SOFTWARE REQUIREMENTS SPECIFICATION

Version 1.1

17.01.2013

MOBCOLL PROJECT

Prepared By: ANDIOS

Murat Öksüzer

Sercan Çidem

Vedat Şahin

Fatih Osman Seçmen

Change History

*A - ADDED M - MODIFIED D – DELETED

VERSION NUMBER	DATE	NUMBER OF FIGURE, TABLE OR PARAGRAPH	A* M D	TITLE OF BRIEF DESCRIPTION
1.0	11.11.2012			Original
1.1	17.01.2013	Figure1 Figure 2	A,	Screenshots from Grifitth and Flinsoft are added
		Preface 2.1 Figure 3 Figure 4 3.2.1 4.1.1 Figure 5	M,	Preface, Product Perspective, User Use Cases, Data Objects and ER Diagram are modified.

Preface

This document contains the system requirements for MOBCOLL project. This document is prepared according to scaled version of the “IEEE Recommended Practice for Software Requirements Specification – IEEE Std 830 – 1998” for CENG491.

This Software Requirements Specification provides a complete description of all the functions and specifications of the MOBCOLL project.

The first section of this document includes purpose, scope, references, definitions and overview of the document.

The second section of this document includes product perspectives, the use cases of MOBCOLL project, assumptions and dependencies. Use cases are documented for helping the requirements to be understood clearly.

The third section of this document includes specific requirements of the MOBCOLL project.

The fourth section of this document includes data model and description of the project.

The fifth section of this document includes team structure, planning and process model of the project.

The sixth section of this document includes conclusion of the project.

Table of Contents

List of Figures

1.0 Introduction

1.1 Problem Definition

People collecting items often had difficulty in keeping track of their collection repository. Storage and location, availability, missing items, wanted items, borrowed items, web and shopping searching are among tasks that a collector might need help.

1.2 Purpose

The purpose of this document is to present a detailed description of the COLLMOB, an android application. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for the acquirer which is the owner in this case.

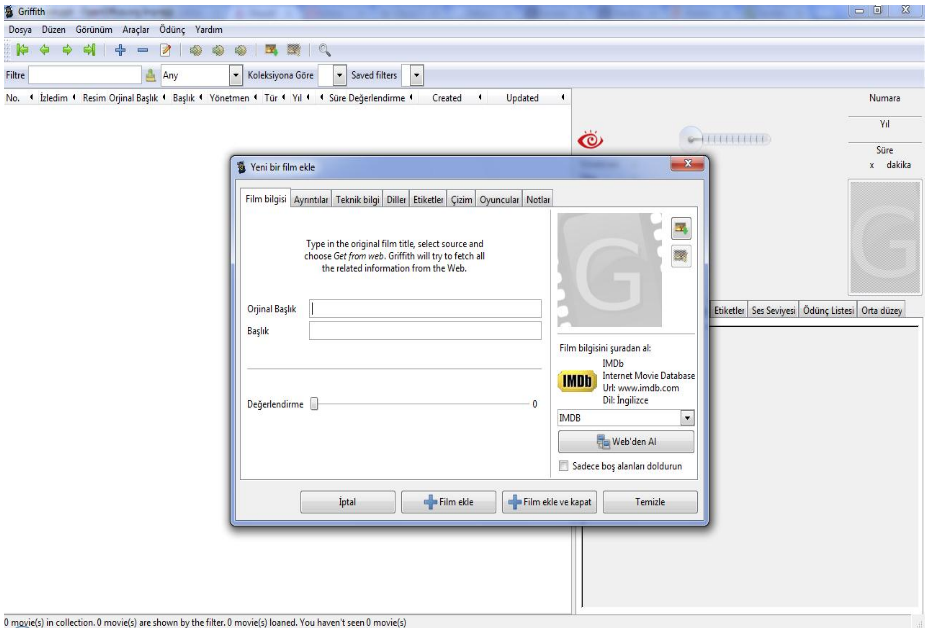
1.3 Scope

This software system will be an android application for an owner of a book collection. This system will be designed to help the owner to:

- Manage the information about owned items
- Track the place of the owned items (on the shelves, notebooks, closets, drawers etc.)
- Image based search of item (based on barcodes or item picture taken by the phone) on repository.
- Tracking of borrows or usage statistics if applicable
- Wanted items management and gift requests.

1.4 User and Literature Survey

There are two projects which are likely to our project. The first one is Griffith media collection manager application that adds items to the collection quickly and easily like typing the film title and selecting a supported source. Also Grifitth media collector can fetch movies and related information about them from Amazon and IMDB. Grifitth have its own database but it does not have mobile application version.


Griffith
Dosya Düzen Görünüm Araçlar Ödünç Yardım
Filtre
Any
Koleksiyona Göre
Saved filters
No. İzledim Resim Orjinal Başlık Başlık Yönetmen Tür Yıl Süre Değerlendirme Created Updated
Numara
Yıl
Süre
x dakika
Yeni bir film ekle
Film bilgisi Ayrıntılar Teknik bilgi Diller Etiketler Çizim Oyuncular Notlar
Orjinal Başlık
Başlık
Değerlendirme
0
Film bilgisini şuradan al:
IMDB
Web'den Al
Sadece boş alanları doldurun
İptal
Film ekle
Film ekle ve kapat
Temizle
Etiketler Ses Seviyesi Ödünç Listesi Orta düzey

Type in the original film title, select source and choose *Get from web*. Griffith will try to fetch all the related information from the Web.

IMDb
Internet Movie Database
Url: www.imdb.com
Dil: İngilizce

0 movie(s) in collection. 0 movie(s) are shown by the filter. 0 movie(s) loaned. You haven't seen 0 movie(s)

Figure 1 – A Screenshot from Grifitth application

The second program is Flinsoft which is a hobby software collecting program to catalogue and storage every collection according to your requirements. But activation is too expensive and it does not have mobile application version.

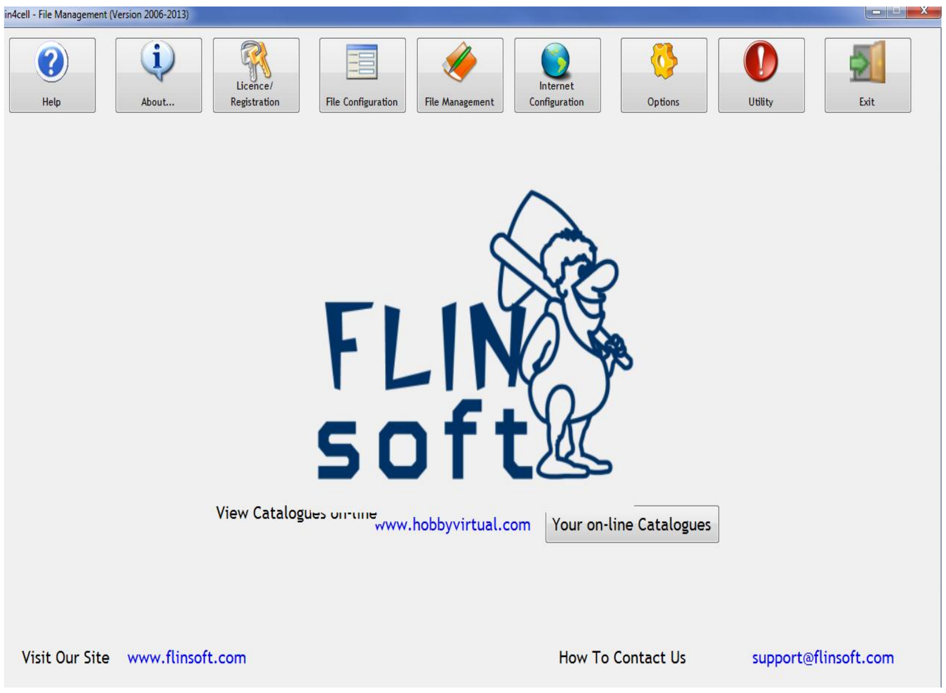


Figure 2 – A Screenshot from Flinsoft application

1.5 Definitions and Abbreviations

Term	Definition
Database	Collection of all the information monitored by this system.
User	The book collector.
Android	A mobile device operating system developed by Google Inc.
Social media	Facebook or twitter.
ISBN	The International Standard Book Number (ISBN) is a unique numeric commercial book identifier.
IEEE	The Institute of Electrical and Electronics Engineers (IEEE) is a professional association headquartered in New York City that is dedicated to advancing technological innovation and excellence.
Software Requirements Specification	A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document.

1.6 References

IEEE STD 830-1998, IEEE Recommended Practice for Software Requirements Specifications IEEE Computer Society, 1998.

1.7 Overview

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

2.0 Overall Description

In this part, background information about specific requirements of the system will be provided briefly. General issues that affect the product and outline of the functional requirements will be mentioned, too.

2.1 Product Perspective

Our system will consist of three components.

1- Android application

2- Restful Web Service

3- Database

Users reach to the functions of the system via android application which is connected to the database by using restful web service. After each operation, the changes shall be reflected to the database. The android application shall deal with the database interaction issues, such as searching and retrieving data from the database.

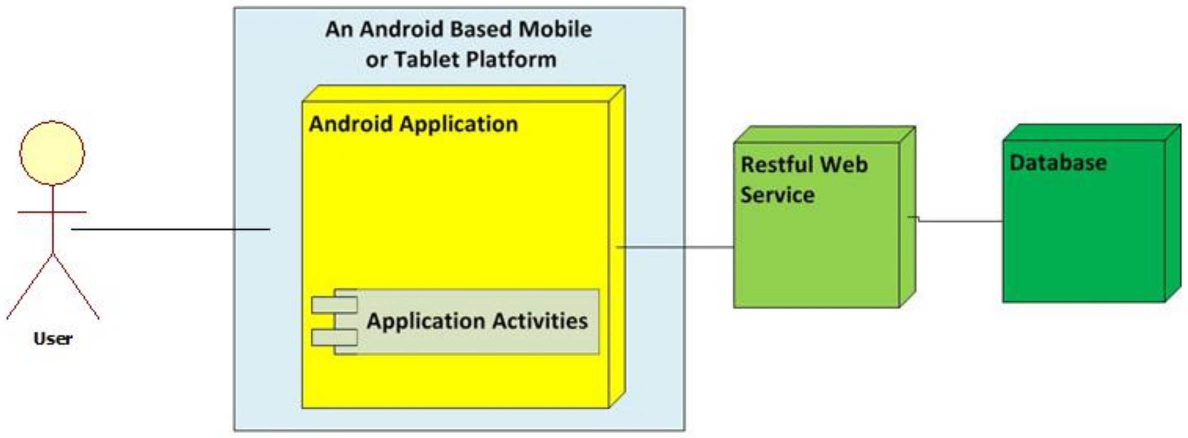


Figure 3 – Product Perspective

2.2 Constraints, Assumptions and Dependencies

- The user is expected to be a book collector.
- We will use ISBN (International Standard Book Number) to identify a book.

- Privacy of users will be protected.

3. Specific Requirements

In this section and its subsections, we will explain all the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements.

3.1 Interface Requirements

Basically the system can have two interfaces. One is between the user and the application. Besides that the user may want to select scenarios from the hard disc and there shall be an interface between the application and the file system.

3.2 Functional Requirements

This section outlines the use cases for user.

3.2.1 User Use Cases

The user has the following sets of use cases:

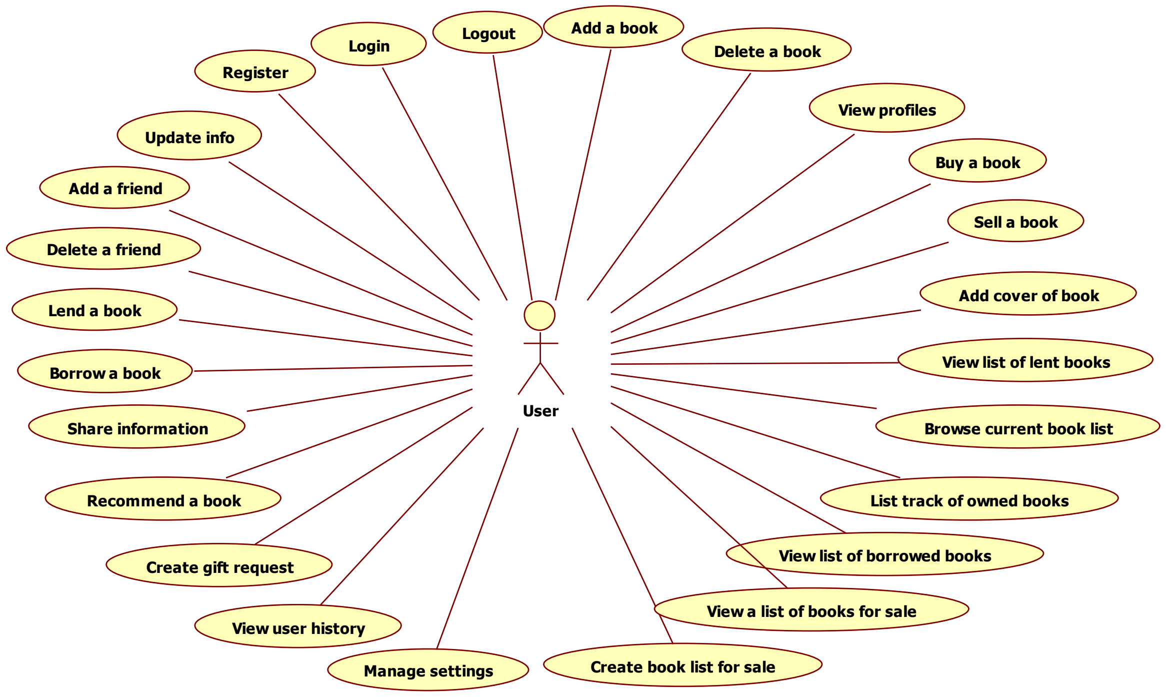
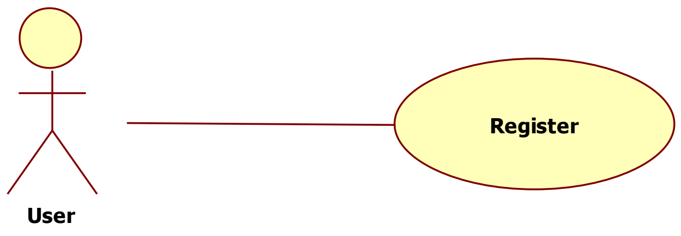


Figure 4 – User Use Cases

3.2.1.1 Use Case: Register

Diagram:



Brief Description

The user enters the personal information i.e. name, surname, username, password, e-mail and phone number and register to the system.

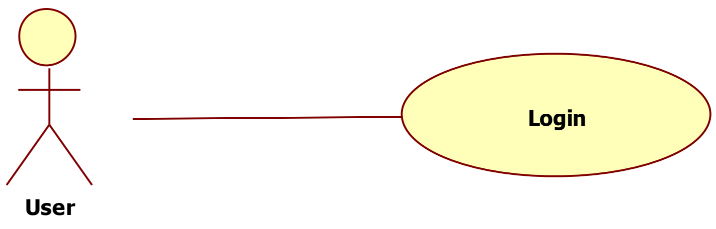
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user selects the register button in the welcome screen.
2. The system provides a list of boxes for user to enter the personal information and a password.
3. The user fills the information gaps, and submits the form by touching the “Sign Up” button.
4. The system creates the user account and returns the user to the MobColl login page.

3.2.1.2 Use Case: Login

Diagram:



Brief Description

For this use case to be initiated, the user has already accessed MobColl application.

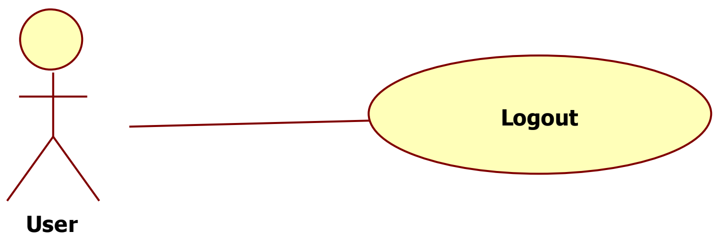
Step-By-Step Description

For this use case to be initiated, the user should start the application.

1. The user enters the e-mail and password and selects login.
2. The system checks the e-mail and password pair.
3. If the e-mail and password pair is correct, the system redirects user to main page of MobColl.
4. If the pair is not valid, it returns to the login page with an error message "E-mail or password is wrong".
5. The user can also use the "Login with Facebook" button to login if he/she has a Facebook account.

3.2.1.3 Use Case: Logout

Diagram:



Brief Description

The user logged into the server or android application before logs out from the system.

Step-By-Step Description

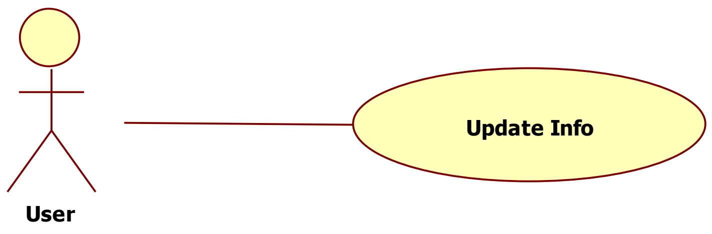
For this use case to be initiated, the user has already accessed MobColl application.

1. The user touches the "logout" button.

2. The system logs out the user and redirects user to login page.

3.2.1.4 Use Case: Update Info

Diagram:



Brief Description

The user changes his/her personal information on the system.

Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user selects “User Profile” page from left menu.

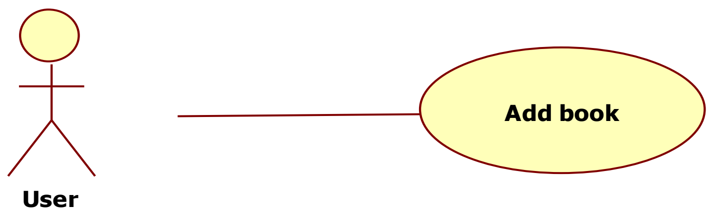
2. The system provides personal information like name, surname, e-mail, password etc. of the user.

3. The user enters new information to user info fields and touches “Save Changes” button.

4. The system saves the new information of the user.

3.2.1.5 Use Case: Add book

Diagram:



Brief Description

The user adds book to his/her list and edit some information about it.

Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user touches “add new book” page from left menu.

2. The system provides a list of boxes for user to enter the information about book.

3. The user should touch the button “Update Information with ISBN”.

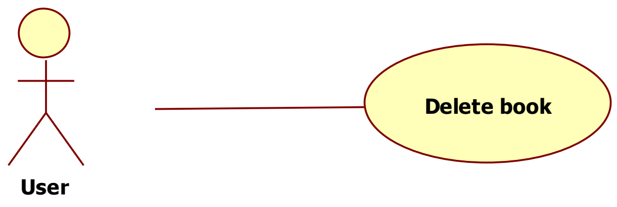
4. Book's barcode is read by the device's camera and information gaps are filled automatically.

5. If the user wants to add some comments about the book like location in the shelf and description, there will be gaps to fill them.

6. In order to add the book to user's book list, “Save Book” button will be at the right top corner.

3.2.1.6 Use Case: Delete a Book

Diagram:



Brief Description

The user deletes a book from his/her list.

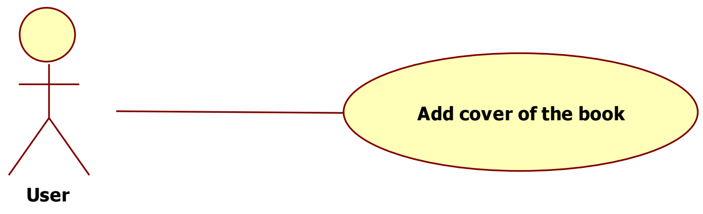
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user touches “My Book List” page from left menu.
2. The system provides a list of books from the left menu.
3. The user should mark the book that is wanted to be deleted.
4. Edit button is touched and in a new screen, some edit options will show up.
5. The user can delete the selected books from “delete” button.

3.2.1.7 Add cover of the book

Diagram:



Brief Description

The user adds cover of the book to book information part of the system by using android application.

Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. While adding new book, there will be buttons “Take a Photo” and “Browse” like “Update Information with ISBN”.

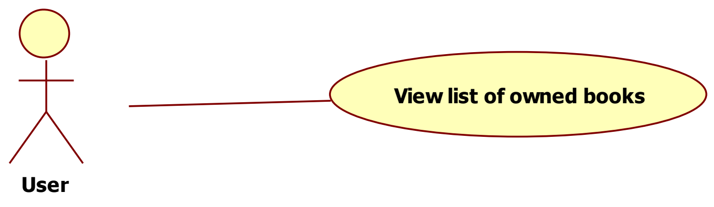
2. When “Take a Photo” is touched, the camera is started and waits for user to take the book's photo.

3. When “Browse” is touched, the application accesses to the device's memory and waits for user to select a picture file that will be added for book's cover in database.

4. After filling other information of the book, all data is saved by the “Save Book” button.

3.2.1.8 Use Case: View list of owned books

Diagram:



Brief Description

The user views list of owned books on the system.

Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

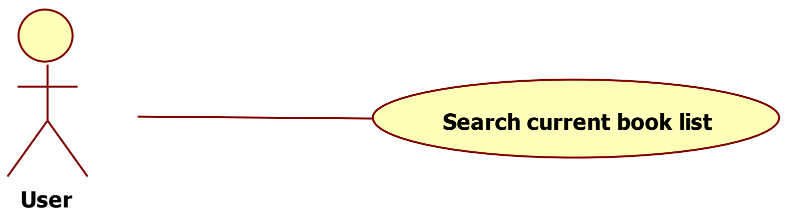
1. The user touches “my booklist” button from left menu.

2. The user views list of owned books in the left menu.

3. The application lists the information of a book selected from the left menu.

3.2.1.9 Use Case: Search current book list

Diagram:



Brief Description

The user searches his/her current book list via a key word.

Step-By-Step Description

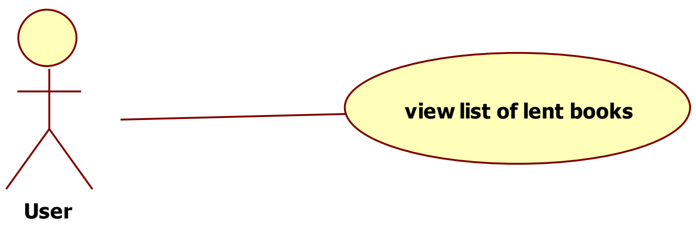
For this use case to be initiated, the user has already accessed MobColl application.

1. The user enters a key word and touches the "search book" button.

2. The system provides a list of books according to entered key word.

3.2.1.10 Use Case: View list of lent books

Diagram:



Brief Description

The user views list of lent books on the system.

Step-By-Step Description

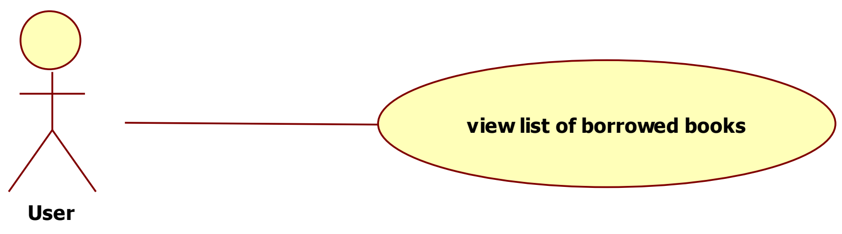
For this use case to be initiated, the user has already accessed MobColl application.

1. The user touches "lent books" button.

2. The user views list of lent books.

3.2.1.11 Use Case: View list of borrowed books

Diagram:



Brief Description

The user views list of borrowed books on the system.

Step-By-Step Description

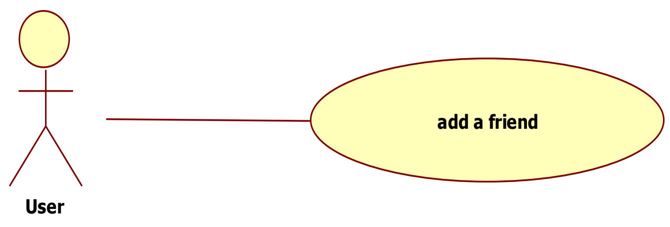
For this use case to be initiated, the user has already accessed MobColl application.

1. The user touches “borrowed books” button.

2. The user views list of borrowed books.

3.2.1.12 Use Case: Add a friend

Diagram:



Brief Description

The user adds a friend to his/her friend list.

Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

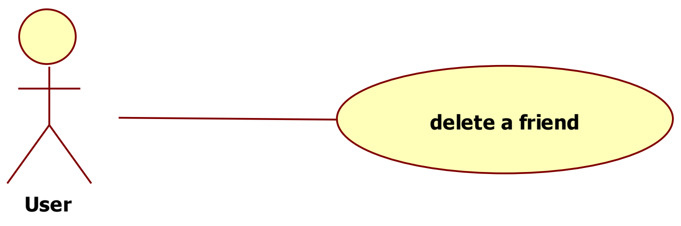
1. The user touches “add friend” button at a profile of a user.

2. The system sends a request to the user.

3. If the user accepts the request, they will be friends.

3.2.1.13 Use Case: Delete a friend

Diagram:



Brief Description

The user deletes a friend from his/her friend list.

Step-By-Step Description

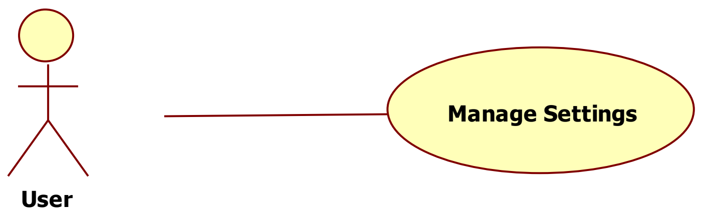
For this use case to be initiated, the user has already accessed MobColl application.

1. The user touches "delete friend" button at a profile of a friend.

2. The system deletes the friendship between these users.

3.2.1.14 Use Case: Manage Settings

Diagram:



Brief Description

The user changes the privacy and account settings.

Step-By-Step Description

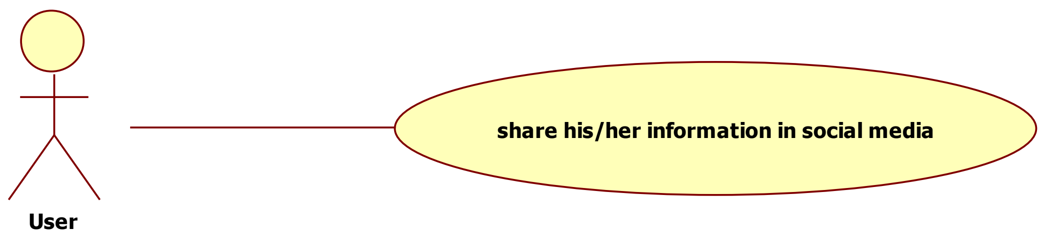
For this use case to be initiated, the user has already accessed MobColl application.

1. User touches the “settings” button placed on the left menu.
2. User updates the privacy level info.
3. User touches the “save” button and quits the settings.

4. The system saves the new settings.

3.2.1.15 Use Case: Share his/her information in social media

Diagram:



Brief Description

The user shares its current book, gift request, borrowed, for sole book lists.

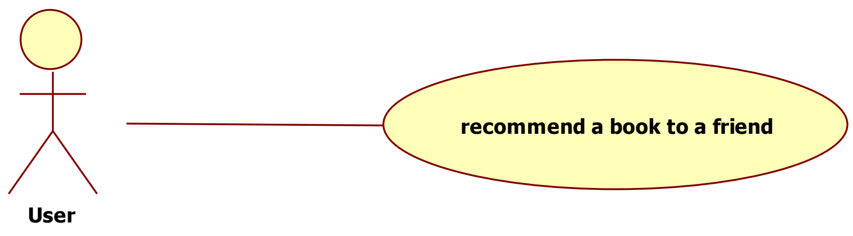
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. User browses the relevant list.
2. User touches the “share” button.
3. The system asks users to select which social media and to give authentication information.
4. User selects which social media to share and gives authentication information.

3.2.1.16 Use Case: Recommend a book to a friend

Diagram:



Brief Description

The user recommends a book it likes to a friend.

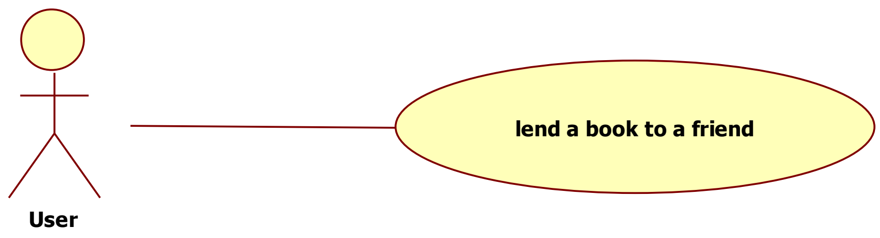
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. User browses his/her owned books by “my book list” button from the left menu.
2. After marking the book, user touches “recommend” button which is placed at the bottom of the screen.
3. The system asks the user to select which friends to recommend.
4. The user selects the friends and approves the operation.
5. The system sends an e-mail to the friend about recommendation.

3.2.1.17 Use Case: Lend a book to a friend

Diagram:



Brief Description

The user gives positive response to a borrow request of a friend.

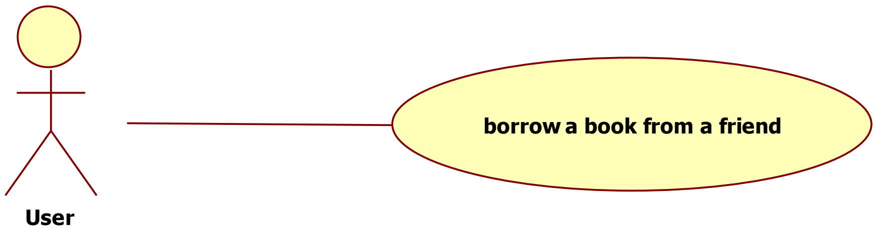
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. A friend of the user sends a borrow request for a book.
2. The user approves the lending operation and gives the book to the friend.
3. The system saves a log into history of the user consist of friend id, time and due date.

3.2.1.18 Use Case: Borrow a book from a friend

Diagram:



Brief Description

The user sends a borrow request to a friend.

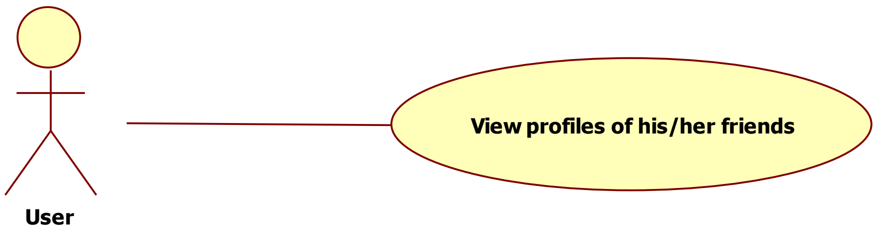
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. “Book Market” button is touched from left menu.
2. The user browses list of books that are selected as “lended” by the oweners.
3. The user likes a book and touches “borrow” button.
4. The system sends a borrow request to the owner of the book.

3.2.1.19 Use Case: View profiles of his/her friends

Diagram:



Brief Description

The user browses the information of a friend.

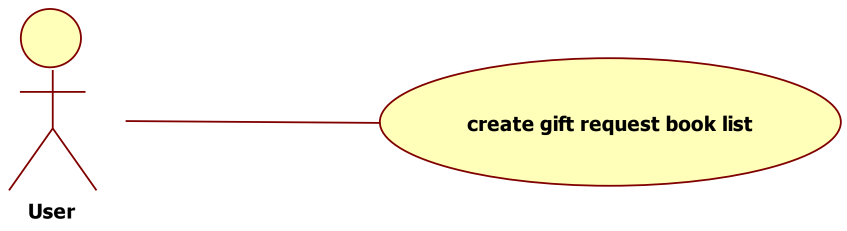
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user browses the user profile of a friend.
2. The user can see owned, gift request or for sale book lists of the friend.
3. On the left menu, there will be a “view profiles” button that provides all enabled lits of users.

3.2.1.20 Use Case: Create gift request book list

Diagram:



Brief Description

The user creates a gift request book list.

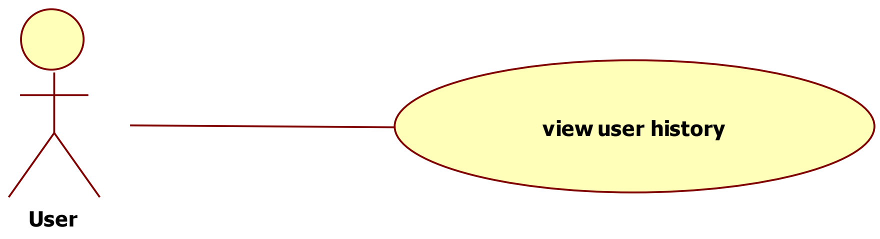
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user goes to “My Book List” page.
2. The list of books owned by the user is shown on the left menu.
3. The user selects the book to create gift request.
4. On the bottom menu, “gift” button is touched.
5. The book is added to user's gift request list that can be seen by the “gift request” button on the “Book Market” page that can be accessed from the left menu.

3.2.1.21 Use Case: View user history

Diagram:



Brief Description

A user can view the history of other users who allow their book lists to be shown by other users.

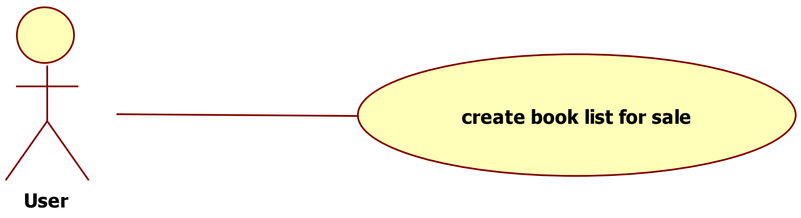
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user selects a friend or any other user allowing own book list to be shown.
2. The searcher user browses the profile of the user.
3. The transaction history of the user is displayed on the screen.

3.2.1.22 Use Case: Create book list for sale

Diagram:



Brief Description

The user creates a list that consists of books which are wanted to be sold.

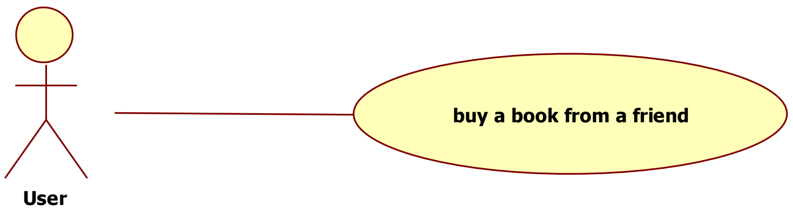
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user goes to “Book Market” page.
2. The user touches the “create book list for sale” button.
3. A new list is created and owned by the user.
4. A book is added to the list by changing the status of a book to “for sale”.
5. Selecting the books as “for sale” is saved to the transaction history.

3.2.1.23 Use Case: Buy a book from a friend

Diagram:



Brief Description

The users can sell books from their friends if they want.

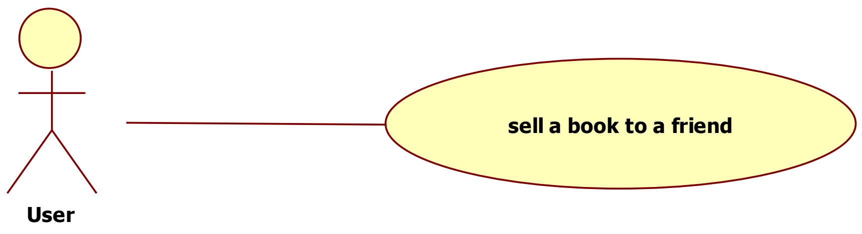
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user goes to the “Book Market” page.
2. The user searches for book using the browser tool.
3. The system retrieves the list of users who has the book wanted.
4. The user can send buy request to the owner of searched book even they are not friends.
5. A buy request is sent by the system to the owner of searched book.

3.2.1.24 Use Case: Sell a book to a friend

Diagram:



Brief Description

The users can sell books to their friends if they want.

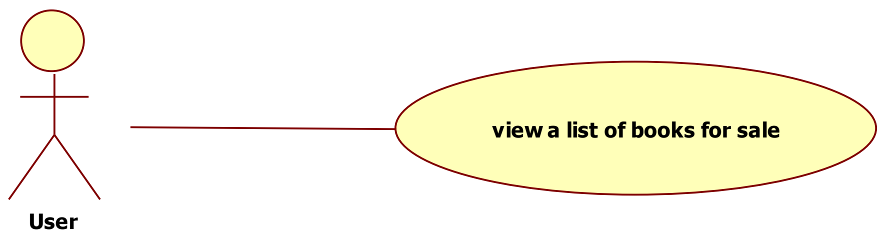
Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user goes to the “Book Market” screen.
2. There should be a request for book that wanted to be sold.
3. The seller user should accept or decline the request.
4. The owner of the book will be changed in database.
5. The sell action will be saved to the transaction history.

3.2.1.25 Use Case: View a list of books for sale

Diagram:



Brief Description

The user views the list of books which are selected as for sale by the user.

Step-By-Step Description

For this use case to be initiated, the user has already accessed MobColl application.

1. The user touches the “Book Market” button.
2. There will be “view for sale list” button to see the books that are selected as for sale by the user.

3.3 Non-Functional Requirements

In order to run the program, any version of Eclipse shall be installed on the computer. In addition, to run Java on a computer, JDK or JVM or JRE (mentioned above) shall be installed too. Minimum system requirements for Eclipse are as follows:

3.3.1 Performance Requirements

1000 people should be able to use web application simultaneously. A user should be able to keep 1000-book information on the system. The admin should be able to use the application interface to manage the system.

The Android application of the product should response user request at most in 2-seconds time.

3.3.2 Design Constraints

We will follow Object-Oriented-Programming paradigm. We will use Java on android application development. The Android version of the device that the app will run on should be 2.3 or higher. Also this application will run without internet connection.

4. Data Model and Description

4.1 Data Description

4.1.1 Data Objects

Book:

id: Identity number given from the database for each book. This attribute is unique for any book.

title: Title of the book.

author: Author of the book.

description: A small description/summary of the book. The user will be able to edit this text.

isbn: A number consisting of 13-digit-number which is identical for the book in every country.

cover_image: Cover image of the book.

publisher: The company that has published the book.

publish_date: The date of publishing.

language: Demonstrates the language which the book is written in or translated to.

category: The general type of the book

type: The specified type of the book.

rating: Average score of the book rated by the users.

location: Indicates the place of the book in the bookshelf.

User:

id: Identity number given from the database for each user. This attribute is unique for any user.

name: The name and surname of the user.

username: User's nickname for the application.

password: The user's password.

e-mail: E-mail of the user that will be used for the communication information.

phone_number: Phone number of the user that will be used for the communication information.

description: A brief description where the user mentions about his/herself

profil_photo: A picture that may be used to face detection login.

Books-of-User :

user_id: Id number that will be taken from the user table indicating the owner of the book.

book_id: Id number taken from the book table indicating which book is owned by the user.

is_favourite: The user has selected the book as a favorite book.

status: The number demonstrating the status of the book. 0 means "owned" and 1 means "for sale"

is_lent: Is the book lent or not.

lent_date: When the lending action happens.

due_date: Due date for returning the book back.
id: Id of the user that takes the book.

Friends-of-user:
user_id, friend_id: The user with "friend_id" is a friend of the user with "user_id" .

Borrowed-book list:
book_id: Id of the book taken from "book" table.

owner_id: Id of the user that gives the book.

borrowed_date: When the borrowing action happens

due_date: Due date for returning the book back.

History:

id: Primary key.

type_id: Process type id.

date_time: When the process occurred.

description: Detailed explanation about process.

user_id: This is the user id of owner of history entry.

Process_types :

id: Primary key

type: Name of the process. It can be lending, borrowing, adding book, deleting book, selling, buying, adding friend, favoring, etc.

Message:

id: Primary key

sender_id: The user who sends the message.

receiver_id: The user who receives the message.

date: The date of the sent message

message: Information which is sent from a sender to a receiver.

Settings:

user_id: Primary key

isSearchabilityPubliclyAvailable: Whether user can be reached by public search or not.

isSearchabilityFriendlyAvailable: Whether user can be reached by a friend of that user or not.

isBookListsPubliclyAvailable: Whether user's book lists can be reached by public search or not.

isBookListsFriendlyAvailable: Whether user's book lists can be reached by a friend of that user or not**.**

isHistoryPubliclyAvailable: Whether user's history can be reached by public search or not.

isHistoryFriendlyAvailable: Whether user's history can be reached by a friend of that user or not.

isContactInformationPubliclyAvailable: Whether contact information of user can be reached by public search or not.

isContactInformationFriendlyAvailable: Whether contact information of user can be reached by a friend of that user or not.

4.1.2. Relationships and Complete Data Model

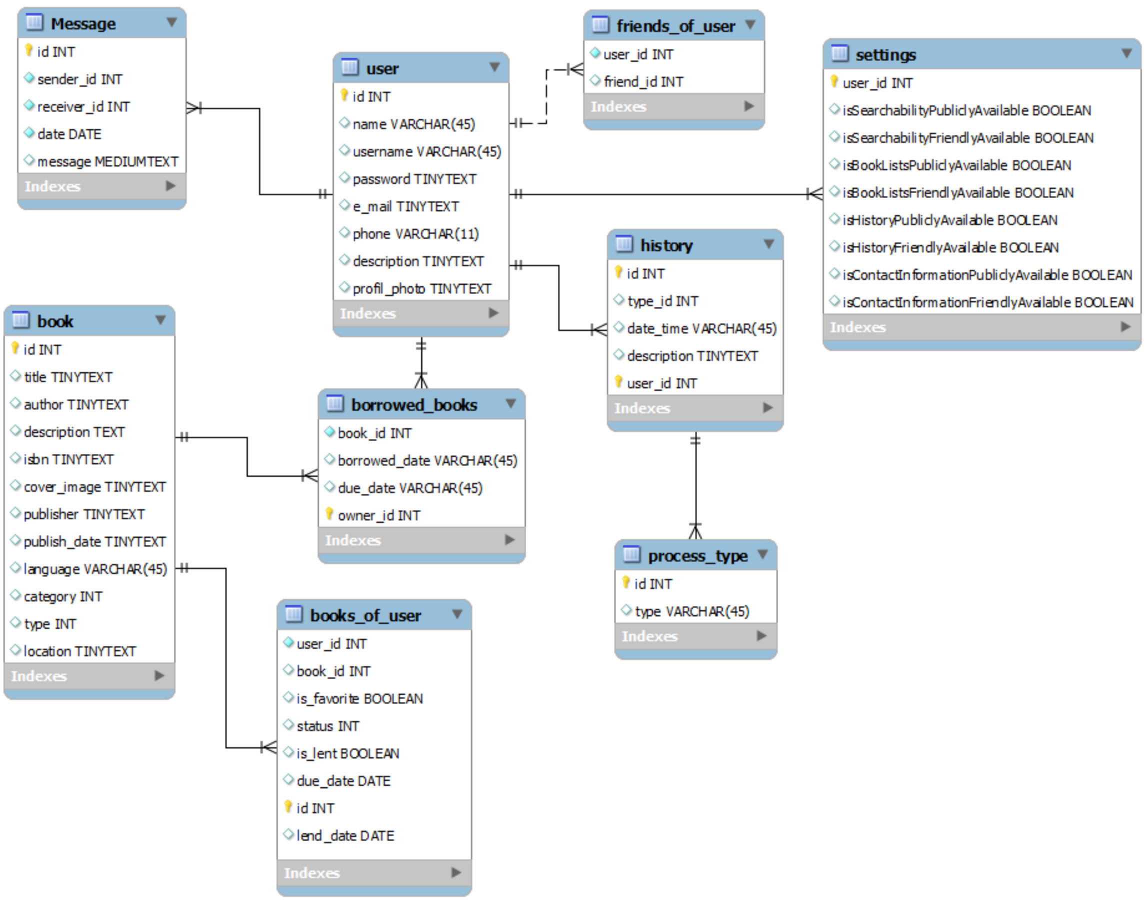


Figure 5 – ER Diagram

5. Planning

5.1 Team Structure

Our team, Andios, consists of four students. Murat Öksüzer (1679448), Sercan Çidem (1679430), Vedat Şahin (1679562), Fatih Osman Seçmen (1679182).

5.2 Estimation (Basic Schedule)

Project (SRS): 11 November 2012

Project (SDD): 22 November 2012

Prototype demo: 20 January 2013

Project (STD): Second Semester

Project (STR): Second Semester

5.3 Process Model

We are using waterfall process model. In general, this model may be considered as having six different phases:

1. Requirements Analysis
2. Design
3. Implementation
4. Testing
5. Installation
6. Maintenance

6 Conclusion

In Conclusion, in this document, we have defined how should be MOBCOLL Project in detail. These are the first plans about the project and when we are implementing the project, some minor details may change or added. These changings will be shown in the update documents.