
Software Requirements Specification for



Prepared by
**Chris Colavita
Claude Daniel
Lucas D'Avila
Patrapee Pongtana**

12/01/2017

Table of Contents

Introduction	3
Purpose	3
Intended Audience	3
Product Scope	3
Overall Description	5
Product Perspective	5
Product Functions	5
User Classes and Characteristics	5
Operating Environment	5
Design and Implementation Constraints	5
User Documentation	6
Assumptions and Dependencies	6
External Interface Requirements	6
User Interfaces	6
Hardware Interfaces	9
Software Interfaces	9
Communications Interfaces	9
System Features	9
Account Recovery	9
Log into Account	10
Search for Friend	10
Search for Book	11
Add Friend	11
Add Book	12
View Book	12
Instant Messaging	12
Request to Borrow	13
Accept/Deny Book Request	13
Return Book	14
Other Nonfunctional Requirements	14
Performance Requirements	14
Safety Requirements	14
Security and Privacy Requirements	14

Software Quality Attributes	15
Business Rules	15
Other Requirements	16

Revision History

Name	Date	Reason For Changes	Version
Initial Specifications	10/30/2017	N/A	0.5
Specification 0.75	12/01/2017	Updates	0.75

1. Introduction

1.1 Purpose

We buy books when someone we know may already have it and are willing to lend it. In such an instance, we may be wasting money on a book we could have simply borrowed from a friend or an acquaintance.

Our system will help mitigate such a scenario and help a user to first check if someone he or she knows already has the book. This program is an application for smartphones that allows users to share books with their friends.

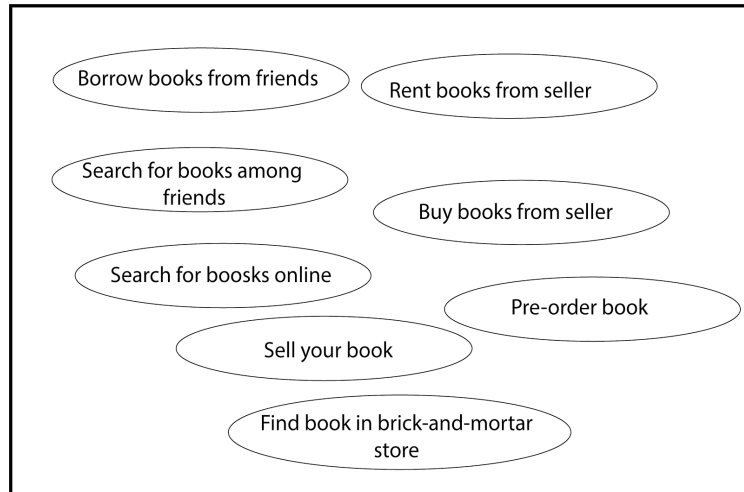
1.2 Intended Audience

This application is intended for anyone that reads physical books. We expect the vast majority of users to be college students - given the cost of college textbooks and temporary nature of their use. However, any book can be shared using this application and the general population might find this application useful when they are looking for a specific book.

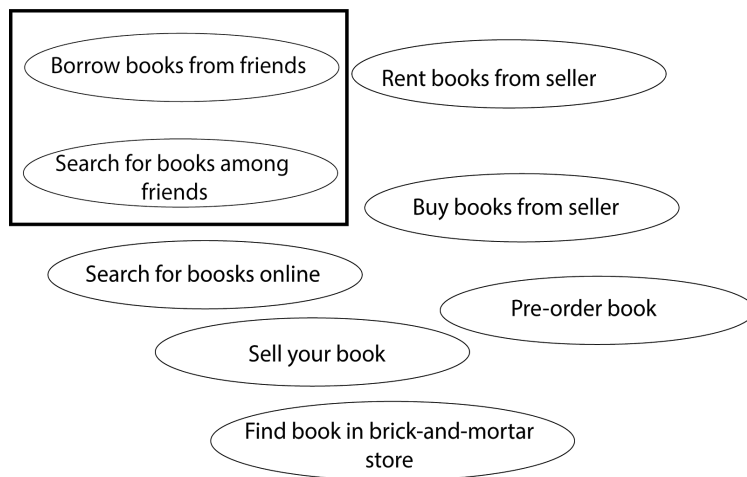
1.3 Product Scope

This system will be in the overall book purchasing and lending domain. This is a large domain with entire sub-industries built around it.

The broad scope of the system will have several diverse fields within the books industry, as shown:



Our system will be of a narrowly defined scope within the overall system, involving only looking for and borrowing books among friends:



This system will help solve the problem defined above. Users will be able to create accounts and add all the books they have into their account. They can mark each of their books as ‘available to lend’, whenever they want. They will then be able to add other users as ‘friends’, hence showing the user all the books the ‘friend’ has available to lend. The user can then request to borrow the book.

Users will be able to go into their app and search for a particular book among their friends. Or they can choose a friends and see all the books that friend has available to lend. Once a request is sent to borrow a book, the receiving user must then approve or reject the request.

A messaging system will also be implemented to allow users to communicate with each other and determine, for example, how long a friend can keep a borrowed book.

2. Overall Description

2.1 Product Perspective

The system is a new self-contained android application. It originates from the perspective of a professor and students who are constantly acquiring new books. These books are expensive, sometimes costing several hundred dollars. This product is designed to eliminate these costs.

2.2 Product Functions

Users will be able to...

- Create an account in *Borrow Me*
- Add books to the list of books that they own
- Search and add friends who also have a *Borrow Me* account
- Search for books to borrow from the user's friends
- Request to borrow a book from a friend
- Accept/Deny borrow requests
- Message other users

2.3 User Classes and Characteristics

The domain of users is any group of friends that wish to share books. The majority of users will be students who wish to share textbooks with friends. They will access the program typically at the beginning of a class semester to borrow textbooks and after the semester ends to return textbooks.

2.4 Operating Environment

The hardware platform will be an Android smartphone. It will require the operating system version of KitKat 4.4 or above. The only additional software components will be Firebase database.

2.5 Design and Implementation Constraints

- The project is under a 5 week time constraint. It must be finished before the end of the semester, December 15th, 2017.
- The project members have other class and work obligations that take away time from the project.
- In person meetings will be mostly limited to class sessions on Tuesday and Thursday nights.
- No project members have prior knowledge of Firebase and must learn to use it relatively bug free within the time constraints.

- Currently programming is tested on an android emulator using Android Studio. Getting the system to work on a physical phone may pose problems.
- Establishing an internal message system.

2.6 User Documentation

- User manual detailing basic usage of the system
- Documentation of the classes likely in form of Javadoc's

2.7 Assumptions and Dependencies

- We are dependent on creating an internal message system. An external system is not currently allowed.
- We are dependent on the free version of Firebase and its constraints such as data limits.
- We assume that most users will be students. It is possible that the system will be dependent on being tied into a school system.
- We assume that users will share legal versions of books.

3. External Interface Requirements

3.1 User Interfaces

The user will first open the login page. There they can choose to create a new account, login, or recover/ reset their login information. Upon logging in, they will be directed to their home page.

The home page will indicate the username at the top with an option to logout. The center of the page will contain a list of the all the currently borrowed books and lent out books. The bottom of the home page will the following buttons: Search for User, Search for Book, Add Book, Remove book, Messages. Each page will have a back button at the top left to return to the home page.

Both of the search pages will contain a search bar at the top to enter the book information or username to search for. The list will below this with the option to borrow a book or add a friend respectively.

Whenever looking at a list of books, there is the option to click the book title to display information about the book. It will display the title, author(s), publisher, owner, edition, and year published.

The Add Book page will be several text input fields for book information. There will be two buttons at the bottom: Add, Cancel. The Remove Book page will be a list of the user's owned books with a button to remove it from the user's list.

The Message page will have a list of the user's friends. The user can click the user to view all the previous messages and send a new one.

The main stages of the app and the user interfaces during those stages are shown below :

Left Screenshot (Login/Sign-up):

- Logo: borrow me
- Fields: username, password
- Buttons: sign in, sign up

Right Screenshot (User Profile):

- Logo: borrow me
- Message: Welcome user1!
- Buttons: My books, My friends
- Book Entries: book 1, book 2, book 3 (rented)
- Bottom Bar: profile, search, add book, add friend, message

The login and sign up screens would be very similar - users enter their username and password if they already have an account or choose a new username and password if they are new users. They are then taken to the profile page where they see the books they have.


Left Screenshot (User Profile):

- Logo: borrow me
- Message: Welcome user1!
- Buttons: My books, My friends
- Friend Entries: friend 1, friend 2, friend 3
- Bottom Bar: profile, search, add book, add friend, message

Right Screenshot (Friend's Page):


- Logo: borrow me
- Message: friend1's page
- Book Entries: book 1, book 2, book 3 (rented), book 4 (rented)
- Bottom Bar: profile, search, add book, add friend, message

The user can click on a friend from to look at all the friend's books.



Search books:

profile	search	add book	add friend	message
---------	--------	----------	------------	---------



add book

name


ISBN

author

edition

profile	search	add book	add friend	message
---------	--------	----------	------------	---------

The user can search for books among friends. On the screen on the right, the user adds a book he or she owns to his or her profile.




add friend

name or username

user 1	add friend
user 2	add friend
user 3	add friend

profile	search	add book	add friend	message
---------	--------	----------	------------	---------



chat with friend1

you

friend1

you

friend1

type message

profile	search	add book	add friend	message
---------	--------	----------	------------	---------

On these screens, the user can add friends and chat with available friends.

3.2 Hardware Interfaces

- Android Operating System
- Implement or emulate a touch screen for data input
- Functional on Android tablets and Android smartphones

3.3 Software Interfaces

- The product will be connecting with Firebase to store data from users.
- The communication between Borrow Me and Firebase will be concurrent style, meaning the data will be update in real time.
- Borrow Me will be on Android platform only.
- Java will be used to implement Borrow me
- Account registration will require email and password

3.4 Communications Interfaces

- Email are required in prior to use the application
- No browser is required
- Firebase is the main environment for users to manage their data
- Users' data will be encrypted by Firebase
- The data transfer rates will be synchronized in real time

4. System Features

4.1 Account Recovery

4.1.1 Description and Priority

Medium Priority. The user will be able to recover their forgotten password. The user will need to know their username and email to then reset their password.

4.1.2 Stimulus/Response Sequences

User presses *Forgot Password?* button → System prompts user to enter your username and email

User enters invalid username or email → System prompts an error message

User enters valid username and email → System prompts user to create new password
→ User enters new password

4.1.3 Functional Requirements

REQ-1: Enter Valid Username

REQ-2: Enter Valid Email

4.2 Log into Account

4.2.1 Description and Priority

High Priority. The user will enter their account information to login and access their account at the *Login* page.

4.2.2 Stimulus/ Response Sequences

User enters valid password and username → Take user to home page
User enters invalid password or username → Send error → Request valid information at login page

4.2.3 Functional Requirements

REQ-1: Validate username
REQ-2: Validate password

4.3 Search for Friend

4.3.1 Description and Priority

Medium Priority. The user will enter the username of another user on their friends list to look them up.

4.3.2 Stimulus/Response Sequences

User enters a username into search bar → Return list of possible matches

4.3.3 Functional Requirements

REQ-1: Multiple users in database
REQ-2: Search database for username

4.4 Search for Book

4.4.1 Description and Priority

Medium Priority. The user will enter the title, author, or ISBN to search for a book available to borrow from one of their friends. This will be done in the search menu and will search through all of the books available to be borrowed based on all of a user's friends.

4.4.2 Stimulus/Response Sequences

User enters a title into search bar → Return list of possible matches
User enters an author into search bar → Return list of possible matches

User enters an ISBN into search bar → Return list of possible matches

4.4.3 Functional Requirements

REQ-1: Multiple books in database

REQ-2: Search database for book title

REQ-3: Search database for book author

REQ-4: Search database for book ISBN

4.5 Add Friend

4.5.1 Description and Priority

High Priority. Users will be able to search for other users and request to add them as a friend. Those users in turn will then be able to accept or deny that friend request. Once two users are friends, they will be able to browse through each other's book selection and make requests to borrow those books.

4.5.2 Stimulus/Response Sequences

User searches for a friend to add by username → returns list of possible matches
→ user presses *Add Friend* button → potential friend receives *Friend Request* →
potential friend accepts/denies request

Potential friend accepts request → user gains access to friend's list of books

Potential friend denies request → user is unable to access other user's list of books

4.5.3 Functional Requirements

REQ-1: Multiple users in database

REQ-2: Search database for users to add

REQ-3: Add friend button

REQ-4: Accept/Deny friend request

4.6 Add Book

4.6.1 Description and Priority

High Priority. Users will be able to add books that they own at home onto their profile for their friends to browse and request to borrow. Once a book has been added, others will be able to see in real time. The user must include a book's title, author, and ISBN.

4.6.2 Stimulus/Response Sequences

User presses *Add Book* button → User gets prompted to enter book information →
User enters book title, author, and ISBN → User presses *Add* → book gets added to
user's profile

4.6.3 Functional Requirements

REQ-1: Add book button

REQ-2: Title field must not be *Null*

REQ-3: Author field must not be *Null*

REQ-4: ISBN field must not be *Null*

REQ-5: Add button can only be pressable when above fields are filled in

REQ-6: Adds book to database

4.7 View Book

4.7.1 Description and Priority

Low Priority. Users will be able to press on any book listed and view details on the book.

4.7.2 Stimulus/Response Sequences

User presses on a book → Details on the book appear

4.7.3 Functional Requirements

REQ-1: Books in database

4.8 Instant Messaging

4.8.1 Description and Priority

Medium Priority. Once a user becomes friends with another, they have the ability to message each other. Messages are located in their own tab and users can even view previous messages.

4.8.2 Stimulus/Response Sequences

User presses new message → User enters which contact to message → User types message and hits send → Contact receives message

4.8.3 Functional Requirements

REQ-1: Create new message

REQ-2: Search database for friends to message

REQ-3: Save messages in database

4.9 Request to Borrow

4.9.1 Description and Priority

High Priority. Users will be able to select a book that is available to borrow and then request to borrow it. Hitting the *Borrow* button will send a notification to the other user with the request. Unavailable books will be marked as so and the user will not be able to request it until it has been returned to the owner.

4.9.2 Stimulus/Response Sequences

User presses *BorrowMe* on the desired book → Contact receives notification for a book request → Book changes from *BorrowMe* to *Pending*

4.9.3 Functional Requirements

REQ-1: Book in database

REQ-2: Book must be available

REQ-3: *BorrowMe* button sends request to user

4.10 Accept/Deny Book Request

4.10.1 Description and Priority

High Priority. Once a user receives a book request he can then choose to accept or deny that request. If the user denies the request the book goes back to being available. If the user accepts the request the book changes to unavailable and a message begins between the two users to set up details to pick up the book.

4.10.2 Stimulus/Response Sequences

User receives book request → User accepts book request → book becomes unavailable → Opens a message between the two users

User receives book request → User denies book request → book becomes available

4.10.3 Functional Requirements

REQ-1: User must receive a request notification

REQ-2: When denied, the book should become available for other users

REQ-3: When accepted, book should become unavailable for other users

REQ-4: When accepted, a message between the two users should begin

4.11 Return Book

4.11.1 Description and Priority

High Priority. Once the user is done using the book he will go onto his profile, select the books borrowed tab, and be able to hit return. This should only be done once the user has actually returned the book to its owner. Then the owner will also receive a notification asking if the book was really returned or not. That user will then hit yes or no.

4.11.2 Stimulus/Response Sequences

User 1 hits *Return Book* → User 2 receives a notification for the returned book

User 2 hits yes → Book becomes available to be borrowed → User 1 receives a notification that the book has been successfully returned

User 2 hits no → Book remains unavailable → User 1 receives a notification that the book hasn't been returned

4.11.3 Functional Requirements

REQ-1: Books that User 1 are borrowing must have a *Return* button

REQ-2: Once *Return* has been pressed User 2 must receive a notification

REQ-3: User 1 must receive a notification based on if User 2 hits Yes or No

5. Other Nonfunctional Requirements

5.1 Performance Requirements

- Application
 - Boot time should be minimized. Preferably, the app should open up instantly, when the icon is clicked.
 - Navigating between the various screens and interfaces of the app should be seamless and lag-free.
- Searches/Insert
 - Inserting books should be seamless for the user. The user should be able to add as many books to his or her account without any noticeable lack in speed.
 - The search must be as quick as possible. Searching will be implemented using either binary search trees or the inbuilt SQL search function.
- Database
 - Should be updated as soon as an action is performed by a user, in real time.
 - When searching for a user, the search should access all of the users in the database with minimal visible lag in speed.

5.2 Safety Requirements

A potential harm or damage that could occur through the use of this system is the theft or damage of books. Borrow Me assumes no responsibility when it comes to the safe return of lent books. This application assumes that users already know each other outside of the app ecosystem - and so users are assumed to be making informed decisions on whom to lend to.

The duration for which a book is borrowed is similarly left to the users to decide, based on the assumption that users know each other.

Also, no private information other than first and last name will be available for display. No email address or any other contact information about a user is shared - even with friends on the app. This provides a further level of safety.

5.3 Security and Privacy Requirements

- The login screen ensures that a user's account is protected from intruders. A user will have to enter the username and the password in order to access the account. In case the password is forgotten, it can be retrieved using a retrieval system with security questions.

- The messaging system is integrated into the application. This ensures that user interactions are not exposed to external systems outside of this app.
- A user will only be able to interact with friends that have already accepted the request. This includes messaging, looking at available books or requesting to borrow a book.
- Cannot share PDFs or other formats that violate copyright laws.
- The username and password should be encrypted for extra security during transmission.
- Information should only be shared between ‘friends’. No interaction whatsoever can take place until a friend request is accepted and a ‘friends’ relationship is established between the two users.
- Non-friends will only be able to see a user’s username and name upon searching for list of people.
- When searching for a book, the search should access all of their friend’s and what books they own in the database. Only friends should be searched. The system must ensure that no books outside of the immediate friends list.

5.4 Software Quality Attributes

- Usability
 - Must be easy to use for users that are not tech savvy..
 - GUI will be kept simple and robust, with a minimalist design.
- Accessibility
 - Everyone will have the same access within the application. In other words there will be no ‘admin access’ within the app itself.
 - Further updates to the app could include options to increase the font size to accommodate the visually impaired.
- Portability
 - As an android app, this software will be portable on smartphones running on the android operating system. Future releases could include a iOS version for Apple’s iPhones and iPads.
- Availability
 - The release version of this application will only be released on the android platform.

- Further releases in the future could include releases on iOS, MacOS, and Windows. Once released on multiple platforms, cross-functionality will be implemented among all the platforms such that users using two different systems will still be able to add each other as friends.

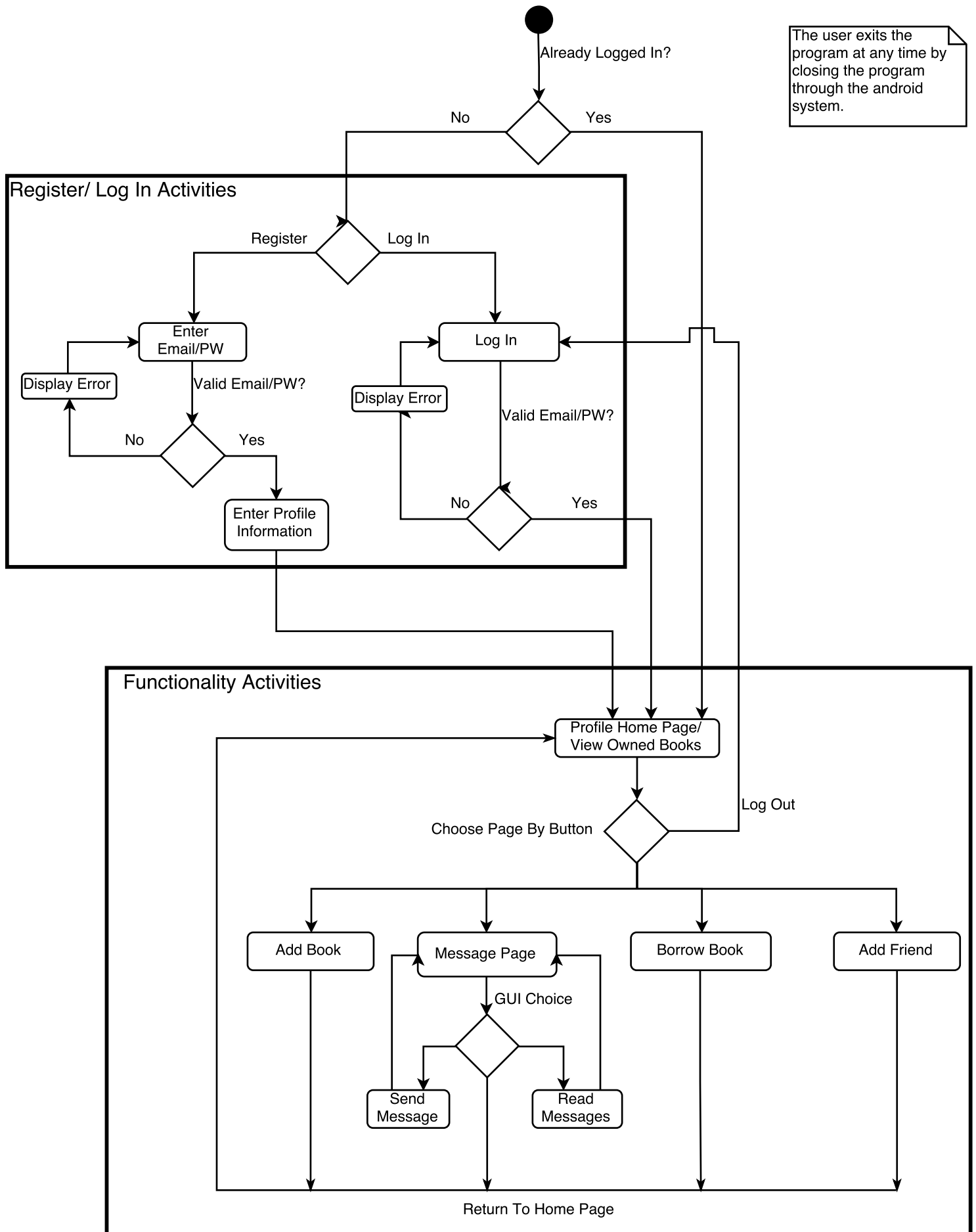
5.5 Business Rules

- Users should only be able to see friends' books that are listed as 'available to lend'.
- A user has to search for a friend using the friend's username, or first and last name.
- A friend request must be accepted before any communication between the users can take place.
- The duration for which a book is borrowed is left for the users to work out amongst themselves. This app assumes that users know each other in real life - we only display the books they have.
- We will make sure that users understand that we assume no responsibility for stolen or damaged books as a result of using this app to borrow or lend books.

6. Other Requirements

- When searching for a user, the search should access all of the users in the database, without constricting the search to specific groups of users.
- The database must be an external system that doesn't rely on the Android framework. This ensures seamless integrations when the application is released on iOS and other platforms in the future.
- The database must ensure that different editions of the same book can be added as separate entities. Users may require a specific edition of a specific book. In turn, users should have the option to choose for a specific edition or all editions of a specific book.

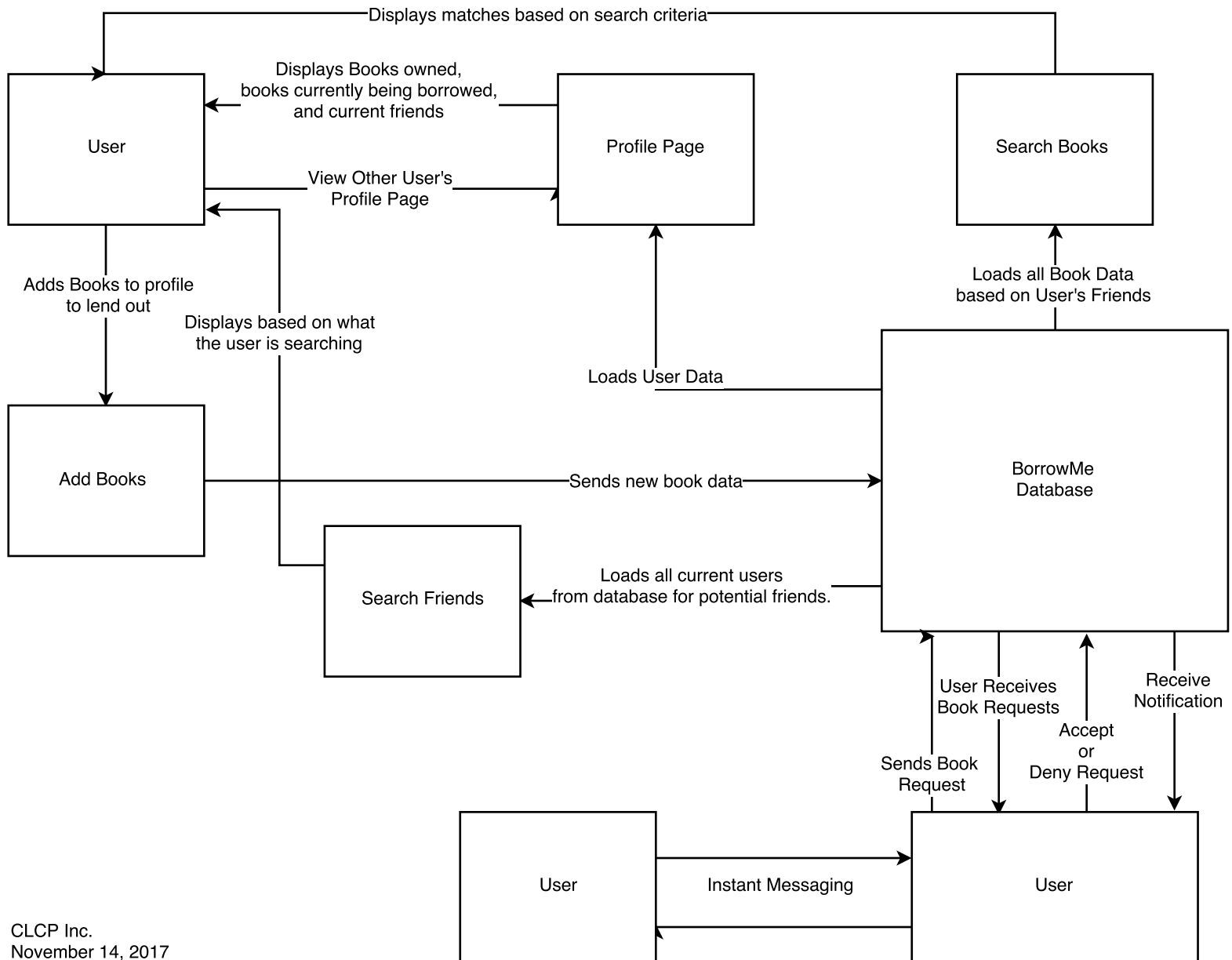
BorrowMe Activity Diagram



BorrowMe

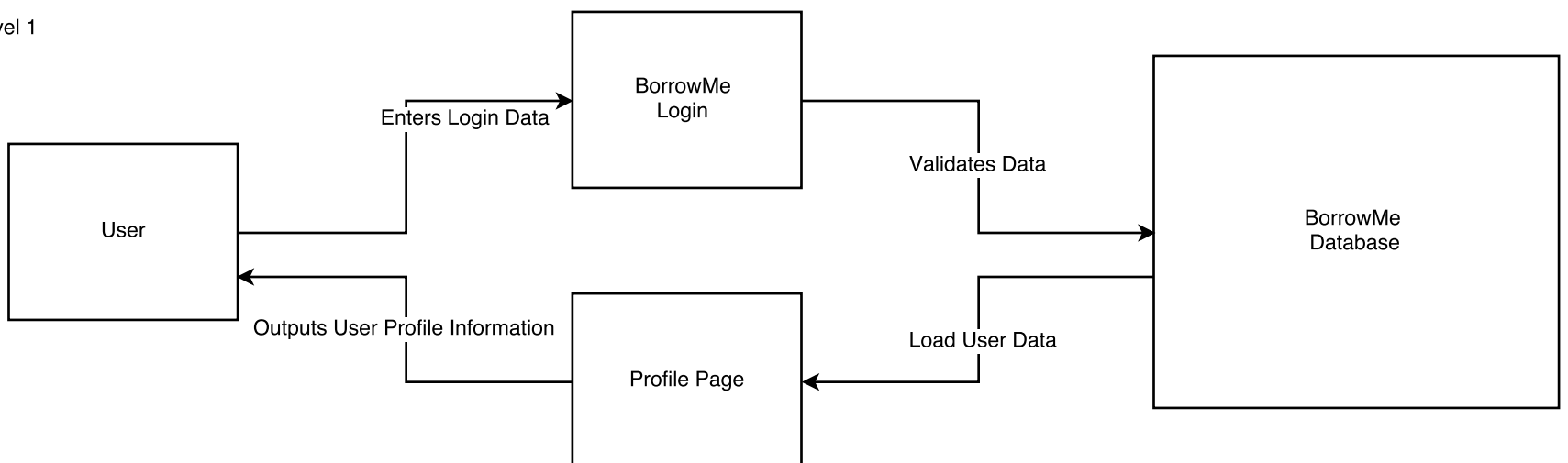
DataFlow Diagram

Level 0

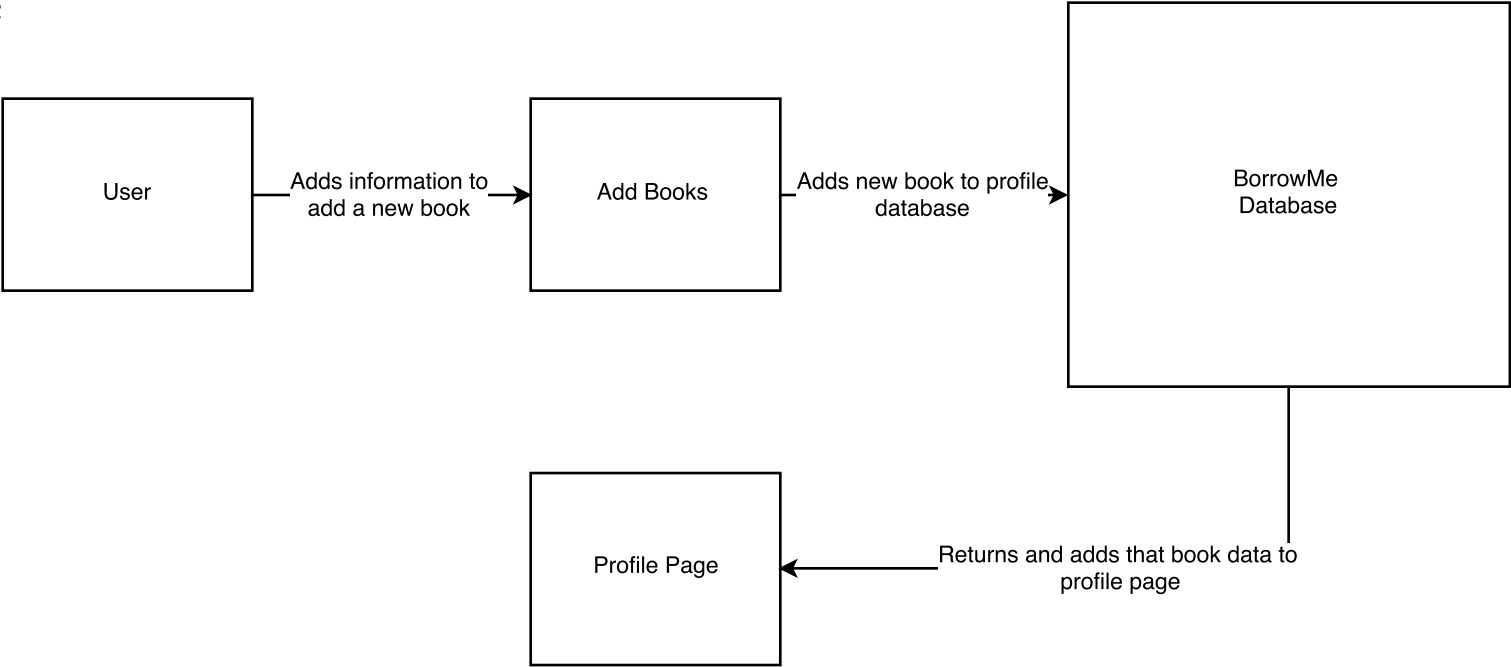


CLCP Inc.
November 14, 2017

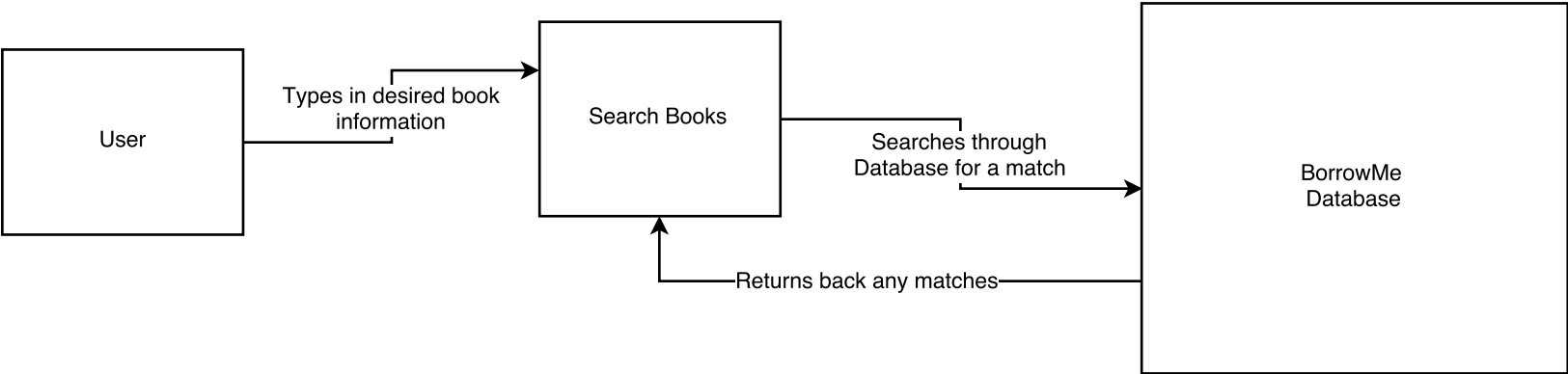
Level 1



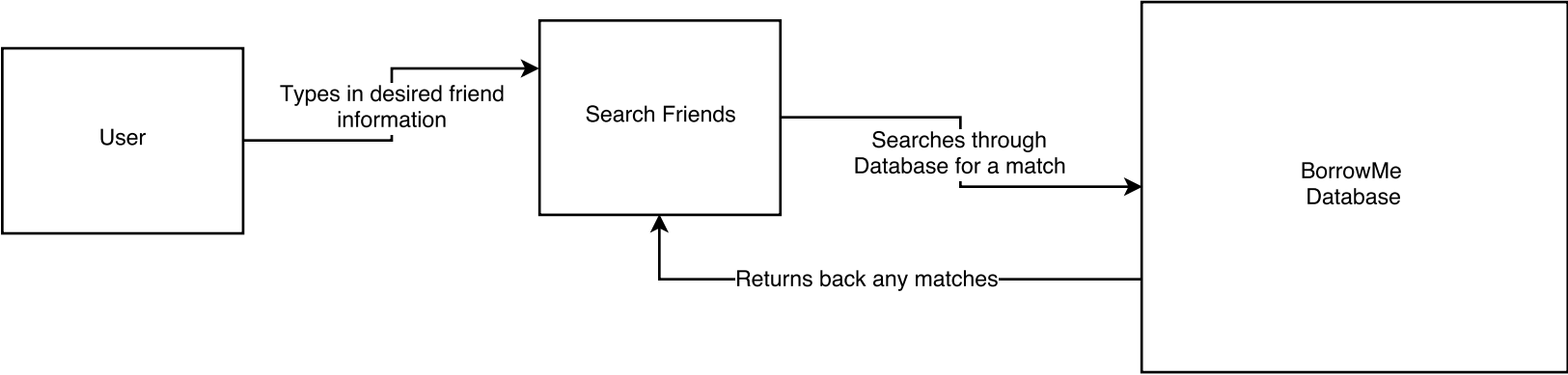
Level 2



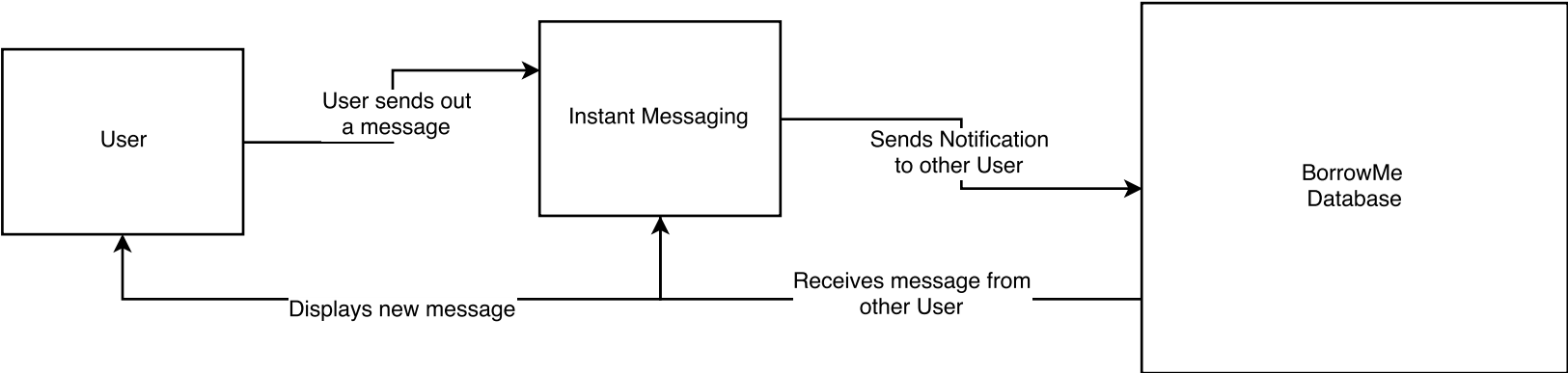
Level 3



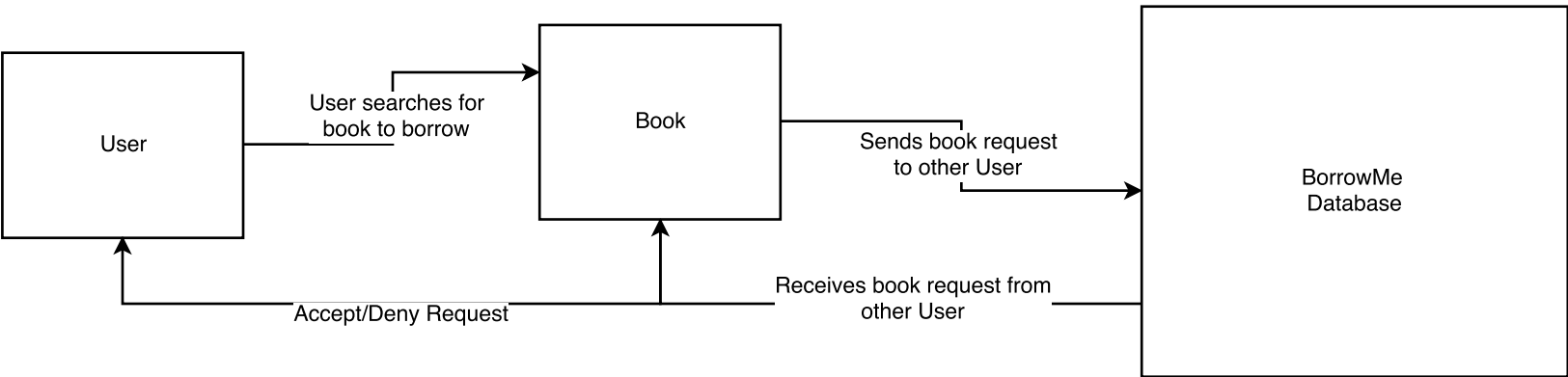
Level 3



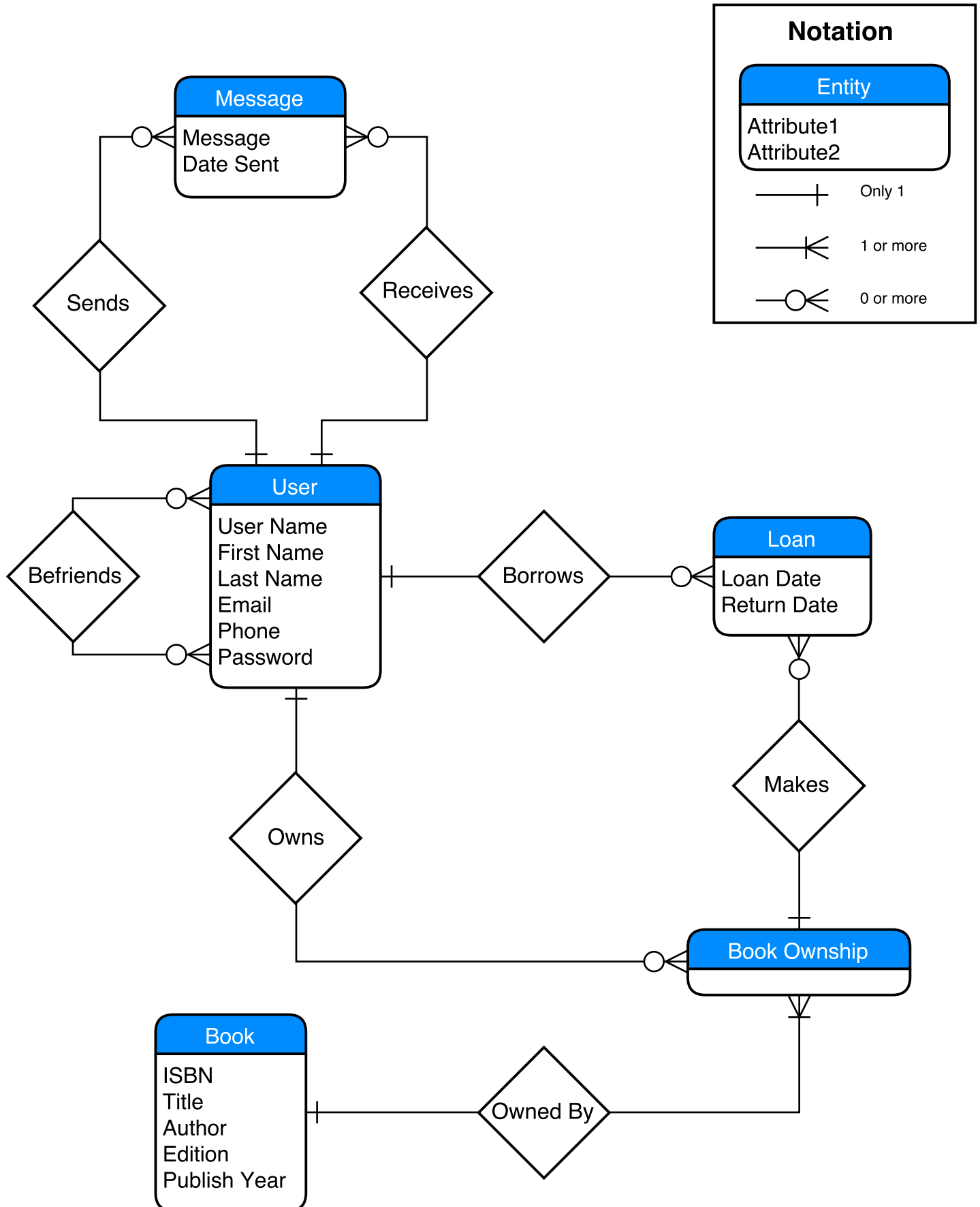
Level 5



Level 6



Entity Relationship Diagram

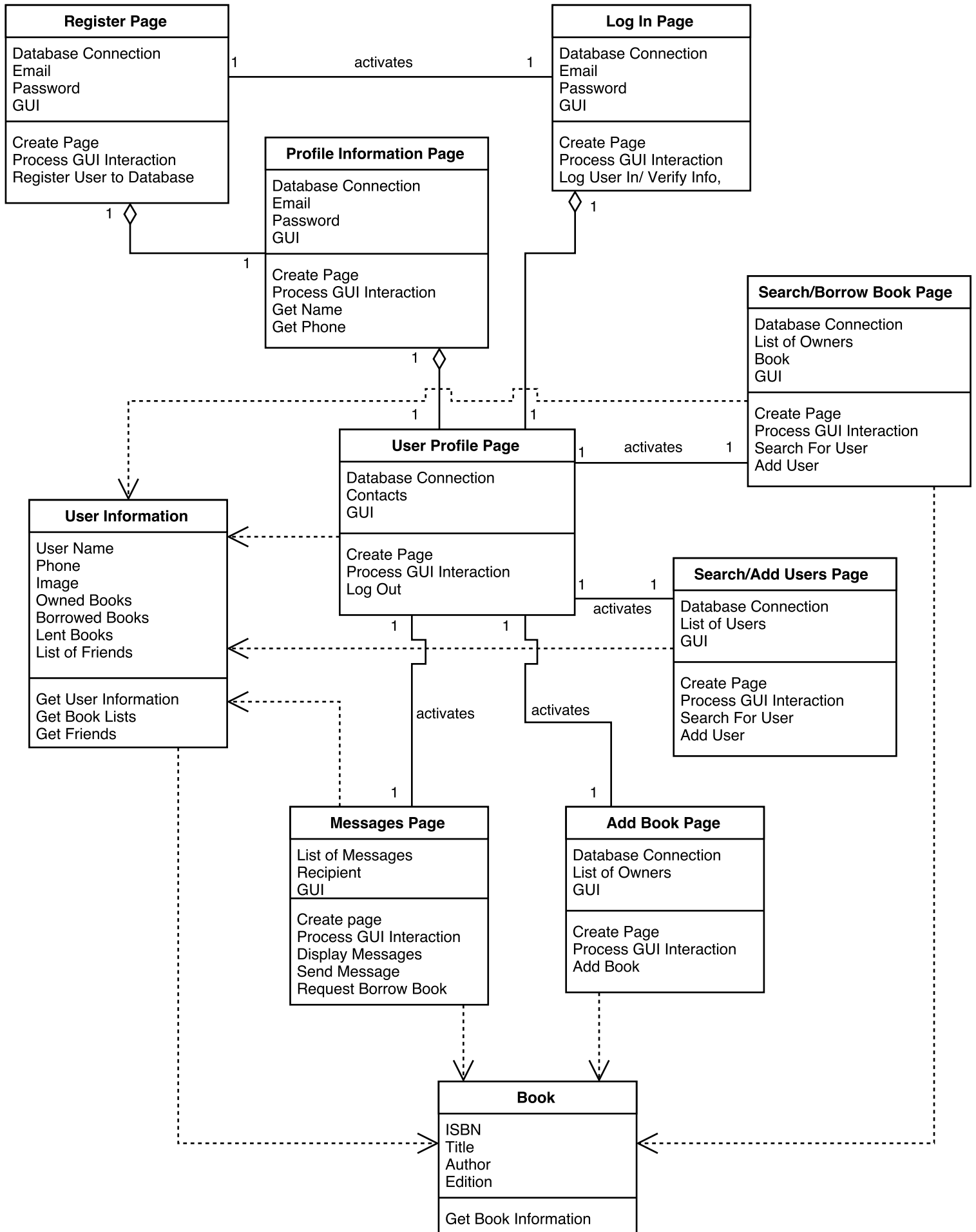


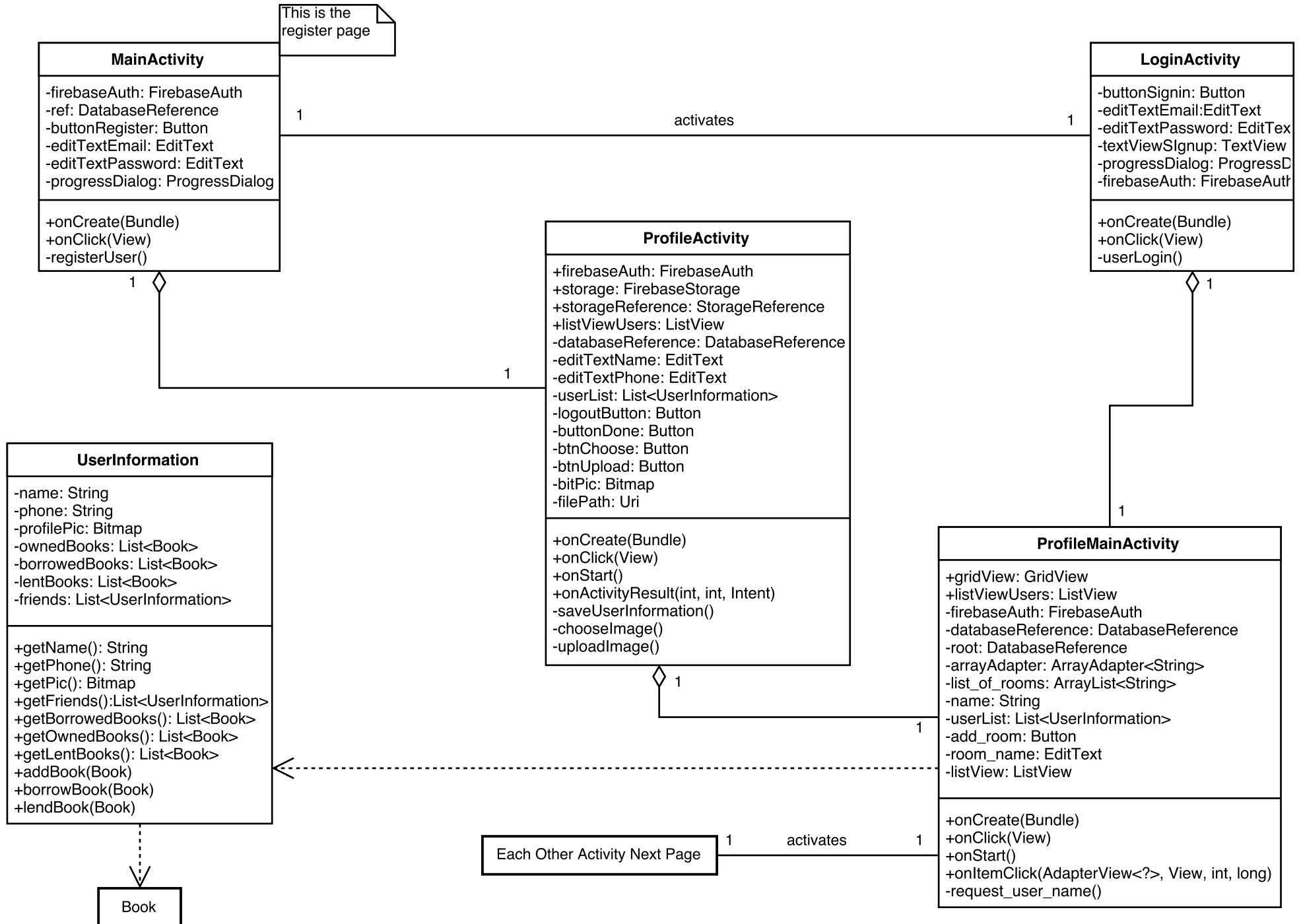
BorrowMe Class Tables

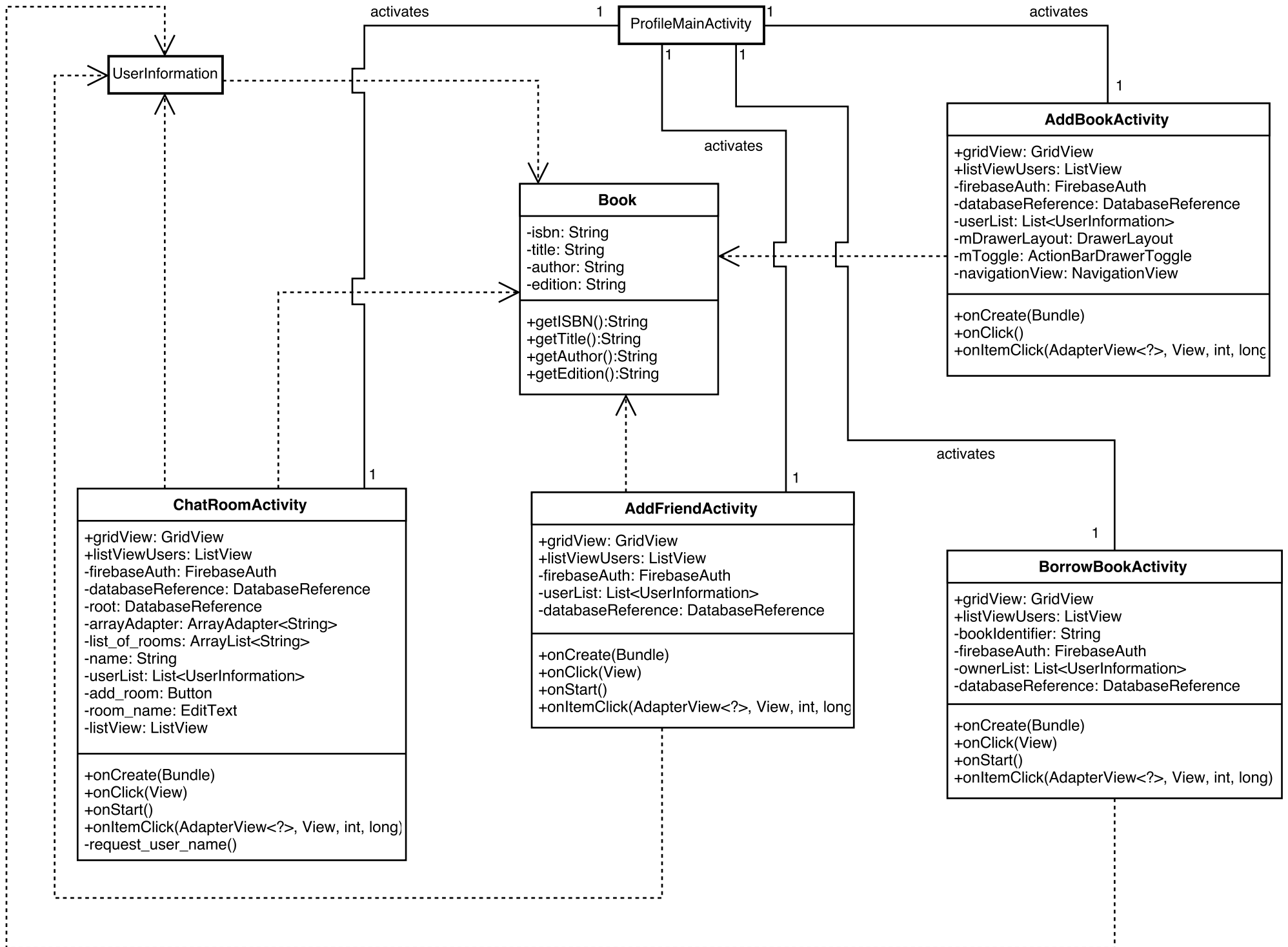
CLASS	NOUNS/ ATTRIBUTES	TYPE
Book	ISBN Title Author Edition	String String String String
User Information	User Name Phone Image Owned Books Borrowed Books Lent Books List of Friend	String String Bitmap List<Book> List<Book> List<Book> List<Users>
Register Page	Database Connection Email Password GUI	FirebaseAuth String String Button, EditText, TextView, ProgressDialog
Profile Information Page	Database Connection Name Phone GUI	FirebaseAuth, FirebaseStorage, StorageReference, DatabaseReference String String Button, EditText, TextView, ProgressDialog, ImageView, BitMap
Log In Page	Database Connection Email Password GUI	FirebaseAuth String String Button, EditText, TextView, ProgressDialog
User Profile/ Home Page	Database Connection Contacts GUI	FirebaseAuth, DatabaseReference List<Users> Button, GridView, ListViewUsers
Search For Book Page	Database Connection List of Owners Book	FirebaseAuth, DatabaseReference List<Users> Book
Add Book Page	Database Connection List of Users GUI	FirebaseAuth, DatabaseReference List<Users> DrawerLayout, ActionBarDrawerToggle, NavigationView, GridView, ListView
Messages Page	List of Messages Recipient GUI	List<String> User DrawerLayout, ActionBarDrawerToggle, NavigationView, GridView, ListView

CLASS	VERBS/ METHODS
Book	Get Book Information
User Information	Get User Information Get Book Lists Get Friend List
Register Page	Create Page Process User GUI Interaction Register User Into Database
Profile Information Page	Create Page Process User GUI Interaction Get Name Get Phone
Log In Page	Create Page Process User GUI Interaction Log User In/ Verify Information
User Profile/ Home Page	Create Page Process User GUI Interaction Log Out
Search For Book Page	Create Page Process User GUI Interaction Search For Book Request Borrow
Add Book Page	Create Page Process User GUI Interaction Add Book
Messages Page	Create Page Process User GUI Interaction Display Messages Send Message

BorrowMe Analysis UML







Test Cases

ID	Input	Expected Result	Actual Result
1	Register: <u>lucas@hotmail.com</u> , password00	add <u>lucas@hotmail.com</u> with password as password00 to database	<u>lucas@hotmail.com</u> with password as password00 was added to database
2	Login: <u>lucas@hotmail.com</u> , password01	Fail Login	Fail Login
3	Login: <u>lucas@hotmail.com</u> , password00	Login Successful	Login Successful
4	Update Profile: 916-337-8463 Computer Science	Display updated information: 916-337-8463 Computer Science	Displays: 916-337-8463 Computer Science
5	Search book: Harry Potter ISBN:1234567890	Book Not Found	Book Not Found
6	Add book: Harry Potter ISBN:1234567890	Added Book: Harry Potter ISBN:1234567890 to profile	Added Book: Harry Potter ISBN:1234567890 to profile
7	Search book: Harry Potter	Book Found, display Harry Potter	Book Found, display Harry Potter
8	Go to Profile	Display new added book: Harry Potter	Display new added book: Harry Potter
9	Add friends: Search PotatoMan	Display PotatoMan	Display PotatoMan
10	PotatoMan was added as a friend	PotatoMan is now displayed in FriendsList	PotatoMan is now displayed in FriendsList
11	Add friends: Search John	User not found	User not found
12	Send Message to PotatoMan: "Hello"	PotatoMan Receives Message: "Hello" and is displayed in his messages	Message sent by <u>lucas@hotmail.com</u> -"Hello" (This is in PotatoMan Profile)

13	Reply to <u>lucas@hotmail.com</u> : "How are you?"	lucas@hotmail Receives Message from PotatoMan "How are you?" and is displayed in his messages	Message sent by PotatoMan -"How are you?" (This is in <u>lucas@hotmail.com</u> profile
14	Search book: ISBN: 1234567890	Book Found, display Harry Potter	Book Found, display Harry Potter
15	Log Out	Log out Successful and taken back to login screen	Log out Successful and taken back to login screen

BorrowMe Specifications and Installation

File structure/implementation:

This program uses the standard android app file structure to organize the source files. The source code is available as several .java files along with .png files of all the images and icons used in the program. No .exe file is not used for this program.

Tools used:

The main tool used for the creation of this app was Android Studio. This allowed us to simultaneously test the app on a virtual android emulator while developing the code. When simple changes had to be made to the code, it was done using a simple text editor such as Sublime Text. Github was used as the main collaboration and version control platform throughout the development of this app.

Installation:

To install and test the system at its current stage, Android Studio is recommended. Most of the source files are .java files and so any Java IDE such as Eclipse or even a simple text editor may be used to view the code. To view the full app functionality, however, Android Studio will have to be used to create an emulator that can recreate the functions of the app on a laptop or desktop screen. Once Android Studio is installed, the whole source code package can be downloaded from github and opened and emulated using Android Studio.

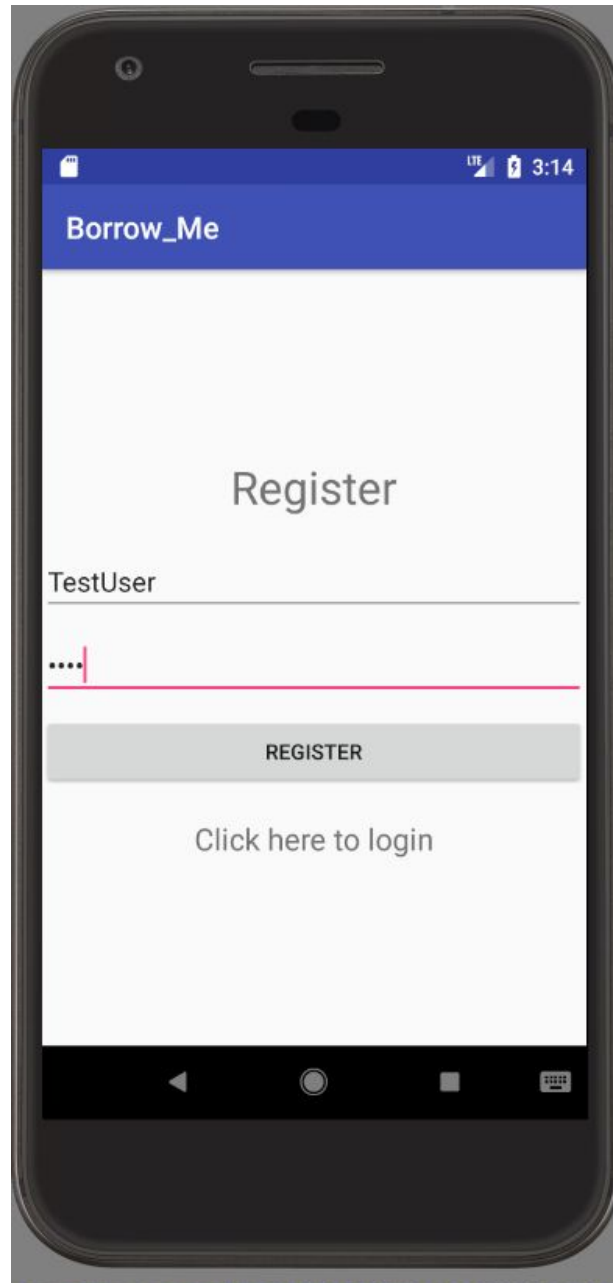
In the future, when the app is submitted and approved into the Android app store, it can be directly downloaded through the app store. All you will have to do is search for "BorrowMe" using the app store search bar.

Prerequisites:

As mentioned above, Android Studio will need to be used to view the full functionality of the app on a laptop or desktop. Android Studio is available for both Mac OS and Windows. BorrowMe is a lightweight app and so it should be possible to run it on older systems without too much difficulty.

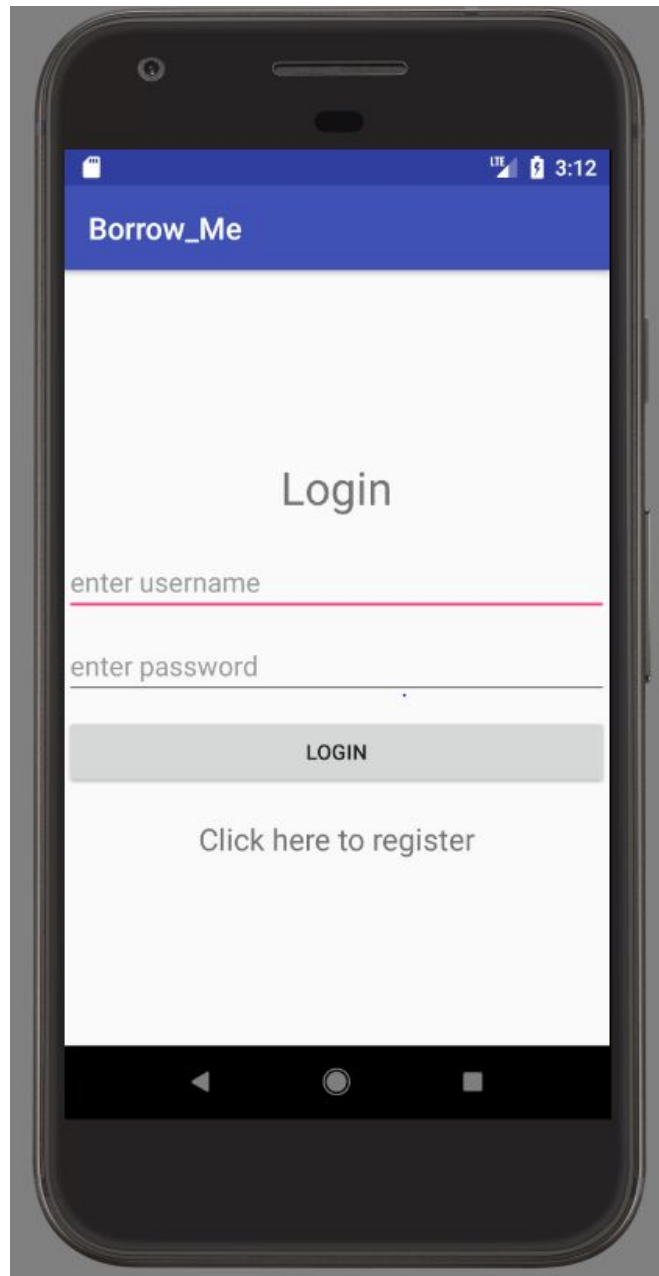
BorrowMe User Directions

1. Open the application.
2. Choose to register an account by pressing on “Click here to register.”
Note: A small pop up will say that the registration was a success.
3. Return to the login screen by pressing on “Click here to login.”



4. Log into the application by correctly typing in your username and password in the appropriate fields.

Note: If you enter invalid information, the application will alert you.

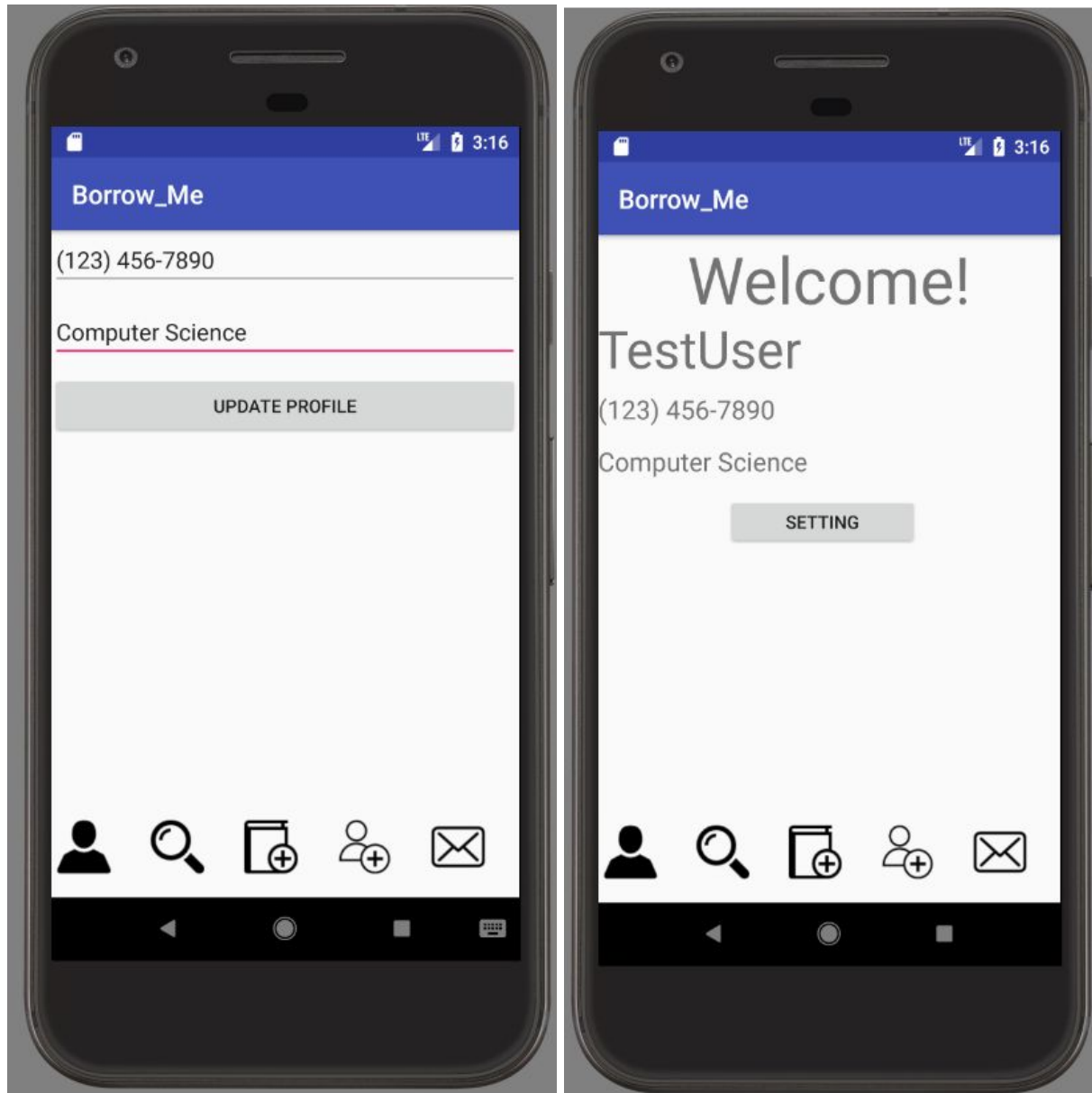


Note: This page is your home page that displays your username and some information about yourself, which are blank by default

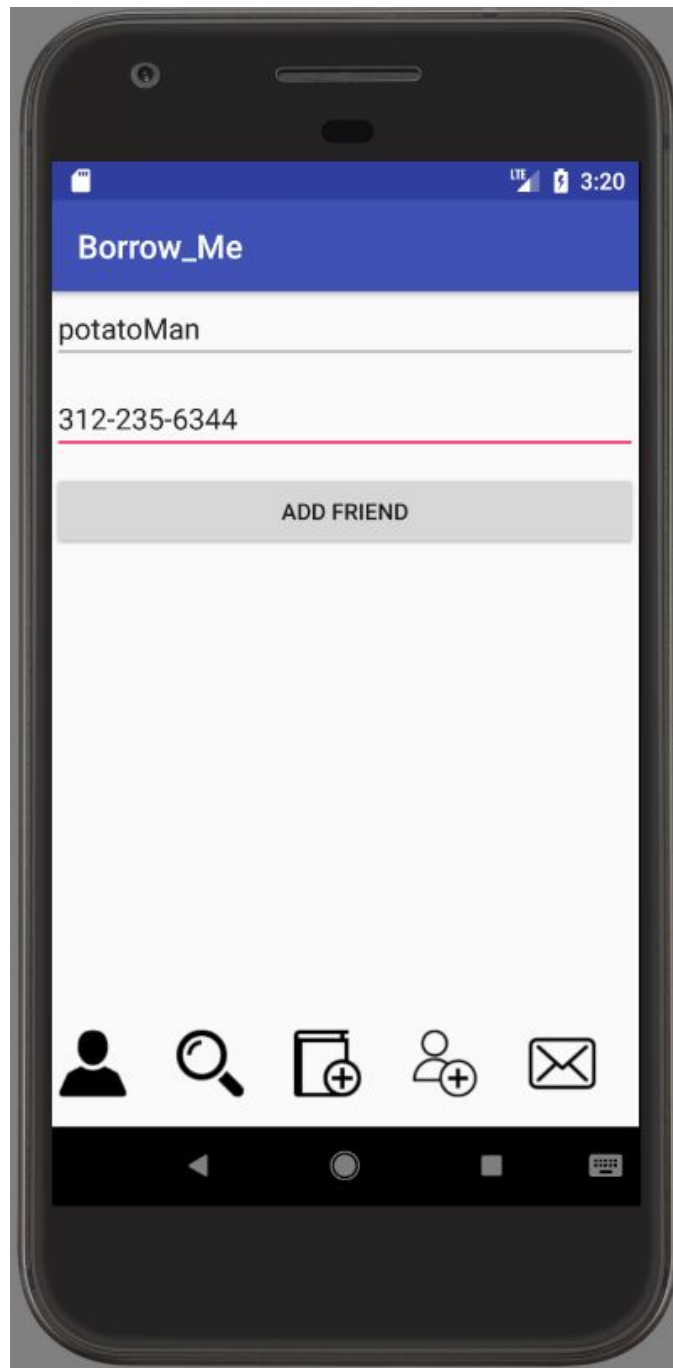
5. Press settings to change your user profile information.



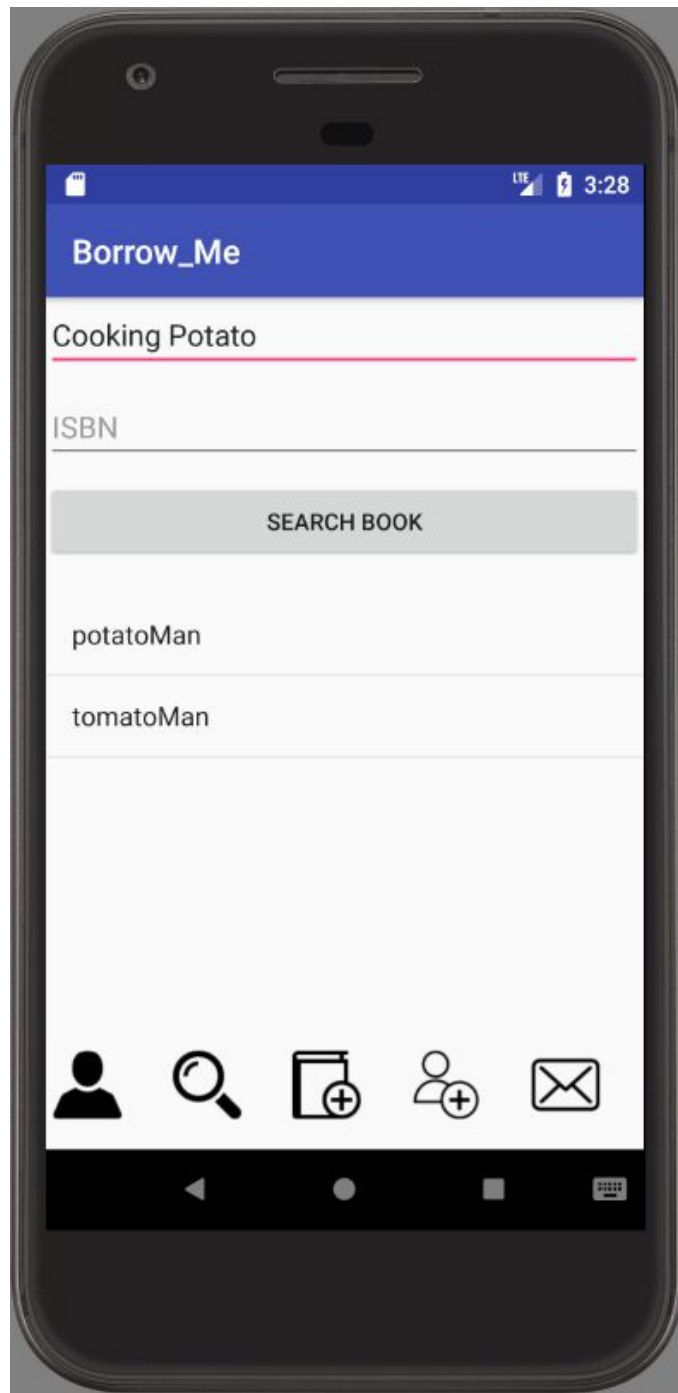
6. Enter your phone number on the first line.
7. Enter your major or field of study on the second line.
8. Press "Update Profile to Return to your home page.



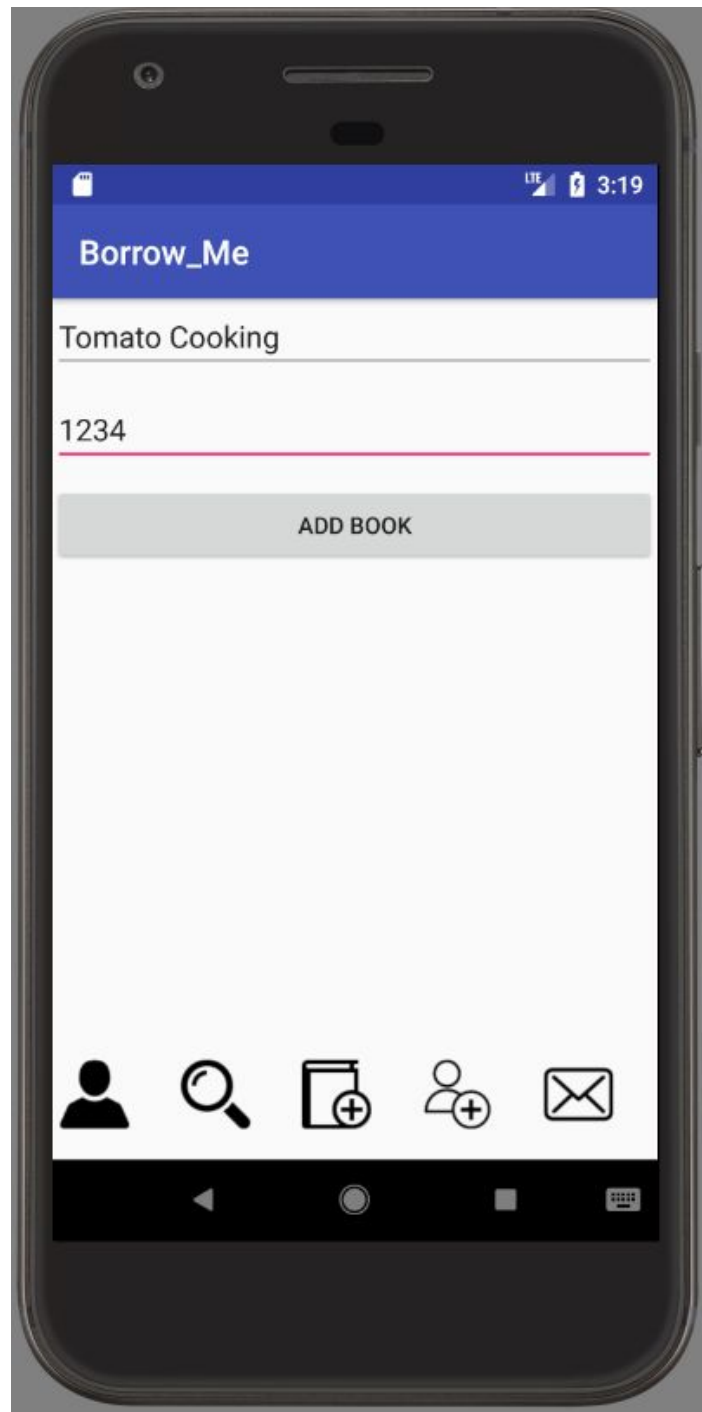
9. Too add a contact, press the bottom person symbol with a plus sign.
10. Type in the person's name on the first line.
11. Type in the person's phone number on the second line and press Add Friend."



12. To search for a book, press the magnifying glass at the bottom.
13. Now enter the book title on the top line and/ or the ISBN on the bottom line.
Note: The bottom results will be those friends of yours that own the book and the message "No Users Found" shall be displayed if no one does.



14. To add a book you own, press the book symbol at the bottom.
15. Type in the book title on the first line.
16. Type in the book ISBN on the second line and press “Add Book.”



17. To send and receive messages with your contacts, press the envelope at the bottom.
18. Press the name of the person you wish to converse with.
19. Type in and send messages with the sideways arrow at the bottom right.
Note: Both parties must be on each other's friends list to see the messages from the other. And in order to return you must use the phone back button.

