Software Engineering Lab

Implementation Report II - Build II

Library Book Management App

Palak Singhal (16CO129)

Sharanya Kamath (16CO140)

March 30, 2018

1. Basic Information

This Library management app is an Android app developed in Android Studio with Intellij IDEA. The programming language used is Java. The front end is developed using xml file format. Java Sdk and Android APIs are extensively used for backend development. SQLite database is used for storing user data and book information. Prerequisite for running this app is Android Studio 3.0 with API 27.

2. Pending / New Functional Requirements (Implemented)

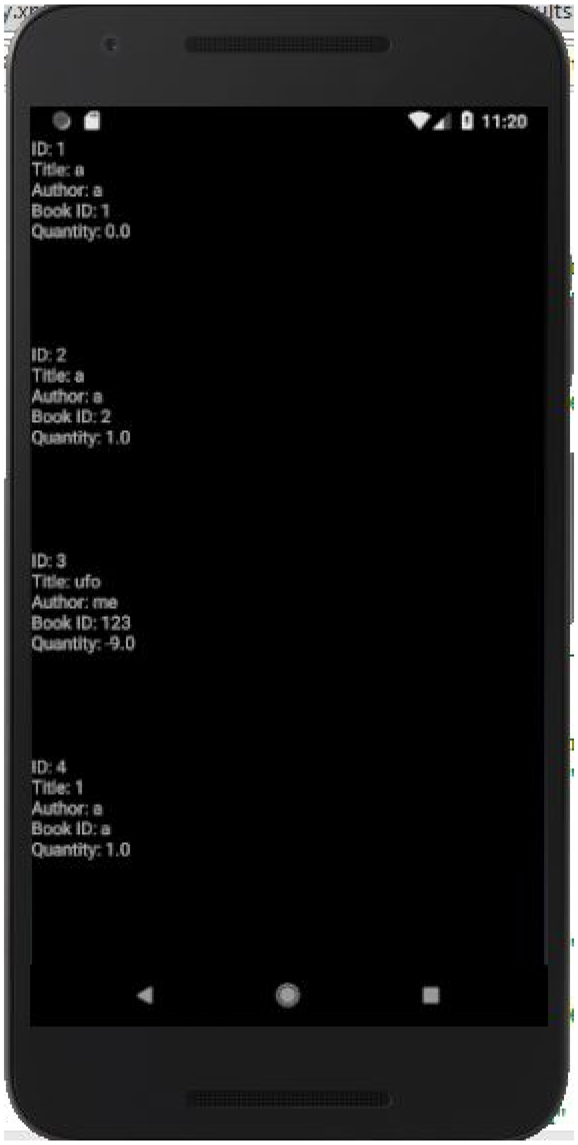
FRID	NAME	Type (Pending/New)	Description
1	Increase Quantity of book by adding existing book to the database	New	Admin can add an already existing book to the database. The quantity of the book will increase in the database.
2	Issue Book	Pending	If the quantity of the book entered by the student to issue is not greater than the quantity available, then student can click on the issue button which would decrease the book count in the database by the quantity and number of books issued by student will increase. The student can go and collect the book from the library anytime.
3	Search window and Book Details	Pending	The search window shows all the books available with their details such as Author, ISBN No., Quantity available etc.
4	Book Not Available	New	If Quantity of the book in the database is less than the quantity entered by the user to issue, it will show “Book Not Available” error message.

3. Functional Requirements (Not Implemented)

FRID	NAME	Description	Why Not Implemented in this Build?
1	Bookmark	If the book requested by the student is currently unavailable then a bookmark option should be provided upon clicking which the book name is stored under the bookmarked books.	Could not debug a minor error which caused the app to stop everytime we tried to run it , hence had to remove the feature for now.
2	Notification button	A button which would display all the important notifications like if the book bookmarked by the student becomes available or if the fine date is approaching etc.	Notification button could have been made but before we implement bookmark feature it isn't of any use.
3	Fine	If the book has not been renewed within given period of time an algorithm to increment the fine each day according to the number of books issued will be calculated and displayed in the user profile.	Could not figure out proper algorithm due to lack of time.

4. Screenshots of Functional Requirements Implemented in this Build

4.1 Inventory of books with their quantity



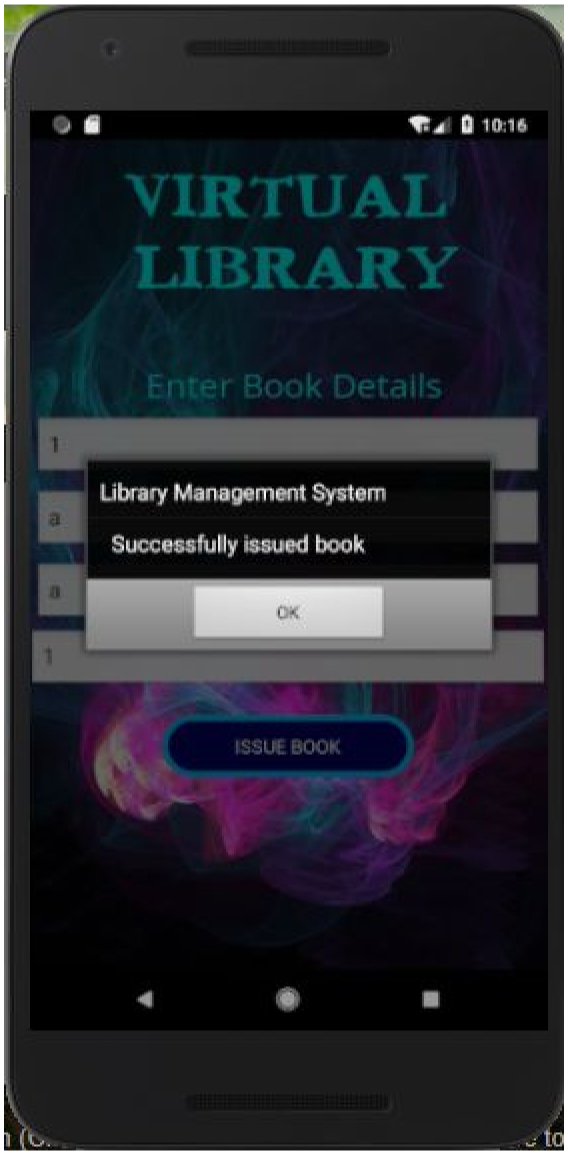
```
ID: 1
Title: a
Author: a
Book ID: 1
Quantity: 0.0

ID: 2
Title: a
Author: a
Book ID: 2
Quantity: 1.0

ID: 3
Title: ufo
Author: me
Book ID: 123
Quantity: -9.0

ID: 4
Title: 1
Author: a
Book ID: a
Quantity: 1.0
```

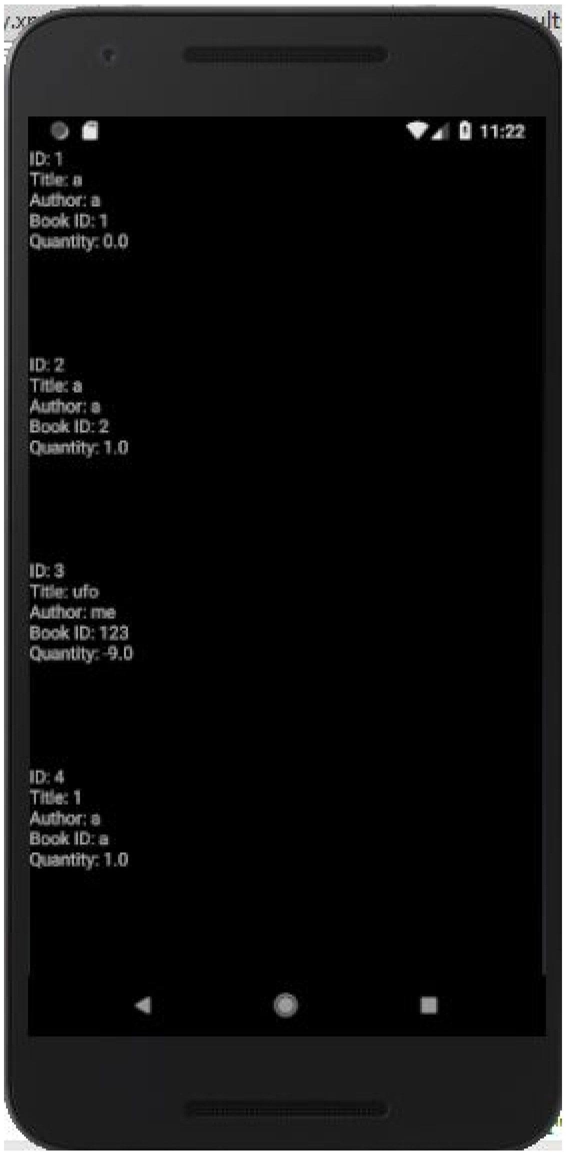
4.2 Successfully Issued Book



Library Management System

Successfully issued book

4.3 Search Window and Book Details



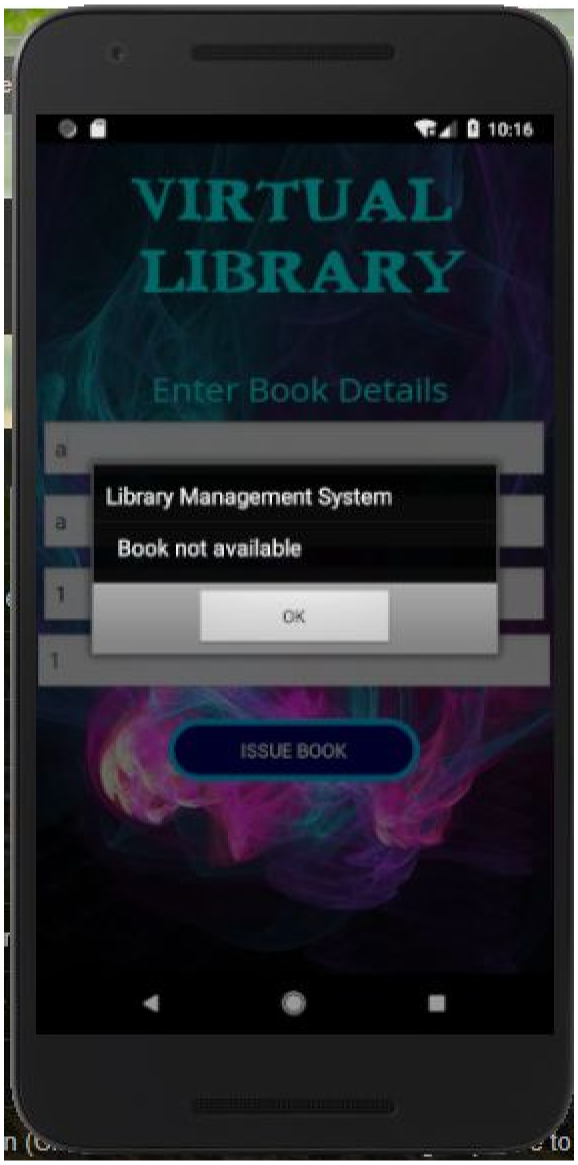
```
11:22
ID: 1
Title: a
Author: a
Book ID: 1
Quantity: 0.0

ID: 2
Title: a
Author: a
Book ID: 2
Quantity: 1.0

ID: 3
Title: ufo
Author: me
Book ID: 123
Quantity: -9.0

ID: 4
Title: 1
Author: a
Book ID: a
Quantity: 1.0
```

4.4 Book Not Available



Library Management System

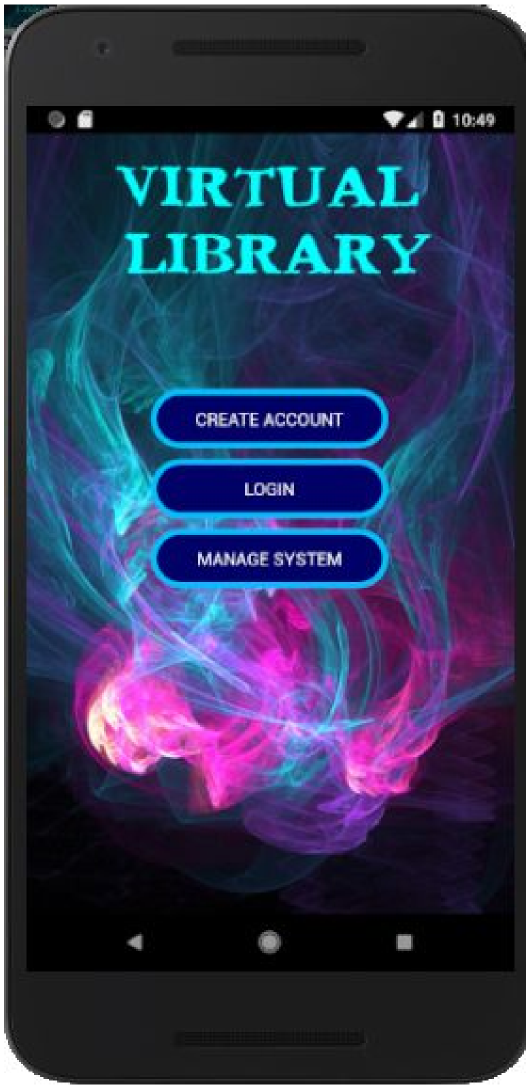
Book not available

5. Non-functional Requirements Considered

NFRID	NAME	Description
1.	Attractive UI	User interface will be made more appealing and attractive.

6. Screenshots of Non-Functional Requirements

6.1 Attractive User Interface



7. Summary of Test Plan

The functional requirements which should be tested are

- Adding books to the database by the admin(Checking whether books get displayed in the inventory).
- Updating book count .(Checking whether book count is updated)
- Issue book feature(Checking whether the count gets decreased after issuing a book)
- Displaying bookmark option in case of unavailability of book.
- Displaying all details correctly when user searches for a book.

The functional requirements which need not be tested are

- Admin login
- User login
- User signup

The non-functional requirements which should be tested are

- Reliability (Used SQLite which queries faster)
- Screen Resolution and Screen Form Factors
- CPU Memory
- Network Conditions

The non-functional requirements which need not be tested are

- Durability
- Usability
- Interrupts from other apps

Types of testing:

Unit Testing

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

Integration Testing

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

Functional Testing

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

System Testing

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

Performance Testing

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

Usability Testing

Usability testing is performed to the perspective of the client, to evaluate how the GUI is user-friendly? How easily can the client learn? After learning how to use, how proficiently can the client perform? How pleasing is it to use its design? This falls under the class of black box testing.

Acceptance Testing

Acceptance testing is often done by the customer to ensure that the delivered product meets the requirements and works as the customer expected. It falls under the class of black box testing.

<u>Testing Tools:</u>

We are not planning to use any automated testing tools.

<u>Expected Outcome of Testing:</u>

Through testing, we expect to detect loopholes in our application. Correcting these loopholes will make our app more secure and reliable.

8. Summary

This report focuses on the work done during the second build of the “LIBRARY BOOK MANAGEMENT APP”. In the first build more focus was on implementing the admin features as compared to the student features. Most of the admin features like logging in as admin, adding the books to the book database, viewing the registered users, viewing the inventory of books present in the library, were completed in the first build. However only few student functionalities like logging in, signing up were implemented. In this build some admin features like adding more copies of the same book and updating that in the database were implemented, and some student features like searching for a book in the database, issuing a book and simultaneously updating the database were implemented. The student can also view all the books present in the database. We highly improved upon some non functional requirements like user interface and made it more attractive and secure. Some features like bookmark, notification button and fine calculation could not be done due to lack of time and expertise.