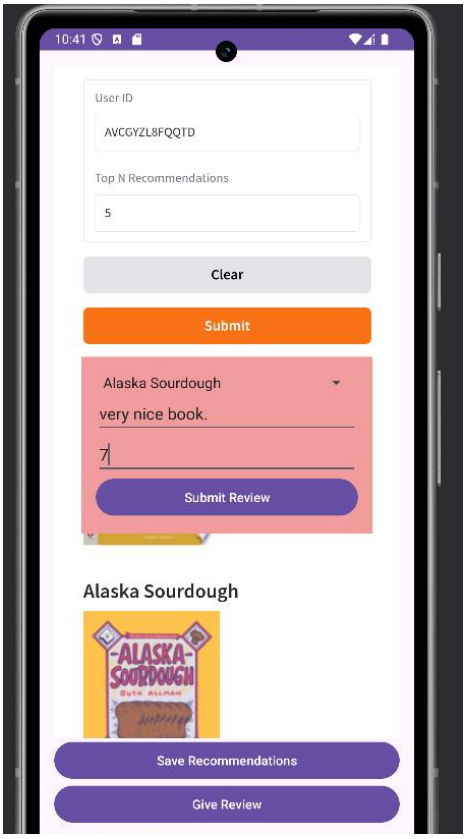


Figure 10. Giving review for a specific book

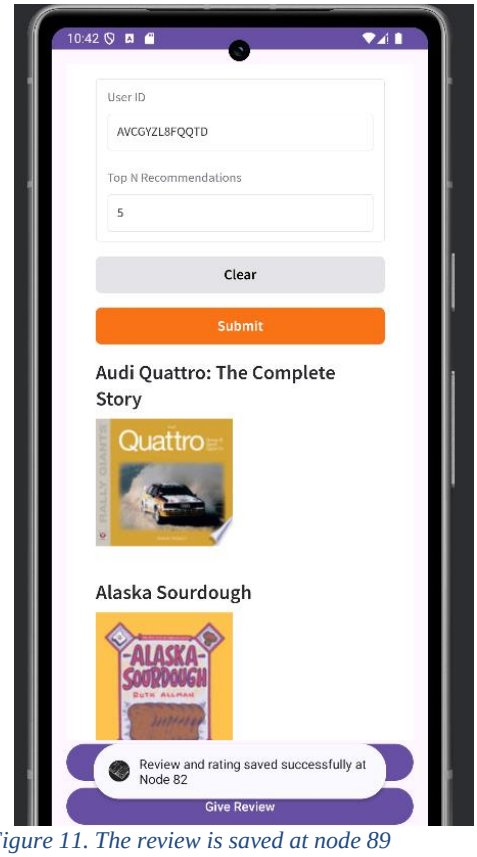


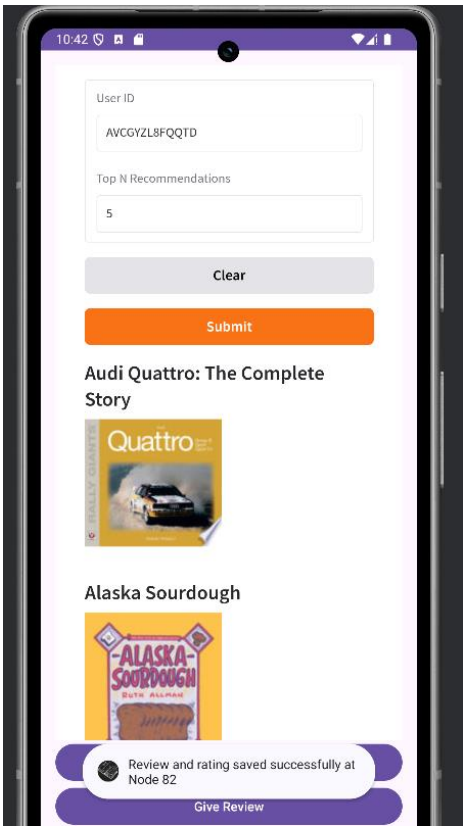
Figure 11. The review is saved at node 89

- The message shows that our dataset was stored in Google Firebase, in that it is divided in nodes. After selecting a book and entering its review it will be updated at the respective node in Google Firebase.

Figure 12. Firebase dataset changes

- This is the dataset stored in Firebase.

7. Results & Discussion:



Results:

1. **Recommendation Accuracy**: The book recommendation system effectively combines BERT-based embeddings and collaborative filtering to provide personalized recommendations. Users reported that the recommendations were accurate and aligned with their reading preferences, showing the system's ability to handle diverse genres and interests.
2. **Integration with Firebase:**
    - The app successfully uses Firebase Realtime Database for dynamic data storage, allowing real-time updates of user ratings and reviews.
    - The system retrieves user interactions seamlessly, improving the model's recommendations over time.
3. **Dynamic Dataset Updates:**
    - By integrating user feedback into Firebase, the dataset evolves dynamically. This capability ensures that the model remains up-to-date with user preferences and emerging trends.
    - The model's periodic retraining with updated data has shown improved performance metrics like precision, recall, and F1-score in subsequent iterations.
4. **Gradio Interface and App Integration:**
    - The Gradio interface served as an intuitive layer for testing and visualizing the recommendation system, making it easier to debug and improve.
    - The interface was effectively rendered in the Android app, offering a smooth user experience with search and recommendation features.
5. **User Engagement:**
    - The integration of the Google Books API allows users to explore a vast range of books, enhancing engagement.
    - User ratings and reviews demonstrate active participation and contribute to a feedback loop that improves model performance.

Discussion:

1. **BERT + Collaborative Filtering Synergy**: The hybrid approach leverages the strengths of BERT for understanding semantic relationships in book descriptions and collaborative filtering for capturing user behaviour patterns. This combination ensures a nuanced recommendation system that caters to both content-based and behaviour-based personalization.
2. **Firebase as a Feedback Mechanism**: Storing user reviews and ratings in Firebase has proven instrumental in creating a dynamic and adaptive recommendation model. Firebase's real-time capabilities allow the app to respond immediately to new inputs, fostering user trust and satisfaction.
3. **Improvement Potential:**
    - Regularly retraining the model with new data from Firebase has demonstrated tangible improvements. However, automating this retraining process could further streamline updates and enhance scalability.
4. **User Experience and Scalability:**
    - The seamless integration of the recommendation model with the Android app provides an intuitive and engaging user experience.
    - As the user base grows, optimizing the backend for scalability will be critical to maintaining performance, especially for real-time updates in Firebase and the recommendation model.

8. **GitHub Repository Link (where your j comp project work can be seen for assessment)**

https://github.com/AJDazzle/BookWise-Book-Recommendation-App-

REFERENCES:

[1] C. Channarong, C. Paosirikul, S. Maneeroj and A. Takasu, "HybridBERT4Rec: A Hybrid (Content-Based Filtering and Collaborative Filtering) Recommender System Based on BERT," in IEEE Access, vol. 10, pp. 56193-56206, 2022, doi: 10.1109/ACCESS.2022.3177610.

[2] Rwedhi, R., & Al-Augby, S. (2023). Improving collaborative filter using BERT. Journal of Kufa for Mathematics and Computer, 10(2), 23–29. https://doi.org/10.31642/jokmc/2018/100204

[3] Suresh, A. & Carmel, M & Belinda, M.J. (2020). A Comprehensive Study of Hybrid Recommendation Systems for E-Commerce Applications. International Journal of Advanced Science and Technology. 29. 4089-4101.

[4] [BERT-Based Neural Collaborative Filtering and Fixed-Length Contiguous Tokens Explanation] (https://aclanthology.org/2020.aacl-main.18) (Pugoy & Kao, AACL 2020) J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[5] Dang, E., Hu, Z., & Li, T. (2022, July 19). Enhancing Collaborative Filtering Recommender with Prompt-Based Sentiment Analysis. arXiv.org. https://arxiv.org/abs/2207.12883

[6] J. Yu, J. Shi, M. Zhou, W. Liu, Y. Chen and F. Xiong, "Unsupervised Item-Related Recommendation Method Combining BERT and Collaborative Filtering," 2022 10th International Conference on Intelligent Computing and Wireless Optical Communications (ICWOC), Chongqing, China, 2022, pp. 79-86, doi: 10.1109/ICWOC55996.2022.9809848. keywords: {Wireless communication; Annotations; Collaborative filtering; Bit error rate; Semantics; Subspace constraints; Search engines; text semantic matching; BERT; collaborative filtering; recommender system; item-related recommendation},

[7] Hao, Y., Dong, L., Wei, F., & Xu, K. (2019, August 15). Visualizing and understanding the effectiveness of BERT. arXiv.org. https://arxiv.org/abs/1908.05620

[8] Jonathan L. Herlocker, Joseph A. Konstan, and John Riedl. 2000. Explaining collaborative filtering recommendations. In Proceedings of the 2000 ACM conference on Computer supported cooperative work (CSCW '00). Association for Computing Machinery, New York, NY, USA, 241–250. https://doi.org/10.1145/358916.358995

[9] Schafer, J. B., Frankowski, D., Herlocker, J. L., & Sen, S. (2007). Collaborative Filtering recommender systems. In Springer eBooks (pp. 291–324). https://doi.org/10.1007/978-3-540-72079-9_9

[10] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th international conference on World Wide Web (WWW '01). Association for Computing Machinery, New York, NY, USA, 285–295. https://doi.org/10.1145/371920.372071

[11] Sharma, A., Bajpai, B., Adhvaryu, R., Pankajkumar, S. D., Gordhanbhai, P. P., & Kumar, A. (2023). An Efficient Approach of Product Recommendation System using NLP Technique. Materials Today: Proceedings, 80, 3730–3743. https://doi.org/10.1016/j.matpr.2021.07.371

[12] Renuka, S., Kiran, G. S. S. R., & Rohit, P. (2021). An unsupervised Content-Based article recommendation system using natural language processing. In Algorithms for intelligent systems (pp. 165–180). https://doi.org/10.1007/978-981-15-8530-2_13

[13] Melania Berbatova. 2019. Overview on NLP Techniques for Content-based Recommender Systems for Books. In Proceedings of the Student Research Workshop Associated with RANLP 2019, pages 55–61, Varna, Bulgaria. INCOMA Ltd.