BookClub

READ YAP REPEAT

IAT 359: Milestone 3
Qiraa Qadri
301454940

Abstract:

Book Club is a social reading app designed to combine the personal enjoyment of reading with the interactivity of social media. The app allows users to track their reading progress, share updates, and connect with friends and family through posts, quotes, and book recommendations. Unlike existing cataloging platforms such as Goodreads, Book Club focuses on connection and engagement. Users can upload photos, comment on posts, and view friends' reading activity in real time. By integrating with Google Books API, Book Club gives users easy access to book information while maintaining a dynamic and personalized experience.

Original Idea:

The initial concept for the project was Book Club, a social reading app designed to combine traditional book-tracking tools with interactive, community-focused features. Unlike platforms such as Goodreads, the goal was to create a more socially engaging environment where users could share their reading progress, post thoughts or quotes, and recommend books to friends.

The target audience included casual and dedicated readers, friend groups, and book clubs looking for a space to share updates and discover new reads. The app aimed to meet these needs by pairing reading lists with social media-style interactions.

The original feature plan included:

- User Authentication for secure individual accounts.
- Book Search powered by APIs such as Google Books or Open Library.
- Interactive Reading Lists categorized by reading status and filterable by parameters like genre.
- A Social Feed for posts, quotes, photos, and reading updates.
- Recommendations and Annotations to let users share notes, progress markers, and reasons for recommending books.
- Progress Tracking through visual elements like cards or progress bars.

From a technical standpoint, the main challenges anticipated were API integration, managing real-time social updates, and organizing the data needed for posts, lists, and user interactions. The planned architecture followed a modular, component-based structure with core screens for Home, My Library, Book Search, and Profile. Cloud storage was expected to support user data, though the specifics were initially left open for further research.

The intended user flow was straightforward:
Login → Home / Search / Library / Profile.

Completion of Features:

By this stage of development, several major components of the original plan have been successfully implemented. The core user interface and navigation system are fully functional, and users can sign up and log in securely using Firebase authentication. The book search feature has also been completed through integration with the Google Books API, allowing users to retrieve metadata such as titles, authors, and descriptions.

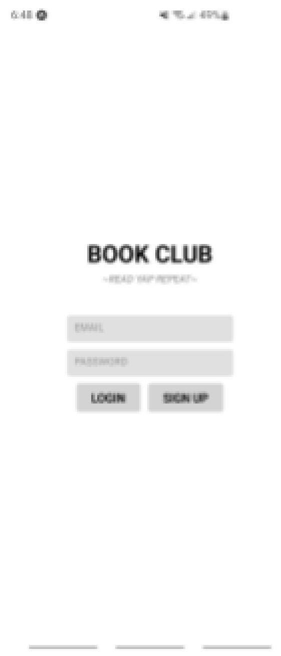
Key social features are in place. Users can view a functional feed displaying posts from other members of their book club, and they can create and share their own posts, including both text and images. Camera access has been added to support photo attachments directly within the app. Other completed features include:

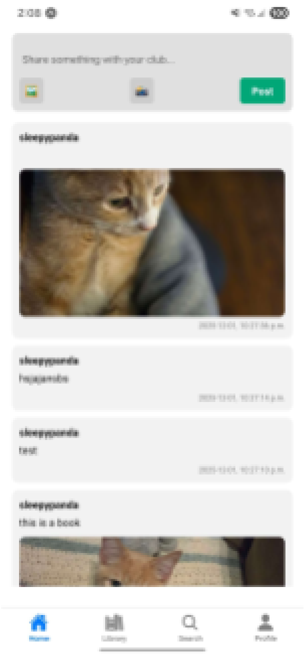
- User Reading Lists: Users can view their “Read,” “Currently Reading,” and “Want to Read” lists directly on the Library screen.
- Profile Information: The profile page now displays user stats and favorite books.
- Social Feed and Post Storage: Users can upload text and image posts that are stored and retrieved reliably.

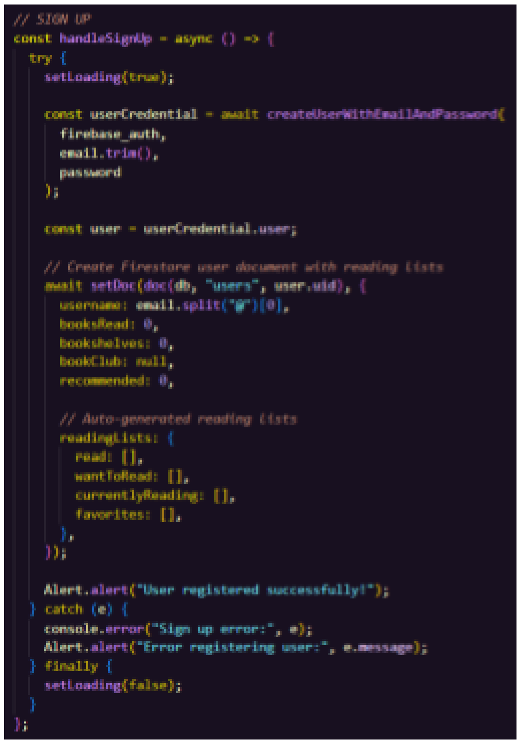
Feature Overview:

User Auth / Log In Screen:

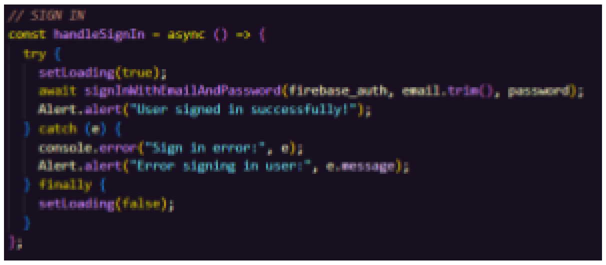
Secure login and sign-up system powered by Firebase.







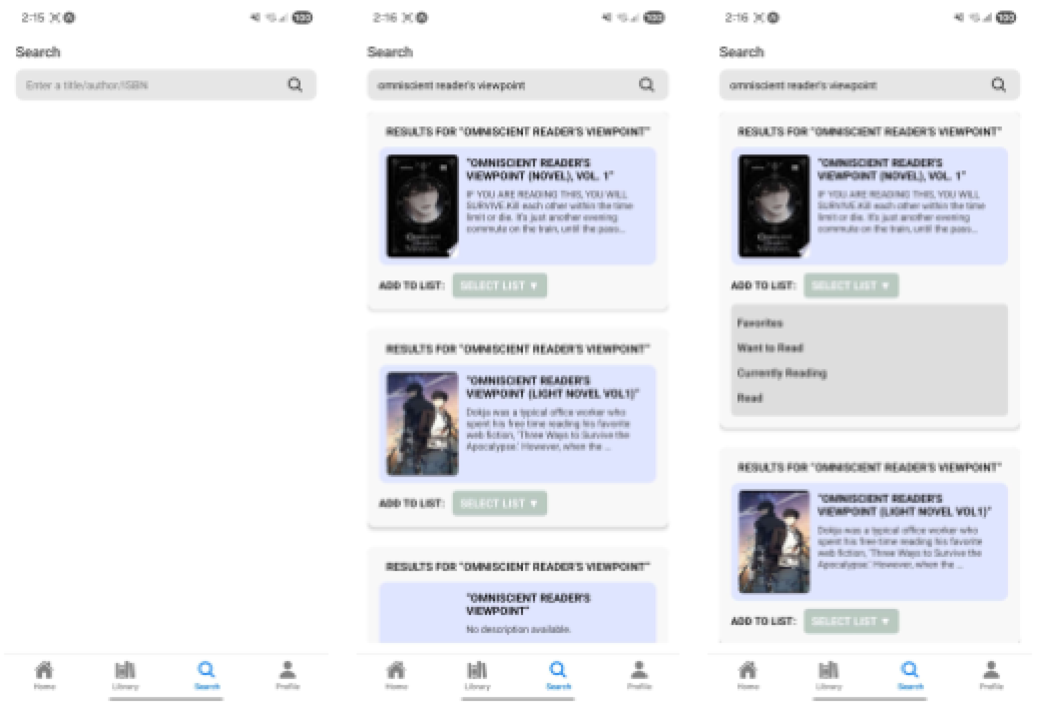
```
// SIGN UP
const handleSignUp = async () => {
  try {
    setLoading(true);

    const userCredential = await createUserWithEmailAndPassword(
      firebase_auth,
      email.trim(),
      password
    );

    const user = userCredential.user;

    // Create Firestore user document with reading lists
    await setDoc(doc(db, "users", user.uid), {
      username: email.split("@")[0],
      booksRead: 0,
      bookshelves: 0,
      bookClub: null,
      recommended: 0,

      // Auto-generated reading lists
      readingLists: {
        read: [],
        wantToRead: [],
        currentlyReading: [],
        favorites: [],
      },
    });

    Alert.alert("User registered successfully!");
  } catch (e) {
    console.error("Sign up error:", e);
    Alert.alert("Error registering user:", e.message);
  } finally {
    setLoading(false);
  }
};
```



```
// SIGN IN
const handleSignIn = async () => {
  try {
    setLoading(true);
    await signInWithEmailAndPassword(firebase_auth, email.trim(), password);
    Alert.alert("User signed in successfully!");
  } catch (e) {
    console.error("Sign in error:", e);
    Alert.alert("Error signing in user:", e.message);
  } finally {
    setLoading(false);
  }
};
```

Book Search:

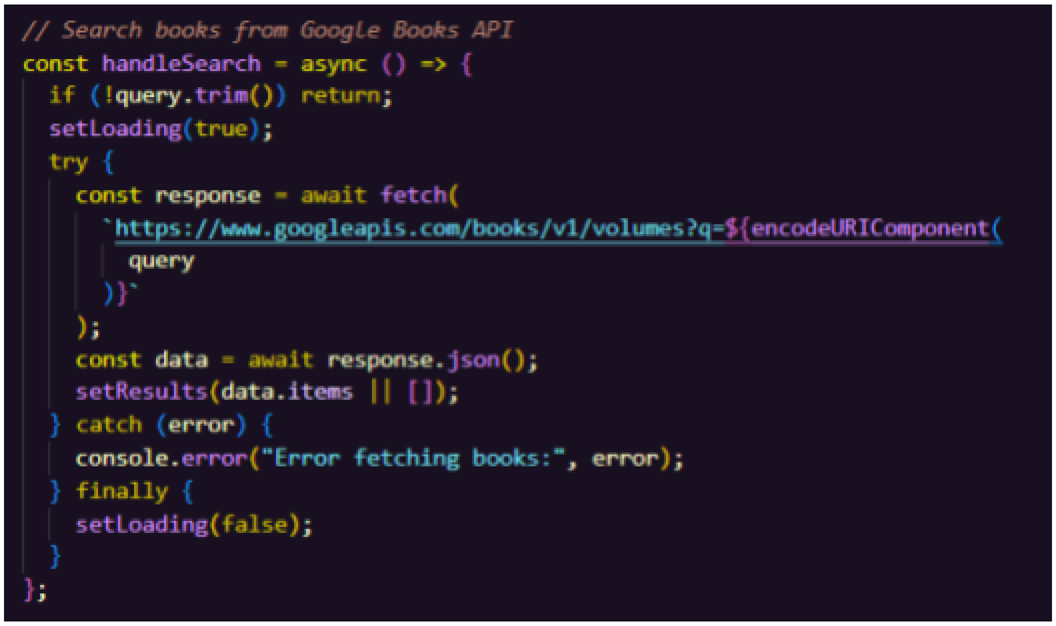
Searches the Google Books database and displays key book details.


Search
Enter a title/author/ISBN
omniscient reader's viewpoint
RESULTS FOR "OMNISCIENT READER'S VIEWPOINT"
"OMNISCIENT READER'S VIEWPOINT (NOVEL), VOL. 1"
ADD TO LIST:
SELECT LIST ▾
"OMNISCIENT READER'S VIEWPOINT (LIGHT NOVEL VOL1)"
"OMNISCIENT READER'S VIEWPOINT"
Favorites
Want to Read
Currently Reading
Read
Home
Library
Search
Profile

IF YOU ARE READING THIS, YOU WILL SURVIVE. Kill each other within the time limit or die. It's just another evening commute on the train, until the pass...

Dokja was a typical office worker who spent his free time reading his favorite web fiction, 'Three Ways to Survive the Apocalypse.' However, when the ...

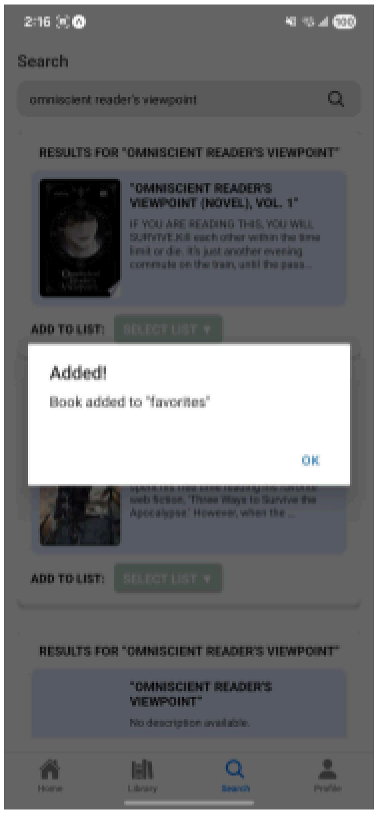
No description available.



```
// Search books from Google Books API
const handleSearch = async () => {
  if (!query.trim()) return;
  setLoading(true);
  try {
    const response = await fetch(
      `https://www.googleapis.com/books/v1/volumes?q=${encodeURIComponent(
        query
      )}`
    );
    const data = await response.json();
    setResults(data.items || []);
  } catch (error) {
    console.error("Error fetching books:", error);
  } finally {
    setLoading(false);
  }
};
```

Reading Lists + Library Screen:

Users can view their Favorites, Read, Currently Reading, and Want to Read shelves.

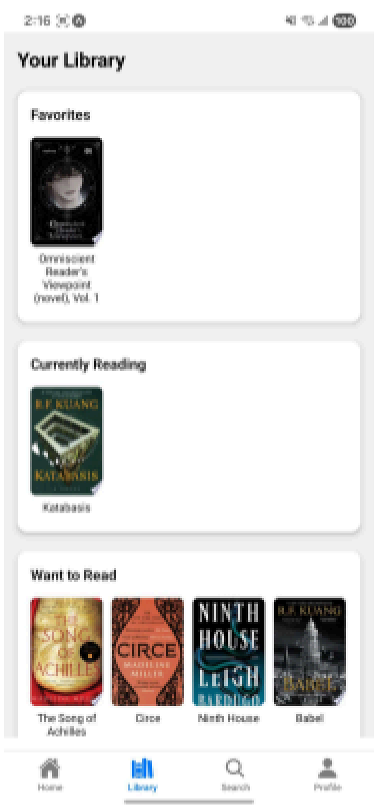

2:16
Search
omniscient reader's viewpoint
RESULTS FOR "OMNISCIENT READER'S VIEWPOINT"
"OMNISCIENT READER'S VIEWPOINT (NOVEL), VOL. 1"
ADD TO LIST:
SELECT LIST ▾
ADD TO LIST:
SELECT LIST ▾
RESULTS FOR "OMNISCIENT READER'S VIEWPOINT"
"OMNISCIENT READER'S VIEWPOINT"
No description available.
Home
Library
Search
Profile

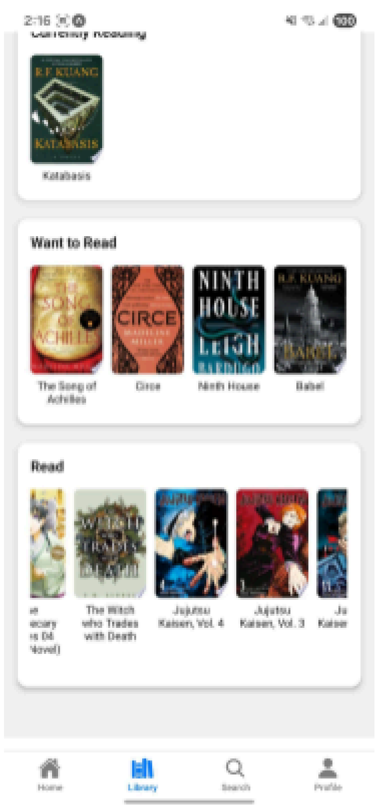
IF YOU ARE READING THIS, YOU WILL SURVIVE.Kill each other within the time limit or die. It's just another evening commute on the train, until the pass...

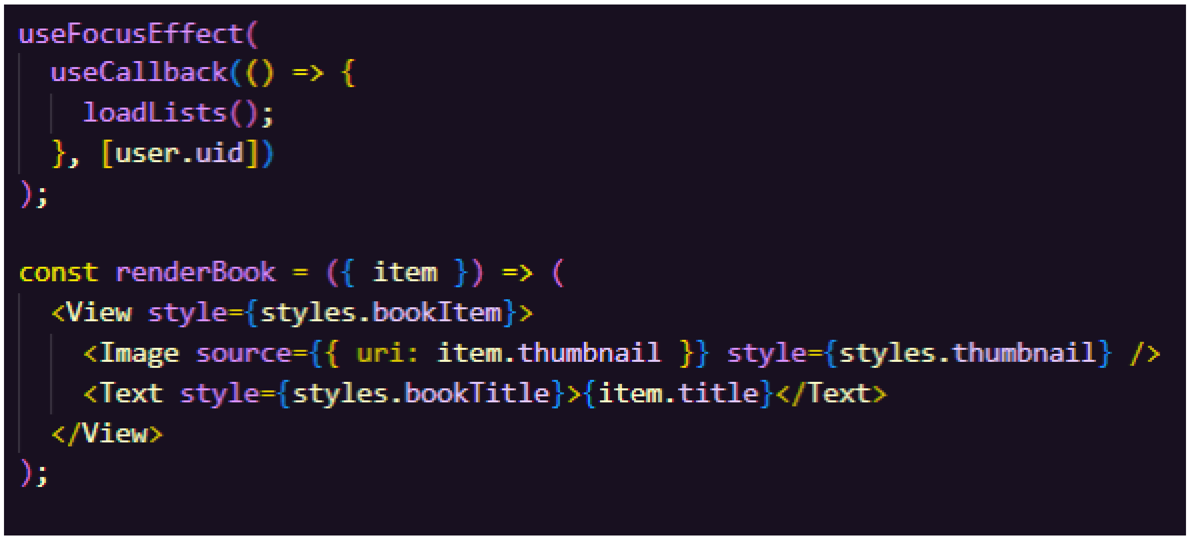
Added!

Book added to "favorites"

OK



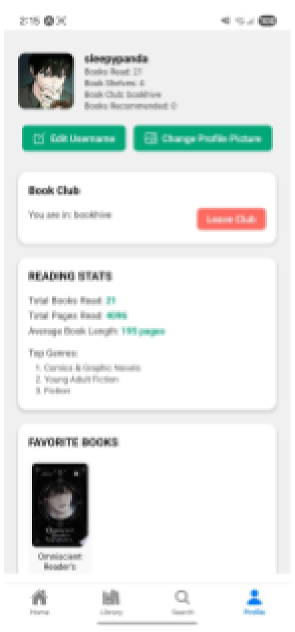




```
useFocusEffect(
  useCallback(() => {
    loadLists();
  }, [user.uid])
);

const renderBook = ({ item }) => (
  <View style={styles.bookItem}>
    <Image source={{ uri: item.thumbnail }} style={styles.thumbnail} />
    <Text style={styles.bookTitle}>{item.title}</Text>
  </View>
);
```

Social Feed:

Shows posts and updates shared by other members of the book club.



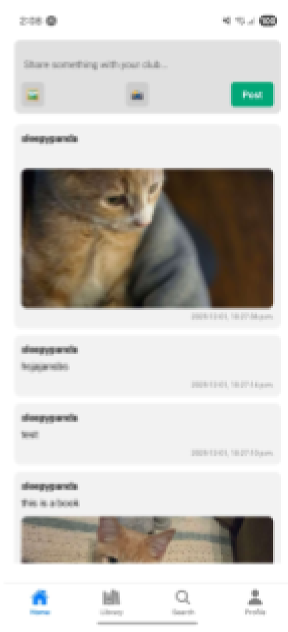
sleepypanda
Books Read: 21
Book Shelves: 4
Book Club: bookhive
Books Recommended: 0

Edit Username
Change Profile Picture

Book Club
You are in: bookhive
Leave Club

READING STATS
Total Books Read: 21
Total Pages Read: 4096
Average Book Length: 195 pages
Top Genres:
1. Comics & Graphic Novels
2. Young Adult Fiction
3. Fiction

FAVORITE BOOKS
Omniscient Reader's



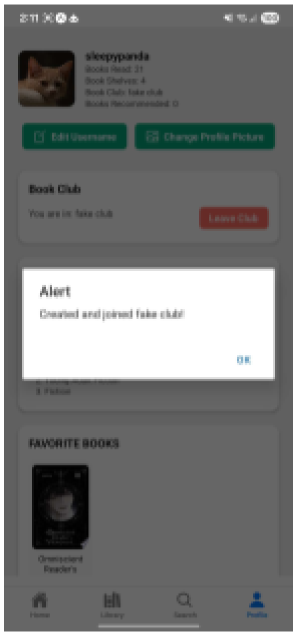
Share something with your club...
Post

sleepypanda

sleepypanda
[illegible]

sleepypanda
test

sleepypanda
this is a book



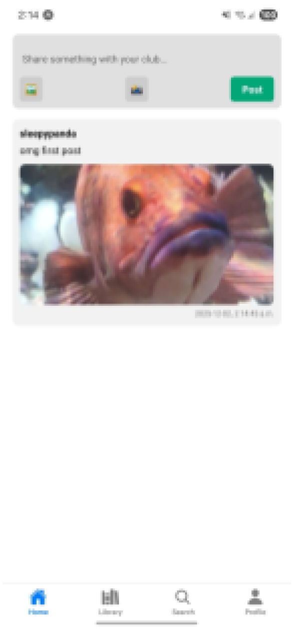
sleepypanda
Books Read: 21
Book Shelves: 4
Book Club: fake club
Books Recommended: 0

Edit Username
Change Profile Picture

Book Club
You are in: fake club
Leave Club

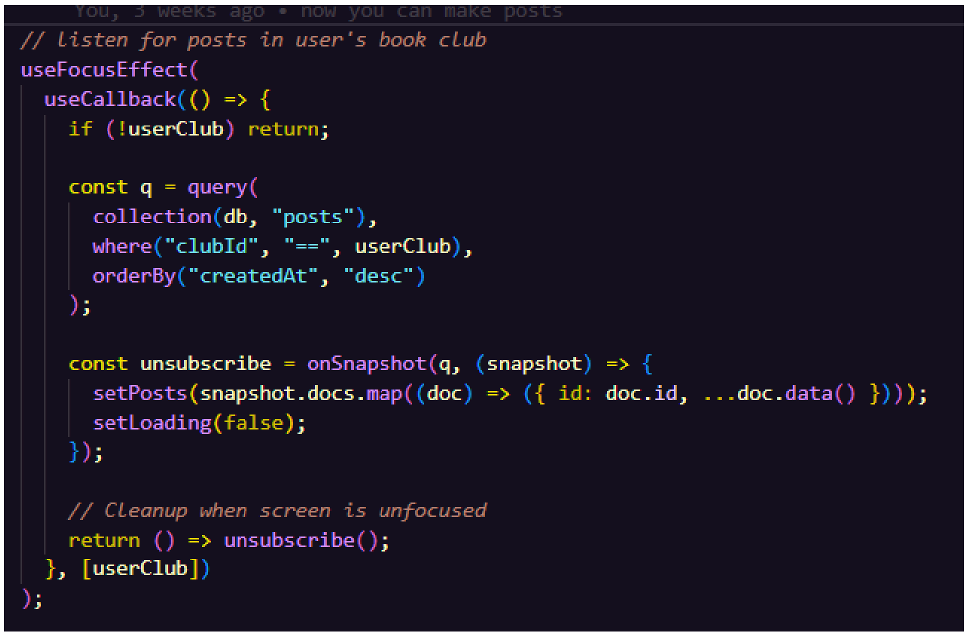
Alert
Created and joined fake club!
OK

FAVORITE BOOKS
Omniscient Reader's



Share something with your club...
Post

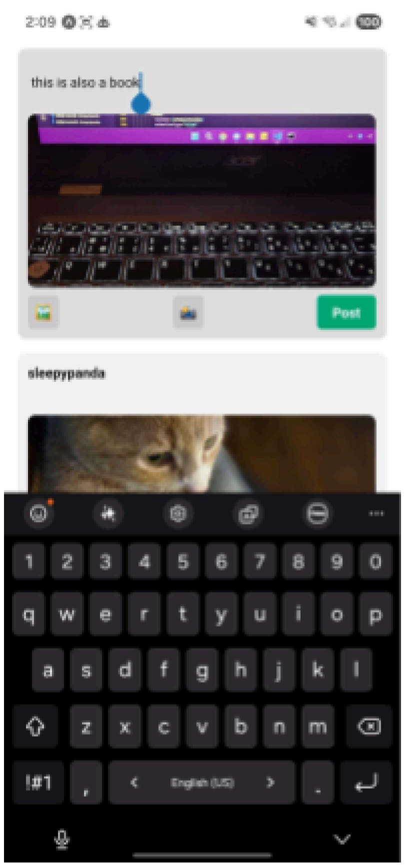
sleepypanda
omg first post

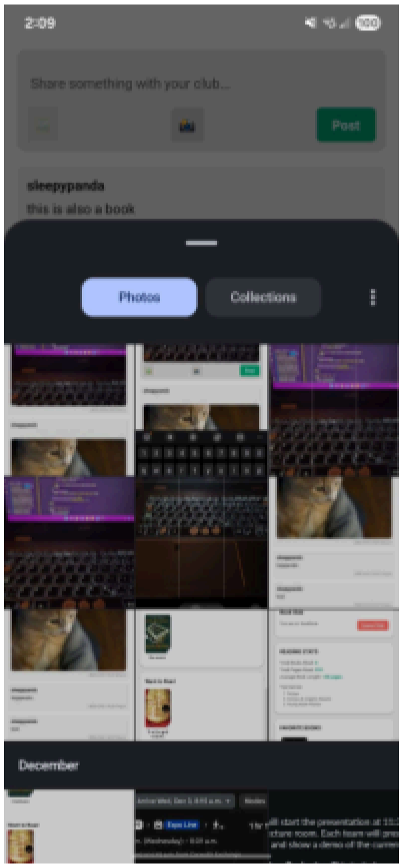


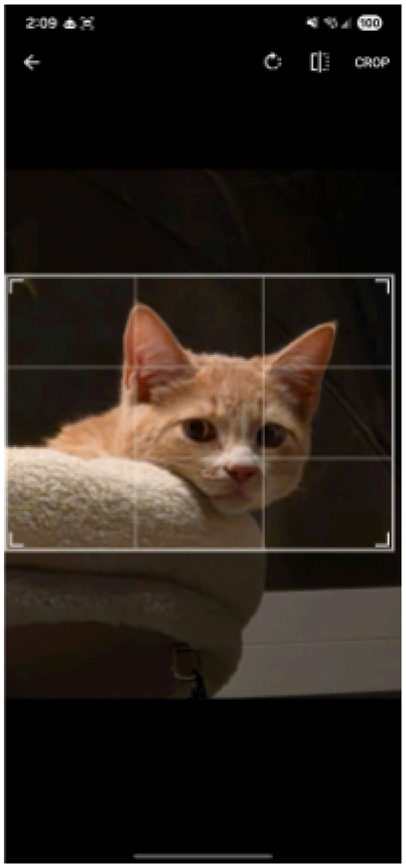
```
        You, 3 weeks ago • now you can make posts
// listen for posts in user's book club
useFocusEffect(
  useCallback(() => {
    if (!userClub) return;

    const q = query(
      collection(db, "posts"),
      where("clubId", "==", userClub),
      orderBy("createdAt", "desc")
    );

    const unsubscribe = onSnapshot(q, (snapshot) => {
      setPosts(snapshot.docs.map((doc) => ({ id: doc.id, ...doc.data() })));
      setLoading(false);
    });

    // Cleanup when screen is unfocused
    return () => unsubscribe();
  }, [userClub])
);
```

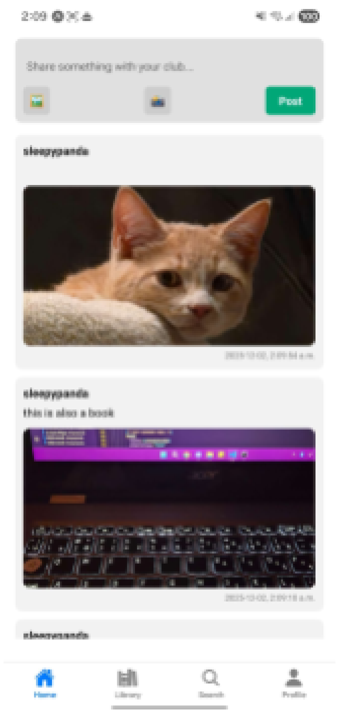
Post Creation:

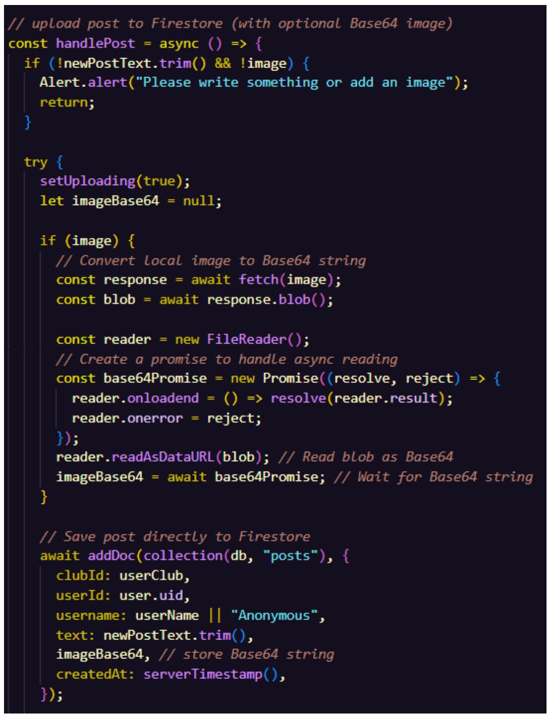
Users can create posts with text or photos using built-in camera access.







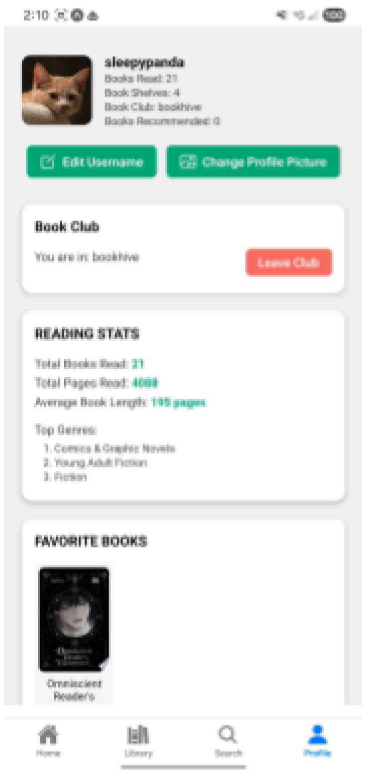




```
// upload post to Firestore (with optional Base64 image)
const handlePost = async () => {
  if (!newPostText.trim() && !image) {
    Alert.alert("Please write something or add an image");
    return;
  }

  try {
    setUploading(true);
    let imageBase64 = null;

    if (image) {
      // Convert local image to Base64 string
      const response = await fetch(image);
      const blob = await response.blob();

      const reader = new FileReader();
      // Create a promise to handle async reading
      const base64Promise = new Promise((resolve, reject) => {
        reader.onloadend = () => resolve(reader.result);
        reader.onerror = reject;
      });
      reader.readAsDataURL(blob); // Read blob as Base64
      imageBase64 = await base64Promise; // Wait for Base64 string
    }

    // Save post directly to Firestore
    await addDoc(collection(db, "posts"), {
      clubId: userClub,
      userId: user.uid,
      username: userName || "Anonymous",
      text: newPostText.trim(),
      imageBase64, // store Base64 string
      createdAt: serverTimestamp(),
    });
```

Profile Page:

Displays user stats and favorite books for a personalized profile.


2:10

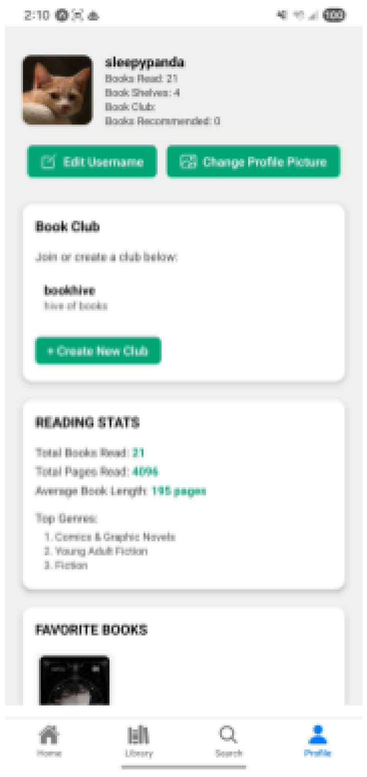
sleepypanda
Books Read: 21
Book Shelves: 4
Book Club: bookhive
Books Recommended: 0

Edit Username | Change Profile Picture

Book Club
You are in: bookhive
Leave Club

READING STATS
Total Books Read: 21
Total Pages Read: 4088
Average Book Length: 195 pages
Top Genres:
1. Comics & Graphic Novels
2. Young Adult Fiction
3. Fiction

FAVORITE BOOKS
Omniscient Reader's


2:10

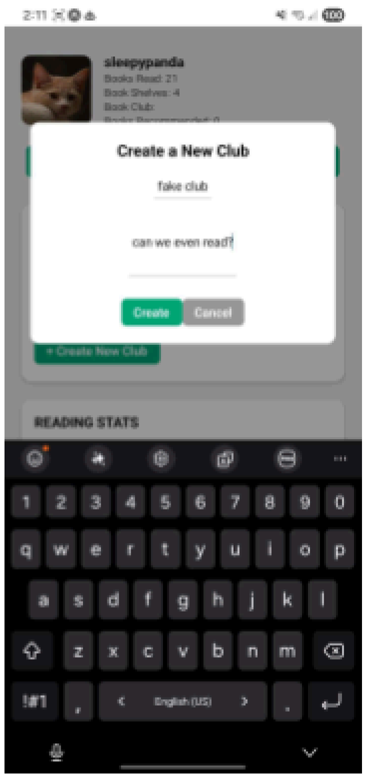
sleepypanda
Books Read: 21
Book Shelves: 4
Book Club:
Books Recommended: 0

Edit Username | Change Profile Picture

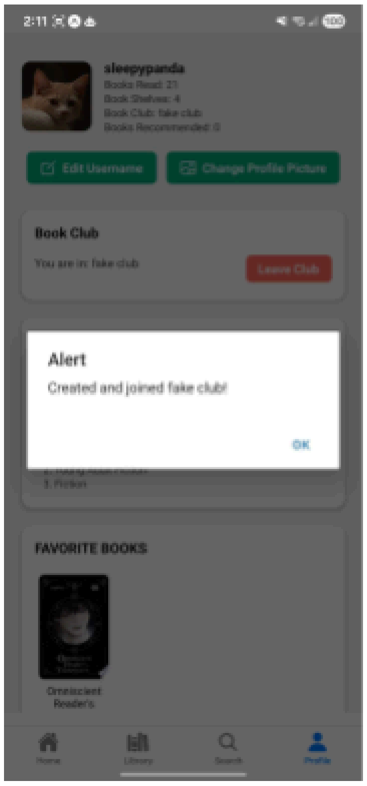
Book Club
Join or create a club below:
bookhive
hive of books
+ Create New Club

READING STATS
Total Books Read: 21
Total Pages Read: 4096
Average Book Length: 195 pages
Top Genres:
1. Comics & Graphic Novels
2. Young Adult Fiction
3. Fiction

FAVORITE BOOKS


2:11

Create a New Club
fake club
can we even read?
Create | Cancel

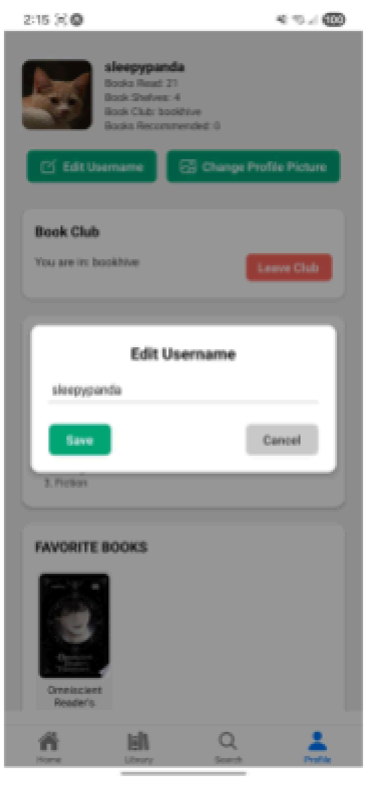

2:11

sleepypanda
Books Read: 21
Book Shelves: 4
Book Club: fake club
Books Recommended: 0

Book Club
You are in: fake club

Alert
Created and joined fake club!
OK

FAVORITE BOOKS
Omniscient Reader's

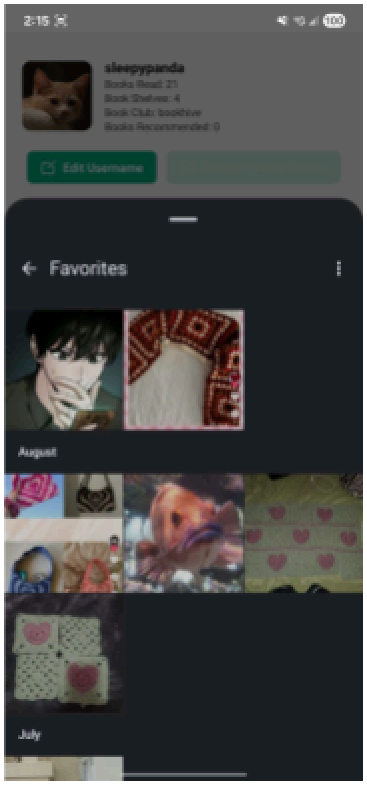

2:15

sleepypanda
Books Read: 21
Book Shelves: 4
Book Club: bookhive
Books Recommended: 0

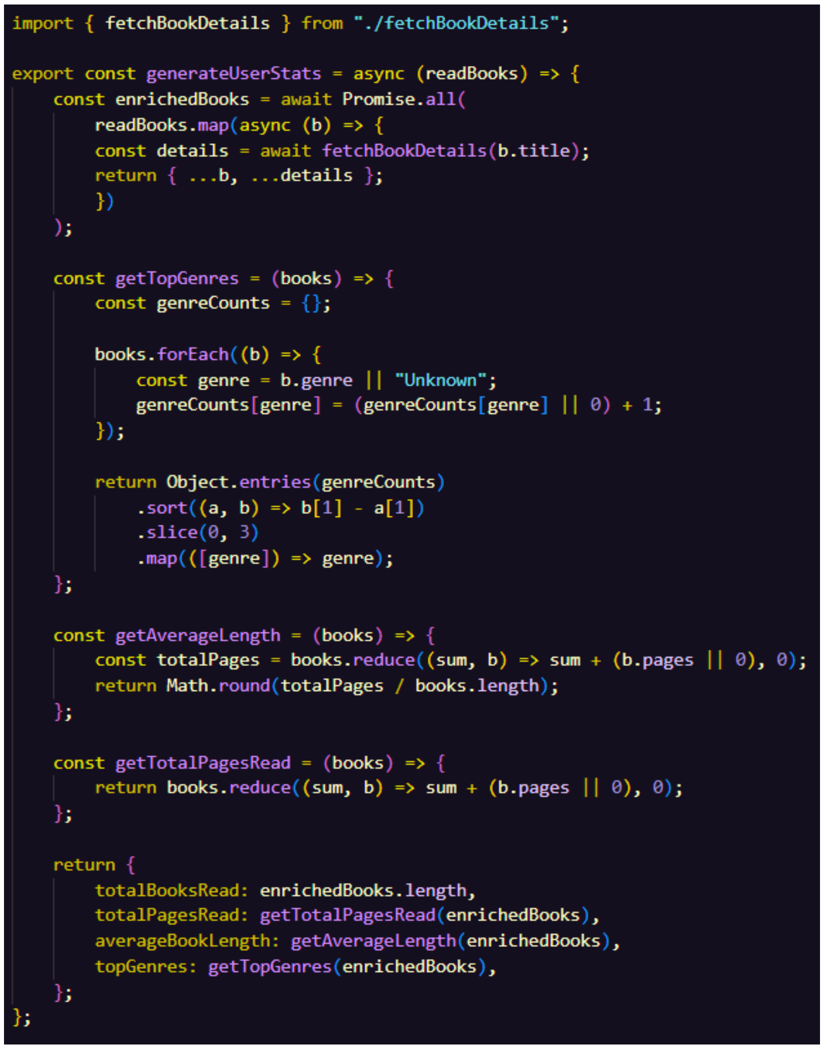
Book Club
You are in: bookhive
Leave Club

Edit Username
sleepypanda
Save | Cancel

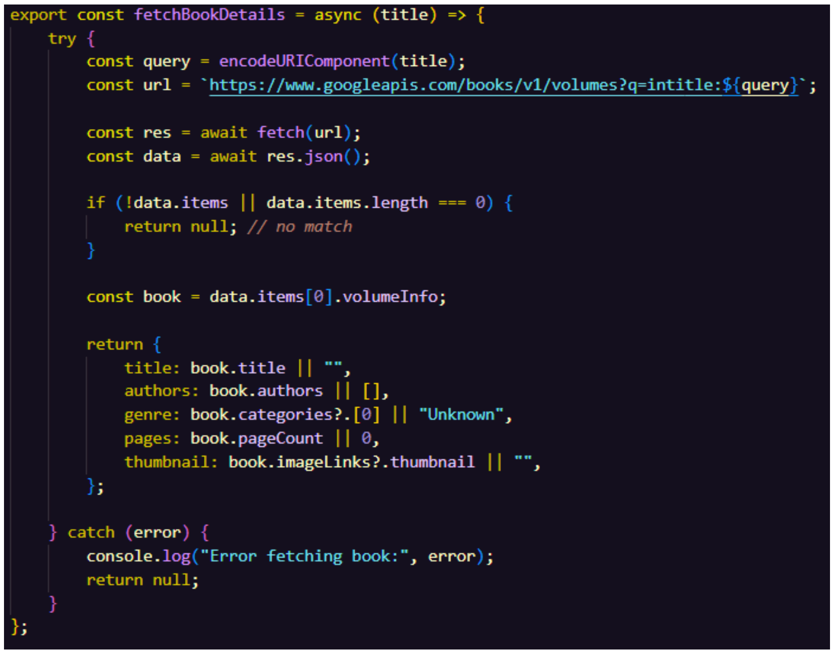
FAVORITE BOOKS
Omniscient Reader's


2:15

sleepypanda
Books Read: 21
Book Shelves: 4
Book Club: bookhive
Books Recommended: 0



```
import { fetchBookDetails } from "./fetchBookDetails";

export const generateUserStats = async (readBooks) => {
    const enrichedBooks = await Promise.all(
        readBooks.map(async (b) => {
        const details = await fetchBookDetails(b.title);
        return { ...b, ...details };
        })
    );

    const getTopGenres = (books) => {
        const genreCounts = {};

        books.forEach((b) => {
            const genre = b.genre || "Unknown";
            genreCounts[genre] = (genreCounts[genre] || 0) + 1;
        });

        return Object.entries(genreCounts)
            .sort((a, b) => b[1] - a[1])
            .slice(0, 3)
            .map(([genre]) => genre);
    };

    const getAverageLength = (books) => {
        const totalPages = books.reduce((sum, b) => sum + (b.pages || 0), 0);
        return Math.round(totalPages / books.length);
    };

    const getTotalPagesRead = (books) => {
        return books.reduce((sum, b) => sum + (b.pages || 0), 0);
    };

    return {
        totalBooksRead: enrichedBooks.length,
        totalPagesRead: getTotalPagesRead(enrichedBooks),
        averageBookLength: getAverageLength(enrichedBooks),
        topGenres: getTopGenres(enrichedBooks),
    };
};
```



```
export const fetchBookDetails = async (title) => {
    try {
        const query = encodeURIComponent(title);
        const url = `https://www.googleapis.com/books/v1/volumes?q=intitle:${query}`;

        const res = await fetch(url);
        const data = await res.json();

        if (!data.items || data.items.length === 0) {
            return null; // no match
        }

        const book = data.items[0].volumeInfo;

        return {
            title: book.title || "",
            authors: book.authors || [],
            genre: book.categories?.[0] || "Unknown",
            pages: book.pageCount || 0,
            thumbnail: book.imageLinks?.thumbnail || "",
        };

    } catch (error) {
        console.log("Error fetching book:", error);
        return null;
    }
};
```

Comparison to Original Idea:

- User Authentication: Complete
  - Secure login and account creation through Firebase.
- Book Search & Metadata Integration: Complete
  - Google Books API integration provides title, author, and description.
- Interactive Reading Lists: Complete
  - Lists visible on the Library screen, categorized by reading status.
- Social Feed: Complete
  - Users can view and interact with posts from friends.
- Post Creation (Text & Images): Complete
  - Users can create posts, including photo attachments via camera.
- Recommendations & Annotations: Partially Complete
  - Users can post thoughts and notes, but detailed annotation features are still under development.
- Progress Tracking: Incomplete
  - Reading progress is not yet displayed in lists, and advanced visual indicators like “library cards” are not yet implemented.
- Profile Page: Complete
  - Displays user stats and favorite books.

Overall, the majority of the core functionalities outlined in the original proposal have been implemented, establishing the foundation for social interaction, reading tracking, and personalized user experiences.

Future Additions:

Looking ahead, several features could enhance the app’s social and personalization aspects:

- Public User Profiles: Allowing users to make their profiles visible to others would encourage community engagement and discovery of shared reading interests.
- Public Bookshelves: Users could create and share bookshelves with the wider community, making it easier to recommend collections of books or find inspiration from others.
- Sending Recommendations: Introducing both general and user-specific book recommendations would provide a more interactive and tailored reading experience.
- Progress Tracking: Expanding the display of reading progress, including visual indicators for completion percentages or milestones, would help users monitor their own reading and share progress with friends.
- Shelf Management: Currently, users cannot add or remove shelves, nor remove books once added. Enabling modification of shelves, including editing, adding, or removing books, would give users more control over their “Currently Reading” and “Want to Read” lists and improve overall usability.

Implementing these features would strengthen the app's social connectivity, personalization, and overall user experience.

Challenges:

During development, several technical challenges arose, the first being image storage in Firebase. Initially, attempts to upload raw image files directly to Firebase caused inconsistent rendering and high storage usage. This was resolved by converting images into Base64-encoded strings before storage, allowing reliable upload, retrieval, and display within the app. This approach also simplified data handling by keeping image information within standard database entries rather than managing separate file storage.

The second general challenge was generating User Stats, as integrating the Google Books API to dynamically generate user reading statistics proved challenging. Retrieving and processing book metadata in a way that could support accurate tracking of completed books, genres, and progress required careful handling of API responses and database synchronization.

Reflection:

Developing this app has been a rewarding and sometimes challenging experience. It was exciting to see an idea move from a concept on paper to a functioning application where users can track books, share posts, and interact with friends. Working on the social and reading-focused features gave me a chance to think carefully about what makes an app engaging and intuitive, from the layout of the library screen to the way posts are displayed in the feed.

At the same time, I learned a lot about the technical side of app development. Integrating Firebase for authentication and image storage, as well as working with the Google Books API, pushed me to problem-solve in ways I hadn't before. There were moments of frustration, particularly around image handling and synchronizing external data with user stats, but figuring out solutions for these issues was incredibly satisfying and taught me the importance of planning for both functionality and user experience.

Overall, this project strengthened my skills in React Native, cloud-based storage, and API integration, while also giving me insight into how social and interactive features can enhance a reading app. It has been a great learning journey, and I'm excited about the potential improvements and new features that could be added in the future. I will definitely continue working on this project even after this class ends because I genuinely look forward to being able to use it someday in the future!