Table of Contents

[CHAPTER ONE 3](#_Toc498956543)

[1.0    PROBLEM DEFINITION 3](#_Toc498956544)

[CHAPTER TWO: ECONOMIC FEASIBILITY STUDY 4](#_Toc498956545)

[2.1 4](#_Toc498956546)

[2.2 AIM AND OBJECTIVES OF STUDY 4](#_Toc498956547)

[CHAPTER THREE: SOFTWARE REQUIREMENT SPECIFICATION 6](#_Toc498956548)

[3.1 SYSTEM REQUIREMENT ANALYSIS AND DESIGN 6](#_Toc498956549)

[3.1.1  USER'S REQUIREMENT 6](#_Toc498956550)

[3.1.2  FUNCTIONAL REQUIREMENTS: 6](#_Toc498956551)

[3.2  USE CASE DIAGRAM 7](#_Toc498956552)

[3.3  SOFTWARE REQUIREMENTS AND QUALITIES: 8](#_Toc498956553)

[3.3.1 BUYERS 8](#_Toc498956554)

[3.3.2  SELLER 9](#_Toc498956555)

[CHAPTER FOUR: DESIGN SPECIFICATION 11](#_Toc498956556)

[4.1  SOFTWARE ARCHITECTURE 11](#_Toc498956557)

[4.2  DESIGN PATTERNS 12](#_Toc498956558)

[4.2.1   FACTORY DESIGN PATTERN 12](#_Toc498956559)

[4.2.2  OBSERVER DESIGN PATTERN 13](#_Toc498956560)

[4.3  CLASS DIAGRAM 15](#_Toc498956561)

[4.4  UML TOOL 16](#_Toc498956562)

[CHAPTER 5: PROGRAMS 17](#_Toc498956563)

[5.1 COMPONENT DIAGRAM 17](#_Toc498956564)

[5.2 DEPLOYMENT DIAGRAM 20](#_Toc498956565)

[5.3 CLASSES AND FUNCTIONS 21](#_Toc498956566)

[5.4 NUMBER OF FILES OF LAYERS/COMPONENT 22](#_Toc498956567)

[5.5 TABLES OF THE SYSTEM DATA 22](#_Toc498956568)

[CHAPTER 6: TECHNICAL DOCUMENTATION 26](#_Toc498956569)

[6.1 PROGRAMMING LANGUAGES 26](#_Toc498956570)

[6.2 REUSED ALGORITHM AND PROGRAMS 26](#_Toc498956571)

[6.3 SOFTWARE TOOLS AND ENVIRONMENTS 26](#_Toc498956572)

[CHAPTER 7: ACCEPTANCE TESTING 27](#_Toc498956573)

[7.1 FUNCTIONAL TESTING 27](#_Toc498956574)

[7.2 ROBUSTNESS TESTING 31](#_Toc498956575)

[7.3  USER INTERFACE TESTING 35](#_Toc498956576)

[7.4  TIME-EFFICIENCY TESTING 38](#_Toc498956577)

[CHAPTER 8: TEAM MEMBER CONTRIBUTION 39](#_Toc498956578)

[8.1    GROUP MEMBER CONTRIBUTION 39](#_Toc498956579)

# CHAPTER ONE

## 1.0    PROBLEM DEFINITION

Buying used books online is something that every student in every university nationwide can relate to, not only because students will be able to get the books cheaper than they do in any bookstore, but the books always comes in good condition which almost likely seem new.

The good thing is that we have a platform which enables students to be able to do all this in the University of Regina through the University of Regina buy and sell Facebook group. We have seen and noticed that there is a lot of problem with this platform hence our desire to come up with something better and much friendly for users.

The application domain of our project is E-commerce. E-commerce can be defined as commercial transaction conducted electronically on the internet, some of the advantages of E-commerce are Buying and selling 24/7, faster buying and selling procedure, no need for physical company set up, Some of the disadvantages will be that there is no guarantee of product quality, mechanical failures can cause unpredictable effects on the total processes (eSalestrack).

Some of the problems that we face with the University of Regina buy and sell was that users find it hard to search for books and if eventually they were able to find the book it took longer than expected to actually locate the book in which they were looking to buy. Another problem was the sorting of the books, books where not sorted properly either by ISBN no, by faculty or by department which in turn affected users ability to search properly for books. Books that have been sold are still being advertised, customers go to buy a book and then you find out that the book you want is no longer available, meanwhile on the platform it still reads as available.  Buyers of books don’t have a wishlist, they can't come back and view books that they wanted to buy but didn't have the resources for at that time.

The scope of this study is to design and develop a platform that makes the buying and selling of used books much more easier and safer for our users, the intended system will make the searching of books as easy as it could be, which will help our users search for books with ease and also get results of the search in real time and on time. The system will also make use of encryption techniques to secure data (i.e. the protection of data from unauthorized modification, destruction or disclosure during input, transmission or output operations), so that only authorized users can view specific information.

# CHAPTER TWO

# ECONOMIC FEASIBILITY STUDY

This section clearly show the method used on how to build the system, e.g. developing a new system, extending or improving on an existing system. Then it presents the benefits of the system for its end users.

## 2.1

We will be improving on an existing system, which is the University of Regina buy and sell Facebook group.

The problems identified with the system put in place at the University of Regina buy and sell Facebook group. That are to be addressed in this project include:

* Lack of a search that does not enable users get easy access to books been searched for.
* Books not arranged well enough which in turn affects searching for books.
* When searching for a book, you don't get the books being searched for or it takes a very long time to actually get what you really looking for.
* Books that have been sold are still being displayed that they are still for sale.
* User interface not friendly enough for users. Too much information on the platform which either makes users confused or takes away their attention from what they really came to accomplish.

## 2.2 AIM AND OBJECTIVES OF STUDY

The objective of this study is to improve on the existing system (University of Regina buy and sell Facebook group) that will try to achieve the following:

* Search in real time and display results fast.
* Looking for a book will be made easier for anybody looking for it. Buyers will be able to search for books either by using ISBN number, faculty, department, subject name e.t.c
* Create security at every level of authorization.
* Books will be taken off system when the user deletes the book.
* Only authorized users can view specific information.
* When buyer wants to purchase a book, he/she will have to email the seller.
* Buyer will be able to add books that he/she wishes to buy at a later time to his/ her wish list.

# CHAPTER THREE

# SOFTWARE REQUIREMENT SPECIFICATION

## 3.1 SYSTEM REQUIREMENT ANALYSIS AND DESIGN

A system requirement is a description of needs and desires for a system. A requirement may describe functions, features (attributes) and constraints. It defines the service the system is to provide and prescribe constraints for its operation.

System design is an iterative process through which requirements are translated into a '"blueprint" for constructing the software. Design is usually represented at a high level of abstraction - a level that can be traced to a specific system objective and more detailed data, functional and behavioral requirements. As design iterations occur, subsequent refinement leads to design representations at much lower levels of abstraction. There are many strategies or techniques for performing systems design, they include: Waterfall model, Case tools, Prototype, RAD/RSD, JAD, RUP and object-oriented methodology (OOM). For this project work, the **WATERFALL** model was adopted.

The user's requirement for the proposed system is discussed in sections below:

### 3.1.1  USER'S REQUIREMENT

These are statements in natural languages together with the diagram of the services the system provides and its operational constraints written for customers.

User's requirement consist of:

* Functional requirement

### 3.1.2  FUNCTIONAL REQUIREMENTS:

This is a statement of services which the system must provide. It could also be referred to as specific requirements.

* Buyer

i. Buy a book

ii. Signup

iii. Login

iv. View book on sale

v. Send email to seller of the book

vi. Buyer should have a wishlist

vii. Buyers search for books in real time.

viii. Buyers should be able to search books using keywords.

ix. Buyer can search by location

* Seller

i. View his/her books on sale

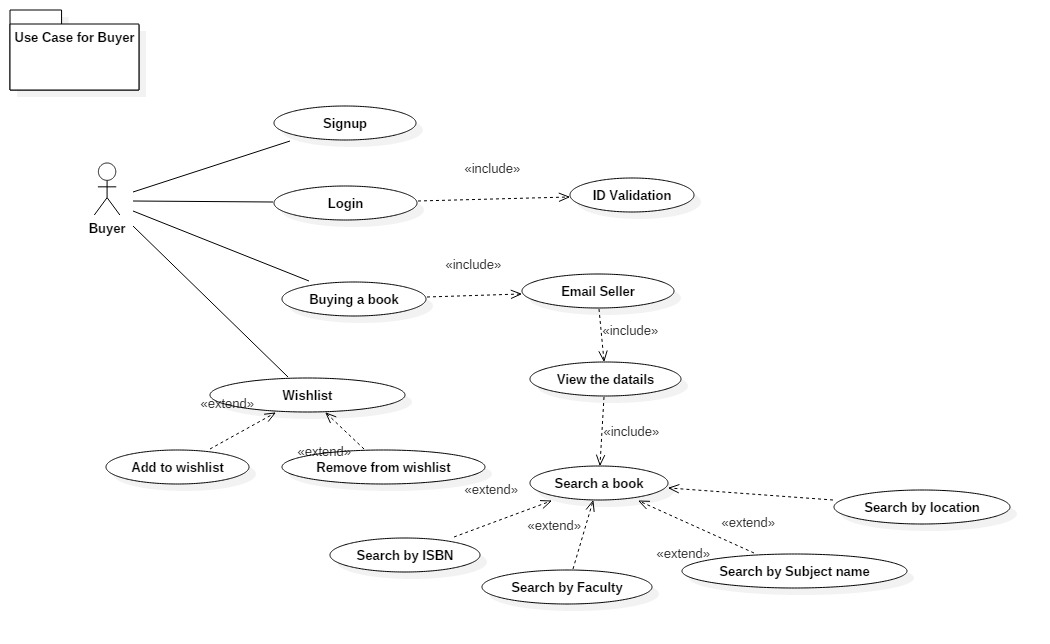
ii. Signup

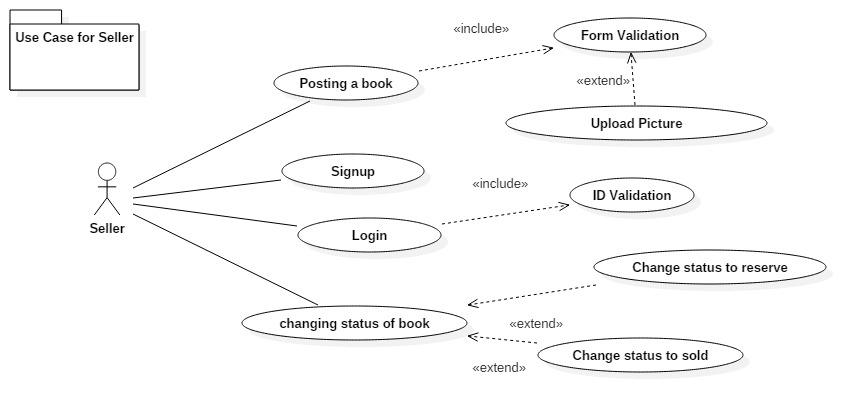
iii. Login

iv. Responsible for changing the status of a book he/she posted to sold.

v. Post books to be on sale

## 3.2  USE CASE DIAGRAM

****

****

## 3.3  SOFTWARE REQUIREMENTS AND QUALITIES:

### 3.3.1 BUYERS

1. correctness:
   1. The search option for the buyer should return the correct information being searched for at the time.
   2. When buyer wants to purchase a book and email the seller, the seller should get the email of the buyer.
   3. When buyer adds a book to wishlist, buyer should be able to see book on the wish list if book is still available.
   4. Buyers should be able to search for books in any geographical location.
   5. If buyer is not logged in, he/she can't email the seller about buying a book.

2.  Robustness:

1. When wrong userID or password is inputted an error message is displayed.
2. If buyer searches for a book that is not available, an error message should be displayed telling the buyer that the book is not available.

3.   User-friendliness

1. After signing up, when buyer wants to login there should be a remember me to ease login in and save time for seller
2. The UI should be suggest options of what the buyer  wants to input as more values comes in.
3. There should be a good browse navigation to ease the searching of books.

4. Time efficiency:

1. When searching for book it should be in real time and should start displaying options once letters are being typed.

### 3.3.2  SELLER

1. Correctness:
   1. The UI page of the seller should return the correct information being searched for at a time.
   2. When the seller posts a book/books people should be able to view it when searched.
   3. When the seller deletes a book/books it should not be on the seller's list or in the system.

1. Robustness:
2. When wrong user ID or password is inputted an  error message is displayed
3. If seller's posts a book and it does not get posted, seller should be notified of the book not getting posted.
4. Seller should be made to sign in before being able to post book for sell

1. User-friendliness:
   1. After signing up, when seller want to login there should be a remember me to ease login in and save time for seller.
   2. Users should be able to receive notifications of any action that has been taken on the users account.
2. Time-efficiency:
   1. When users posts a book it should be posted the same time the seller clicks post.

5.   Security:

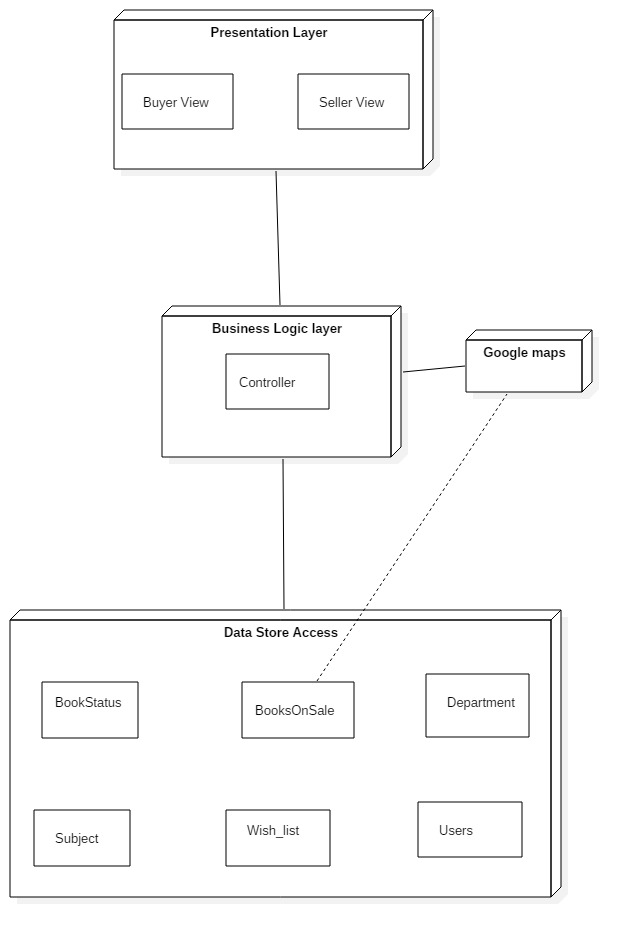
1. Sellers information should be encrypted and only viewed by him/her.

# CHAPTER FOUR

# DESIGN SPECIFICATION

## 4.1  SOFTWARE ARCHITECTURE

The UOFL will utilize the three (3) Tier Architecture. We are using the three (3) Tier Architecture because it allows loose coupling within the different layers of the architecture i.e if a layer needs to be changed. You can do so without impacting the other layers. It also provides an ease of maintenance of the code base, managing presentation code and business logic separately, so that a change to business logic, for example, does not impact the presentation layer.



## 4.2  DESIGN PATTERNS

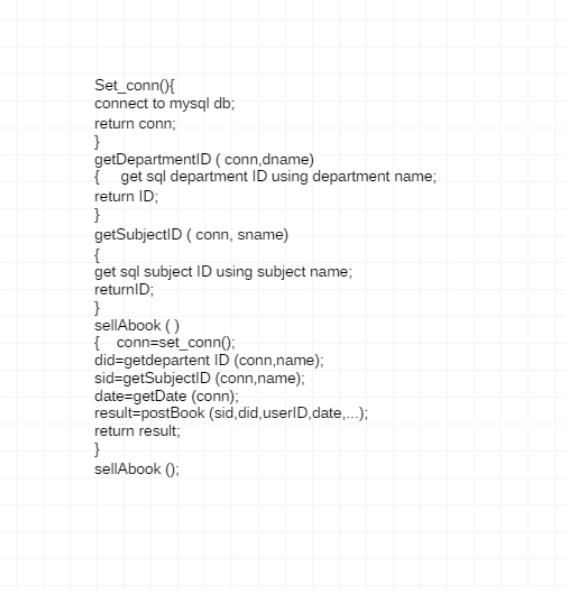
In this project, we have decided to go with two (2) design patterns which are:

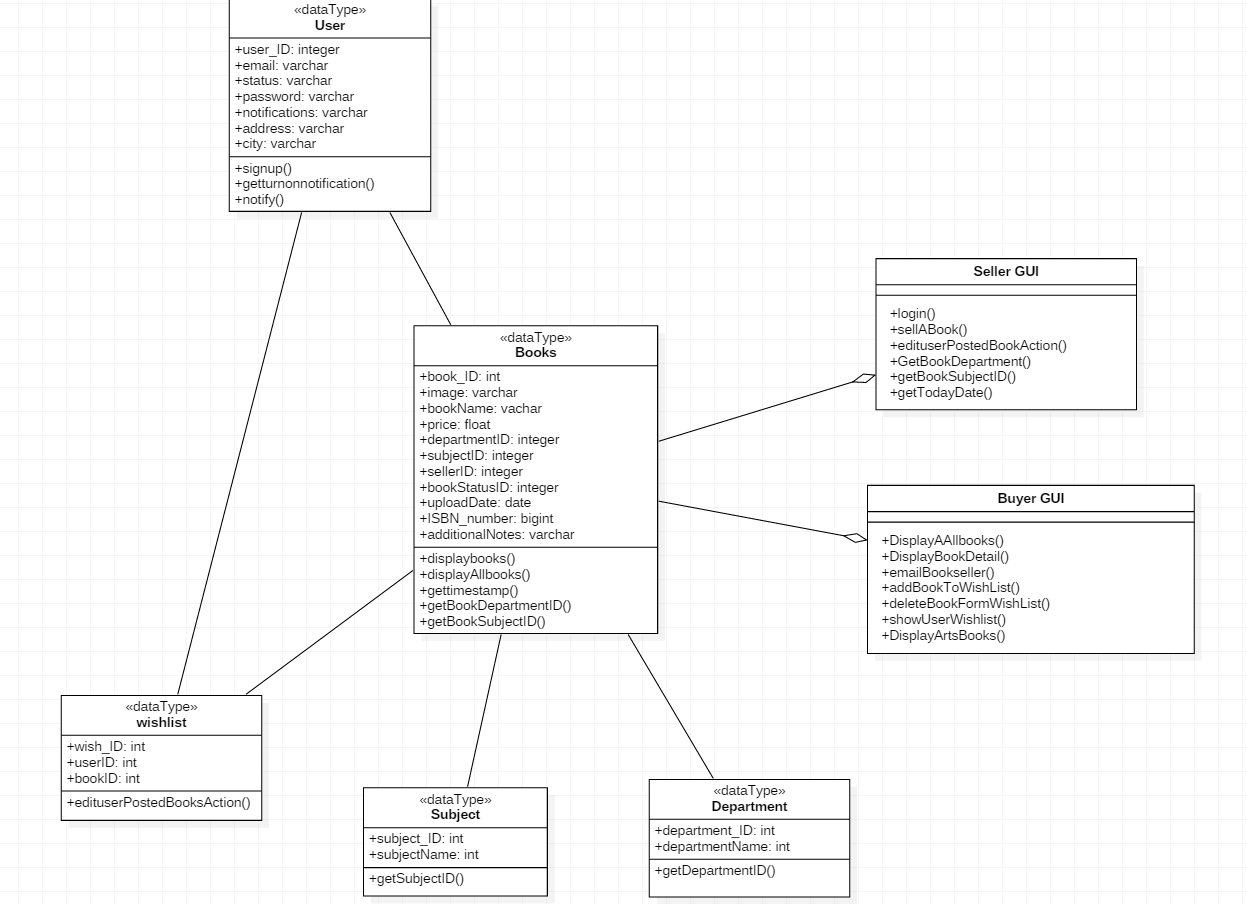
* Factory design pattern
* Observer design pattern

### 4.2.1   FACTORY DESIGN PATTERN

The Factory Design Pattern was chosen because it solves a problem which is, how do we quickly and easily instantiate numerous types of objects in the parent class? We are going to need to introduce a method that allows low coupling between the instantiation of new data objects and the parent class. It is not feasible to have 25-30 individual states having some sort of instantiation code in the parent class. We would have extremely high coupling between the main class and all of the legend states. This would make it difficult to add and remove new types of objects states, thus taking away from the extendibility of the software.

To solve this method we have chosen to use the Factory Method Design Pattern. This design pattern gives us low coupling that we need, so that we can adhere to our requirement of extensibility. It helps to make our design more customizable. It also provides us with a high level of abstraction for instantiation of data objects we want to export. Finally we will be able to create any type of data object through a factory.

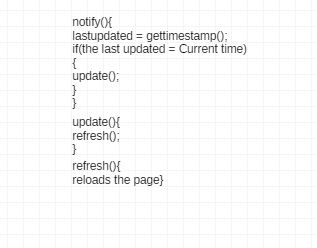


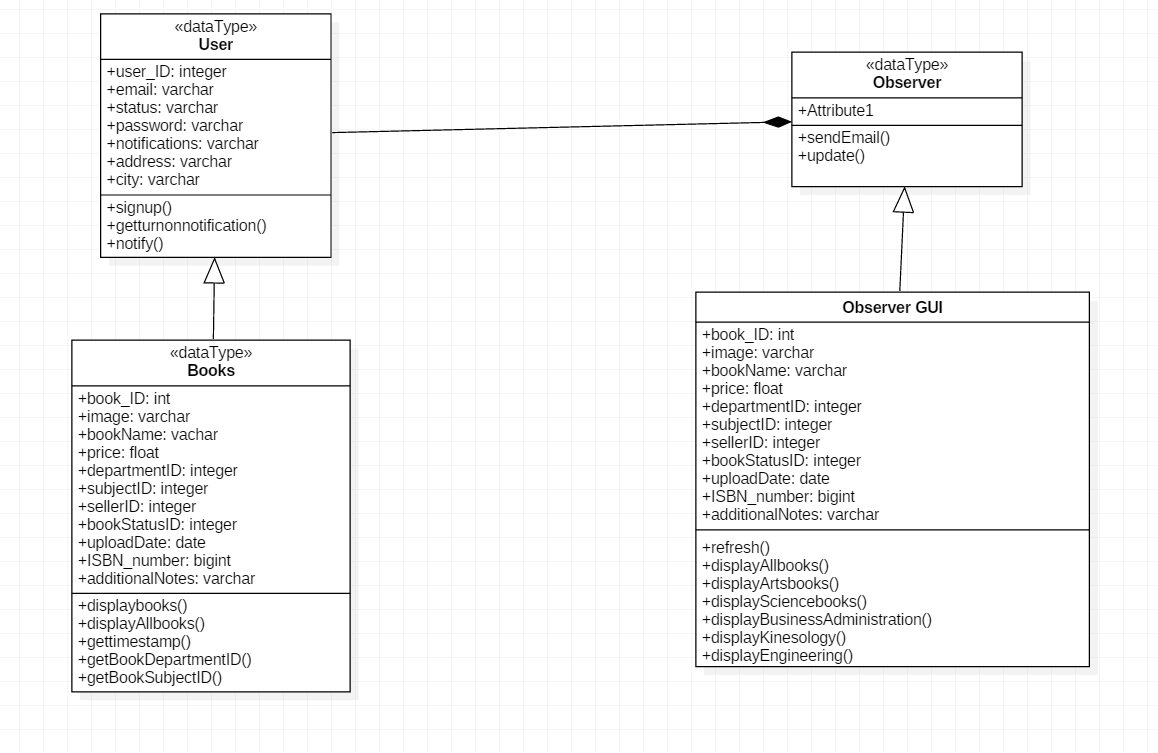


### 4.2.2  OBSERVER DESIGN PATTERN

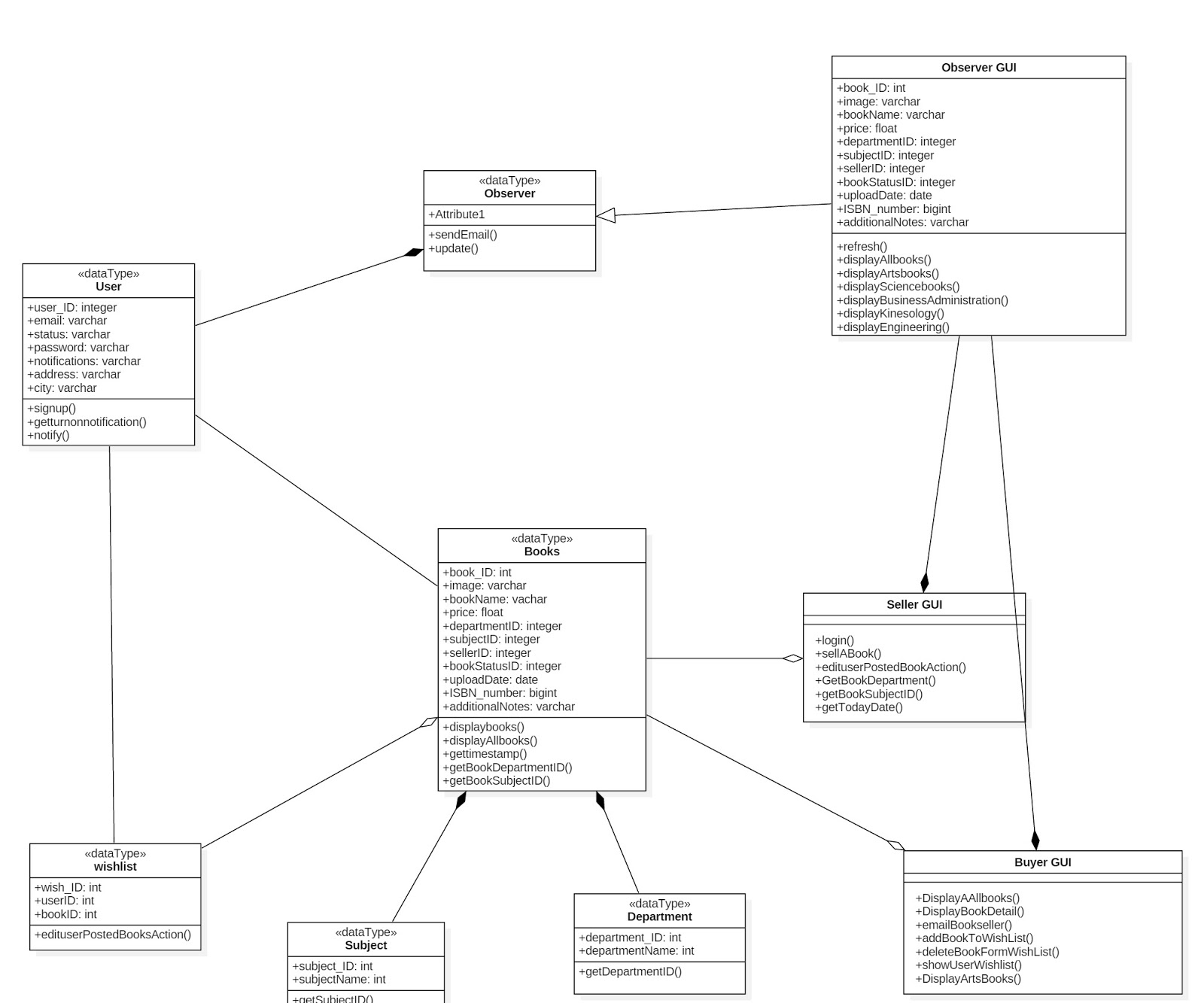
The Observer Design Pattern was another obvious choice, as it would provide us with low coupling between objects. However, the classic observer pattern was not sufficient for our purposes. It would be inefficient to attach a list of observers to every object in the game world. Furthermore, attaching observers to subjects would be very difficult to do without losing low coupling. As such, we decided to switch to the event handler and delegate method of implementing the observer design pattern.

To solve this problem we decided to use the Observer Design Pattern because it helps define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically. It also helps in that when observers register themselves with the Subject as they are created. Whenever the Subject changes, it broadcasts to all registered Observers that it has changed, and each Observer queries the Subject for that subset of the Subject's state that it is responsible for monitoring.





## 4.3  CLASS DIAGRAM

****

## 4.4  UML TOOL

The UML tool for this project is StarUML. StarUML is an open source UML modeling application. The pros and cons of this application is as follows:

**Pros**

* Supports most of the diagrams specified in UML 2.0.
* Very rich feature set and formatting options.
* Ability to generate source code from the UML diagram.
* Reverse engineer the existing code into UML diagrams.
* Supported languages : C , C# and Java.
* Fast load time/execution time compared with other UML tools.
* Familiar *Visual Studio* like user interface.
* Supports exporting diagrams into JPG / XMI formats.

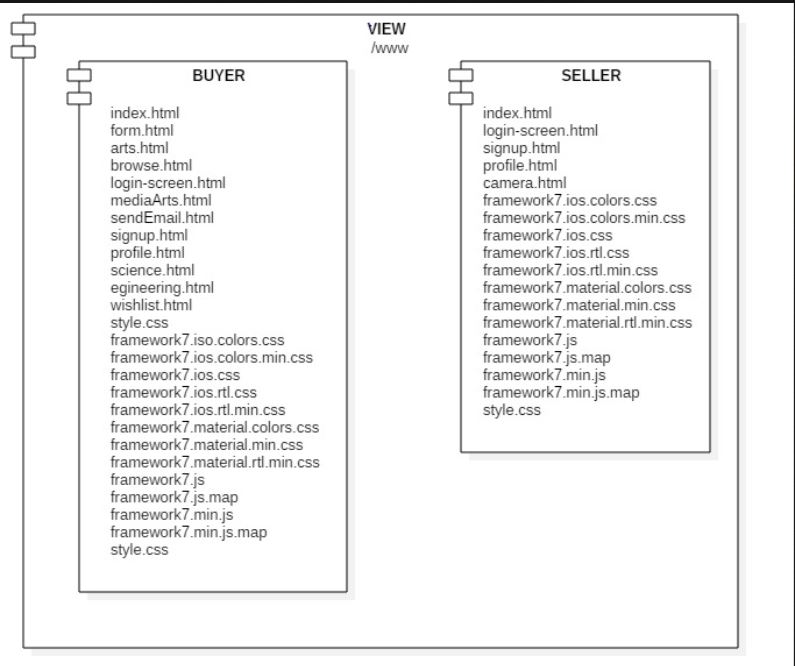
Cons

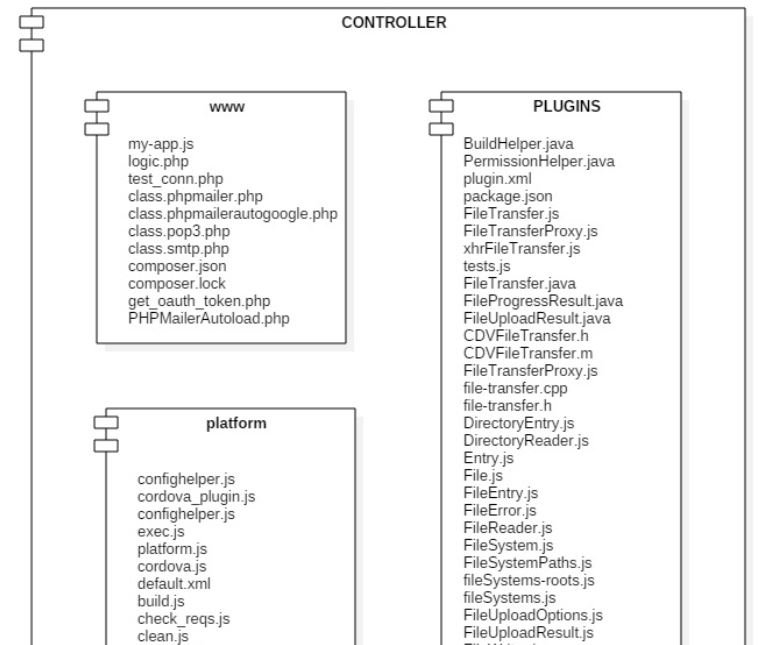
* Does not supports exporting diagrams into SVG format

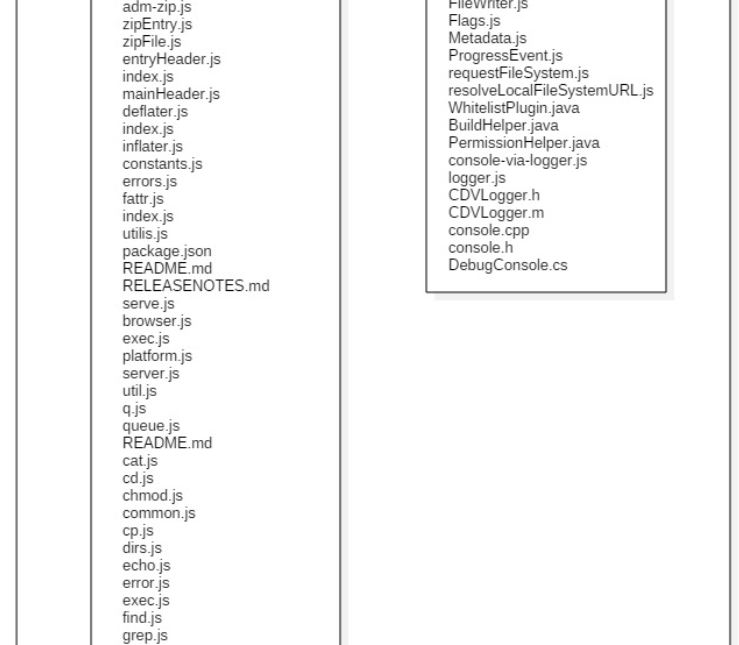
# CHAPTER 5

# PROGRAMS

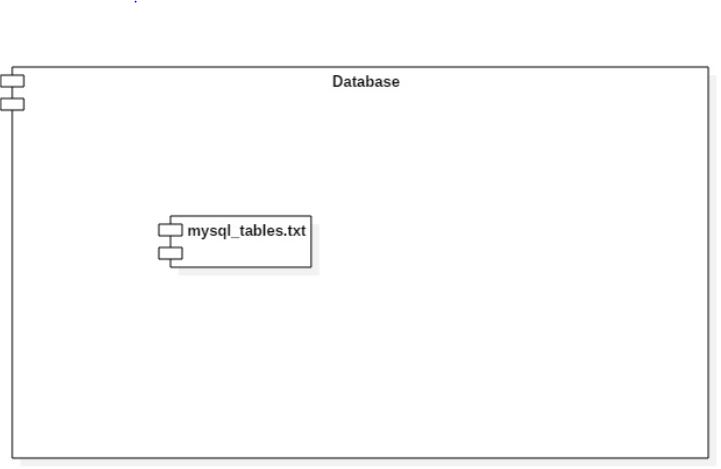
## 5.1 COMPONENT DIAGRAM

****

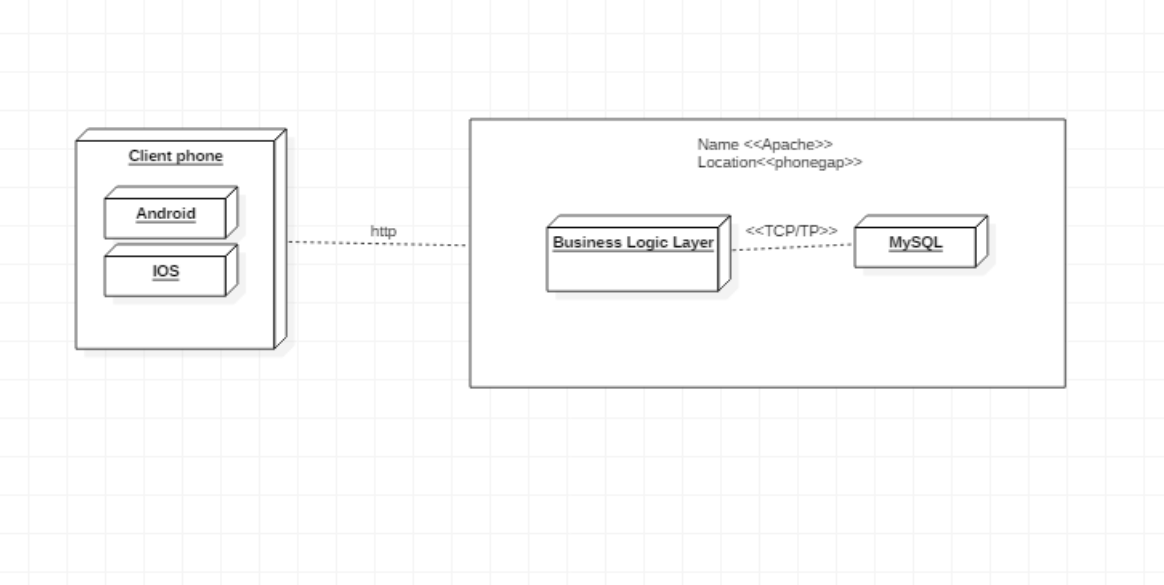
****

****

****



## 5.2 DEPLOYMENT DIAGRAM



## 5.3 CLASSES AND FUNCTIONS

The classes and functions that we implemented are:

**Functions**:

Signup ( );

Login ( );

SellABOOK ( );

DisplayAllbooks ( );

emailBookSeller ( );

addBookSeller ( );

addBookToWishlist ( );

edituserPostedBooksAction ( );

deleteBookfromWishlist ( );

showuserWishlist ( );

displayArtsBooks ( );

getBookDepartment ( );

GetBookDepartment ( );

getBookSubjectID ( );

getTodayDate ( );

**Classes:**

**Users:**

userID: int

Email:varchar

Status: varchar

Notifications: varchar

Address:varchar

City: varchar

**Books:**

BookID:int

Image:varchar

bookName: varchar

Price:float

Subject:int

sellerID: int

bookstatusID:int

Uploadstate:date

ISBNnumber:int

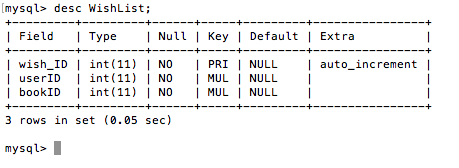
Additionalnotes:varchar

## 5.4 NUMBER OF FILES OF LAYERS/COMPONENT

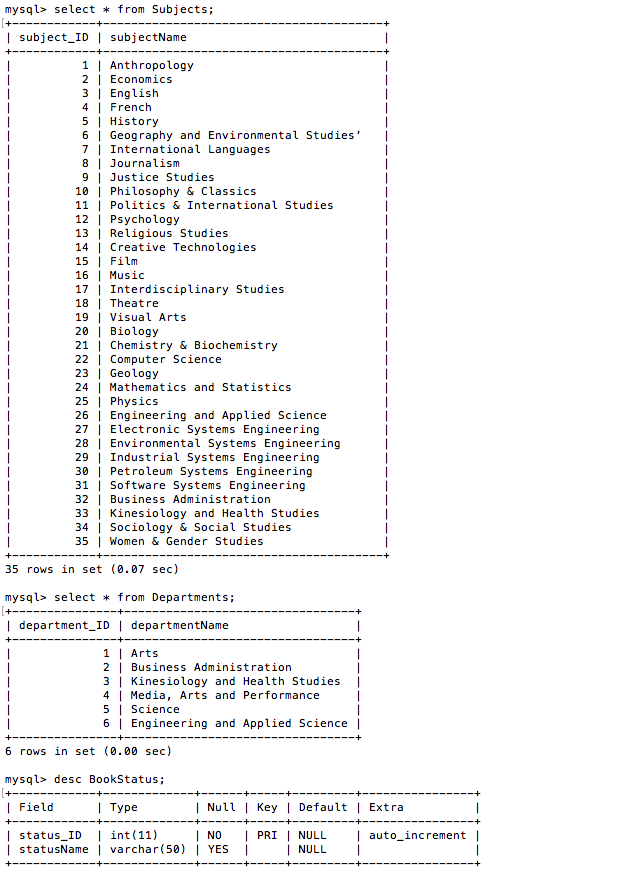
The total number of files we have on our layer and component are 166

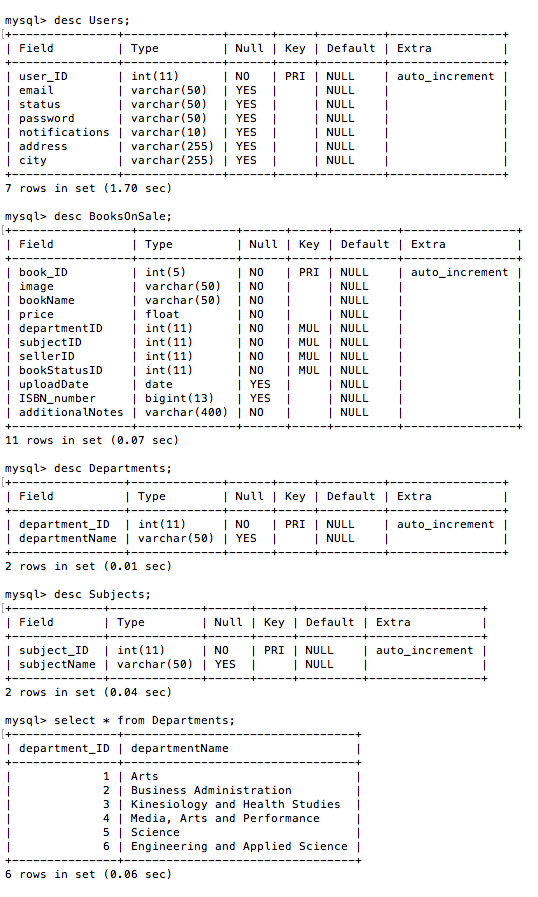
## 5.5 TABLES OF THE SYSTEM DATA

The screenshot of all the tables of the system data are below:









# CHAPTER 6

# TECHNICAL DOCUMENTATION

## 6.1 PROGRAMMING LANGUAGES

The programming languages for this project are:

* Javascript
* Php
* Sql
* Java

## 6.2 REUSED ALGORITHM AND PROGRAMS

We reused some algorithms and programs

* Google Maps
* Phonegap
* Php mailer
* Framework7

## 6.3 SOFTWARE TOOLS AND ENVIRONMENTS

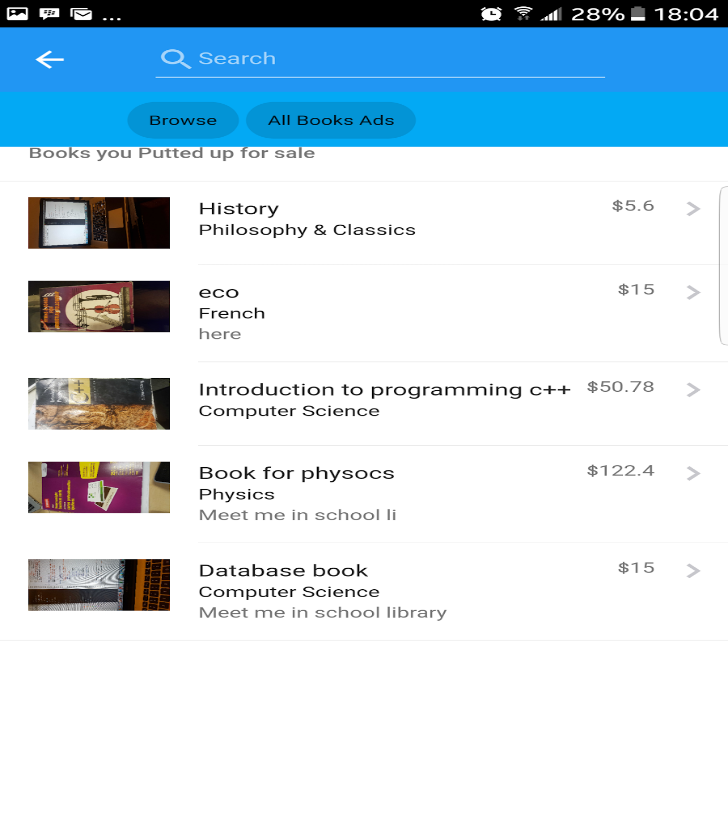
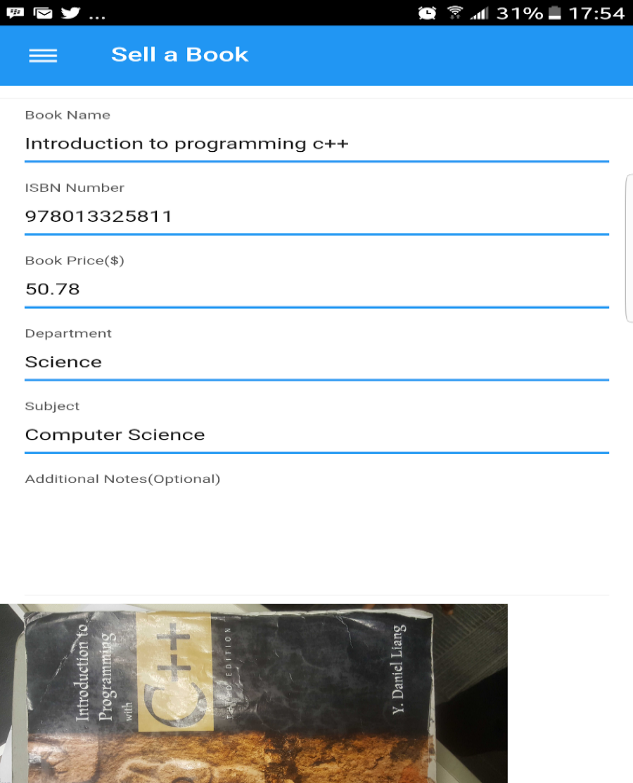
* Bracket
  + Use: This was used for programming the applications
  + Benefit: It allowed us to see how our program looks in real-time
* Phonegap
  + Use: This was used as a server for emulation for our phone
  + Benefit: It allowed us to easily deploy our app
* StarUML
  + Use: This was used for all the drawings made in this document.
  + Benefit : It allowed use to easily make our diagrams and edit it when need be

# CHAPTER 7

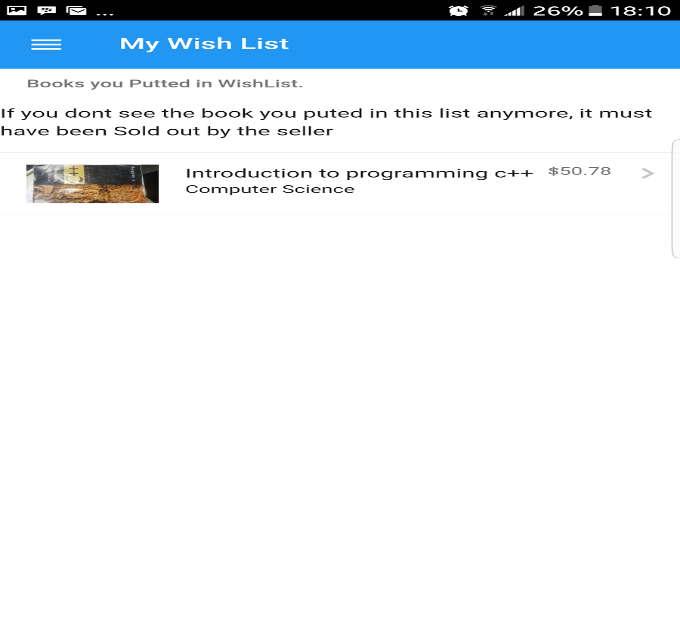
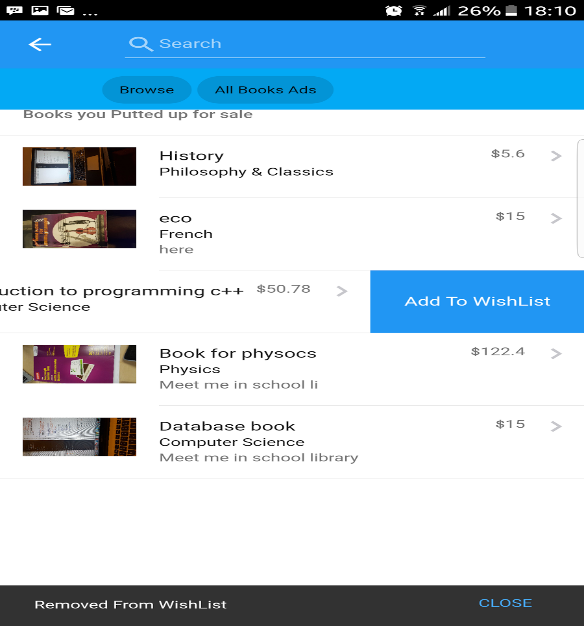
# ACCEPTANCE TESTING

## 7.1 FUNCTIONAL TESTING

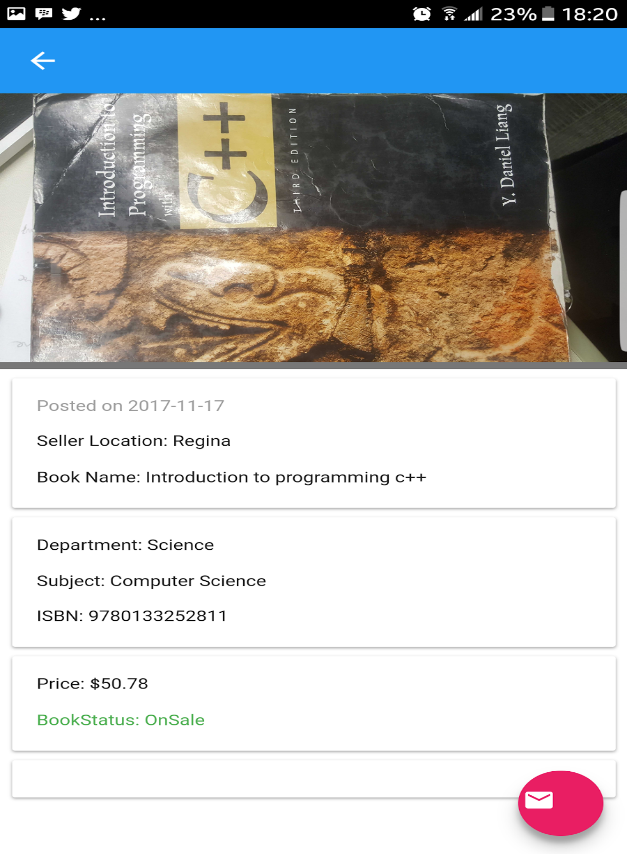
**Action:** when sellers post a book, they should be able to see the books that was posted in the all book displayed page

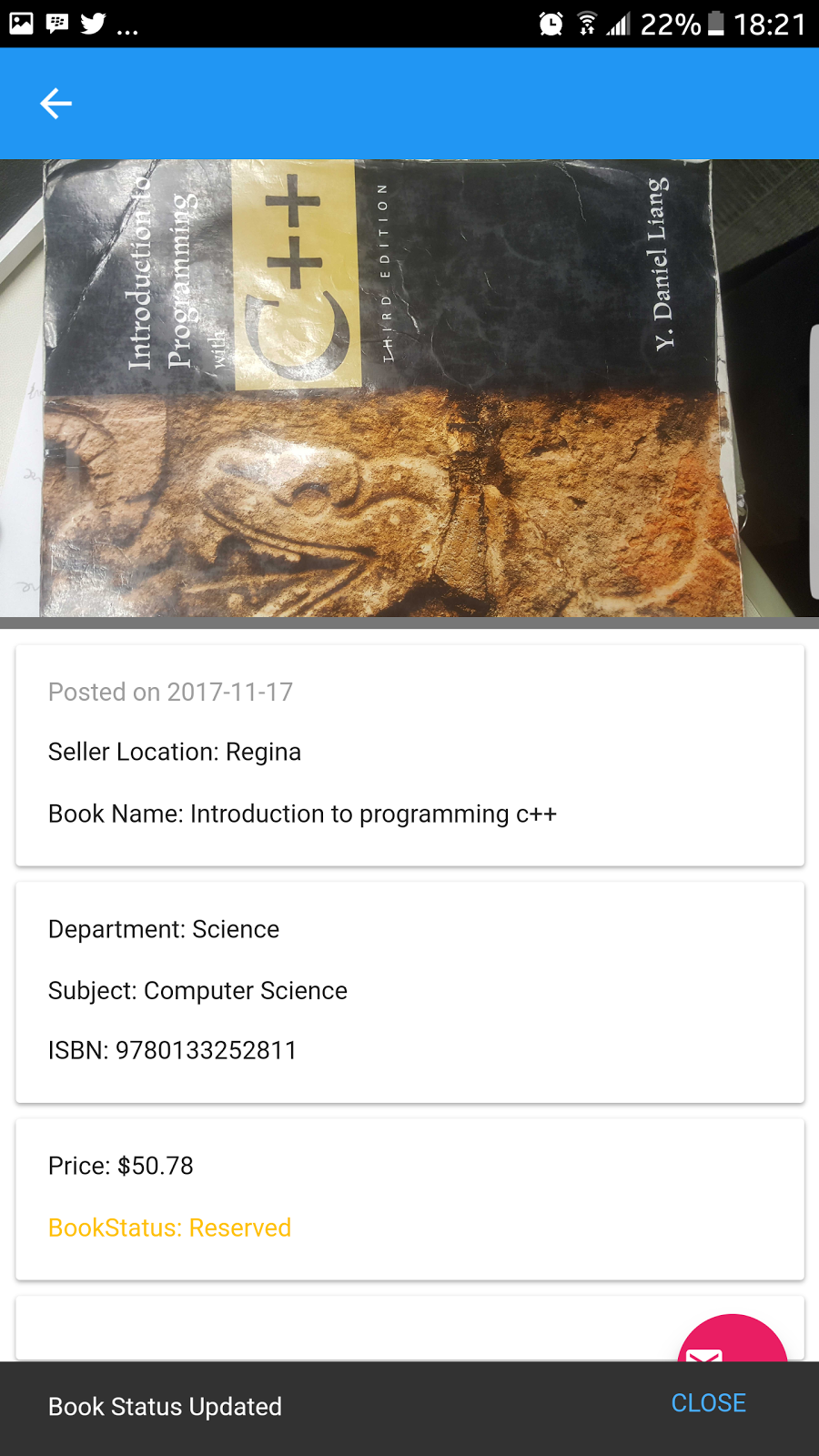


**Action:** when a buyer adds a book to his/her wish list. The user should always be able to view it in his wishlist.

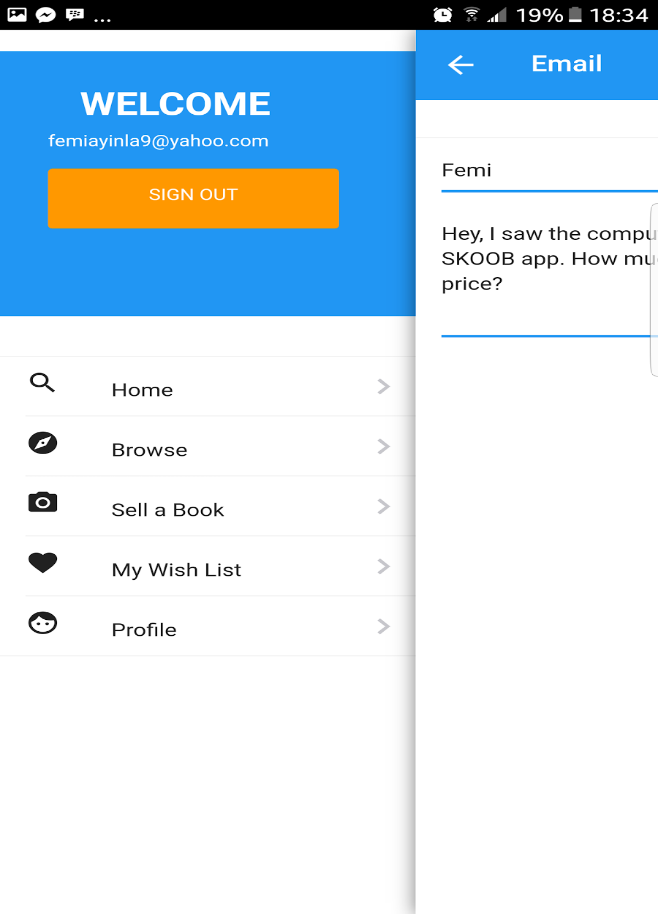


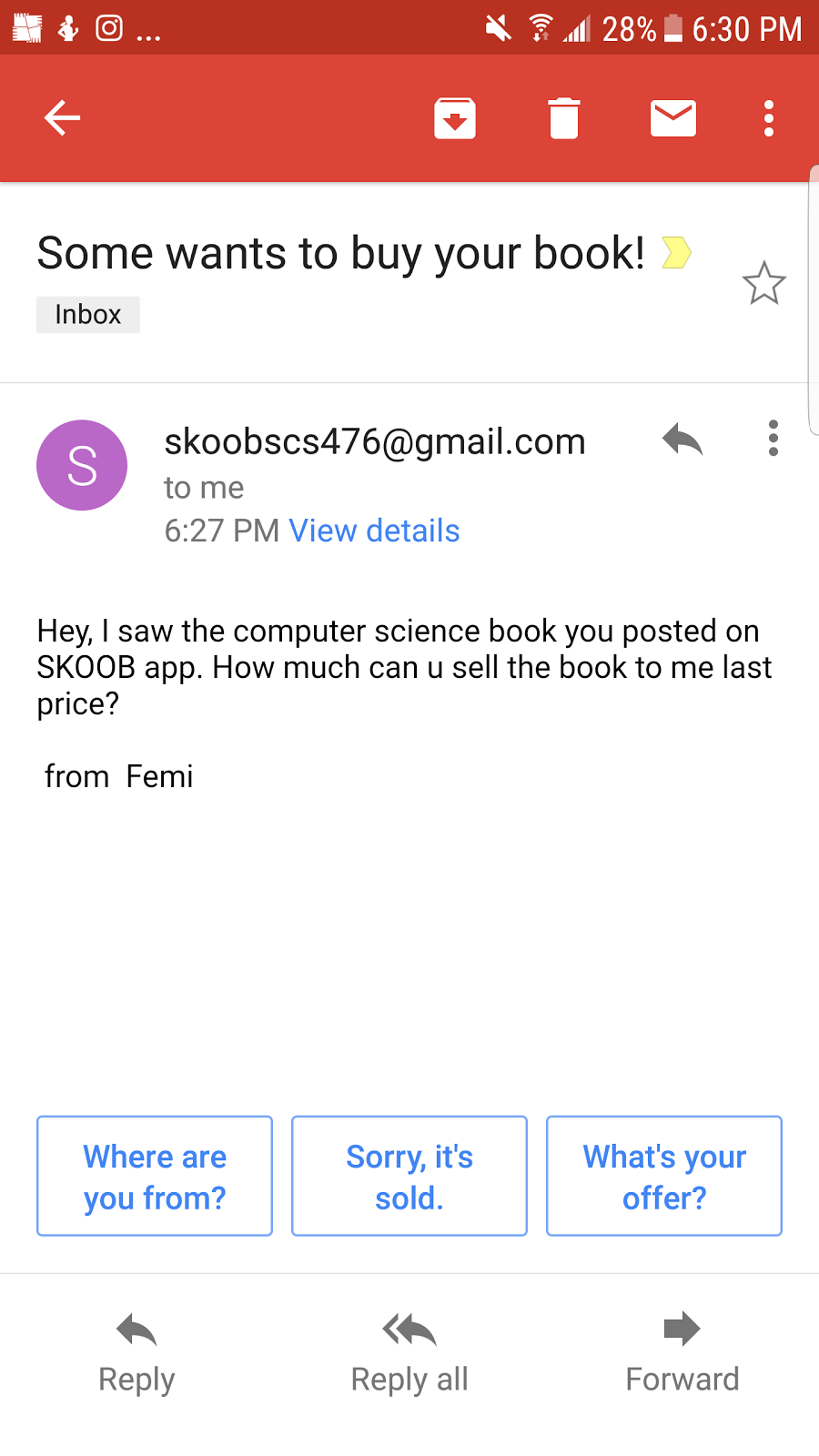
**Action:** the seller should be able to change the status of a book. When book is either being sold or reserved for someone else.



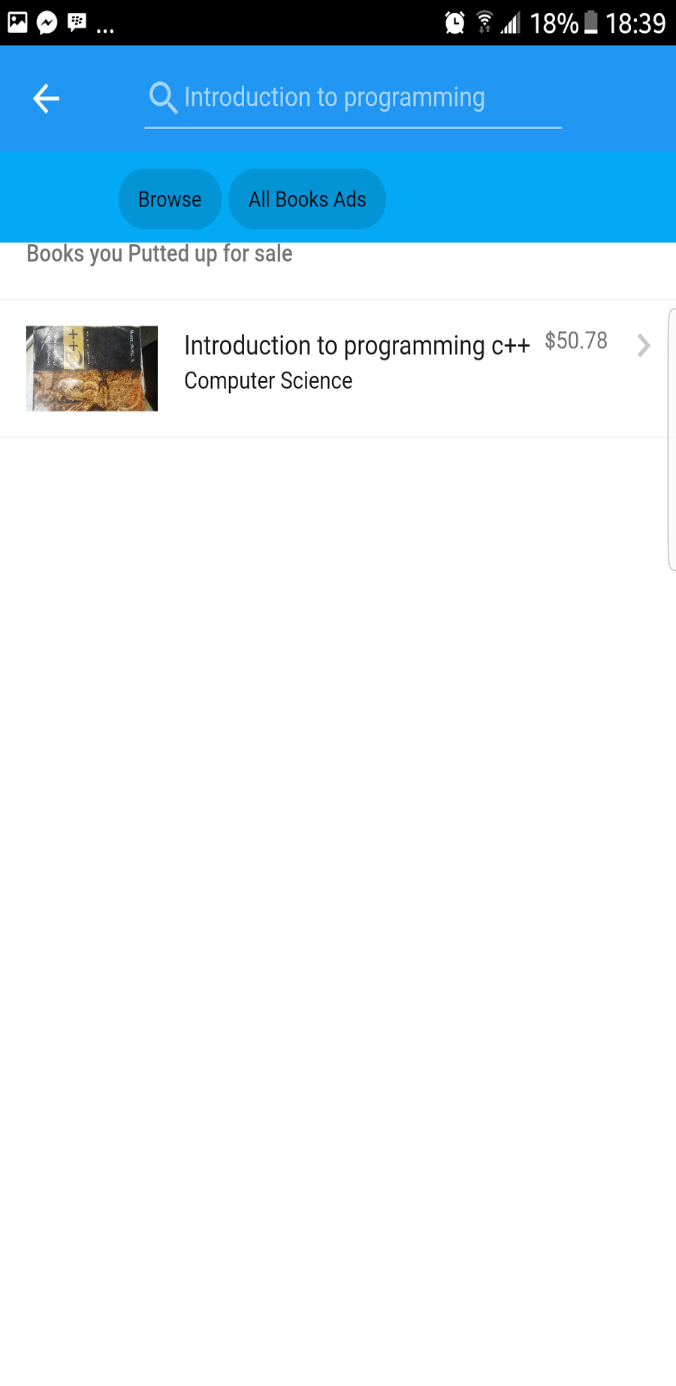
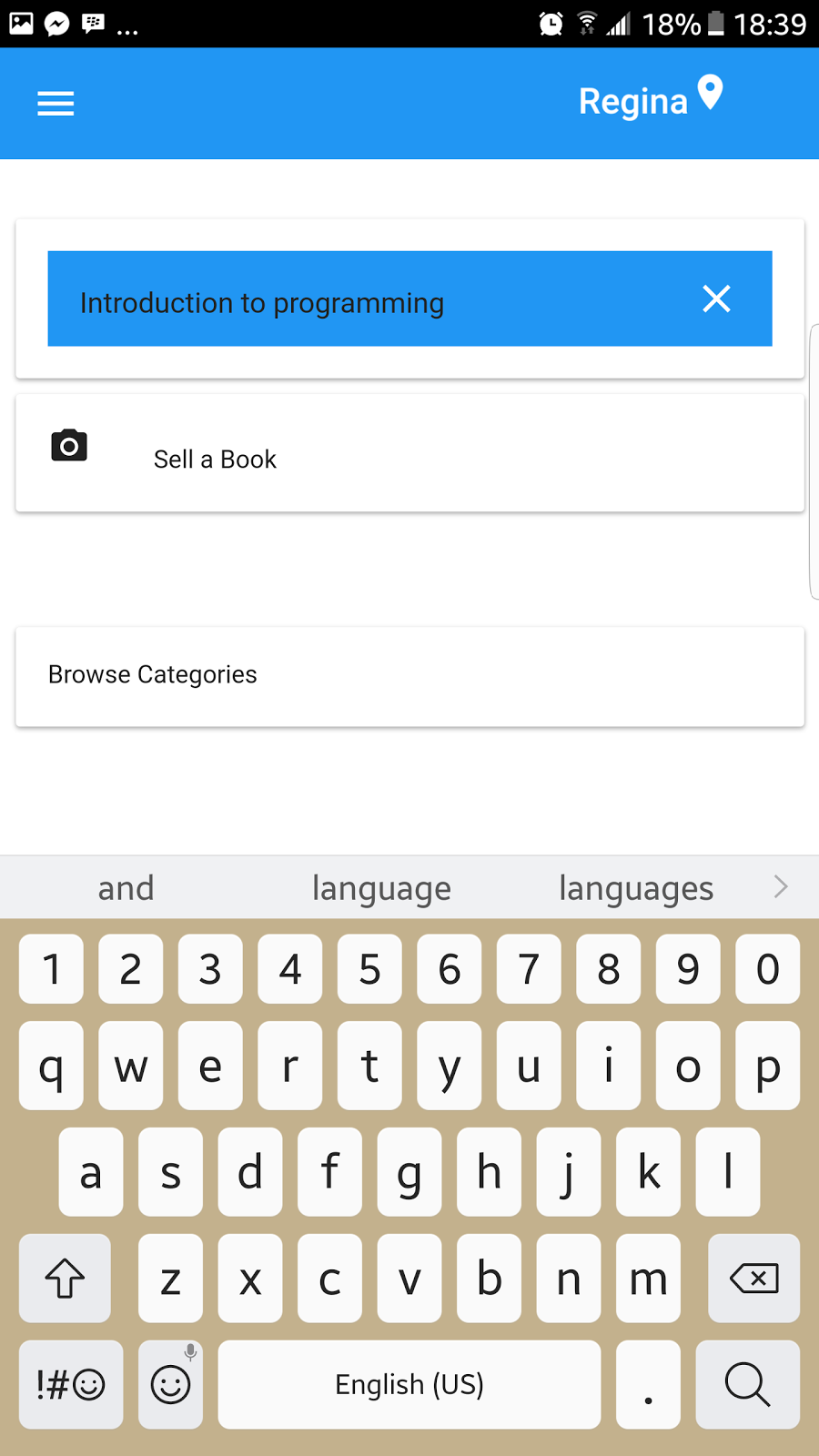


**Action:** when buyer wants to buy a book. He/ she can send an email to the seller. Which should be received successfully by the seller.



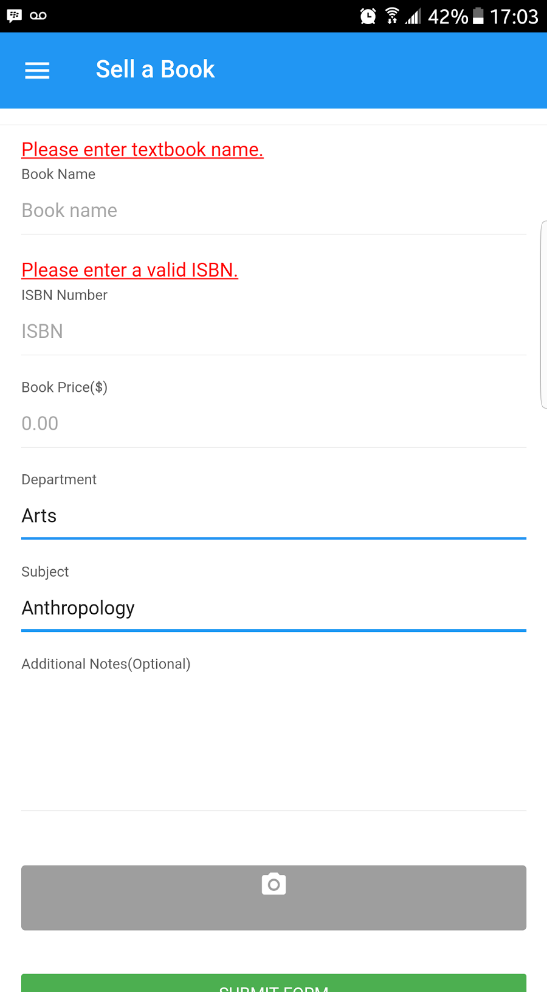
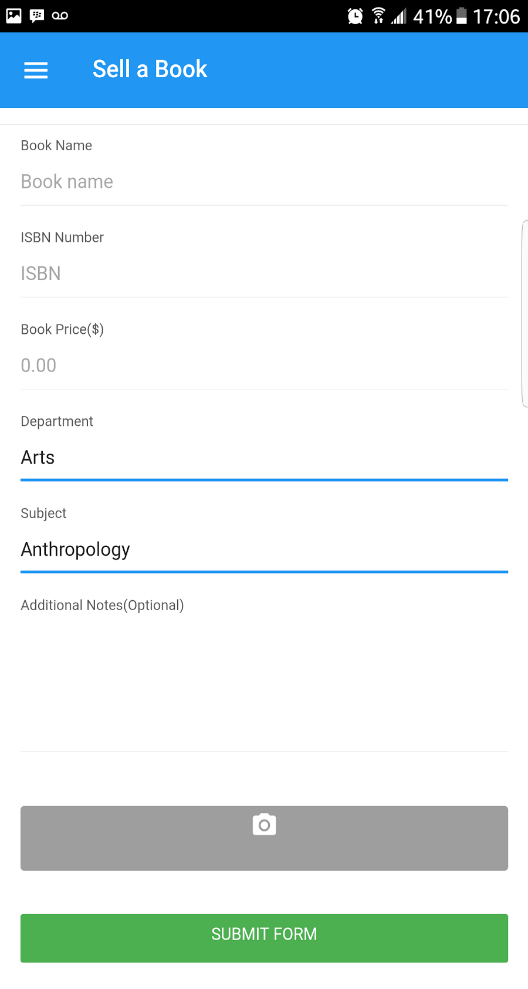
****

**Action:** when you search for a book, the book being searched for should appear, so far we have it in our database.

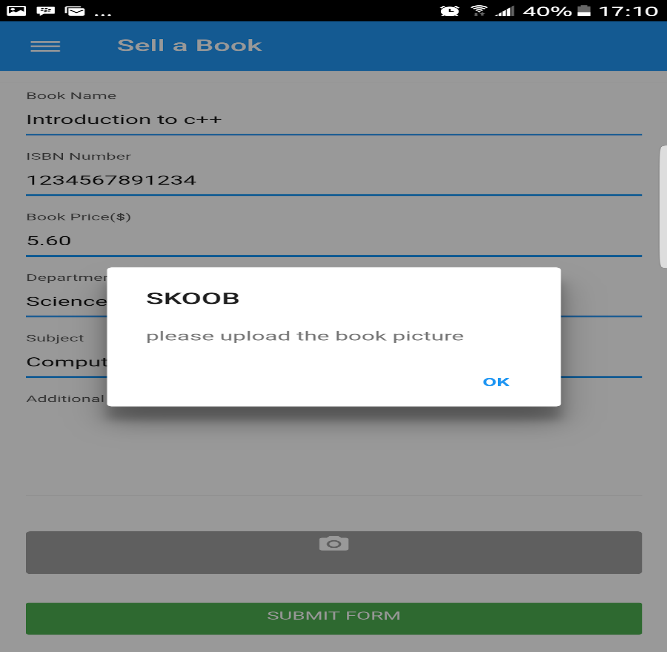


## 7.2 ROBUSTNESS TESTING

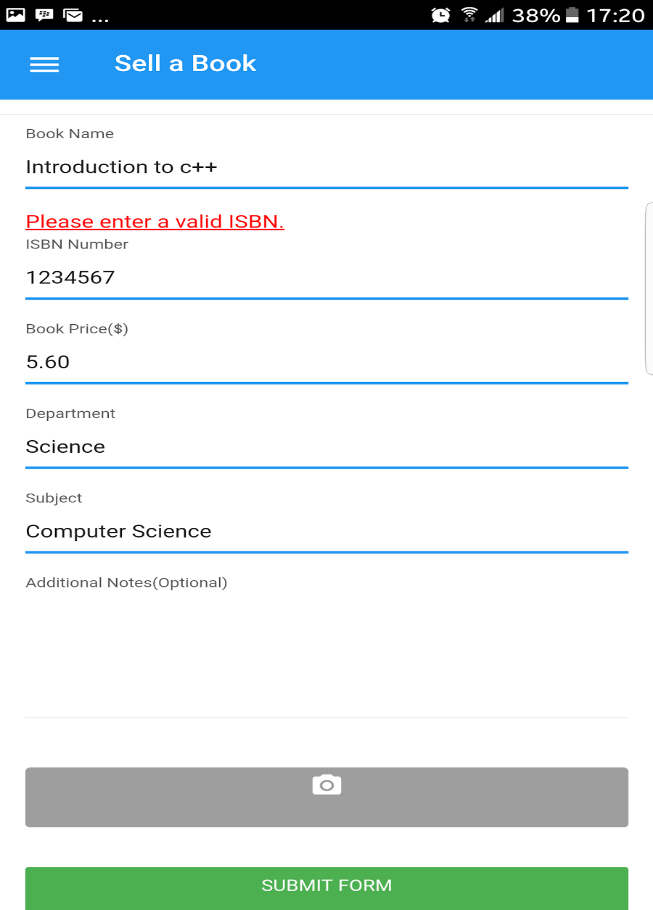
**Action:** trying to submit a book without a book name or ISBN no. You will receive an alert letting you know that you can't sell a book without filling out the information.



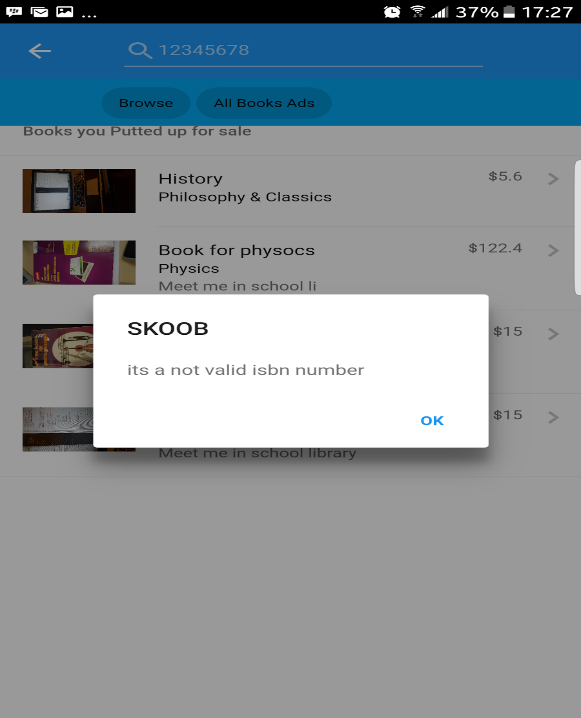
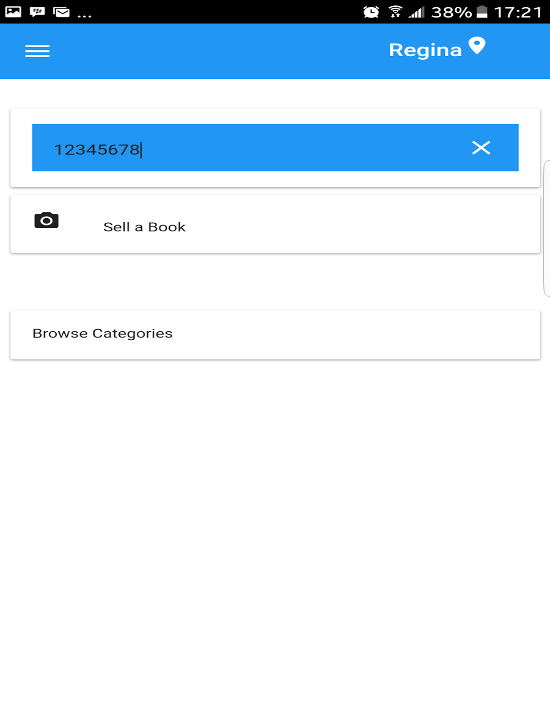
**Action:** trying to submit a book without any book picture. You will receive an alert letting you know you need to upload a picture before the posting of your book can be successful.



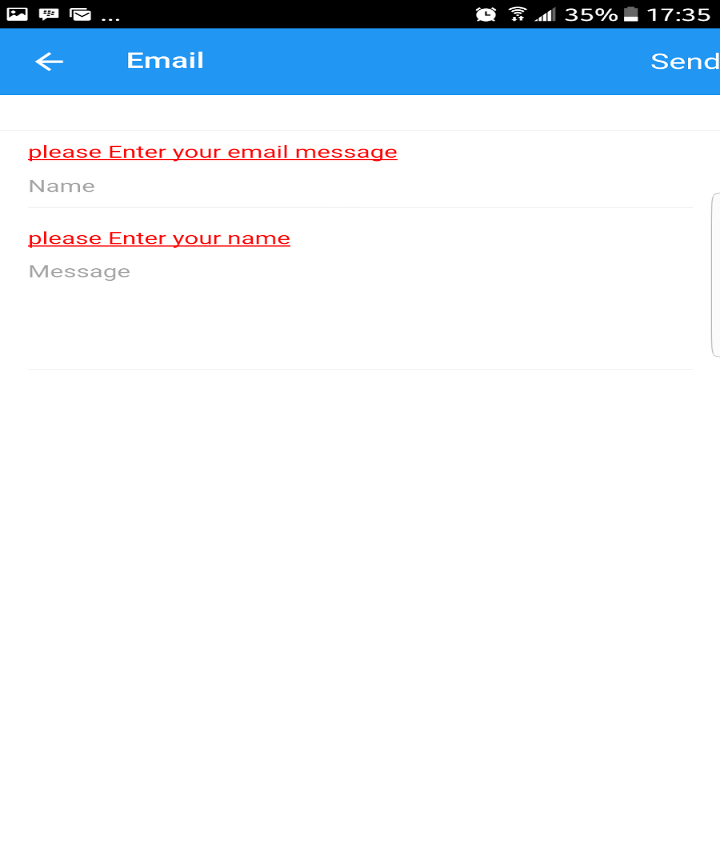
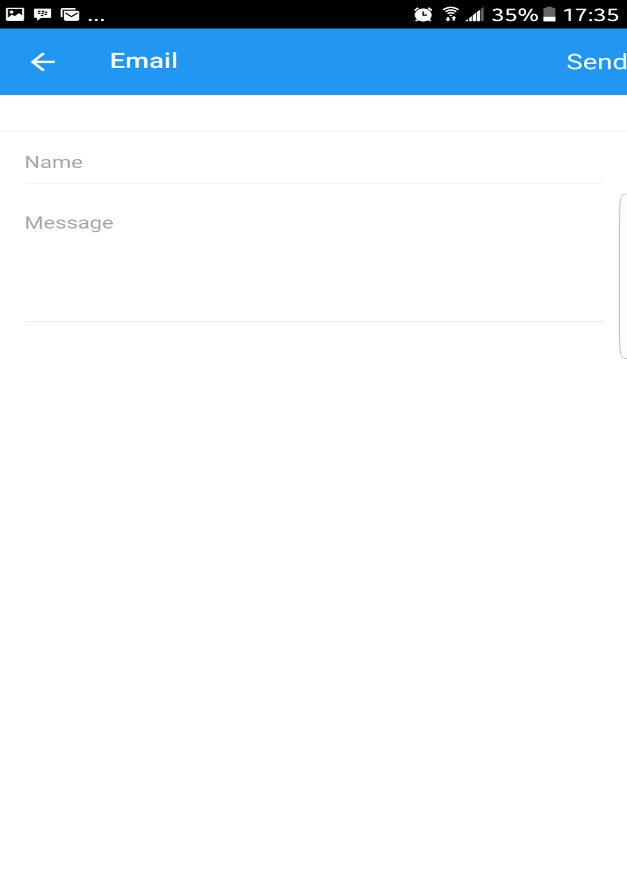
**Action:** trying to input an invalid ISBN no when selling a book. You will get an alert letting you know that the ISBN no inputted is wrong.



**Action:** when trying to search for a book using an invalid ISBN no, you will receive an alert. Letting the user know that the ISBN no entered is invalid.

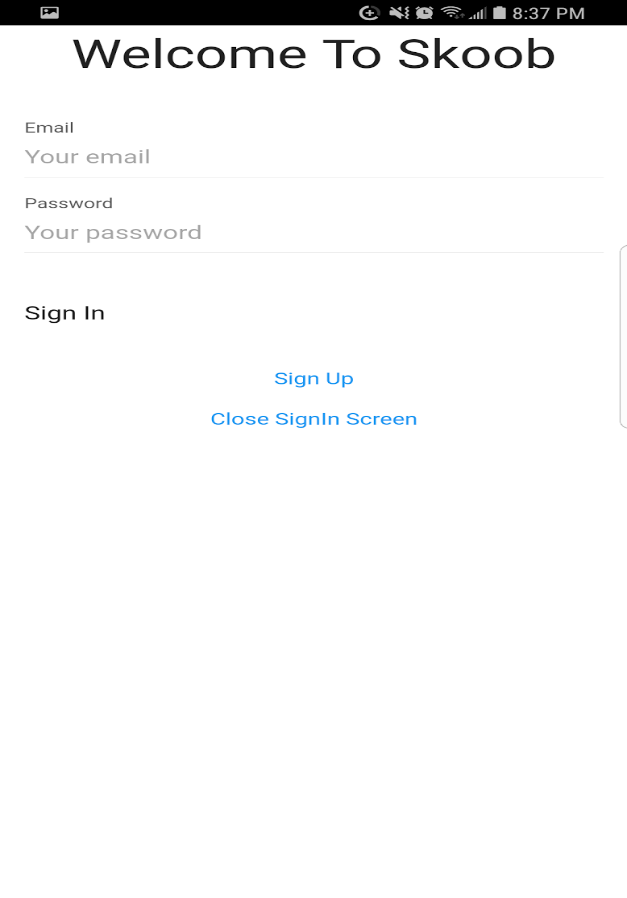
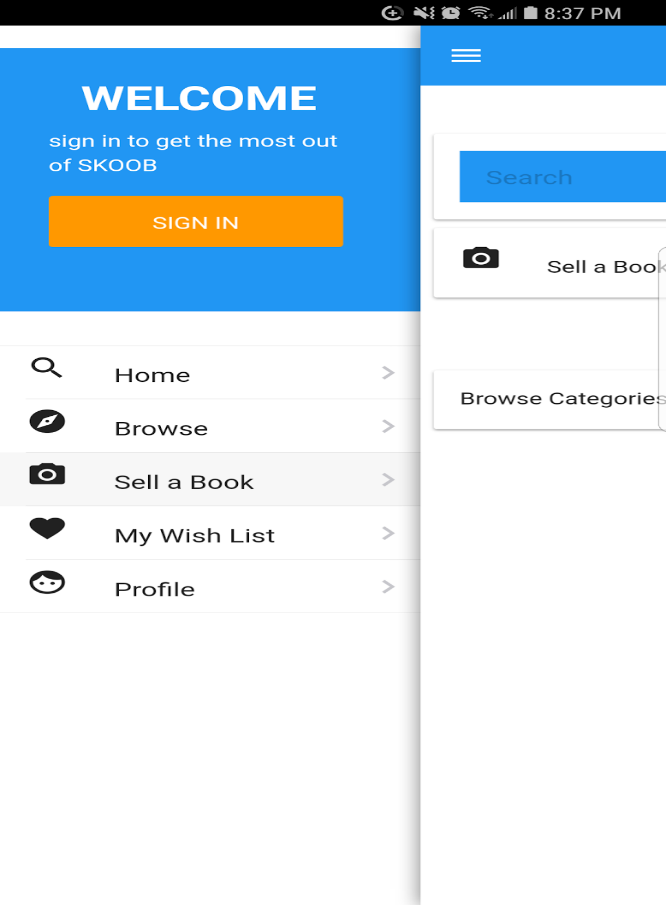


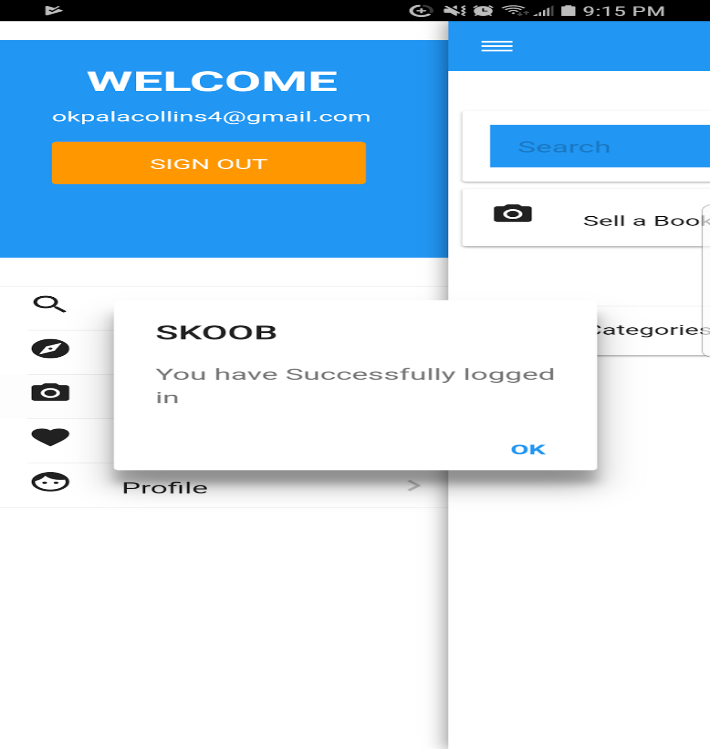
**Action:** sending an empty email to the seller. The system won't allow that to happen, therefore prompting the user to enter a text to send to the seller of the book.



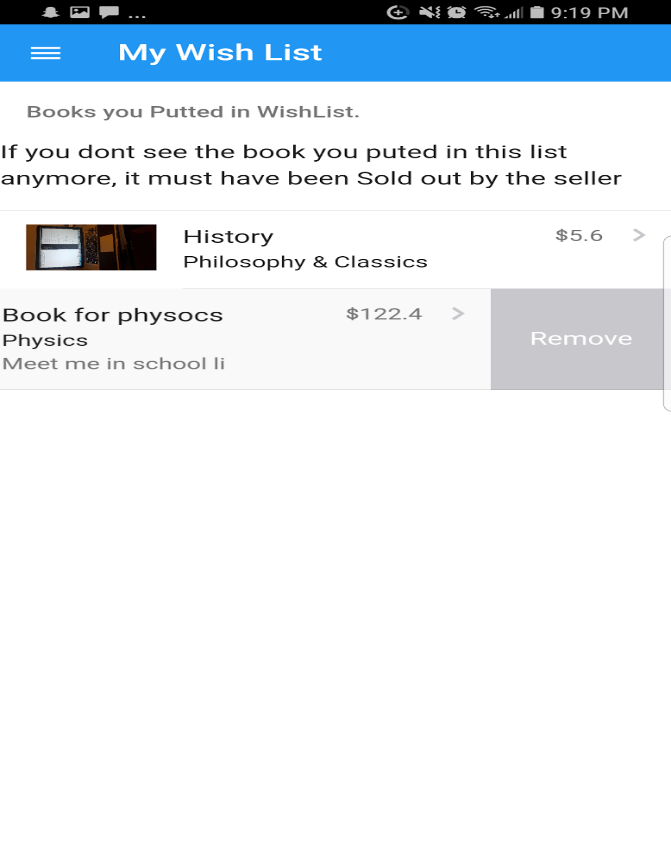
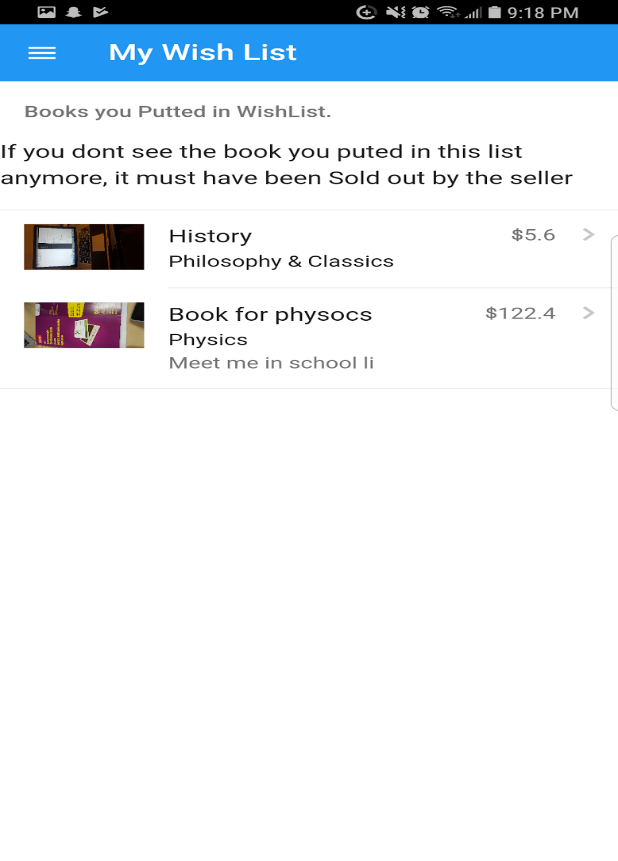
## 7.3  USER INTERFACE TESTING

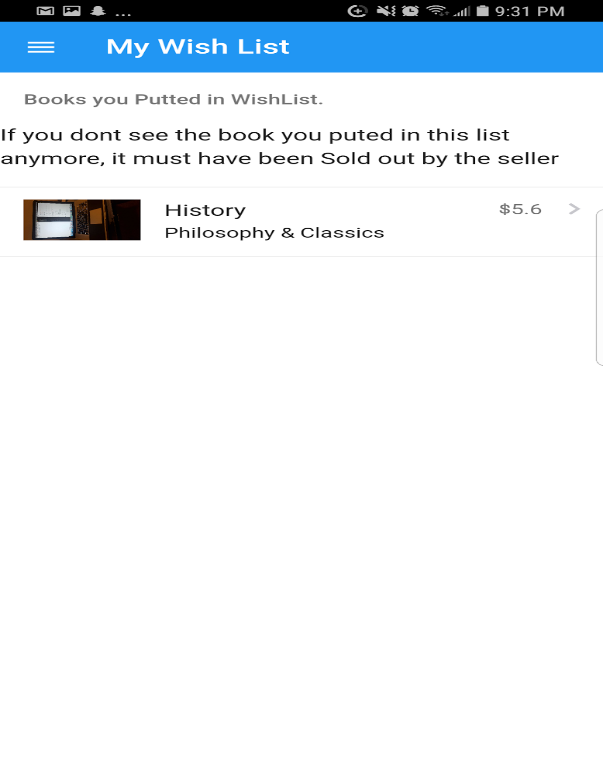
**Action:** if user isn’t signed in, user won’t be able to access some of our services. If user tries to access our services, user will be prompted to sign in.



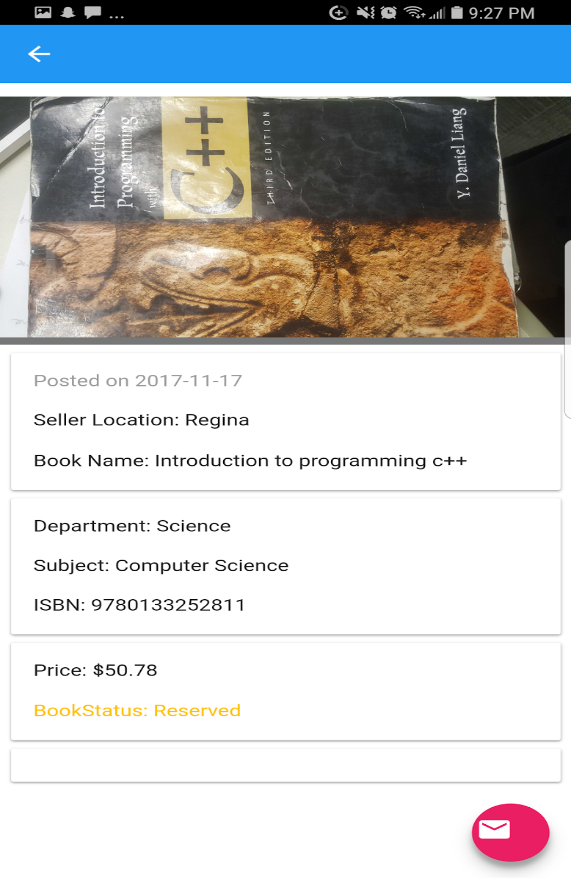


**Action:** users should be able to view his/her wishlist and also delete books from wishlist.

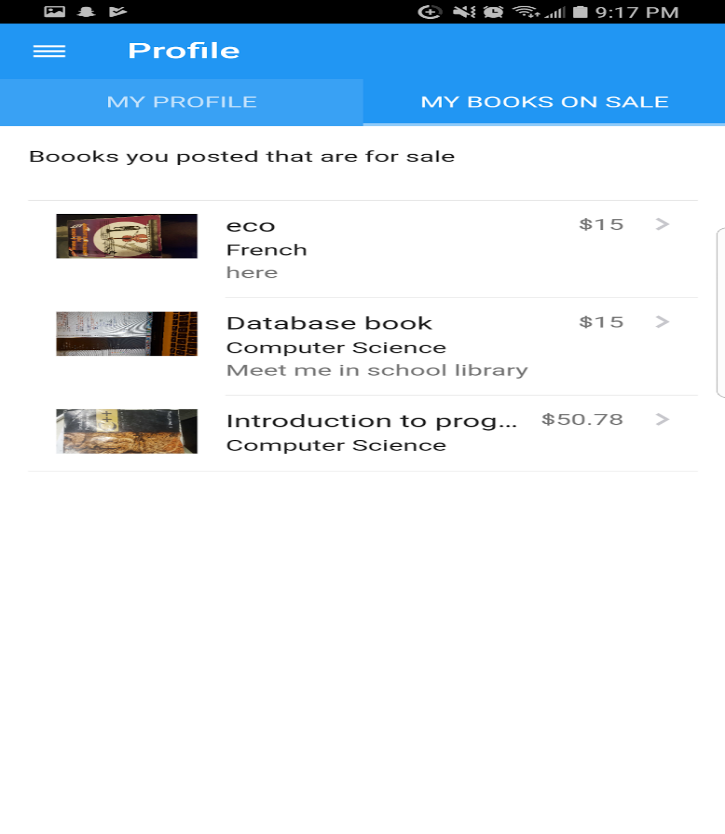
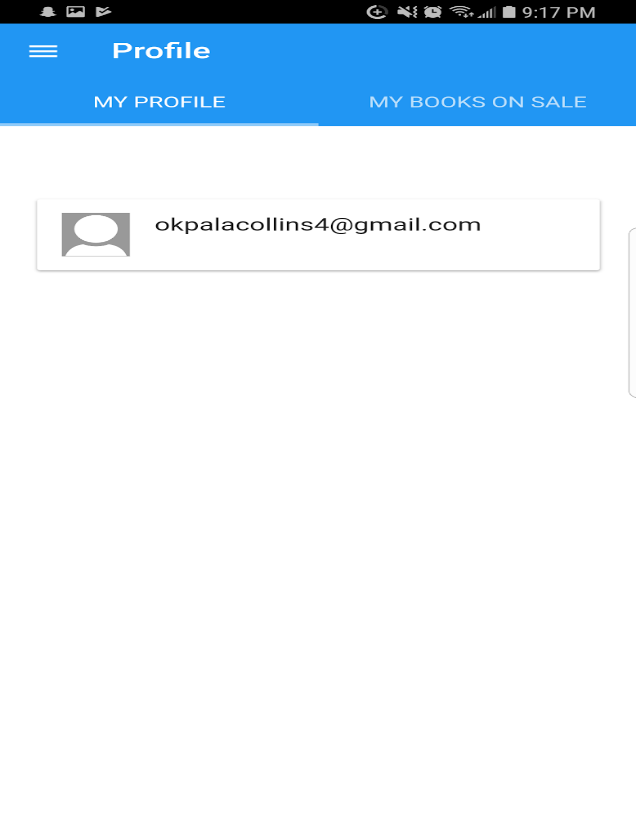




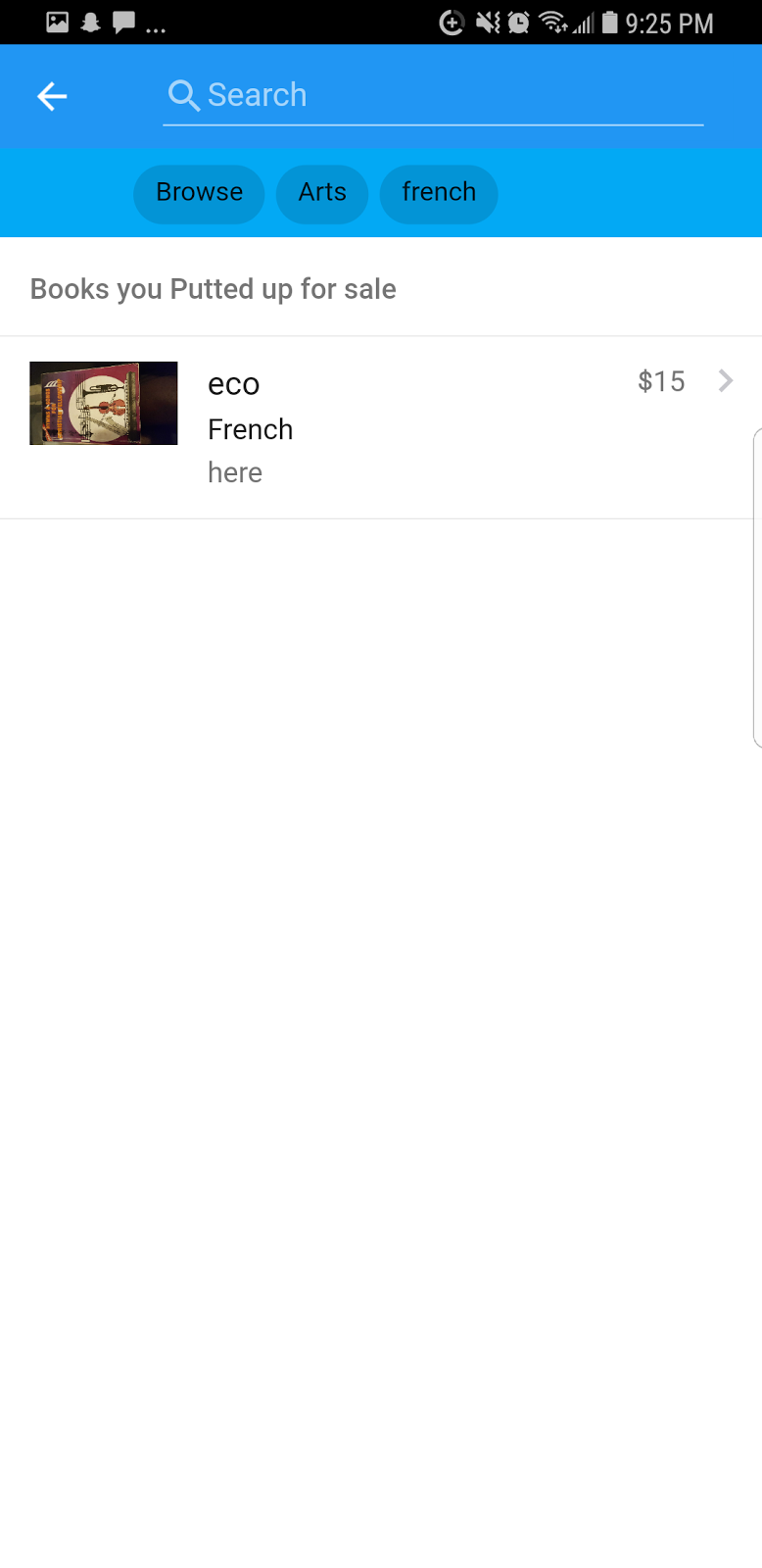
**Action:** users should be able to search for and view all the details about a particular book.



**Action:** users should be able to see their profiles and also check all books that they have posted for sale.

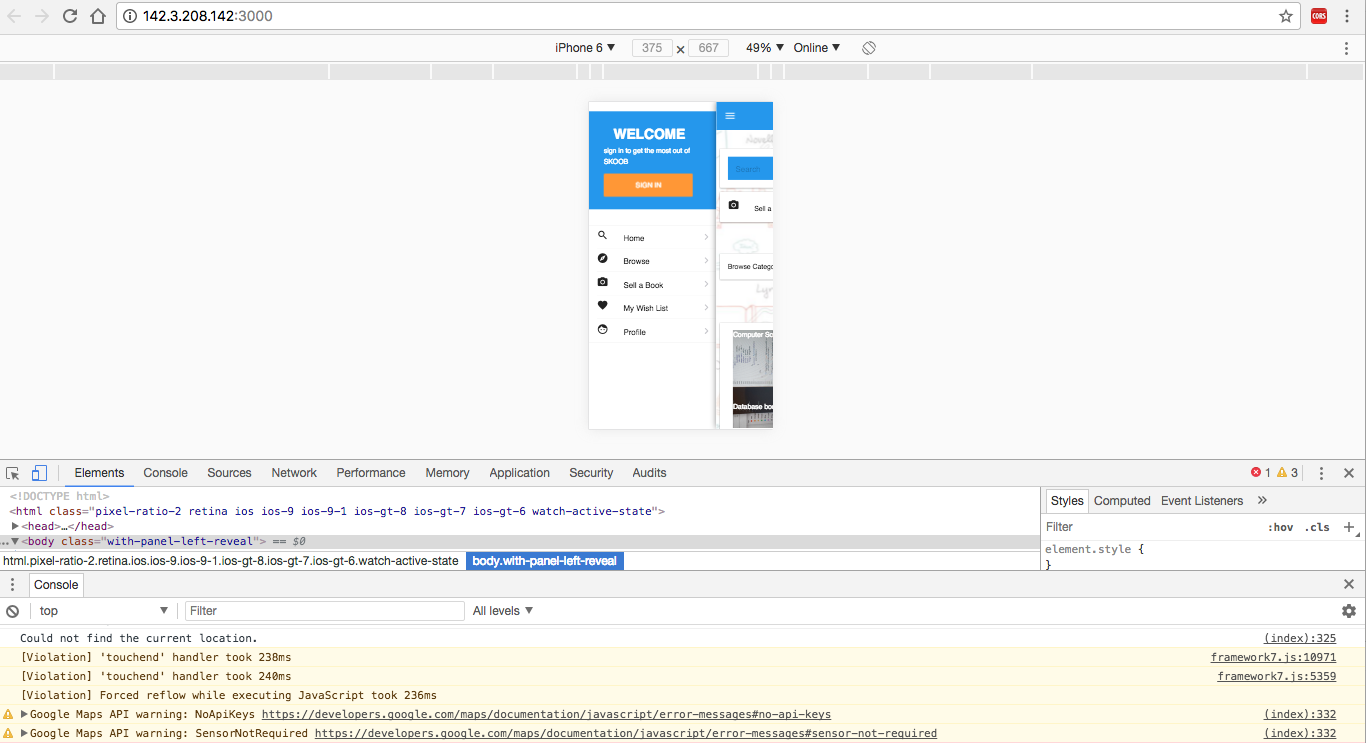


**Action:** users should be able to browse through books using subjects. For example you can browse for French books. i.e you want to learn French.



## 7.4  TIME-EFFICIENCY TESTING

Action:



# CHAPTER 8

# TEAM MEMBER CONTRIBUTION

## 8.1    GROUP MEMBER CONTRIBUTION

**Chidiebere Okpala**

* CODES
* SOFTWARE REQUIREMENT SPECIFICATION
* DESIGN SPECIFICATION
* TECHNICAL DOCUMENTATION
* ACCEPTANCE TESTING

**Temitope Omoyefa**

* SOFTWARE REQUIREMENT SPECIFICATION
* DESIGN SPECIFICATION
* PROGRAMS
* TECHNICAL DOCUMENTATION
* ACCEPTANCE TESTING
* TEAM MEMBER CONTRIBUTION

**Junia Ayinla**

* VALIDATION
* PROBLEM DEFINITION
* ECONOMIC FEASIBILITY STUDY
* SOFTWARE REQUIREMENT SPECIFICATION
* DESIGN SPECIFICATION
* TECHNICAL DOCUMENTATION
* ACCEPTANCE TESTING

References

<https://veerasundar.com/blog/2008/11/staruml-and-argouml-compared/>

<https://www.izenda.com/blog/5-benefits-3-tier-architecture/>

<https://sourcemaking.com/design_patterns/observer>

<https://sourcemaking.com/design_patterns/factory_method>

<http://docs.phonegap.com/en/edge/cordova_geolocation_geolocation.md.html#Geolocation>

<https://framework7.io/docs/get-started.html>

<https://www.tutorialspoint.com/framework7/>

<https://github.com/PHPMailer/PHPMailer> <http://docs.phonegap.com/en/edge/cordova_camera_camera.md.html>