



# **PUBLIC LIBRARY INFORMATION MANAGEMENT SYSTEM**

Developing new service delivery methods

Mohammed Al-Musawi

Master's thesis  
August 2014  
Degree Programme in  
Information Technology

TAMPEREEN AMMATTIKORKEAKOULU  
Tampere University of Applied Sciences

## **ABSTRACT**

Tampereen ammattikorkeakoulu  
Tampere University of Applied Sciences  
Degree programme in Information Technology

Mohammed Al-Musawi:  
Public Library Information Management System  
Developing New Service Delivery Methods

Master's thesis 94 pages, appendices 12 pages  
August 2014

---

With the advancement of mobile technologies, and with the expanding use of smart phones, the need to have a wider range of these methods has become evident. The objective of this thesis was to develop new methods to deliver library services to the public, and to illustrate how a mobile application interacts with the central library database, which, in turn, is managed through a web application by library staff. This thesis illustrates some of the options for the development of the service delivery methods and provides a basis for their further expansion.

The approach to the subject was practical, and started with the exploration of the background of the library system in Finland and the existing services in the libraries, and the needs of library customers. The possibilities offered by the Windows Phone 8 platform were scrutinized, as it is the most widely used smart phone operating system in this country. The project was implemented in two distinctive parts, namely the web application part, and the mobile application one. Several programming languages and tools were used in the implementation to achieve fluency in the communication between the different parts of the project, and to enhance user experience.

The thesis shows that it is possible to create new service delivery methods based on the Windows Phone 8 operating system, and that a system can be implemented in which the mobile application and the web application communicate fluently and efficiently with the central database, creating an integrated information management system.

New service delivery methods create an opportunity for the libraries to enhance customer service and, consequently, to benefit from a wider customer base, and to concentrate resources away from routine tasks. For customers, these new methods open new options of discovering and using library services, and provide flexibility in the management of their library accounts. The system is able to be developed further, and new functions and features can be added easily to the existing framework, such as an e-book borrowing option. The system can also be customized to better meet specific user requirements.

---

Key words: mobile development, web development, information management system, library

## **FOREWORD**

First of all, I would like to express my deepest gratitude to my family and loved ones for the support and patience during the process of implementing this project and writing this thesis. Without their support this thesis would not be in existence.

Secondly, my sincere thanks go to the supervisor of the thesis process at the Tampere University of Applied Sciences, Mr Tony Torp, for his valuable comments and guidance.

Tampere, 17 August 2014

Mohammed Al-Musawi

## CONTENTS

1	INTRODUCTION.....	7
2	PROJECT IDEA AND JUSTIFICATION.....	8
2.1	Background.....	8
2.1.1	Libraries in Finland.....	8
2.1.2	Mobile technology.....	9
2.2	Project benefits.....	10
2.2.1	For library users.....	10
2.2.2	For the libraries.....	11
3	SYSTEM OVERVIEW AND USE CASES.....	13
3.1	Overview of the system.....	13
3.2	Administration web application use cases.....	14
3.2.1	Login.....	15
3.2.2	Forget password.....	15
3.2.3	Contacts.....	16
3.2.4	Services.....	17
3.2.5	Library's pictures.....	18
3.2.6	Members.....	19
3.2.7	Employee.....	20
3.2.8	Library's events.....	21
3.2.9	Search for an item.....	21
3.2.10	Add media item.....	22
3.2.11	Renew a loan.....	23
3.2.12	Reserve item.....	24
3.2.13	Reservation list and reservation request list.....	24
3.2.14	Check out item.....	25
3.2.15	Check in item.....	26
3.2.16	Sign out.....	27
3.3	Customer mobile application use cases.....	27
3.3.1	Login.....	28
3.3.2	Search for item.....	29
3.3.3	User Account.....	30
3.3.4	Get libraries' events.....	30
3.3.5	Send feedback.....	31
3.3.6	User reservation.....	32
3.3.7	Reserve items.....	32
3.3.8	User loan.....	33

3.3.9	Renew.....	33
3.3.10	Lend and borrow .....	34
3.3.11	Log out .....	36
4	SYSTEM FEATURE AND IMPLEMENTATION.....	37
4.1	Administration web application features and implementation .....	37
4.1.1	Employees feature.....	39
4.1.2	Members feature.....	43
4.1.3	Services feature .....	45
4.1.4	Media items feature.....	46
4.1.5	Reservation management feature .....	53
4.1.6	Check out/in feature .....	54
4.1.7	Contacts feature.....	56
4.1.8	Events feature.....	57
4.1.9	Pictures feature.....	59
4.2	Customer mobile application features and implementation.....	60
4.2.1	My account feature.....	63
4.2.2	Feedback feature .....	64
4.2.3	Events feature.....	64
4.2.4	My items feature .....	66
4.2.5	Search feature.....	68
4.2.6	Lend / Borrow feature .....	70
4.3	Database implementation.....	74
5	SYSTEM ARCHITECTURE.....	79
5.1	Server side architecture.....	79
5.2	Client side architecture .....	80
5.2.1	Mobile application architecture.....	80
5.2.2	Web application architecture.....	81
6	DEVELOPMENT ENVIRONMENT .....	84
6.1	Server side.....	84
6.2	Client side .....	85
6.2.1	Mobile phone application.....	85
6.2.2	Web application .....	87
7	FUTURE DEVELOPMENT .....	88
8	DISCUSSION .....	91
	REFERENCES.....	93
	APPENDICES .....	95
	Appendix 1. Database SQL script .....	95

**GLOSSARY**

APACHE	HTTP Server software
CD	Compact disc
CSS	Cascading Style Sheets
DB	Database
DBAs	Database Administrator
DVD	Digital video disc or Digital versatile disc
EER	Enhanced entity-relationship model
HR	Human Resources
HTML	Hypertext Markup Language
HTTP	Hypertext Transport Protocol
JavaScript	Scripting language
JQuery	JavaScript library
JSON	JavaScript Object Notation
Library barcode	A unique number given by the library for each item in their library
ICT	Information and Communication Technology
ISBN	International Standard Book Number
ID	Identifier, a symbol which uniquely identifies record or an object
PHP	Server-side scripting language to make dynamic web pages
UI	User Interface
RDBMS	Relational Database Management System
SDK	Software Development Kit
SQL	Structured Query Language
SSN	Social Security Number
TAMK	Tampere University of Applied Sciences
Wi-Fi	A popular technology that allows an electronic device to exchange data or connect to the Internet wirelessly using radio waves.
XAML	Extensible Application Markup Language
XML	Extensible Markup Language

## 1 INTRODUCTION

Smart phones and devices exist in almost every pocket these days. And most Finns use the services of a library regularly, according to research. This project was designed to combine these two aspects, and to make it easier for the consumer and the library system alike to access and provide these services using the latest technology available.

The project is implemented in two parts, namely the mobile application and the web application. They interact and communicate with a central database, creating a platform to develop customer experience and the work of library staff further.

The mobile application like a "library in a pocket", allowing the customer, among other things, to search, reserve, and even borrow library items whenever and where ever he or she may be. This is done through an application downloaded to a smart phone with an Internet connection.

The customer has the ability not only to see what is available in the library close to home, but the whole library network connected to the system. In a way, this part of the project is more of a "library network in a pocket".

The web application is where the libraries manage their basic functions and respond to and act upon customer requests. Libraries can, for example, add an employee or manage their stock of items, advertise events held in the library, manage reservations, and receive feedback, with each library able to manage its own information, only.

For the libraries, the project provides the opportunity to widen the customer base, to advertise events held in libraries, and, as the project allows customer more independent management of their own library account, to redirect existing resources to better understand and meet customer needs and requirements.

This project shows how to implement an integrated database for libraries, but it could be utilized for many other purposes, as well. The mobile application part was implemented on the Windows Phone 8 platform, in which such applications have not been implemented before, at least in Finland.

## 2 PROJECT IDEA AND JUSTIFICATION

### 2.1 Background

#### 2.1.1 Libraries in Finland

Finland has traditionally invested in literacy. As early as in the 17<sup>th</sup> century, reading skills were regarded as something required from all. By the year 1880, practically all the Finns, 97.6% of the adult population, were able to read (Kirjastot.fi 2007). Therefore, it is hardly surprising that Finland currently has one of the widest public library networks in the world, and the system is used widely. According to the statistics, the Finns are avid readers and library users: in 2013 the total annual lending was 93 million items (over 17 per capita), the annual number of library visits was 51 million (10 per capita) and the Internet services of the libraries were used 53 million times a year. (Ministry of Education and Culture 2013.)

There are three different types of libraries in Finland:

- Municipal public libraries which offer free access to cultural and information sources for all irrespective of their place of residence and financial standing. There is a public library in every Finnish municipality (304), and most of them also have branch libraries (487) and bookmobiles (148). The 18 municipal libraries also operating as provincial libraries provide information and interlibrary services in their regions. Libraries also contribute to the development of virtual and interactive web services and their educational and cultural content. Customers have free access to computers and Internet in all the libraries.
- Research libraries, comprising university, polytechnic and special libraries, serve higher education, learning and research. Students use public and research libraries side by side. University libraries offer free access to everyone, not only university students and staff. Polytechnics have their own libraries and information services. Teaching-related information services are provided and financed by the local education authority.



- School libraries. Schools and other educational institutions provide library and information services to their pupils and students. In this, they cooperate with public and other libraries. (Ministry of Education and Culture 2014.)

In Finland, libraries are free of charge to all the people. Library and information services promote equal access to education and culture, reading and art appreciation, constant development of knowledge, citizenship skills, internationalisation, and lifelong learning. Libraries also contribute to the development of virtual and interactive web services and their educational and cultural content. (Libraries.fi 2014.)

### **2.1.2 Mobile technology**

As the information revolution continues to unfold, libraries will experiment with mobile devices and services to support the information needs of their users wherever they may be.

Smart phones and other similar mobile devices are increasingly popular, which creates a wide user base for a mobile library application. A total of 61% of Finns aged between 16 and 74 had a smart phone in 2013, and the penetration of smart phones is very high in the younger age groups, with up to 81% of those aged 25 to 34 having one (Suomen virallinen tilasto (SVT): Väestön tieto- ja viestintätekniikan käyttö 2013).

People are using their mobile phones and tablets increasingly to access municipal and other services (Center for Digital Government 2014). Therefore, it is important for the libraries, as well, to keep up with the developments. When the services are available in a way that is attractive to today's user, they will remain popular, and their presence and spread will be guaranteed.

While Android and iOS still dominate the markets in hand-held devices, in Finland, Windows Phone has managed to pass its two main rivals, and, in 2013, constituted 35% of new sales, ahead of Android a 33% and iOS at 30% (Nygrén Toni 2013).

In business, Windows Phone has an even bigger lead, with 50% of sales into enterprise being Windows Phones in this country in 2013. The new, low-priced handsets are expected to further strengthen Windows Phone's progress in Finnish organizations.

According to Market Vision leading analyst Toni Nygrén (2013), the Finnish smartphone platforms in the market differs from other markets. In most other countries, the Android platform dominates the smartphone market, while in Finland the Windows Phone is taking a strong foothold due to the Microsoft/Nokia Lumia family.

## **2.2 Project benefits**

### **2.2.1 For library users**

Libraries are always looking for ways to improve and expand their services for the users. With the wider public able to use mobile technologies, libraries need to keep up-to-date to make it easier for the public to access their media items such as books, newspapers, and DVDs.

Nowadays, library users are demanding more and more services that suit their needs, and they face many obstacles in receiving them. For example, library users:

- are not always in possession of their library card;
- want an easy way to follow or check their library accounts and their loans anywhere and anytime;
- wish to search for media items and find out in which library they are available, whenever they have a free moment, for example when using public transport;
- want to reserve the items that they want anywhere, anytime and from any library in the system;
- wish to track their reservations in an easy way;
- wish to renew their loans without visiting the library or its website;
- wish to check events at libraries, such as book reading events and courses, anytime and everywhere to get the opportunity to attend these events;
- wish to receive more services to manage their own transactions, including borrowing for the library without the need to ask the library personnel for help

or using automated check-out, or borrowing from other library members anytime and anywhere without visiting the library.

As a result, in this project, the researcher designed and developed a mobile phone application for library users. Library mobile application is like a library in a pocket, a mobile service application for municipal libraries, available to anyone with a valid library card and a mobile phone using Windows Phone 8 platform.

The application allows the user to search for available items (books, CD/DVD, magazines and newspapers) in stock throughout the library network, specifying the library in which the item could be found. The application also permits the user to renew a loan, send feedback to the system manager, see the user's account (items on loan and their return date, and possible late fees), reserve an item from the library network and receive it from the library of the user's choice, track the reservation status, and borrow from a friend (another user with a borrowed item to allow for direct peer borrowing without passing through the library physically), or lend to a friend (lending to another user, in the same way as borrowing from a friend). Additional functions include, for example, information about events in municipal libraries, such as visits by authors or new acquisitions.

With the application, when in the library, the user can also scan a barcode of a book or other item to borrow, making the borrowing faster and more convenient. The fact that this is a mobile application instead of one for a workstation makes this possible.

### **2.2.2 For the libraries**

First of all, this mobile phone application, added to the library services, will allow the users to manage their own transactions, and inevitably increase the users' activity towards the library. This will help to reduce the manual labour for library personnel, allowing them to focus on other user services. Also, the application allows the librarians to respond or act more quickly to specific user requests. In addition, libraries will be able to advertise events, thus increasing the participation in them, and, consequently, obtain more users.

It is cost-efficient for the libraries to make the user have the opportunity to self-manage their own transactions, without the need to buy (and maintain) more automated devices.

Secondly, each library has its own database to store and manage its data. They can also have their own website for the management and for the users, although this needs a lot of effort and personnel to maintain.

In this project, the researcher designed and developed a central database for the libraries, combining all their data, and this included a web application for the administration. The web application is implemented to serve all the libraries. However, each library is able to control only their own information, such as adding or removing their own events, services, contacts, and library images, adding media items, or managing employee profiles. Some of the functions in the web application can be performed by any library in the system, such as preparing the reserved media items to the users through making the reservation service faster and more convenient, which consequently will enhance the libraries' performance for this service.

### 3 SYSTEM OVERVIEW AND USE CASES

#### 3.1 Overview of the system

This system contains different types of users: library customers, librarians, and managers of the libraries within the system. The system is divided into two applications or two user interfaces, depending on the user roles inside the system (figure 1). Librarians and managers of the libraries which are part of the system use the web application to administer their library and to implement many other functions which are explained in this chapter (Administration web application). Library customers use the mobile application to manage their accounts and to follow their activities and events in the libraries (Customer mobile application). In addition, there are many other functions and features which are explained in this chapter in more detail.

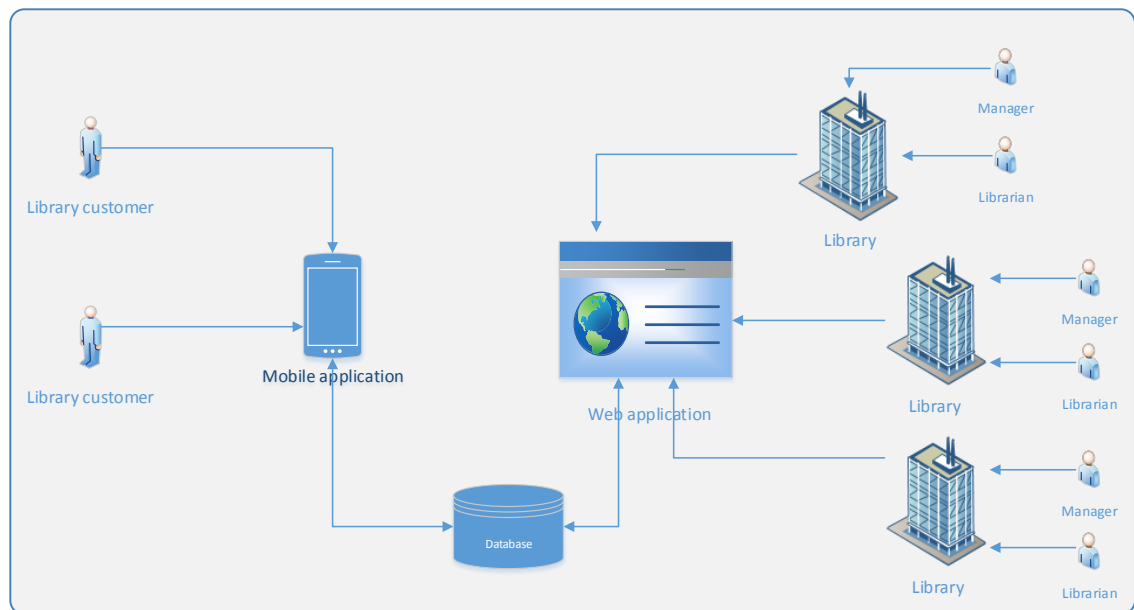


FIGURE 1. Overview of the system

The applications are connected in real time with the libraries' central common loan management database as shown in the figure 1 above.

### 3.2 Administration web application use cases

The web application is used by the librarians and the management of several libraries to manage different functions of the libraries. The information of all the libraries is centralized in one database, and access to this information is limited by user roles, which include librarian and library manager. Figure 2 below explains the use cases of the web application. It shows what kind of functions the web application users can access in the application depending on their roles. It also shows how these functions interconnect and what the scenarios behind them are.

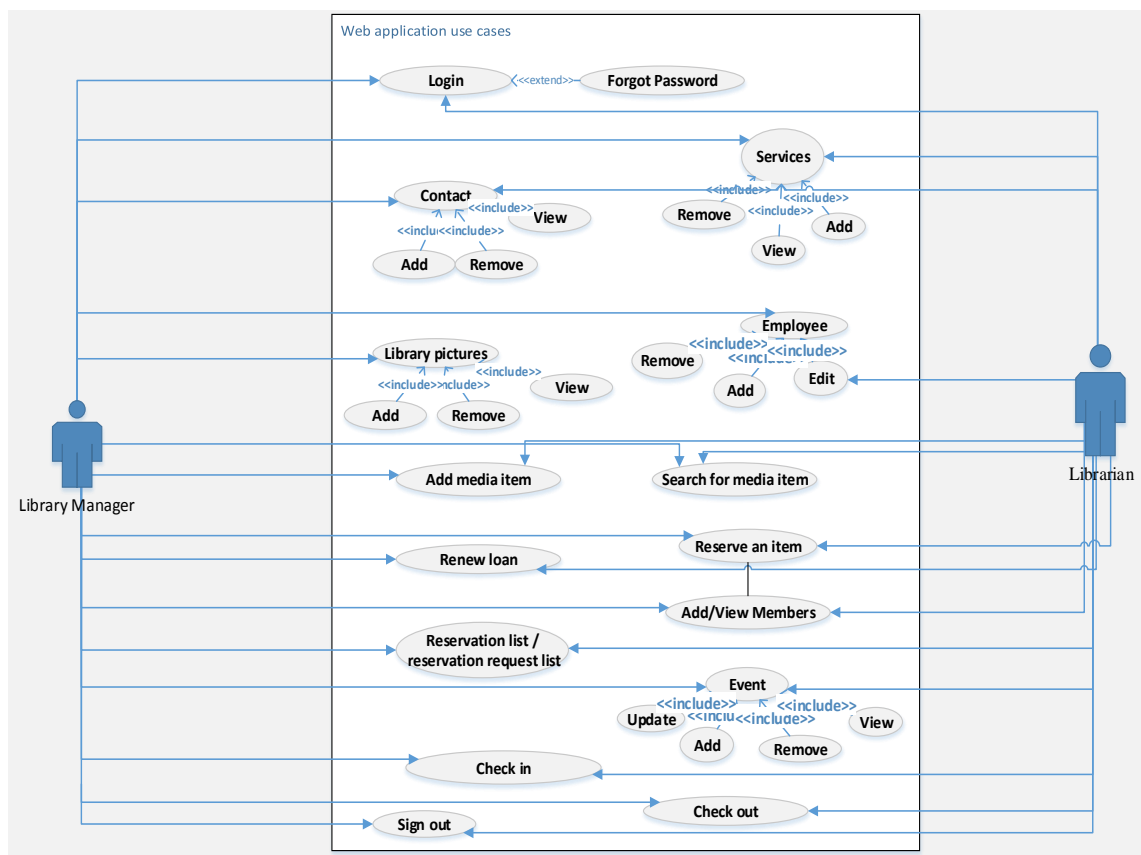


FIGURE 2. Web application use cases

The use cases illustrated in the figure 2 above are explained in more details in the sections below.

### 3.2.1 Login

**Description:** This use case is used to check the correctness of a given username and password, in order to allow the user to access this application. The user name and password are already stored in the DB in the server.

**Preconditions:** The user must have a valid username and password.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user fills in the user name and password, the information is sent to the server for validation by clicking on the 'login' button.

**Key scenarios:**

1. The user provides the username and password.
2. The account information is sent to the server for validation by clicking on the 'login' button.
3. Based on the provided information the server responds with either true or false.
4. If 'true' the application displays the main menu of the appropriate access level.
5. If 'false' the user will get a message prompting him/her to check the information entered.

**Post conditions:** Successful login and open the session.

**Outcome:** The user can access the system and use the features of the application. Also, unauthorized users are prevented from using or accessing the system. There is also auto direction to the appropriate user interface and functions that the user, depending on his/her role, is authorized to access.

### 3.2.2 Forget password

**Description:** In case the user forgot his/her password, this use case is used to provide the user with his/her password.

**Preconditions:** The user must have a valid username and email address registered in the system.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user fills in the user name or email address, the information is sent to the server for validation. The password will then be sent to the user's email address.

**Key scenarios:**

1. The user provides the username or email address given at registration.
2. The account information is sent to the server for validation by clicking 'forget password' button.
3. Based on the provided information the server responds with either true or false.
4. If 'true' the application sends the password to the user's email address.
5. If 'false' the application shows a message prompting the user to check the entered data.

**Outcome:** The user receives his/her password to his/her registered email. The registered email is used for security purposes to prevent unauthorized attempts to obtain users' password.

### 3.2.3 Contacts

**Description:** This use case is used to view, add, and remove contact numbers of libraries which are part of the system. Each library is able to modify its own data, only.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user clicks on a button to see the list of contact information of his/her library. Another button is used to remove the contact information. To add a contact, information is first entered, and approved by clicking a button.



**Key scenarios:**

1. In case of ‘view contact’, the user clicks on the ‘contact’ button and the application sends a request to receive the contact information from the database in the server.
2. In case of ‘remove contact’, the user clicks on a button and the application sends a request to remove the contact information from the database in the server, after which the system shows the updated list of contacts.
3. In case of ‘add contact’, the user must enter the contact title and its number. With a click of the ‘add contact’ button, the application sends the information to be stored in the DB. After this, an updated list of contacts is shown. If the user does not fill in the new contact information and clicks on ‘add contact’ button, the application displays a message to prompt the user to enter the required information. In addition, if the information entered does not fulfill the necessary criteria (for example entering text in number fields), the system informs the user of this mistake.

**Outcome:** The user can control the contact information of his/her own library.

### 3.2.4 Services

**Description:** This use case is used to view, add, and remove services that a library which is part of the system offers for the customers. These services include, for example, scanner, Wi-Fi and color printer.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user clicks on a button to see the list of services offered by his/her library. Another button is used to remove a service. To add a service, information is first entered, and approved by clicking a button.

**Key scenarios:**

1. In case of 'view service', the user clicks on 'Service' button and the application sends a request to receive a list of services from the database in the server.
2. In case of 'remove service', the user clicks on a button and the application sends a request to delete the chosen service from the database in the server. After that the application shows the updated list of services.
3. In case of 'add service', the user must enter the information about the new service. With a click of a button, the application will send the information for validation to and storage in the DB. After this, the system displays the updated list of services. The user receives a message if there is a problem or error in data entered.

**Outcome:** The user can manage the list of services available in his/her library.

### 3.2.5 Library's pictures

**Description:** This use case is used to view, add, and remove pictures of libraries which are part of the system. Each library is able to modify its own data, only.

**Preconditions:** Successful login.

**Actor(s):** Manager.

**Basic flow of events:** The user clicks on a button to see the pictures of his/her library. Another button is used to remove a picture. To add a picture, a file needs to be selected, and approved by clicking a button.

**Key scenarios:**

1. In case of 'view pictures', the user clicks on 'Pictures' button and the application sends a request to receive a list of the pictures of the user's library from the server.
2. In case of 'remove picture', the user clicks on a button and the application sends a request to remove the selected picture from the system, by removing the file of

the picture from the server and the path of this file from the database. After this, the system displays the updated list of pictures.

3. In case of 'add picture', the user must select a new picture file and then click on the 'add' button. The application then sends the file path to be stored in the DB, and uploads the picture file into a specific folder inside the server. After this, an updated list of pictures is displayed.

**Outcome:** User can manage the pictures of his/her own library.

### 3.2.6 Members

**Description:** This use case is used to view or add a customer profile.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user clicks on the 'members' button from the main menu to navigate to the 'members' page which allows the user to search for information about a specific member. The user can also add a new member or customer to the system.

**Key scenarios:**

1. In case of 'view customer profile', the user enters the customer card number or social security number, then clicks on the 'search' button to send a request to receive the customer profile from the database of the system.
2. In case of 'add customer', the user clicks on the 'add' button to receive a special form to add a new member to the system. The user should enter all the required information: Name, SSN, date of birth, address, email, pin and number of the customer's library card. After this, the user clicks on a button to send a request to store this profile in the database. The application checks and validates all the fields which were entered by the user before storing them in the database. The application displays error messages in case of any problems with the data.

**Outcome:** The user can check and manage customer profiles and their loans easily. In addition, the user can add a customer to the system from any library which is part of the system.

### 3.2.7 Employee

**Description:** This use case is used to view, edit, remove and add an employee profile.

**Preconditions:** Successful login.

**Actor(s):** Librarian with limitation and Manager.

**Basic flow of events:** The user clicks on the 'employee' button from the main menu to navigate to an employee's page to perform the following functions, depending on the user's access rights: view an employee's details, edit an employee's details, remove or add an employee.

**Key scenarios:**

1. In case of 'view employee profile', a user with a manager's access rights is able to view all the information in the employees' profiles in his/her library. A user with a librarian's access rights is able to view his/her own information, only.
2. In case of 'edit employee profile', a user with a manager's access rights is able to edit all the information in the employees' profiles in his/her library. A user with a librarian's access rights is able to edit limited parts, such as address, phone number, etc., of his/her own information, only.
3. In case of 'remove employee profile', only a user with a manager's access rights is able to access this function. This function can only be used to remove an employee in the manager's own library.
4. In case of 'add employee profile', only a user with a manager's access rights is able to access this function. This function can only be used to add an employee in the manager's own library.

### 3.2.8 Library's events

**Description:** This use case is used to view, add, edit and remove events at a library which is part of the system, such as language courses or storytelling events.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user clicks on the 'events' button from the main menu to navigate to the 'events' page which allows the user to perform the following functions, view an event's details, edit an event's details, remove or add an event.

**Key scenarios:**

1. In case of 'view event', the user clicks on a button and the application sends a request to receive the list of events of the user's library from the database in the server. The user clicks on the desired event to obtain all the information pertaining to that event.
2. In case of 'edit event', the user enters the updated information and clicks on a button to send a request to update the event. The new information is stored in the database after the system checks and validates it.
3. In case of 'remove events', the user clicks on a button and the application sends a request to remove the selected event from the database in the server. After this, an updated list of events is displayed.
4. In case of 'add event', the user must enter all the details of the new event. After this, with a click of a button, the application validates the data and sends it to the database for storage. After this, an updated list of events is displayed.

**Outcome:** The user can manage the events which are available in his/her library.

### 3.2.9 Search for an item

**Description:** This use case is used to search for item such as book, CD or eBook. The details of the items are already stored in the database.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user selects the search type and fills in the required information in the search item text box. The information is then sent to the server for validation and to obtain a list of items fulfilling the search criteria, with details.

**Key scenarios:**

1. The user selects the type of search preferred: 'classic' search by item title, author name or ISBN, or 'advanced' search where the user is able to specify additional criteria to limit the search, such as the language of the item or item type.
2. In both cases, the user should provide a partial or complete media title, author name, or ISBN.
3. The information is then sent to the server to check for a possible information match in the database.
4. The result is a list of items in case of a match, and 'No title' in case no match is found.

**Outcome:** The user is able to search for items in the libraries which are part of the system.

### 3.2.10 Add media item

**Description:** This use case is used to allow an authorized user to add an item, such as a book, a CD, or a magazine, to the selection of his/her library.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user types or scans the ISBN of the media item and clicks on the 'check item' button. The application checks if the database of the system already has

information about this item. Should there be none, the user must enter all the information required by the system. Should the item already exist in the database, the user only needs to enter the specific information pertaining to the specific copy.

**Key scenarios:**

1. The user enters the ISBN of the media item and clicks on a button.
2. The system checks if the selected item is already available in the central database. This is both to prevent the duplication of items in the database.
3. In case the item is not available in the database, the user enters all the details about this item.
4. In case the item is available in the database of the system, the user only needs to add the library-specific barcode, and the location of the copy.
5. The user receives a message in case of any issues with the validation of the provided information.

### **3.2.11 Renew a loan**

**Description:** This use case is used to send a request to renew a loan of one of the items.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user clicks on the ‘renew’ button, the system sends information about the customer and the item to the server to check if the customer can renew the loan or not.

**Key scenarios:**

1. The system sends information about the customer and the item to the server.
2. The system checks if something prevents the renewal, that is if the item was renewed 5 times in a row, or the item was reserved by another user.
3. In case nothing prevents the renewal, then renewal is done for 2 weeks. In the opposite case, the system informs the user of the reason for the denial of the renewal.

**Outcome:** The user is able to renew a customer's loan easily and get informed about possible reasons preventing the renewal.

### 3.2.12 Reserve item

**Description:** This use case is used to reserve an item to a library customer from the system.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user selects the item that the customer wants to reserve. The application asks the user to choose a library from which the reserved item will be collected. The reservation information will be stored in the database.

**Key scenarios:**

1. The user clicks on a button to reserve an item.
2. The user chooses the library from which the customer wants to receive the item.
3. The user receives a conformation about his/her request.
4. The application sends the reservation information to the database in the server to be stored.

**Outcome:** The user is able to reserve items for the customer if the item is on loan.

### 3.2.13 Reservation list and reservation request list

**Description:** This use case is used to obtain the list of pending reservations from the user's own library, should the item be available there, to prepare the reservations for delivery, and to check the list of the prepared items of the user's own library.

**Preconditions:** Successful login.



**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user selects one of two buttons to either view and prepare pending reservation, or to view completed reservations.

**Key scenarios:**

1. In case of 'reservation request list', the user clicks on a button and the system sends a request to obtain the list of items with their details.
  - The list will contain only the items which are available in the user's library.
  - The user enters the library barcode of the item.
  - The user clicks on the 'done' button to send the request to the database and to change the reservation status from 'request' to 'done', and the item status from 'available' to 'not available' in the database.
2. In case of 'reserved list', the user clicks on a button and the application sends a request to obtain the list of reserved items prepared by the user's library with their details.

**Outcome:** The user can manage reservations easily.

### 3.2.14 Check out item

**Description:** This use case is used to check out items from the libraries and put them in the customers' accounts.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user enters the customer's library card number and the library barcode of the desired media item. The system retrieves the information about the item from the database and then registers the item to the customer's account.

**Key scenarios:**

1. The user enters the customer's library card number.
2. The user enters the library barcode of the desired media items one by one.
3. The user clicks on the 'ok' button to obtain the details of the item.
4. The user clicks on the 'check out' button, after which and the system sends a request to the database to add these items to the customer's account and change the item status to 'not available in the library' inside the database.

**Outcome:** The user is able to manage the check-out of items for the customers easily.

**3.2.15 Check in item**

**Description:** This use case is used to check in the returned items to the libraries.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user enters the library barcode of the returned media items. The system retrieves the details of the items, and calculates the possible total late fee.

**Key scenarios:**

1. The user enters the library barcodes of the returned media items one by one.
2. The user clicks on the 'ok' button to obtain the details of the item with the possible late fee.
3. The user clicks on the 'check in' button and the system sends a request to the database to remove these items from the customer's account and to change the item status inside the database.
4. If the returned item originates from another library, the system informs the user where the item should be sent.

**Outcome:** The user is able to manage the check-in of items for the customers easily.

### 3.2.16 Sign out

**Description:** This use case is used to sign the user out of the application.

**Preconditions:** Successful login.

**Actor(s):** Librarian and Manager.

**Basic flow of events:** The user clicks on a button, the system deletes the user information from the local storage and ends the session for this user, then navigates to the login page.

**Key scenarios:**

1. The user clicks on the 'sign out' button.
2. The system deletes the user information from the local storage and ends the session.
3. The system navigates to the login page.

**Outcome:** Ending the session prevents unauthorized access to the user account.

### 3.3 Customer mobile application use cases

The mobile application is used by the customers to access their library account, namely the loans, reservations, and other data pertaining to the library system. Figure 3 below illustrates the use cases of the mobile application. It shows the different functions of the mobile application which the user is able access, and how these functions interconnect. It also shows the customer requirements.

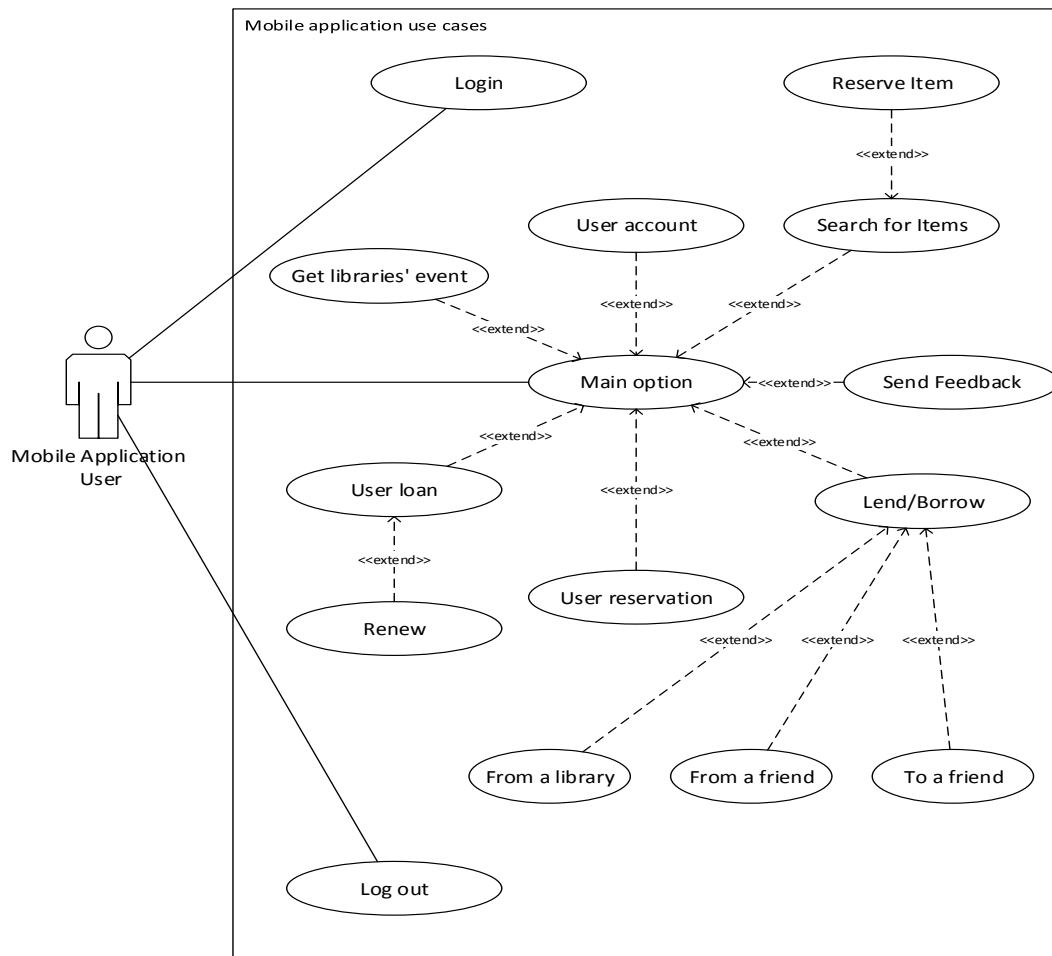


FIGURE 3. Use cases of the mobile application

Each use case has its own scenarios and conditions. In the next sections, each use case with its functions and features inside the application is explained in detail.

### 3.3.1 Login

**Description:** This use case is used to check the correctness of a given user card number and password, and to allow the user to access this application. The user card number and password are already stored in the DB in the server.

**Preconditions:** The user must have a valid card number and password.

**Basic flow of events:** The user fills in the user card number and password, the information is sent to the server for validation.

**Key scenarios:**

1. The user provides the user card number and password.
2. The account information is sent to the server for validation.
3. Based on the provided information the server responds with either true or false.
4. If 'true' the application navigates to the main menu and stores the user information in the local storage of the mobile device for auto login until the user logs out.
5. If 'false' the user will receive a message prompting him/her to check the information entered.

**Post conditions:** Successful login.

**Outcome:** The user can access the system and use the features of the application. Also, unauthorized users are prevented from using or accessing the system.

### 3.3.2 Search for item

**Description:** This use case is used to search for an item, such as a book, a CD or a magazine by entering its title or its author's name, in full or in part. The details of the items are already stored in the database.

**Preconditions:** Successful login.

**Basic flow of events:** The user selects the search type and fills in the required information in the search item text box. The information is then sent to the server for validation and to obtain a list of items fulfilling the search criteria, with details.

**Key scenarios:**

1. The user selects the preferred search type, either by item title or by author's name.
2. The user provides partial or complete media title or author's name.
3. The information is sent to the server to check for a possible information match in the database.

4. The result is a list of items in case of a match, and 'No title' in case no match is found.

**Outcome:** The user is able to search for items in the libraries which are part of the system.

### 3.3.3 User Account

**Description:** This use case is used to display user account details: the library card number, name of the customer, and the total late fee on his/her loans, if any.

**Preconditions:** Successful login.

**Basic flow of events:** The application retrieves user information that has been saved from local storage after login, the information is sent to the server for obtaining his/her possible total late fee.

**Key scenarios:**

1. The user clicks on the 'my account' button.
2. The application retrieves the user's library card number, and his/her name.
3. The information is sent to the server to check for any possible late fees in the user's account.
4. If there is a fee, the application calculates the total fee and displays it.

**Outcome:** The user obtains an electronic version of his/her library card and information about a possible total late fee.

### 3.3.4 Get libraries' events

**Description:** This use case is used to obtain a list of events in the libraries' which are part of the system, and their details, such as courses or book reading events.

**Preconditions:** Successful login.

**Basic flow of events:** The user clicks on a button and a request is sent to the server. The list of events will appear, and the user clicks on the event, after which the information is sent to the server for retrieving the event details.

**Key scenarios:**

1. The user clicks on the 'events' button.
2. The application sends a request to obtain the events, if any, and their details from the database in the server.

**Outcome:** The user is able to receive information about the libraries' events.

### 3.3.5 Send feedback

**Description:** This use case is used to send feedback about the application to the system manager.

**Preconditions:** Successful login.

**Basic flow of events:** The user fills in only the body of the message and clicks on the 'submit' button. The application sends this message to a special email account used by the system manager.

**Key scenarios:**

1. The user fills in the body of the message.
2. The user chooses from which email address he/she wants to send the message from.
3. The application sends the feedback to a special email account.

**Outcome:** The system manager receives feedback from the users of the application of their customer experience. As a result, it is possible to enhance the application's performance and to better match the users' needs.

### 3.3.6 User reservation

**Description:** This use case is used to obtain a list of the user's own reservations of items, and their details. In addition, the reservation status function allows the user to track his/ her reserved item.

**Preconditions:** Successful login.

**Basic flow of events:** The user clicks on a button and a request is sent to the server, the list of the user's reservations appears with their basic details such as item title, type, in which library the item will be available to be picked up, and the reservation status.

**Key scenarios:**

1. The user clicks on a button or navigates to the reservation page.
2. The application sends a request to obtain information about the user's reservations and their details from the database in the server.

**Outcome:** The user can see his/her own reservation list and track the reservation status.

### 3.3.7 Reserve items

**Description:** This use case is used to reserve an item from the system.

**Preconditions:** Successful login.

**Basic flow of events:** The user selects the item that he/she wants to reserve. The application asks the user to choose a library from which to receive the reserved item. The reservation information is then stored in the database.

**Key scenarios:**

1. The user clicks on a button to reserve an item.
2. The user chooses the library that he/she wants to receive the item from.
3. The user receives a conformation about his/her request.



4. The application sends a request to store his/her reservation to the database in the server.
5. Navigate to the user navigation list.

**Outcome:** The user is able to reserve items without going to the library and to choose the library which he/she prefers to pick up the items at.

### 3.3.8 User loan

**Description:** This use case is used to retrieve a list of the user's loans and their details.

**Preconditions:** Successful login.

**Basic flow of events:** The user clicks on a button and a request is sent to the server, the list of the user's loans appears, after which the user clicks on the item on loan and the information is sent to the server for obtaining details of that loan.

**Key scenarios:**

1. The user clicks on a button.
2. The application sends a request to obtain a list of the user's loans and their details from the database in the server.

**Outcome:** The user can follow the status of his/her loans.

### 3.3.9 Renew

**Description:** This use case is used to send a request to renew the loan of one of the items.

**Preconditions:** Successful login.

**Basic flow of events:** The application sends information about the item to the server to check if the user is able to renew the loan or not.

**Key scenarios:**

1. The user clicks on the ‘renew’ button.
2. The application sends information about the item to the server.
3. The application checks if something prevents the renewal, such as the item was renewed 5 times in a row, or the item was reserved by another user.
4. In case nothing prevents the renewal, the renewal is done for 2 weeks. In the opposite case, the application informs the user of the reason for the denial of the renewal.

**Outcome:** The user is able to renew his/her loan without visiting the library.

**3.3.10 Lend and borrow**

**Description:** This use case is used to lend an item to or borrow from a friend, or to borrow from a library.

**Preconditions:** Successful login.

**Basic flow of events:** The user fills in the item barcode, and the respective library card number and password, depending on the case. In case of ‘lend to friend’, the friend’s card number and password are needed. The information is sent to the server to check if the operation is possible.

**Key scenarios:**

1. The user clicks on the ‘lend/borrow’ button.
2. The user navigates to the respective operation page.
3. The user provides the item barcode by typing or scanning it.
4. The user provides his/her password.
5. In case of borrow to a friend, the user’s friend’s library card number is entered.
6. In case of lend to a friend, the user’s friend’s library card number and password are entered.
7. The account information is sent to the server for checking if there is a late fee on this item and if the item was reserved by another user.

8. Based on the provided information the server responds with either 'true' (there is a late fee or there is a previous reservation) or 'false' (no late fee and no previous reservation on this item).
9. If 'false', the application removes this item from or adds it to the user loans in the database.
10. If 'true', the operation is not possible, and a corresponding message is displayed.
11. In case of 'borrow from the library', there is no need to check the late fee or reservation. But after the operation is done, the system changes the book status from 'available' to 'not available' in the library.

The flow diagram below shows the steps of this use case.

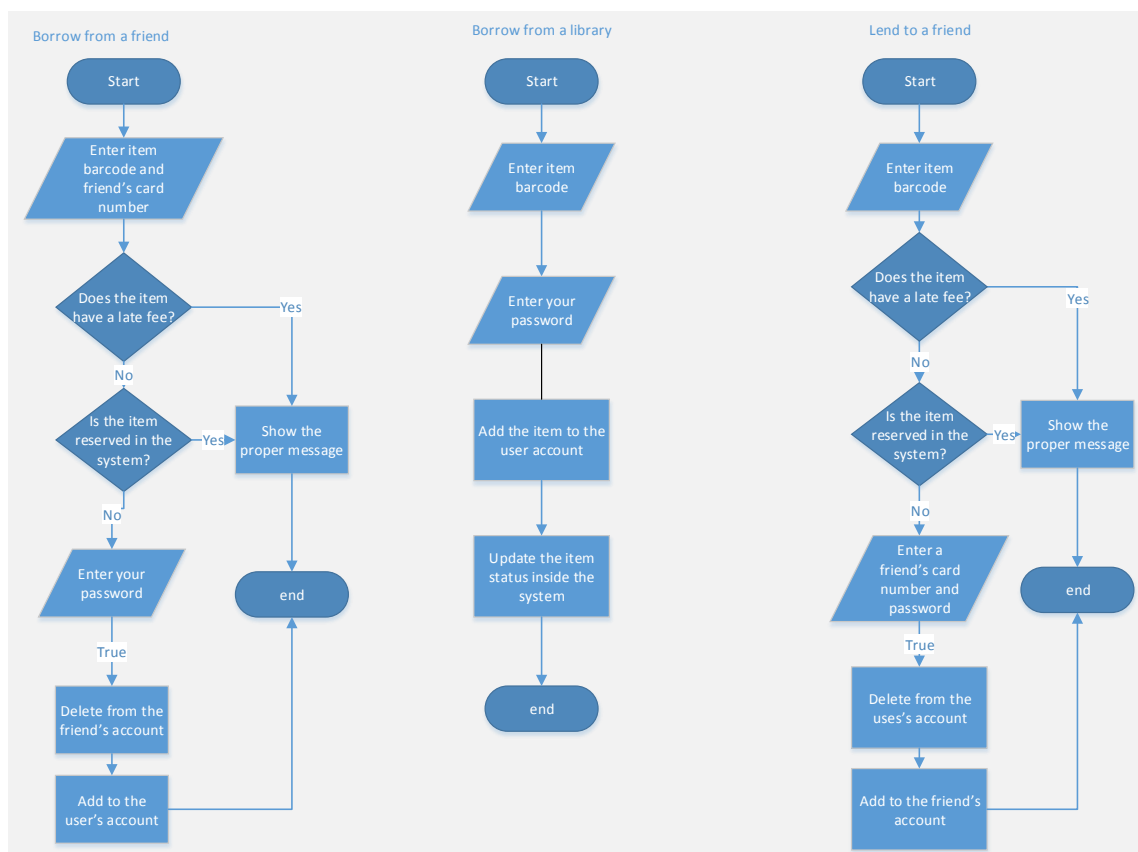


FIGURE 4. Use case of the lend and borrow function

**Outcome:** The user is able to borrow and lend items more independently without the assistance of a librarian, offering possibilities of increased automation.

### 3.3.11 Log out

**Description:** This use case is used to sign the user out of the application.

**Preconditions:** Successful login.

**Basic flow of events:** The application deletes the user information from the local storage of the mobile device, then navigates to the login page.

**Key scenarios:**

1. The user clicks on the 'log out' button.
2. The application deletes the user information from the local storage of the device.
3. The application navigates to the login page.

**Outcome:** Unauthorized access to the user's account is prevented by logging out.

## 4 SYSTEM FEATURE AND IMPLEMENTATION

The system is divided into two main software components, namely the administration software and the customer software, which are explained in detail in the sections below.

### 4.1 Administration web application features and implementation

The administration web application is web-based, used by library staff on a stationary device, although, in principle, it could also be used through a web browser on a mobile device with Internet access. This application is used to administer and manage many of the libraries' functions, as described below.

To start using the application, the user logs in, as shown in figure 5 below.

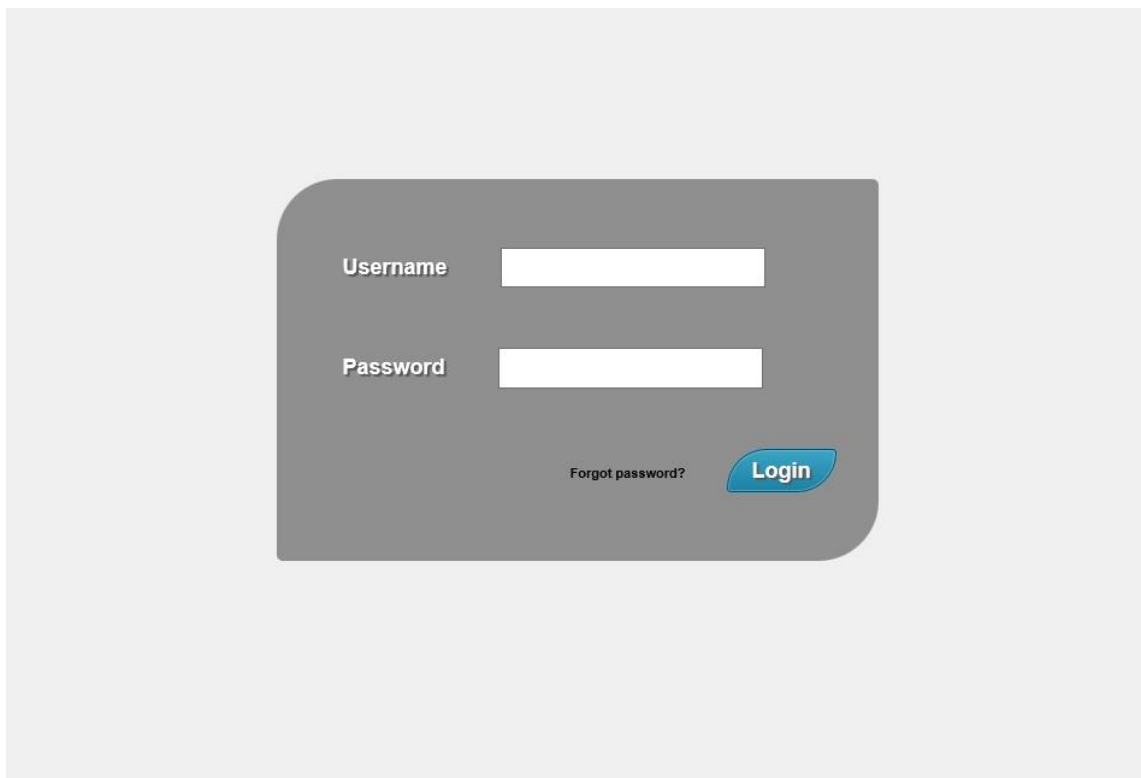


FIGURE 5. Login page

By entering the correct username and password the user can access the application. Should the username and/or password be incorrect, a message is displayed to that effect, prompting the user to try again.

In case the user has forgotten his/her password, he/she is able to retrieve the password through a link on the login page (figure 6).

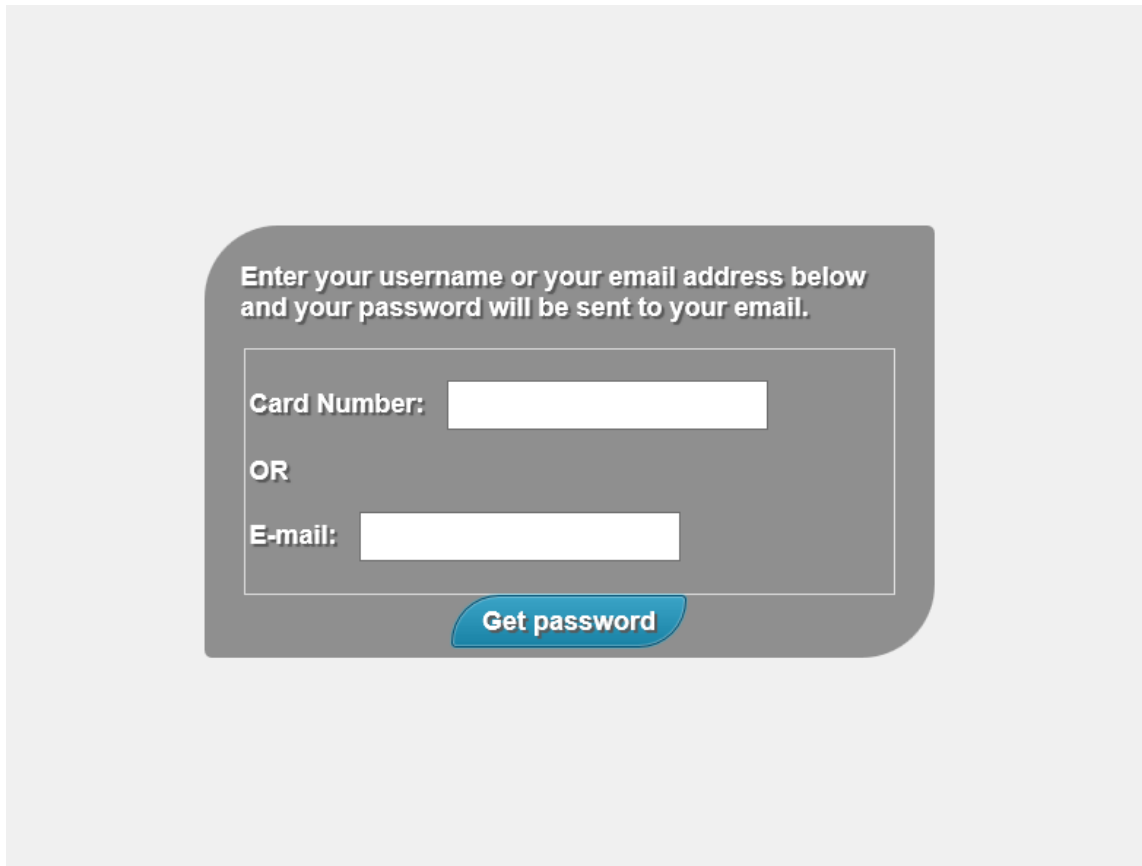


FIGURE 6. Forgot password page.

The password is then sent to the email address registered for the user in the system. The following is the email message which the user receives (figure 7).



FIGURE 7. Email message with account information.

In case the username and/or password were entered incorrectly, an email is not sent. Instead, a message to that effect is displayed on the 'forgot password' page.

After successful login, the user is directed to the main menu and is able to use the functions according to his/her role, i.e. Librarian or Manager (figure 8).

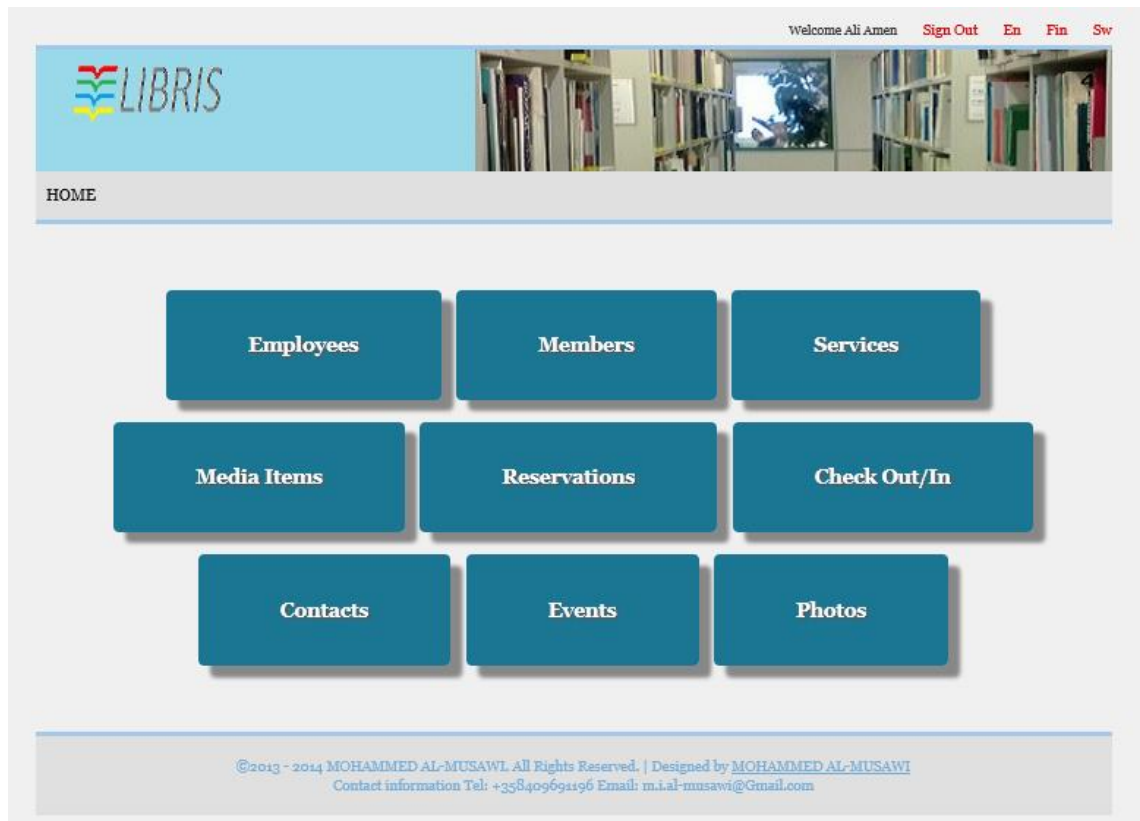


FIGURE 8. The main menu

From the main menu the user is able to navigate to different pages to perform different functions. These features are described below in detail. The name of the user is displayed on the top of the page in all the features, including the main menu.

#### 4.1.1 Employees feature

Both the Manager and the Librarian are able to use this feature. However, the level of access to information depends on the user's role. The Manager has access to the details of all employees in his/her library (figure 9). The Manager is able to modify most of the information and add or remove employees.

Welcome Marko Jäminki [Sign Out](#) [En](#) [Fin](#) [Sw](#)

**LIBRIS**

HOME [ADD EMPLOYEE](#)

### Employees

[Brand Tom](#)  
[Campbell Justin](#)  
[Jäminki Marko](#)

### Basic Information

Name	Justin Campbell
SSN	2222220000
Date of Birth	1979-04-24
Position	<input type="text" value="Librarian"/>
Salary	<input type="text" value="3000€"/>
Street Address	<input type="text" value="Almankatu 2 C 34"/>
Postal Code	<input type="text" value="00750"/>
City	<input type="text" value="Helsinki"/>
Email	<input type="text" value="justin@yahoo.com"/>
Primary Phone Number	<input type="text" value="040-111111102"/>
Secondary Phone Number	<input type="text"/>
Other Phone Number	<input type="text"/>

FIGURE 9. Employees page, Manager's main view

By clicking on the employee's name on the left, the Manager is able to see the employee's details and modify most of the information fields, if needed. Moreover, the Manager is able to delete employees. From the link 'Add employee' the Manager can navigate to a page where he/she is able to add a new employee by filling in the required fields which are marked with an asterisk (figure 10). The new employee is saved into the system by pressing the 'Add' button. The system will validate the information and indicate with a message if some of the required fields are missing, or if the information entered does not correspond to certain criteria (e.g. phone number must be numbers, only, or the name should have at least two characters).

The system also automatically generates a username and password for the new employee and sends them to his/her email. The username consists of the last two numbers of the SSN, the first letter of the first name, and the last name in its entirety. If the username pre-exists in the system, the system adds, to the end of the username, a random number between 10 and 99. The password is a random number between 1000 and 9999.



LIBRIS

HOME ADD EMPLOYEE

## Employees

[Brand Tom](#)  
[Campbell Justin](#)  
[Jäminki Marko](#)

First Name \*

Last Name \*

SSN \*

Birthday (YYYY-MM-DD)

Position \*

Salary

Street Address \*

Postal Code \*

City \*

Email \*

Primary Phone Number \*

Secondary Phone Number

Other Phone Number

©2013 - 2014 MOHAMMED AL-MUSAWI. All Rights Reserved. | Designed by MOHAMMED AL-MUSAWI  
 Contact information Tel: +358409691196 Email: m.i.al-musawi@gmail.com

FIGURE 10. Add employee page

If the Manager chooses to delete an employee, a message pops up urging the Manager to confirm the deletion (figure 11).

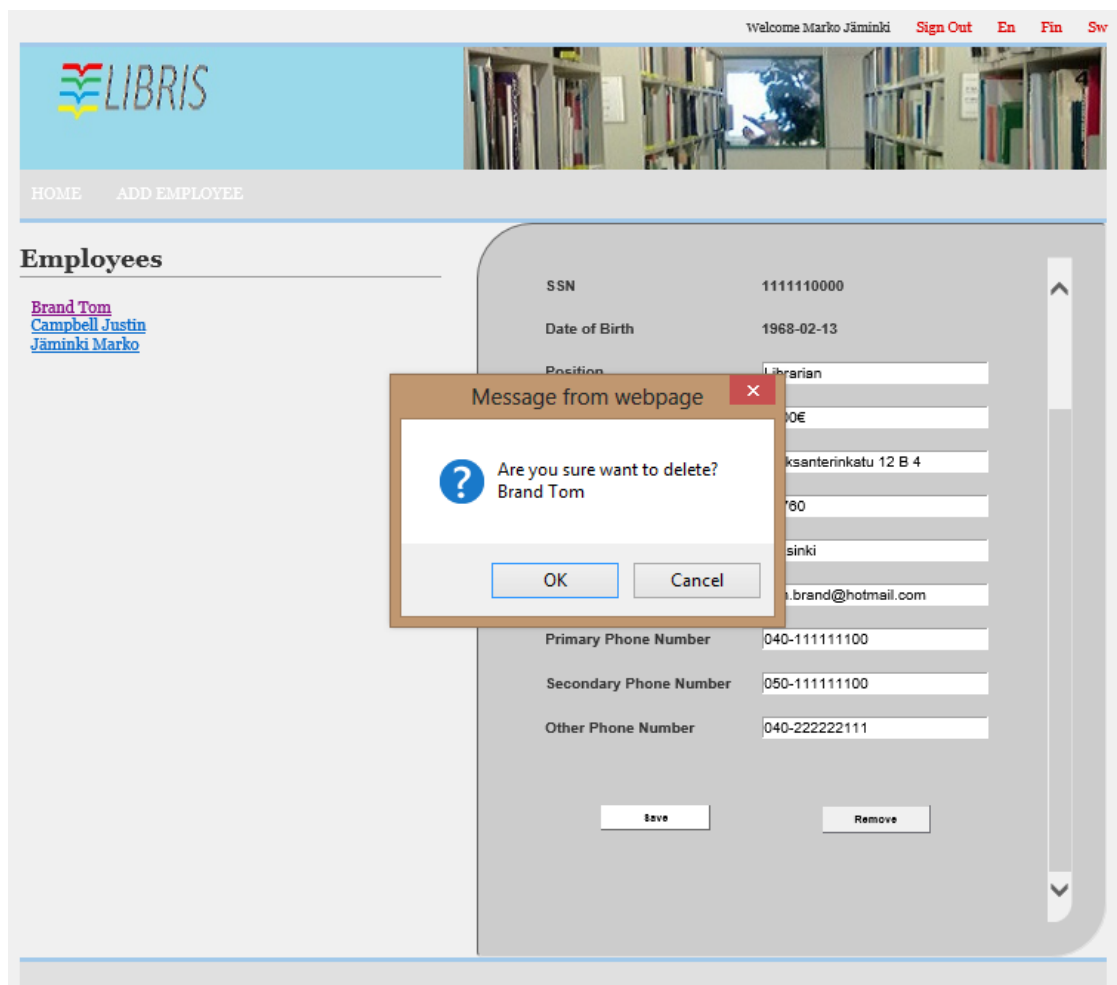


FIGURE 11. Delete employee confirmation request

The Librarian's main view (figure 12) differs from the Manager's one, as the Librarian is able to view only his/her own information which he/she is able to modify in part. The Librarian cannot, for example, modify information regarding the position or salary, whereas a Manager can. In addition, a Librarian cannot delete his/her profile, whereas a Manager can. The functions to add or remove employees, or to view their names or details, do not exist in the Librarian's view.

LIBRIS

HOME

## Employees

[Amen Ali](#)

### Basic Information

Name	Ali Amen
SSN	2222221111
Date of Birth	1983-04-13
Position	<input type="text" value="Librarian"/>
Salary	<input type="text" value="3000€"/>
Street Address	<input type="text" value="Kirvestie 100"/>
Postal Code	<input type="text" value="00730"/>
City	<input type="text" value="Helsinki"/>
Email	<input type="text" value="aliamen@hotmail.com"/>
Primary Phone Number	<input type="text" value="040-123456789"/>
Secondary Phone Number	<input type="text"/>
Other Phone Number	<input type="text"/>

Save

©2013 - 2014 MOHAMMED AL-MUSAWI. All Rights Reserved. | Designed by MOHAMMED AL-MUSAWI  
Contact information Tel: +358409691196 Email: m.l.al-musawi@gmail.com



FIGURE 12. Employees page, Librarian's main view

If the Librarian changes any information in the fields on the right, the system validates the information and checks that it corresponds to certain criteria (e.g. email includes an '@' sign and other email format details). If it does not, the system prompts the user to enter valid data where appropriate.

#### 4.1.2 Members feature

The 'Members' page is used by both the Librarian and the Manager. By entering a member's (i.e. customer's) library card number or SSN the user is able to access and edit information about the customer, and to view his/her loans. As illustrated in the figure 13 below, users are also able to renew the customer's loans from this page, and to add a new member to the system. If the user wishes to renew a loan, the system checks the feasibility of this operation, as in the use case described in chapter 3.2.11.

Welcome Marko Jaminki [Sign Out](#) [En](#) [Fin](#) [Sw](#)

[HOME](#) [ADD MEMBER](#)

### Search for Member

20000018866299 By

**Card Number** 20000018866299

**Name** Timo Mattila

**SSN** / / / / / / / / / /

**Date of Birth** 09-08-1974

**Phone Number**

**Street Address**

**Postal Code**

**City**

**Email**

**Membership date** 10-07-2014

---

**Loans**

Item	Type	Status	Due Date	Renewed	
ESCAPE FROM CAMP 14	Book	Renewed	22-07-2014	1/5	<a href="#">Renew</a>

©2013 - 2014 MOHAMMED AL-MUSAWI. All Rights Reserved. | Designed by MOHAMMED AL-MUSAWI  
Contact information Tel: +358409691196 Email: m.lal-musawi@gmail.com

FIGURE 13. Members page

From this page the users are also able to navigate by clicking on the ‘Add member’ link to add new members to the system. On this page the users can first check if the member is already registered in the system by entering the applicant’s SSN. If the applicant is not registered, the registration is done by filling in the form (figure 14).

HOME    ADD MEMBER

**Search for Member**

By SSN

**Basic Information**

Card Number \*

First Name \*

Last Name \*

SSN \*

Birthday (DD-MM-YYYY) \*

Phone Number \*

Street Address \*

Postal Code \*

City \*

Email \*

PIN \*

FIGURE 14. Add member page

After the user fills in all the fields in the form, the system checks that all fields are filled in, and, in some cases, if they are filled in the correct format (e.g. email should include the '@' mark and other details, phone numbers include numbers, only). If they are not, the system prompts the user to enter valid data where appropriate.

### 4.1.3 Services feature

The 'Services' page can be accessed by the librarian and the manager. On this page the user is able to see which services his/her library offers. Moreover, the user has the option to add or remove individual services available in his/her library (figure 15).

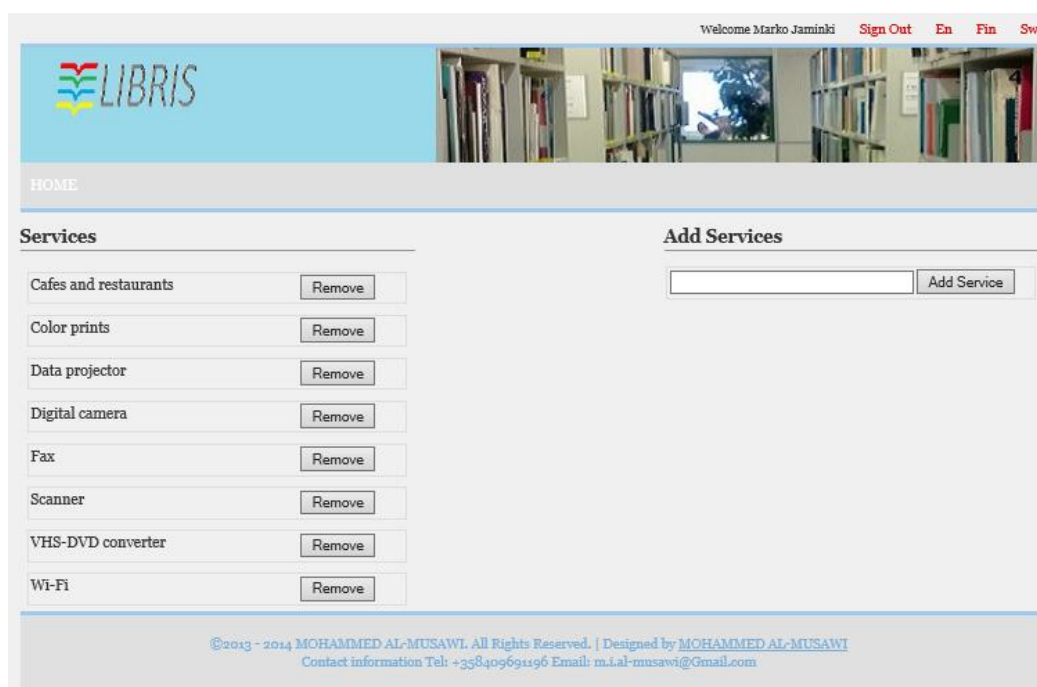


FIGURE 15. The Services page

The services available in the user's library are listed on the left. To add a service the user enters the name of the service in the appropriate field on the right.

#### 4.1.4 Media items feature

Both the Manager and the Librarian are able to use this feature with equal access rights. On the main page the user is able to search for media, or add media items into the system.

If the user wishes to add a media item into the system, after clicking the 'Add media' link, a page appears where he/she can enter the details of the media item. First, the user is able to check if the item is already in the system. This is done by entering the ISBN of the item in the first field (figure 16). If it is, a new page is opened, where the user only needs to enter the Library Barcode and Call Number, i.e. shelf location (figure 17). If not, a message prompts the user to fill in all the required fields.

HOME SEARCH FOR MEDIA

## Add Media

ISBN \*

---

Library Barcode \*

Call Number \*

Title \*

Edition \*

Year (YYYY) \*

Language \*

Media Type \*

Collection \*

Classification \*

Publisher name \*

---

## Author

Author No1 : First name \*  Last name \*

---

FIGURE 16. Add media page

If the item does not exist in the system, the user enters the required information, such as Library Barcode, Title, Edition, etc. in the form. The required fields are marked with an asterisk. In addition, some parameters are selected from a drop-down list. If the item has multiple authors, these can be added by clicking the ‘Add author’ button. After the user presses the ‘Add media’ button, the system validates the information and prompts the user if some fields are missing or not selected.

The system adds the author’s/authors’ and the publisher’s names to the database only if they do not already exist there in order to prevent multiple entries of the same name.

Welcome Marko Jäminki [Sign Out](#) [En](#) [Fin](#) [Sw](#)

**LIBRIS**

HOME SEARCH FOR MEDIA

### Add Media

ISBN \*

Library Barcode \*

Call Number \*

©2013 - 2014 MOHAMMED AL-MUSAWI. All Rights Reserved. | Designed by MOHAMMED AL-MUSAWI  
Contact information Tel: +358409691196 Email: m.i.al-musawi@Gmail.com


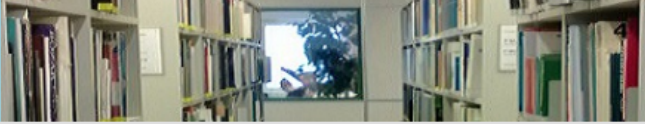
FIGURE 17. Add pre-existing media page

In case the media item already exists in the system, the user enters the library's own barcode and shelf location (Call Number). The required information is marked with an asterisk, and if it is not entered, a message prompts the user to enter it (figure 17).

On the main page, the user is also able to search for media items in all the libraries which are included in the system. There are two ways to search for items, namely the quick or classic search, which can be conducted by title, ISBN, or author, or the advanced search, which gives the user more options to limit the search (figure 18).



Welcome Marko Jaminki [Sign Out](#) [En](#) [Fin](#) [Sw](#)

[HOME](#) [ADD MEDIA](#)

## Search for Media

By

[Advanced Search](#)

- 1- [ESCAPE FROM CAMP 14/Harden Blaine](#)  
2012  
Edition:1
- 2- [Finnish for foreigners 1/Aaltio Maija-Hellikki](#)  
1987  
Edition:18
- 3- [Istanbul Nights/Dunya Yeni](#)  
2012  
Edition:1
- 4- [Kuuntele ja opi suomea/Paavilainen Ulla,Vuorio Nina](#)  
2010  
Edition:1
- 5- [Sanopa muuta/Manner Saaga,Nurmi Irmeli](#)  
2008  
Edition:1

Total 7 Record(s) : 2 Page(s) : 1 [ 2 ] [Next>>](#)

©2013 - 2014 MOHAMMED AL-MUSAWI All Rights Reserved. | Designed by [MOHAMMED AL-MUSAWI](#)  
Contact information Tel: +358409691196 Email: [m.l.al-musawi@gmail.com](mailto:m.l.al-musawi@gmail.com)

FIGURE 18. Classic search page

In the classic search the user enters the searched item's information fully or partially in the search field and selects the search mode (by title, ISBN, or author). After clicking on the 'Search' button, the user sees all the items stored in the database and matching the search, five items per page, as shown in the figure 18 above.

HOME SEARCH FOR MEDIA

### Advanced Search

s By Title

Library ANY

Language ANY

Media Type ANY  
Blu-ray  
Book  
Cassette  
CD  
DVD  
E-book  
Map  
Videocassette

Collection

Classification

Search Classic Search

- 1- [ESCAPE FROM CAMP 14/Harden Blaine](#)  
2012  
Edition:1
- 2- [Finnish for foreigners 1/Aaltio Maija-Hellikki](#)  
1987  
Edition:18
- 3- [Istanbul Nights/Dunya Yeni](#)  
2012  
Edition:1
- 4- [Kuuntele ja opi suomea/Paavilainen Ulla.Vuorio Nina](#)  
2010  
Edition:1
- 5- [Sanopa muuta/Manner Saaga.Nurmi Irmeli](#)  
2008  
Edition:1

Total 7 Record(s) : 2 Page(s) : 1 [ 2 ] Next>>

©2013 - 2014 MOHAMMED AL-MUSAWI. All Rights Reserved. | Designed by MOHAMMED AL-MUSAWI

FIGURE 19. Advanced search page

Apart from the title, ISBN and author, in the advanced search the user can define the search by library, language of the item, media type, collection, and/or classification. The user is free to choose one or several of these criteria. After clicking on the ‘Search’ button, the user sees all the items stored in the database and matching the search, as shown in the figure 19 above.

By clicking on the item link the user navigates to a page where he/she can see the details of the item and its availability in the different libraries with the option to reserve the item in the library of the customer’s choice to be delivered to any library which is part of the system (figure 20).

The screenshot shows the LIBRIS website interface. At the top, there is a navigation bar with the LIBRIS logo and a search bar. Below the navigation bar, there is a banner image of a library interior. The main content area is divided into two sections: 'Item details' and 'Availability information'.

**Item details:**

Author(s):	Parefäinen Ulla, Vuorio Nina
Title :	Kiinniele ja opi suomen
Year:	2010
Edition:	1
ISBN:	9789517924122
Item Type:	CD
language:	English
Classification:	Other
Collection:	Languages, language courses
Publisher:	FINN LECTURA

**Availability information:**

Library name: Pasila  
 Location: Kelloilta 9, Helsinki  
 Call Number: 11  
 Number of Items: 4 (4 Available)

Library name: Malmi  
 Location: Ala-Malmin tori 1, Helsinki  
 Call Number: 15  
 Number of Items: 1 (Available)

FIGURE 20. Item details and availability information page

The item details include information about the author or authors, year of publication, ISBN, language, and other details as shown in the figure 20 above. The availability information includes the name of the libraries which carry the item, the address of the libraries, the shelf information (Call Number), and the number of items the library carries with the information on how many copies are currently available.

The user is able to reserve the item by clicking on the 'Reserve' button, after which a reservation page opens (figure 21).

FIGURE 21. Reservation page

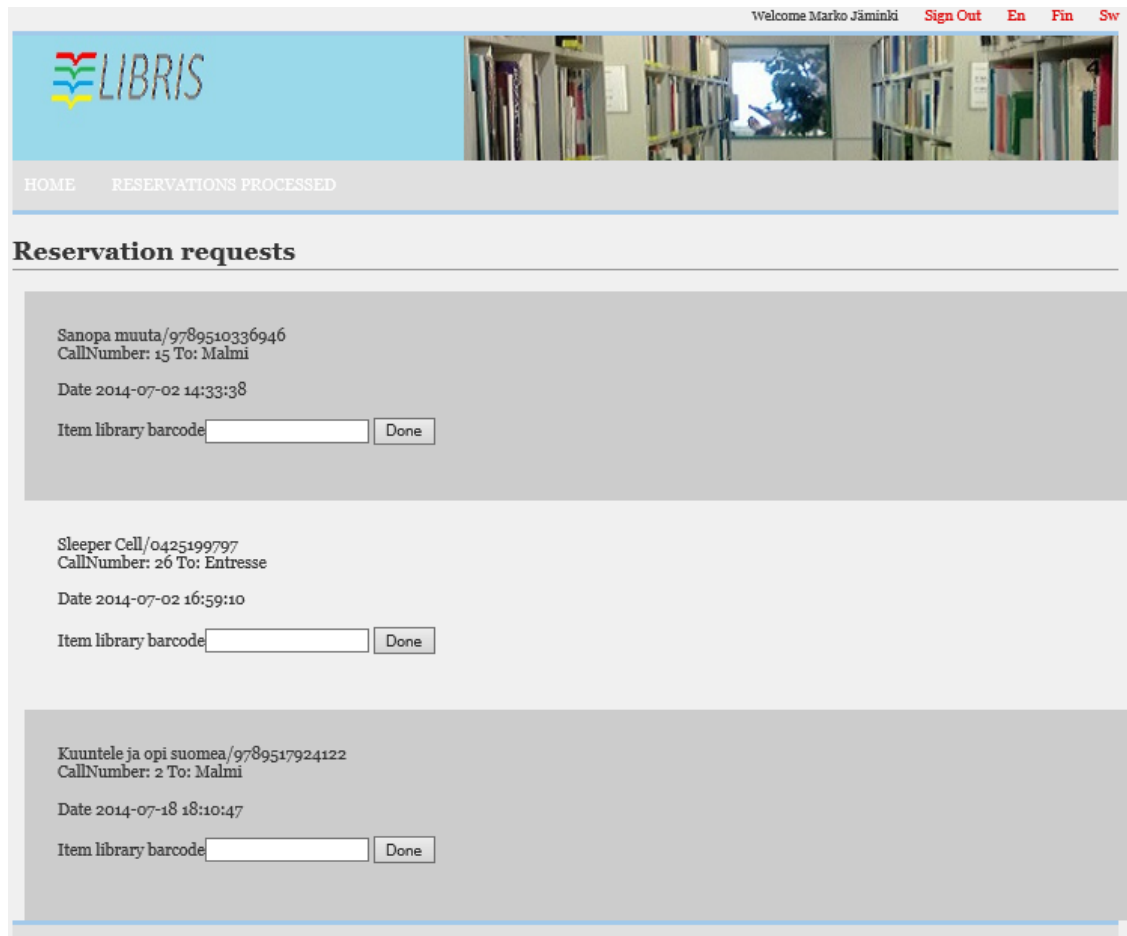
On the reservation page the user enters the customer's library card number and selects the library from which the customer wishes to collect the item.

FIGURE 22. Reservation completed message

After the reservation is completed, the user receives a message to that effect, as shown in the figure 22 above.

#### 4.1.5 Reservation management feature

Both the Manager and the Librarian are able to access and use this feature. In this feature the user can see all the processed and pending reservations of media items in his/her own library. On the reservation request page (figure 23) the pending reservations are displayed, with items currently available in the library, only, which the user is required to process.



Welcome Marko Jäminki [Sign Out](#) [En](#) [Fin](#) [Sw](#)

**LIBRIS**

HOME [RESERVATIONS PROCESSED](#)

### Reservation requests

Sanopa muuta/9789510336946  
CallNumber: 15 To: Malmi  
Date 2014-07-02 14:33:38  
Item library barcode:

Sleeper Cell/0425199797  
CallNumber: 26 To: Entresse  
Date 2014-07-02 16:59:10  
Item library barcode:

Kuuntele ja opi suomea/9789517924122  
CallNumber: 2 To: Malmi  
Date 2014-07-18 18:10:47  
Item library barcode:

FIGURE 23. Reservation requests page

The 'Reservation requests' page displays the main information about the item, including, for example, the shelf location, and to which library the item should be sent. This assists the user to process the request. The requests are listed in order of date and time, with the oldest request first. When the user wishes to process the reservation request, he/she enters the item's library barcode and clicks on the 'Done' button. After this, the item is moved to the 'Reservations processed' list (figure 24), and the copy of the item is no longer available in the library. The user can access this page by clicking on 'Reservations processed' link.

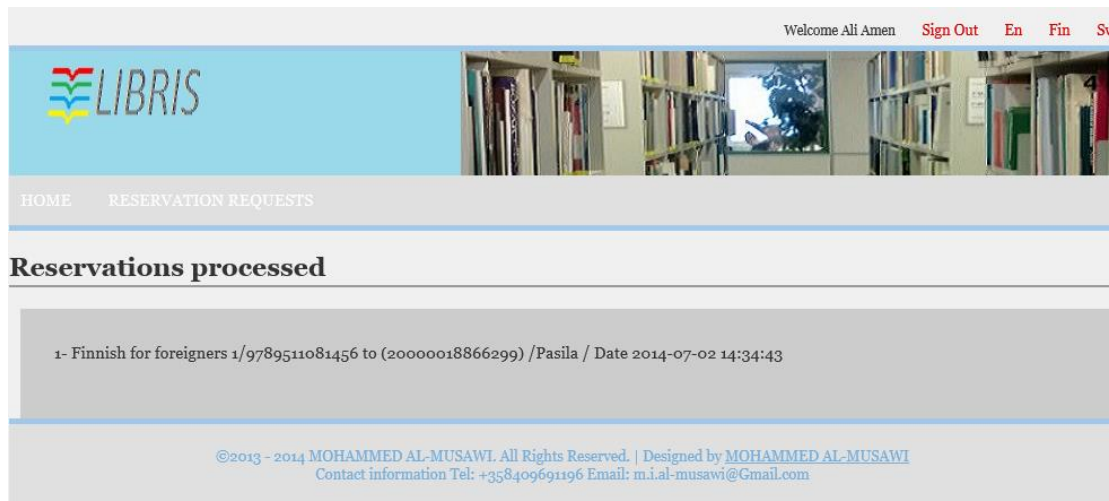


FIGURE 24. Reservations processed page

The ‘Reservations processed’ page indicates the processed item’s name, library barcode of the item, library card number of the customer, to which library the item should be sent, and the date and time of the processing.

#### 4.1.6 Check out/in feature

Both the Manager and the Librarian are able to use this feature. The ‘Check out’ page is used to lend items to the customer (figure 25). The user enters the customer’s library card number and the item’s library barcode. To add an additional item, the user only needs to enter a new library barcode. At the end, the user clicks on the ‘Check out’ button to finalize the transaction, and the system changes the availability information of the lent items automatically. It also adds the items to the customer’s account, with due dates counted automatically.

Welcome Ali Amen [Sign Out](#) [En](#) [Fin](#) [Sw](#)

**LIBRIS**

HOME CHECK IN

### Check out

Member card number

Library barcode:

**1- Finnish for foreigners 1/Book/9789511081456**

**2- Kuuntele ja opi suomea/CD/9789517924122**

**Total number of items = 2**

©2013 - 2014 MOHAMMED AL-MUSAWI. All Rights Reserved. | Designed by MOHAMMED AL-MUSAWI  
Contact information Tel: +358409694196 Email: m.i.al-musawi@gmail.com

FIGURE 25. The Check out page

When the customer returns items to any of the libraries in the system, the user clicks on the 'Check in' link, where he/she only needs to enter the library barcode of the item, as the system recognizes which copy of the items was lent to which customer. For additional items, another barcode is entered. After the user clicks on the 'Check in' button, the system automatically changes the availability information of the returned items and removes them from the customer's account. In case of delayed return after the due date, the system calculates the late fee for each item, and displays the total late fee (figure 26).

Welcome Ali Amen [Sign Out](#) [En](#) [Fin](#) [Sw](#)

**LIBRIS**

HOME CHECK OUT

### Check In

Library barcode:

---

**1- ESCAPE FROM CAMP 14/Book/9780143122913/2014-08-06 /2.00€**

**2- Istanbul Nights/DVD/002233/2014-08-02 /0.00€**

**Total fee on items = 2.00 €**

©2013 - 2014 MOHAMMED AL-MUSAWI. All Rights Reserved. | Designed by MOHAMMED AL-MUSAWI  
Contact information Tel: +358409691196 Email: m.i.al-musawi@Gmail.com

FIGURE 26. Check in page

If needed, the user can cancel the operation by clicking on the ‘Cancel’ button.

#### 4.1.7 Contacts feature

The ‘Contacts’ page can be accessed by the librarian and the manager. On this page the user is able to see the contact details of his/her library. Moreover, the user has the option to add or remove individual contacts (figure 27).



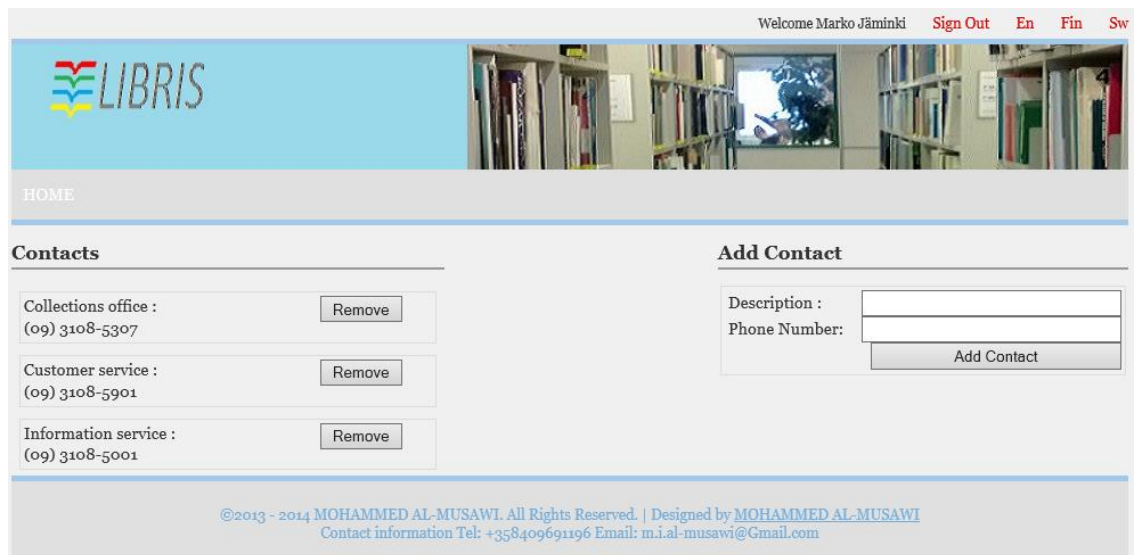


FIGURE 27. Contacts page

The contact numbers of the user's library are listed on the left. To add a contact the user enters the description of the contact and the phone number in the appropriate fields on the right.

#### 4.1.8 Events feature

Both the Manager and the Librarian can access this feature. It is used to view, delete, or add events in the user's own library. The events can range from Finnish courses to storytelling or author visits, for example. On the 'Library's events' page the user can see the events entered into the system (figure 28). He/she is also able to see and modify details of the events, or to delete the event altogether.

Welcome Marko Jäminki [Sign Out](#) [En](#) [Fin](#) [Sw](#)

**LIBRIS**

HOME [ADD EVENT](#)

### Library's Events

[Finnish Course 12-10-14](#)

[Storytelling 15-10-14](#)

### Event details

Event Name \*

Date (DD-MM-YYYY) \*

Start Time (HH:MM) \*

End Time (HH:MM) \*

Description \*

A perfect way to introduce children to exciting stories!




FIGURE 28. Library's events page

By clicking on the name of the event on the left, the user can see a form on the right side of the screen with the details of the event. It is also possible to modify these details and to change an image. Fields marked with an asterisk are the required fields, and certain format requirements exist (e.g. 24-hour format for the time). When the user clicks on the 'Update' button, the system validates the information and prompts the user in case some of the information entered is missing or not in the correct format.

By clicking on the 'Add event' link the user navigates to a blank form, where he/she enters the details of a new event (figure 29).

FIGURE 29. Add event form page

Fields marked with an asterisk are the required fields, and certain format requirements exist (e.g. 24-hour format for the time). To select the picture file for the event, the user clicks on the 'Browse' button. When the user clicks on the 'Add' button, the system uploads the photo and validates the entered information. The system also prompts the user in case some of the information entered is missing or not in the correct format. After a new event is added, it is automatically displayed on the events list on the left.

#### 4.1.9 Pictures feature

Only the Manager is able to access and use this feature. The Librarian only gets a blank page without photos or options. This feature is used to view, delete, or add photographs of the library into the system (figure 30).

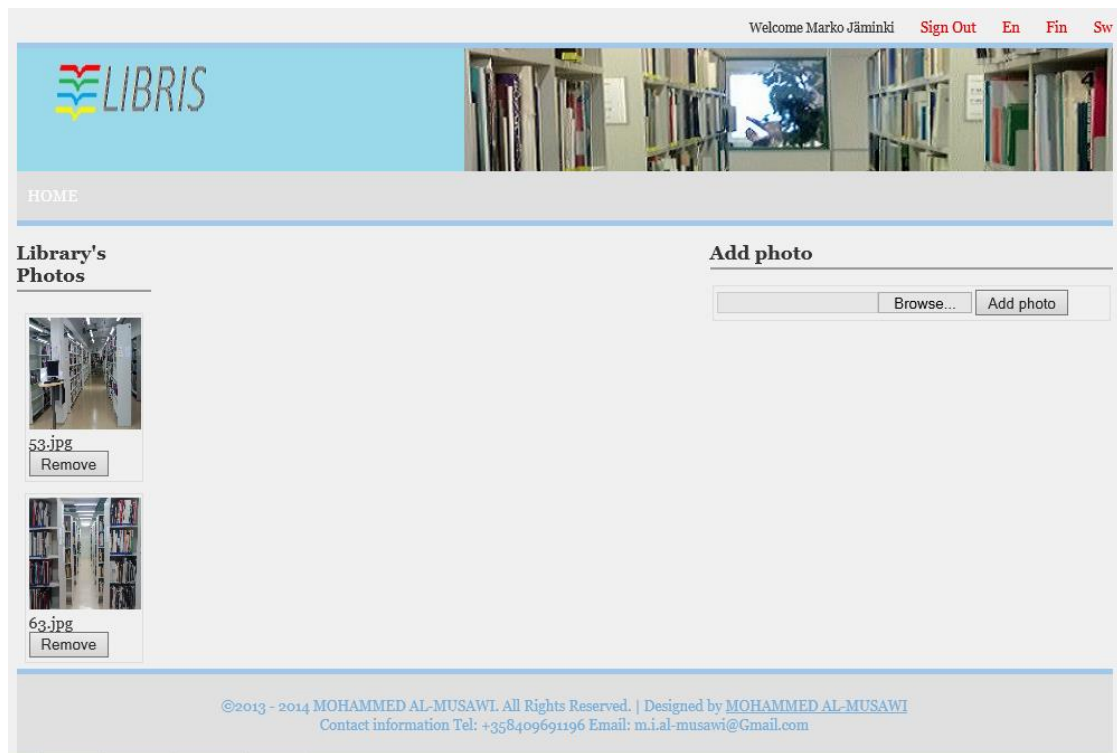


FIGURE 30. Library's photos page

Photos already uploaded are displayed on the left side of the screen. These can be removed individually by clicking on the 'Remove' button under each photograph. To add a new photo, the user should select the picture file by clicking on the appropriate field on the right side of the screen, and click 'Add photo'. After a successful upload, the photo is automatically displayed on the left.

## 4.2 Customer mobile application features and implementation

The customer mobile application is designed and implemented for Windows Phone devices. The application is for the use of library customers, only, and it is connected to the Internet in real time. The application is called 'LIBRIS' and the icon contains the system logo (figure 31.A).

When the user opens the application, the application checks if the mobile device is connected to the Internet or not. If not, the application displays a message 'Check your connection' to the user (figure 31.B). If the device is connected, and the user is accessing the application for the first time, the application displays the login page. For successive logins, as the user details are stored, the main page is displayed directly.

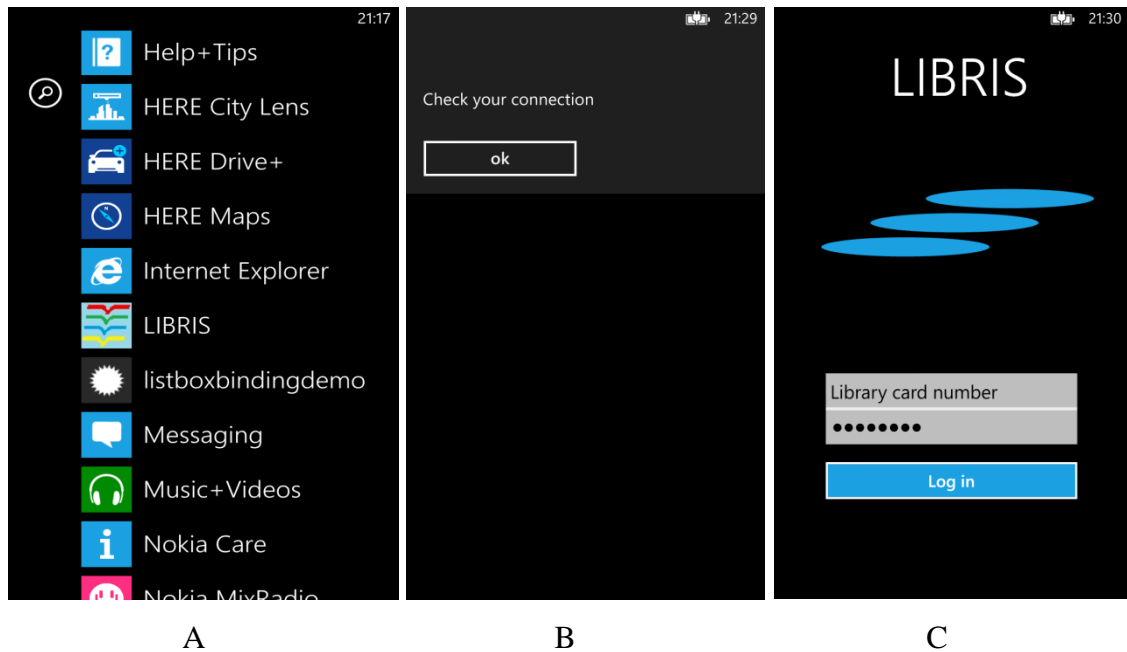


FIGURE 31. (A) Windows Phone application list. (B) Internet connection message. (C) Application login page

The user needs to enter his/her library card number and password or pin in the login page to be able to use this application and benefit from its functions (figure 31.C). The application checks the information which is provided by the user against the user information stored in the central database of the system. If the information is not correct the application displays a message ‘Invalid Username and Password’ on the screen of the device (figure 32). If the information is correct the application navigates to the main menu which has icons for all the functions that are available to the user.

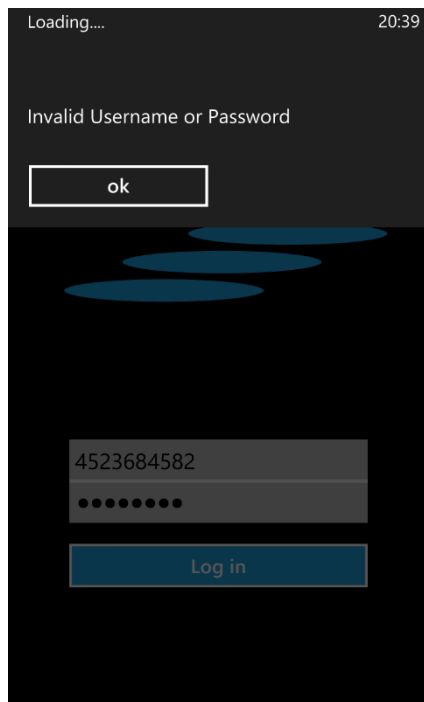


FIGURE 32. Invalid login information

The main menu consists of six buttons and an application bar. The buttons are meant for navigation to different pages to perform different functions (figure 33).

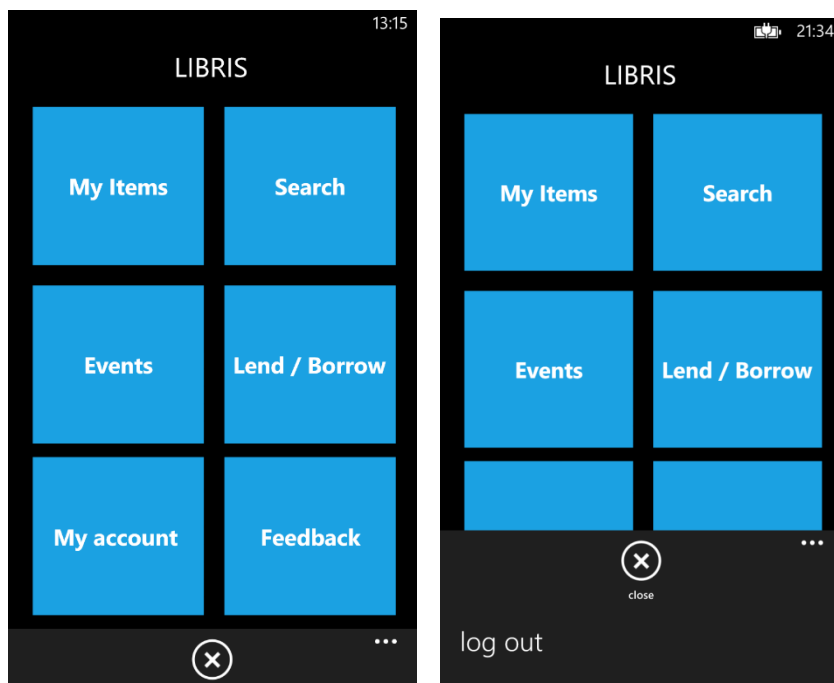


FIGURE 33. Main menu page

The application bar contains two buttons, one for closing the application while staying logged in (i.e. the user does not need to log in again when opening the application the

next time), and another for logging out (i.e. the login information is erased from the device memory). After logout the application navigates to the login page. The features of the main menu are explained below.

#### 4.2.1 My account feature

From the main menu the user can check his/her basic account information. It contains the same information as a normal library card, namely which the user name, and the card number. In addition, it displays the total late fee due from the customer (figure 34).



FIGURE 34. My account page

This feature could be used as an electronic library card. In addition, the user can check his/her late fee. If the user has a late fee, the device vibrates when the user opens this page to alert him/her of the fee, and the 'Total Fee' line is displayed in red. In case there is no late fee, the line is green and the device does not vibrate.

### 4.2.2 Feedback feature

One of the important things for libraries is to get feedback from the users about their experiences of using the library services, the application and the whole of the system. This helps the libraries to improve the user experience and also to better satisfy customer needs. This feature allows the user to send feedback to the library network easily and quickly.

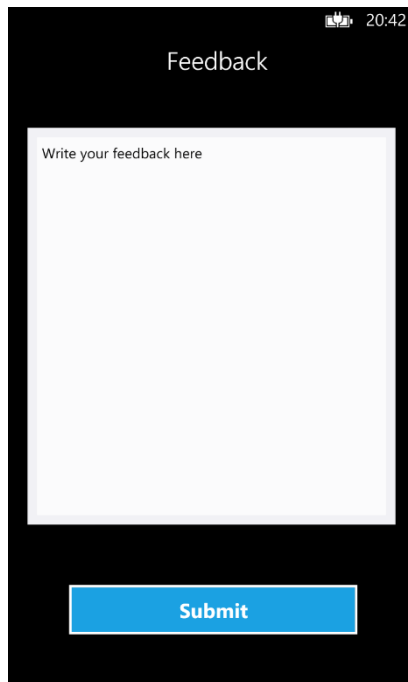


FIGURE 35. Send Feedback page

To send a feedback message, the user only needs to write his/her message and click on the 'Submit' button (figure 35). After this, the application prompts the user to select from which email account he/she likes to send the message. After this, the feedback is sent to a specific email in the system.

### 4.2.3 Events feature

This feature allows the user to view a list of all the events which are announced by the libraries. Information about each event on the list includes the event name, in which library the event take place, and the starting date and time (figure 36).





FIGURE 36. Events page

If the user is interested in one of the events on the list, the user can click on the desired event and the application displays all the details about the selected event (figure 37).



FIGURE 37. Event details page

This feature keeps the user up-to-date with the upcoming events in the libraries. In addition, the libraries benefit from this feature through an increased number of participants in the events.

#### 4.2.4 My items feature

This feature allows the user to view his/her loans and reservations, and to renew the loan of any item. The loan list shows all the media items that the user has borrowed from the system, including their main details, namely the title of the item, the year of publishing, the type of the item, and, most importantly, the due date of the item to avoid delay fees. Should the user have no loans, a message to that effect is displayed (figure 38).

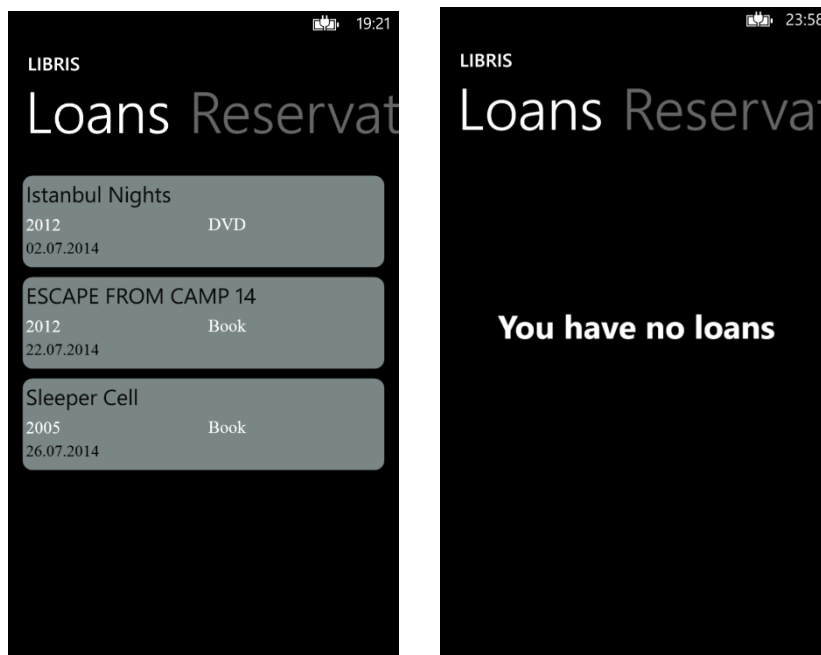


FIGURE 38. User's loans page

By clicking on one of the items on the list the user can see additional details about the selected item, namely the author(s) of the item, the late fee, and the language of the item (figure 39).



FIGURE 39. Loan item details page

The user has the ability to renew the loan of the selected item by clicking on the 'Renew' button. After this, the user receives a message to inform him/her if the renewal of the item was done successfully or not, depending on the renewal use case conditions mentioned in the chapter 3.3.9 (figure 40).

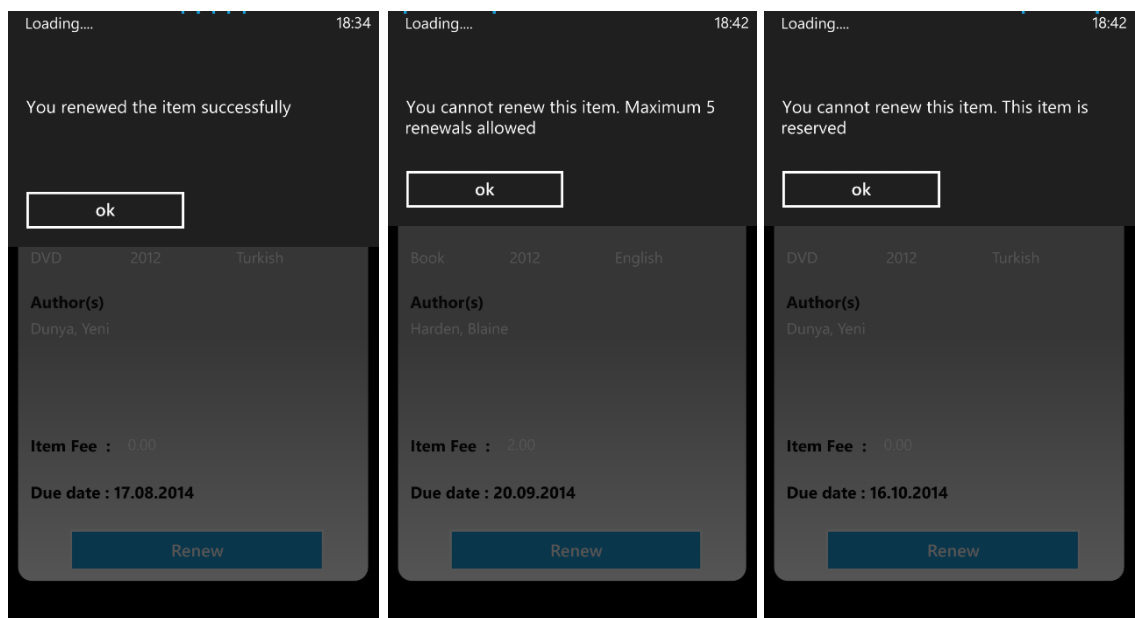


FIGURE 40. Renewal messages

The reservation list displays the items that the user has reserved from the system, including the main details about each item, namely the title of the item, the year of

publishing, the type of the item, which library the user has chosen to receive the item from, and, most importantly, the status of the reservation to allow the user to track the reservation status, i.e. if it is ready to be picked up from the desired library or not. Should the user have no reservations, a message to that effect is displayed (figure 41).

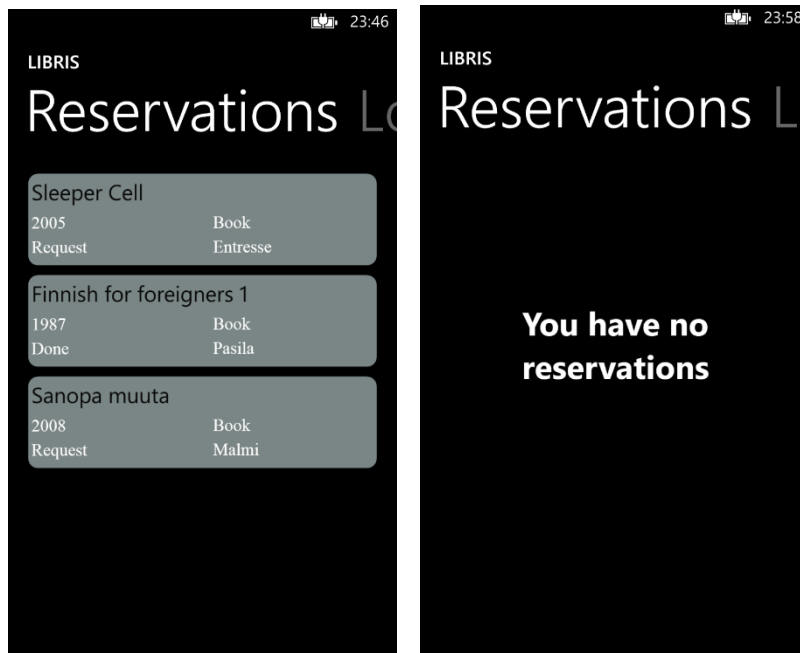


FIGURE 41. Reservations page

On the list, the word ‘Request’ indicates that the reservation is pending, and the word ‘Done’ means that the item is ready to be picked up from the library mentioned on the list.

#### 4.2.5 Search feature

This feature allows the user to search for items from the central database which includes all the items of all the libraries which are part of the system. The user has two choices for searching, namely by the title of the item, or by the name of the item’s author. The user provides partial or complete media title or author name and presses on the ‘Search’ button. The application displays a list of items that match the user’s entry, including the main details about the item, namely the title of the item, the year of publishing, the type of the item, the language of the item, and the edition of the item (figure 42).

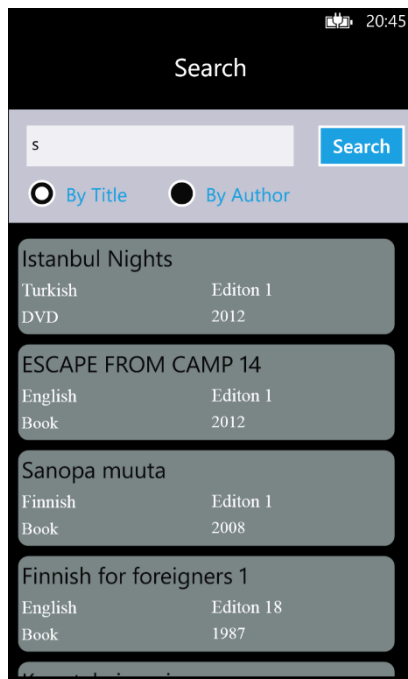


FIGURE 42. Search page

The user can select any of the items on the list by clicking on it, after which the application displays additional details about the item, i.e. the name(s) of the author(s), and in which library or libraries the item is available. Moreover, the user has an option to reserve the selected item immediately through the application (figure 43).

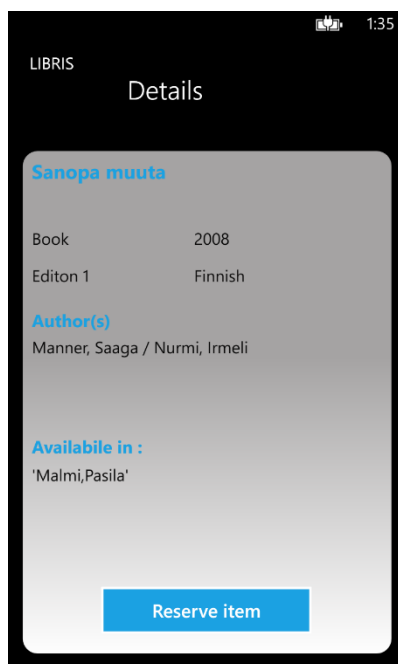


FIGURE 43. Item details and reservation page

If the user chooses to reserve the selected item by clicking on the ‘Reserve item’ button, the application asks the user to select the desired library to receive the item from (figure 44). By letting the user have access to all the library system, but to pick up the items from the library which is most convenient for him/her, the user has access to a wider selection of items and is not limited to the selection of the library next door.

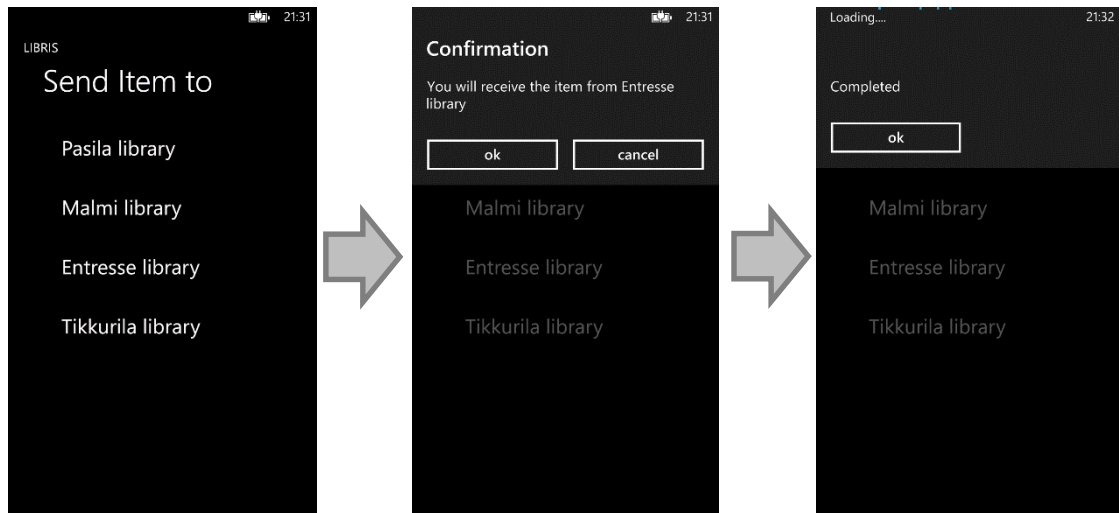


FIGURE 44. Selecting the library from which to pick up the desired item

After the user selects the desired library, the application displays a confirmation message to the user of his/her choice. If the user confirms the chosen library, the application automatically sends a reservation request to the database, and displays a message to inform the user that the reservation request is completed. The user acknowledges this by pressing ‘ok’, after which the application navigates to the user’s reservations list page.

#### 4.2.6 Lend / Borrow feature

This feature allows the user to borrow an item from a library directly, or from a friend who also has a library card. In addition, the application allows the user to lend an item from his own account to a friend with a library card.

This feature is like an automated check-out function without the need to have a librarian present. Some details of the functions and error messages are described below, without exhaustive lists of all the details in each case, as the functions resemble each other.

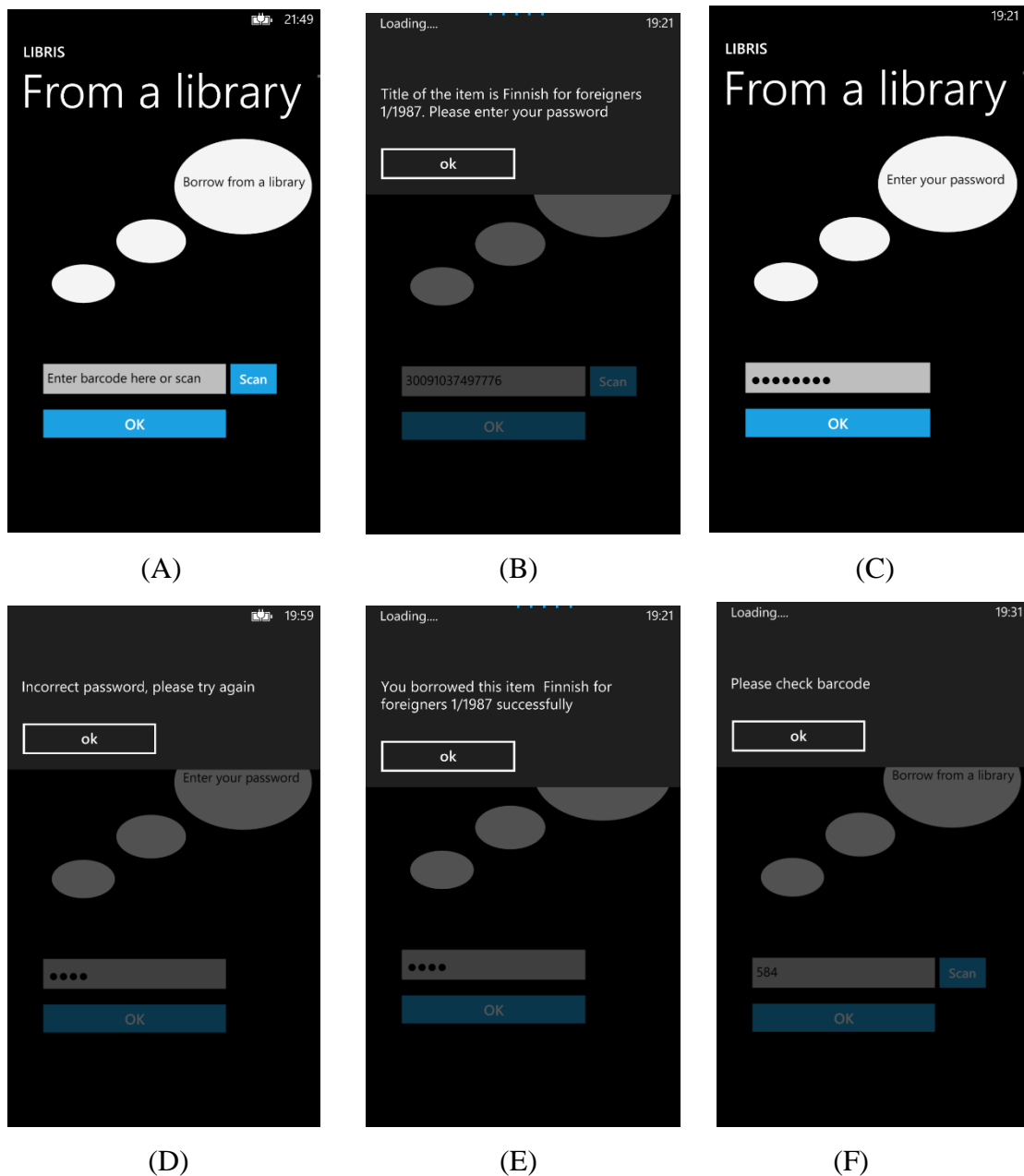


FIGURE 45. Borrow from a library page, with steps

In case the user borrows an item from the library, he/she only needs to enter or scan the library barcode of the desired item (figure 45.A). The application checks the barcode and displays a confirmation message to the user containing the title of the item and the year of publishing (figure 45.B). In addition, the application indicates what it is needed from the user for the next step. The application asks the user to enter his/her password to identify the account holder and to prevent misuse (figure 45.C). In case of an invalid password, a message to that effect is displayed (figure 45.D). After that, the user receives a message to inform him/her that the item has been borrowed successfully (figure 45.E). The item is automatically added to the user's account and its availability

status in the system changed. Should the user enter an invalid barcode, the application prompts him/her to check the barcode (figure 45.F).

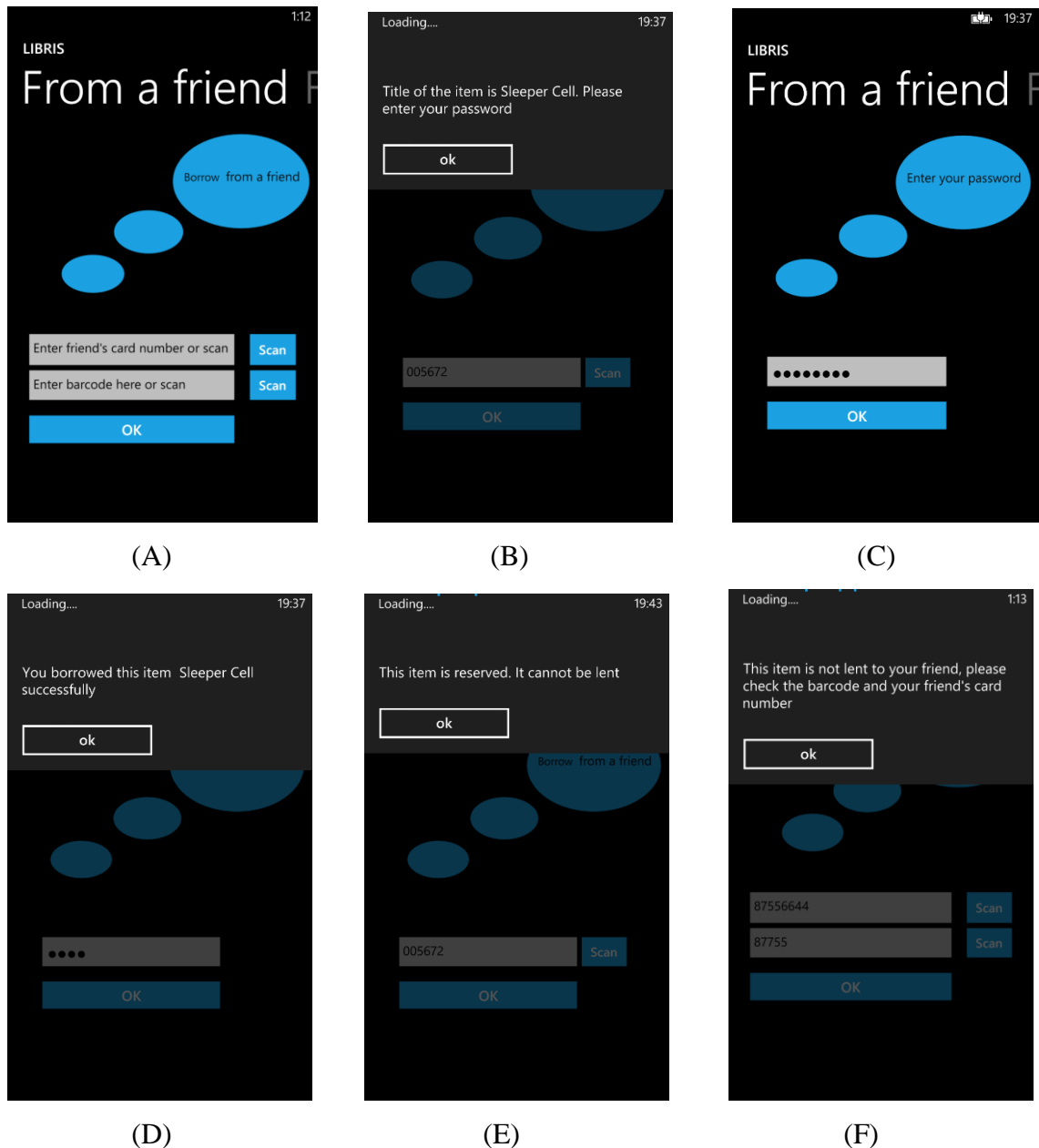


FIGURE 46. Borrow from a friend page, with steps

In case the user borrows an item from a friend who is also a registered customer with a library card, the application asks the user to enter or scan the library barcode of the desired item and the friend's library card number (figure 46.A). The application first checks if the item exists in the database and if the friend has the item on loan, and displays a confirmation message to the user containing the title of the item and the year of publishing (figure 46.B). In addition, it shows what it is needed from the user for the next step. The application also checks simultaneously the friend's account, if the system



accepts the user to borrow this item or not. See lend and borrow use case in chapter 3.3.10 to know the conditions of this function. If everything is accepted, the application asks the user to enter his/her password to identify the account holder and to prevent misuse (figure 46.C). If the password is incorrect, the user is prompted to check it, as in the case of borrowing from a library. The user then receives a message to inform him/her that the selected item has been borrowed successfully (figure 46.D).

If all the conditions for a successful borrowing are not met, for example if the item is reserved by someone else, a message to that effect is shown (figure 46.E). As another example, if the barcode entered is incorrect, or the item was not lent to the friend, the application informs the user to that effect (figure 46.F).

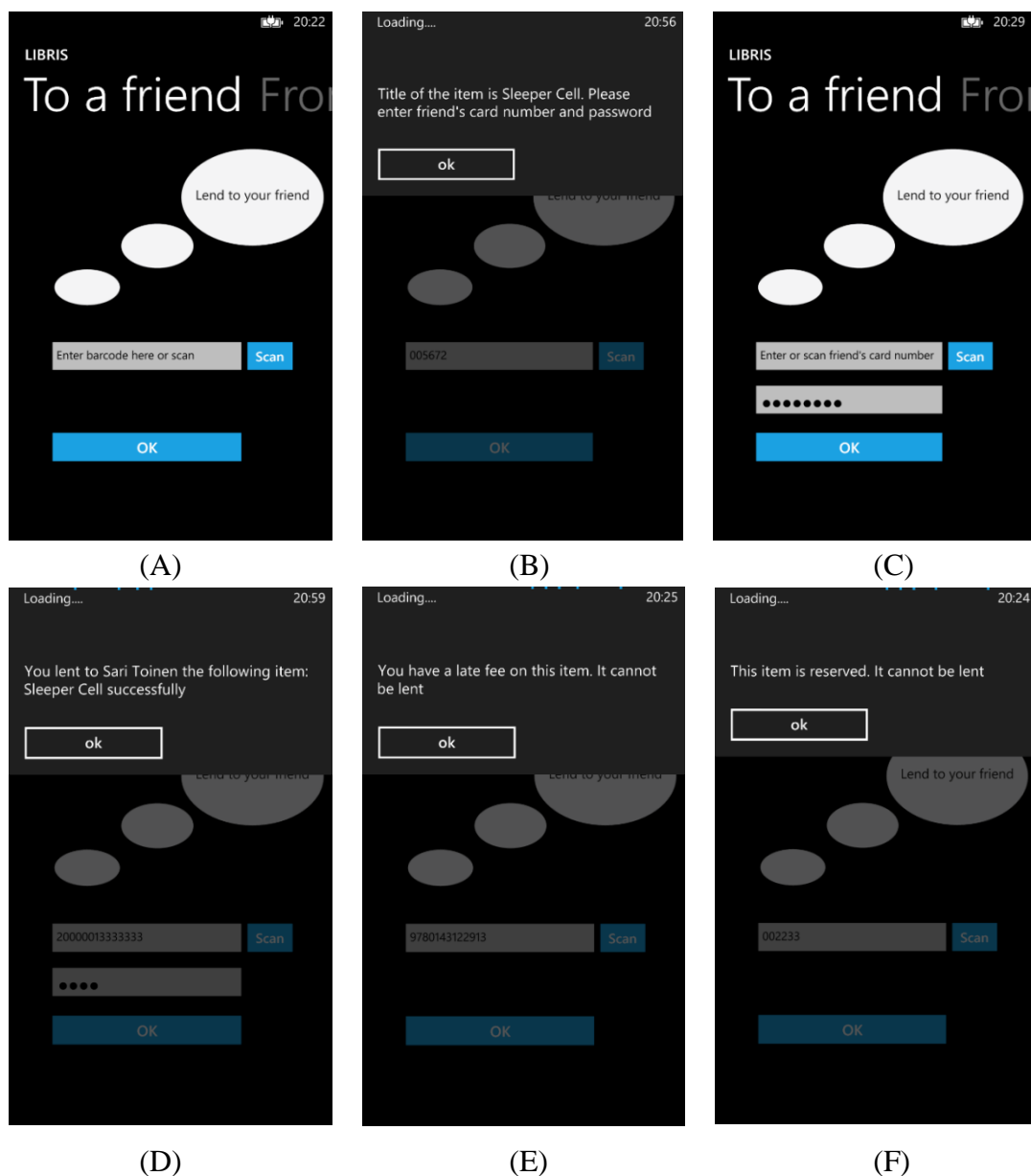


FIGURE 47. Lend to a friend page, with steps

In case the user lends an item to a friend who is also a registered customer with a library card, the application asks the user to enter or scan the library barcode of the desired item (figure 47.A). The application checks the barcode and verifies simultaneously that all the conditions to lend the item to a friend are met. See lend and borrow use case in chapter 3.3.10 to know the conditions of this function. If all conditions are met, the application displays a confirmation message to the user containing the title of the item, and prompts the user to enter the friend's library card number and password (figure 47.B), which are entered to prevent misuse (figure 47.C). If the password is incorrect, the application prompts the user to check it, as in the case of borrowing from a library. After that, the user receives a message to inform him/her that the selected item has been lent to a friend successfully (figure 47.D).

If all the conditions for a successful lending are not met, for example if the item has a late fee, a message to that effect is shown (figure 47.E). As another example, if the item is already reserved by someone else, the application informs the user to that effect (figure 47.F).

### **4.3 Database implementation**

In this project, the database is used to store all the data in the system which can be accessed through both the web application and the mobile application of this project. The database helps to organize and to manage a library or a group of libraries joined together under one system. MySQL is used as a relational database management system (RDBMS), and MySQL Workbench tool was used to design, develop and create SQL script (as seen in Appendix 1) of the database.

The database consists of eighteen entities which are connected to each other through relationships. The enhanced entity-relationship model below (figure 48) shows the relationship between the entities and the structure of each entity inside the database. The purpose of each entity is as follows:

**Libraries:** Represents the basic data of the libraries which are part of the system. This entity contains the name, the address, the geographical location, and the email addresses of the libraries, and gives a unique ID to each library in the system.

**Service:** Contains the services available in the libraries. This entity includes the name of the service (such as Colour printing, Scanner, Wi-Fi, etc.) and, should the service be available in a specific library, the ID of that library.

**Image:** Each library has its own images, namely pictures or photographs. In this entity, the names of these picture files are saved and used to create the path indicating the location of the pictures in the server. Each image has a unique ID, and the entity links to the ID of the library to identify which images belong to which library. The benefit of storing only the path or the name of the image, and not the image file itself, inside the database is to limit the size of the database and the consequent reduction of speed to access it.

**LibraryNumber:** Contains the different phone numbers of the libraries, such as the customer service numbers or the information services number.

**Event:** Each library has its own events, such as language courses, storytelling events, or author visits. In this entity, the information about these events is stored. The entity consists of the name of the event, the description about it, the time and date of the event, and the name of the image file for this event. Through the library ID, events are attributed to the specific libraries.

**Employee:** Contains information (such as names, addresses, user names, passwords and etc.) of all the employees of the libraries which are part of the system (figure 48).

**Members:** Contains personal information of the members, i.e. customers, of the libraries, such as their library card number, passwords, names, etc. (figure 48). In addition, this entity includes the SSN of the employee who registered a specific member into the system.

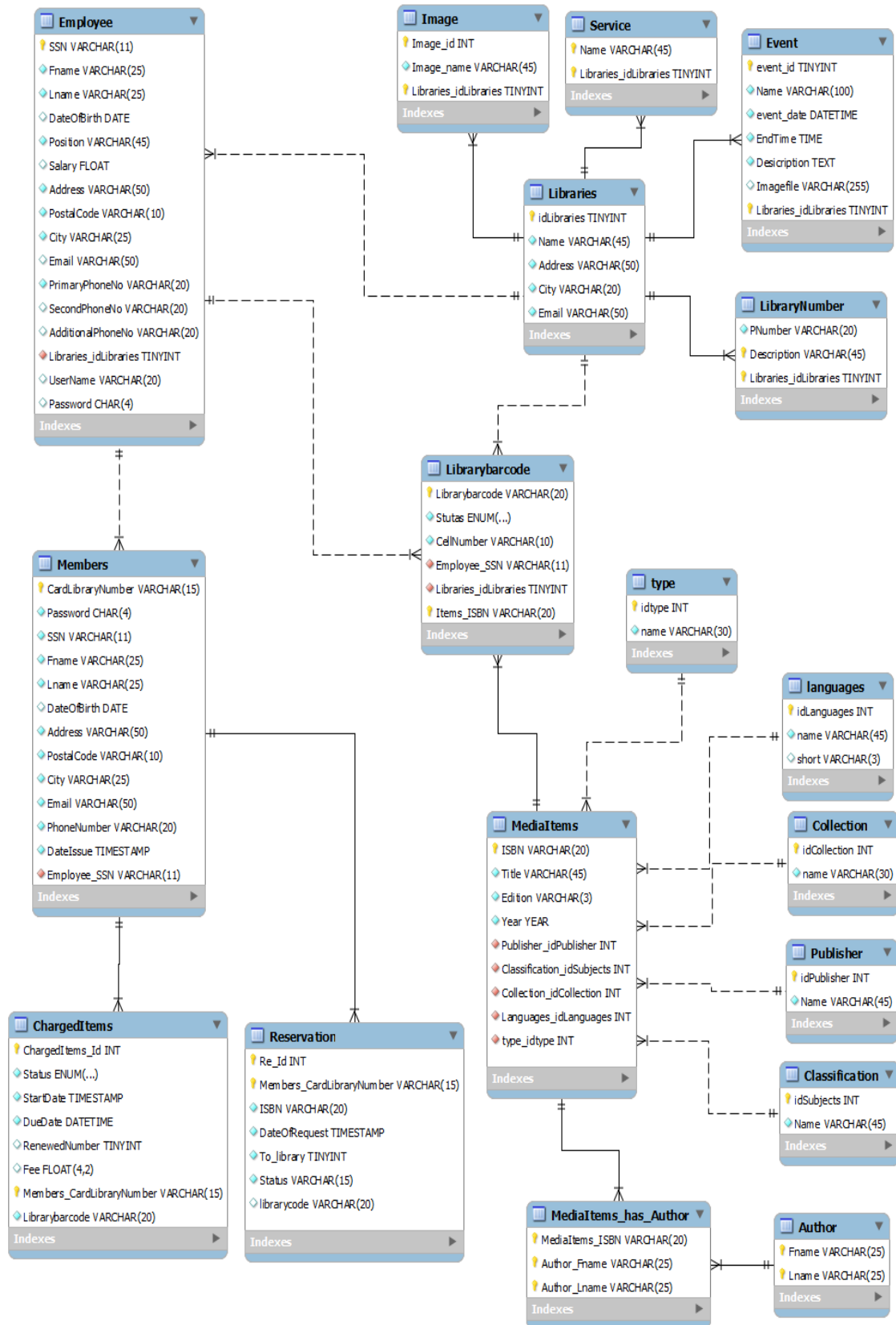


FIGURE 48. Enhanced entity-relationship model (EER) of the database

Librarybarcode: Each media item has its own ISBN, but the ISBN is the same for all the copies of the same version of the media item. As a result, libraries need their own code

for each copy of media items to identify from which library the copy originates. In this entity, the library codes are stored, together with other information concerning a specific media item, such as the status of this copy (available in the library or not), the number of the shelf the item can be found on, and the item's ISBN. Moreover, this entity includes information derived from other entities, such as which library the item belongs to, and who added this copy to the database.

**ChargedItems:** Consists of information of the media items (such as member card number and the item's library code) borrowed by the members. In addition, it includes information on the status of the media item. Firstly, the entity indicates if a member has borrowed it for the first time, or if the loan has been renewed and, if so, how many times. Secondly, it also indicates the start date and due date of the loan, and a late fee, if applicable.

**Reservation:** When the member reserves an item from the system, the information of the member's request will be stored in this entity. The basic information stored includes the member's card number, the ISBN of the item, and the date of the reservation request. In addition, the entity includes information about the library to which the reserved item should be sent, and the status of the reservation (e.g. waiting, processed, done) to make it easier for the member to track their reservation. Moreover, after the reservation has been processed by the library, the item library code will be stored here to identify the specific copy of the item.

**MediaItems:** This entity includes information about the media items, such as the ISBN, the title of the item, the edition, and the year of publishing. Other important information is obtained from the linked entities through relationships between them, such as the language of the item, the classification of the item, authors, publishers and the genre.

**Languages:** Contains a list of languages in which media items are available.

**Collection:** Contains a list of all the genres available and as specified in the system (e.g. fiction, non-fiction, music, etc.).

**type:** Contains a list of the types of media items (e.g. CD, book, DVD, magazine, etc.) which are available in the libraries that are part of the system.

**Publisher:** Contains a list of all the publishers of the media items.

**Classification:** Contains details of the classification of the media items.

**Author:** Contains a list of all the names of authors of the media items.

**MediaItems\_has\_Author:** Media items can have more than one author and vice versa. As a result, many-to-many relationships between the MediaItems entity and the Author entity are required. This entity allows for such relationships.

## 5 SYSTEM ARCHITECTURE

The system is divided into two sides (figure 49); namely the client side which includes the mobile application and the web application, and the server side which contains the web pages and the central database which has all the information of the libraries included in the system.

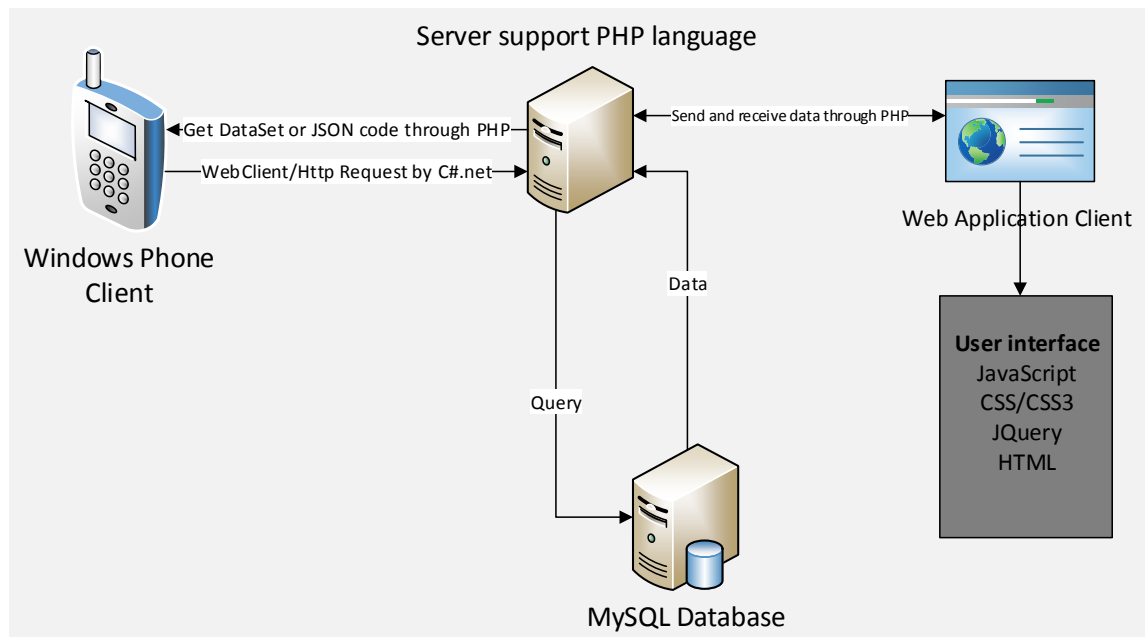


FIGURE 49. System architecture and techniques used.

The figure 49 above illustrates the numerous techniques and tools used to implement the system. These are explained in more detail in this chapter and in chapter 6.

### 5.1 Server side architecture

PHP web pages and MySQL database are used to implement the server side in order to communicate between the mobile application or the web application and servers to insert, modify and retrieve data from the database. In addition, the server side contains a special folder to store image files.

To insert and retrieve data from MySQL database, SQL queries were used. Some of these queries were integrated with PHP pages, while others, such as counting the late

fee every day for the customer, were performed automatically inside the database system.

The main task of the PHP pages is to send an SQL query to the MySQL database, and to retrieve the data from the same. Another task is to validate the user's input to ensure the quality of the information provided by the user.

## **5.2 Client side architecture**

In this system, there are two type of clients, namely the library's customer who is using the mobile application side, and the library managers and librarians who are using the web application side.

### **5.2.1 Mobile application architecture**

The mobile application is implemented on Windows Phone devices which operate on Windows Phone 8 operating system. The mobile application client side communicates with the server side using c# language to send a POST request for uploading and downloading information from the database using HTTP protocol. The request connects to a specific web page written in PHP on the server side. The task of the web page is to send an SQL query to the MySQL database, and to receive the data from the same. The data is then converted by PHP web page into JSON or DataSet string, depending on the original request. The client then receives the information from the server using C#.

DataSet is used for simple requests needed for sending and retrieving small data set, such as requests to check if the login details are correct, whereas more complicated requests use JSON, such as requests to obtain detailed information about a specific media item, for example a book.



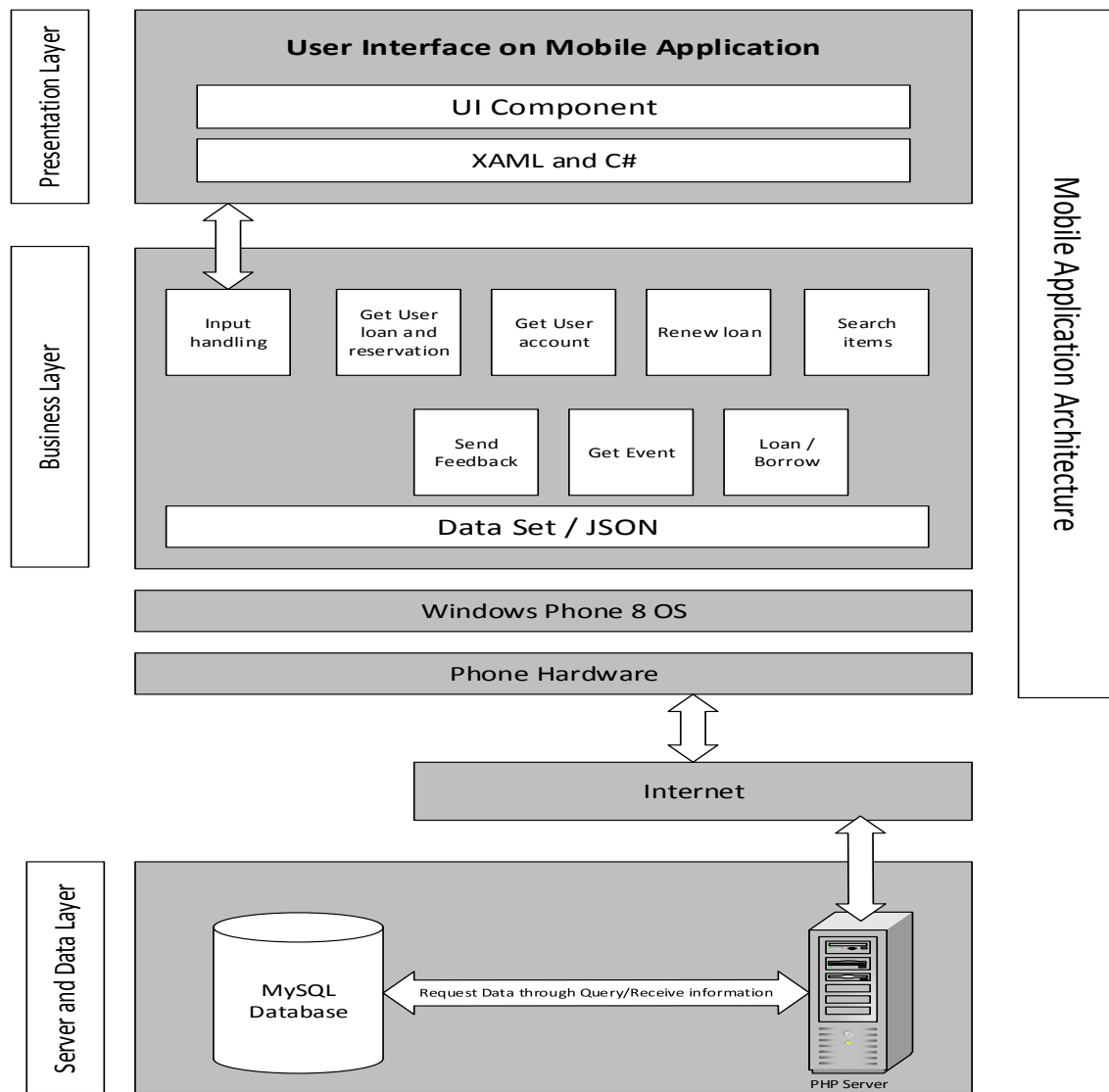


FIGURE 49. Mobile application architecture in layers

The mobile application's architecture in layers, as shown in the figure 49 above, can be explained as follows. The presentation layer is basically what the user sees on his/her device. C# and XAML languages were used to develop and implement this layer. The business layer illustrates the functions of the application and the data format for sending and receiving information within the application. The server and data layer includes the PHP web page and MySQL database which the mobile application communicates with.

### 5.2.2 Web application architecture

The web application client side consists of web pages that were written in different programming languages to create a suitable user interface to satisfy user needs, and to send POST requests to the server to obtain or modify information from the database

using HTTP protocol. The request connects to a specific web page written in PHP on the server side. The task of the PHP pages is to send an SQL query to the MySQL database, and to receive the data from the same. Moreover, it validates user input to ensure the quality of the information provided by the user. Web browsers are used to make the interaction between the user and the web pages, allowing the user to perform the required functions from any device that supports web service and web browsing.

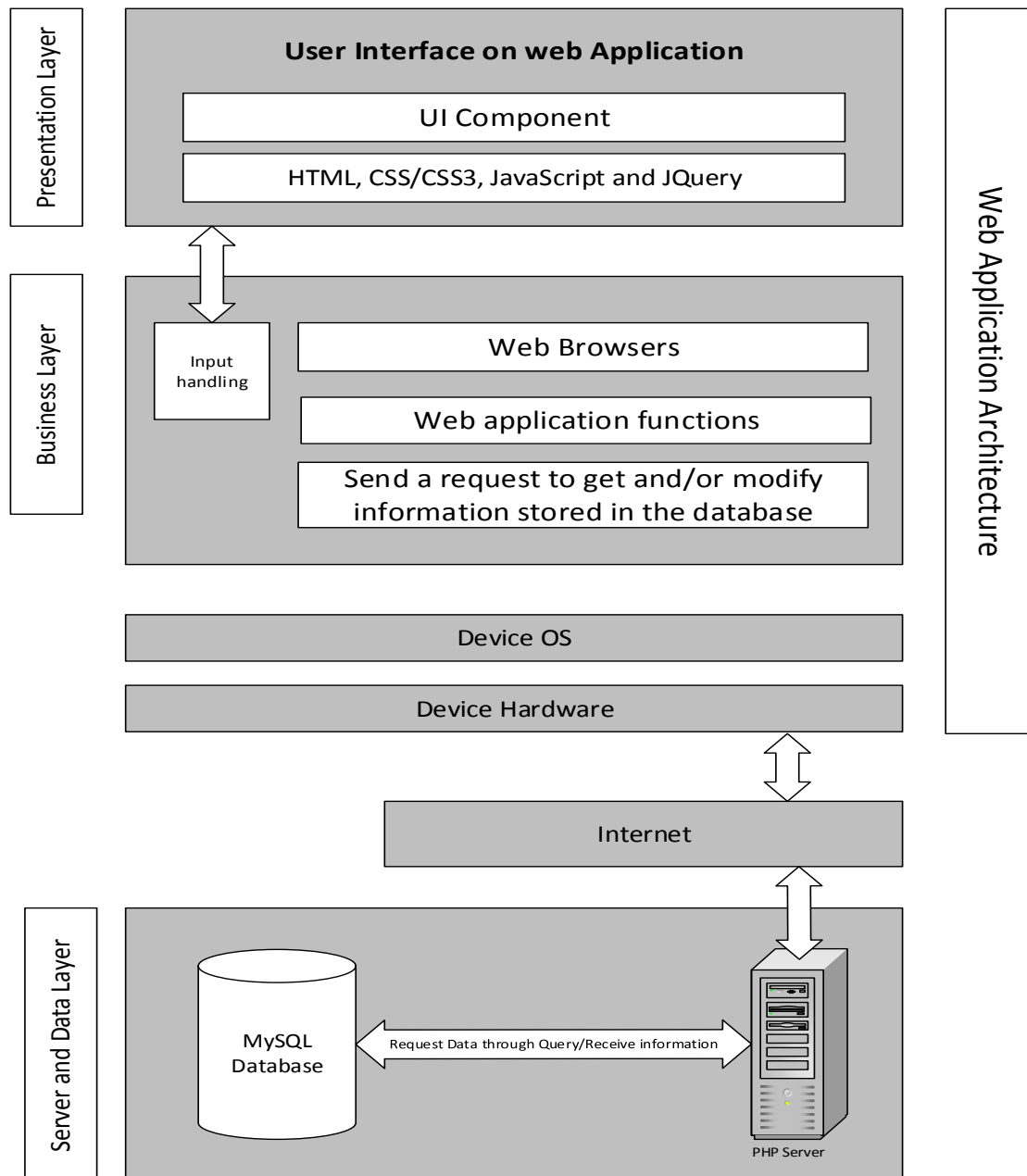


FIGURE 50. Web application architecture in layers

The web application's architecture in layers, as shown in the figure 50 above, can be explained as follows. The presentation layer is basically what the user sees on his/her device's browser. HTML, CSS, JavaScript and jQuery languages were used to develop

and implement this layer. The business layer illustrates the functions of the application and the HTTP request needed for sending and receiving information within the application. The server and data layer includes the PHP web page and MySQL database which the web application communicates with.

## **6 DEVELOPMENT ENVIRONMENT**

Different tools and techniques were used to develop and to implement this system. In the following sections, these tools and techniques, and the reasons behind using them, are explained in more detail.

### **6.1 Server side**

For developing the web pages, PHP 5 was selected in this system. PHP is an open-source server-side scripting language. Although PHP was mainly designed for web development, it can also be used as a general-purpose programming language. PHP code can be mixed with HTML code in a simple manner. PHP is, in fact, one of the most commonly used languages for web and web application development. In addition, PHP is freely available, and it is an excellent language choice when it comes to building dynamic websites that interact with databases.

The database of the system is MySQL database which was implemented and designed using the MySQL Workbench tool. MySQL Workbench is a unified visual tool for database architects, developers, and DBAs. MySQL Workbench provides data modeling, SQL development, and comprehensive administration tools for server configuration, user administration, backup, and much more (MySQL 2014). MySQL Workbench is available on Windows, Linux and Mac OS X and it is an open-source tool. MySQL database is a relational database management system (RDBMS), the world's most popular open source database. SQL language is used inside the PHP web pages to insert, delete, and retrieve information from MySQL database. The PHP MySQL programming enables PHP web developers and programmers to perform script installation, script repair as well as manage every front and back end activity, hassle free (AddCMS 2013).

In the system, to develop and test the PHP web pages and MySQL database performance, an Apache HTTP Server and MySQL database were needed. XAMPP was used to obtain these tools and to have a local host server which helps to develop the application of this system. XAMPP is an open source cross platform web server

solution stack package, including the Apache HTTP Server, MySQL database, PHP and Perl programming languages.

## **6.2 Client side**

### **6.2.1 Mobile phone application**

In the system, the Windows Phone application was developed by using Visual Studio 2012 tool which provides a comprehensive collection of developer tools and services. Visual Studio is an integrated development environment (IDE) from Microsoft which helps to create apps for the Microsoft platform, as well as web sites, web applications and web services. Mainly, C# programming language was used for coding the mobile application, and partially XAML was used for the design view of the application.

Blend 2012 for Visual studio was used in to design the mobile application contains. XAML is the language behind the visual presentation of the mobile application which developed and designed in Microsoft Blend (Microsoft Developer Network 2014).

One of the most important tools was Windows Phone Software Development Kit 8.0 (SDK) which provides the necessary tools which are needed to develop apps and games for Window Phone. The SDK should be included in the Visual Studio to be able to be used.

JavaScript Object Notation (JSON) was used in the mobile application for data-interchange format to receive the desired information from the database through PHP web pages in the server side. XML could be used for this purpose, but as JSON is a lightweight data-interchange format and it is easy for the humans to understand, read and write. Moreover, JSON is easy for machines generate. For example, in this system with one single command in PHP, the JSON code with the desired data can be created.

Another technique was used in the mobile phone application for real-time video scan of a barcode, which could be used in several functions in the mobile application. To develop this option in the Windows Phone application, an additional library needed to

be added to the application 'WP7.ScanBarcode' which allows performing a real-time video scan of a barcode (Hertrich Stephanie 2011). This library is used to scan all types of one-dimensional barcodes.

```
private void Scan_Click(object sender, RoutedEventArgs e)
{
    WP7.ScanBarcode.BarCodeManager.StartScan(
        // on success
        (b) => Dispatcher.BeginInvoke(() =>
        {
            if (lend1.SelectedItem == one)
            {
                Fcard.Text = b;
            }
            else if (lend1.SelectedItem == two)
            {
                Itemcode.Text = b;
            }
            else lib_barcode.Text = b;

            NavigationService.GoBack();
        })),
        // on error
        (ex) => Dispatcher.BeginInvoke(() =>
        {
            if (lend1.SelectedItem == one)
            {
                Fcard.Text = "Not found. Please try again";
            }
            else if (lend1.SelectedItem == two)
            {
                Itemcode.Text = "Not found. Please try again";
            }
            else lib_barcode.Text = "Not found. Please try again";
            NavigationService.GoBack();
        })),
        // Please, All 1D
        BarcodeFormat.ALL_1D);
}
```

FIGURE 51. Barcode scan code in the mobile application

The code the figure 51 above shows how to use the barcode library and its functions. When the user clicks on the 'scan' button, the application calls the barcode library and uses the 'StartScan' function to begin the real-time video scan. If the operation is done successfully, the scanned barcode is inserted into a special text box for further use. If it is not, the application displays a message about the error. Also, from the code, it can be seen that the barcode format is all types of one-dimensional barcodes.

### 6.2.2 Web application

In the system, the web application was developed by using different techniques and programming languages. HTML is a standard markup language which is interpreted by web browsers. HTML elements form the building blocks of all websites. They also create interactive forms. HTML is also used in this system to structure the web pages. Cascading Style Sheets (CSS) were used to shape and manage the layout and the look of all the HTML elements and their content. The CSS code can be integrated into the HTML code, but in this system, another option was used, namely to keep the CSS code as a separate file, with a reference to the file written in the HTML code. CSS is a style sheet language used for creating a more distinct look and style for the document written in a markup language. It is a powerful tool to implement and improve the user interface and user experience.

In the web application, JavaScript code was used to develop and to display confirmation messages to the user after the user performs a specific function. Also, JavaScript was used to navigate from one page to another. JavaScript, which is a dynamic computer programming language, allows for the quick and straightforward development of these kinds of functions. JavaScript library, which is called 'jQuery', was used in this application, mainly to dynamically add input form fields, such as forms for adding more than one author's name. JQuery is a fast, compact, versatile, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works with many browsers (jQuery 2014).

The web application in this system has three kinds of web development pages. Firstly, there are pages which are purely coded for the client side in HTML, JavaScript, and other techniques. Secondly, there are pages coded purely for the server side in PHP and SQL. And thirdly, there are pages which combine the two.

## 7 FUTURE DEVELOPMENT

There are numerous features that can be added relatively easily to the system, both on the mobile application side and on the web application side. These features can range from simple technical additions and modifications to those with wider implications to the whole library system and how it is used today.

One of the main development options would be to extend further the reach of library services with the help of modern technologies, even in remote locations without a library nearby. The one with possibly the widest implications would be allowing users to download e-books, with certain limitations. The service would be accessible, through an application, anytime and anywhere, with any device connected to the Internet, such as tablets, smartphones, PCs with Windows 8 etc. The downloaded e-books would be available for reading for a limited period of time even without access to the Internet. They would not be able to be shared.

The e-book option is especially relevant as, according to a 2013 Global eBook Survey, roughly 30% of respondents in the four largest Nordic countries expected to buy an e-reading device in the next 12 months (Norway 32.4%, Sweden 32.1%, Finland 29.2%, and Denmark 28.7%). More importantly, the same survey found that between 46.4% and 60.2% of respondents in the same countries expected their e-book consumption to reach 50% or more of all of their book consumption in the next three years, as illustrated in the figure 52 below. (Bookboon 2013.)

Other more minor development options for the mobile application include making the application available in other languages, such as Swedish and English, and to make the application available in other mobile operating systems and platforms, such as Android and iOS, which have a wider reach than Windows Phone 8. In addition, adding a barcode to the customer's library card would make it easier to use it as an electronic card (possibility to scan the barcode with the mobile device). This would eliminate the need for special equipment at self-service check-out counters in the libraries.



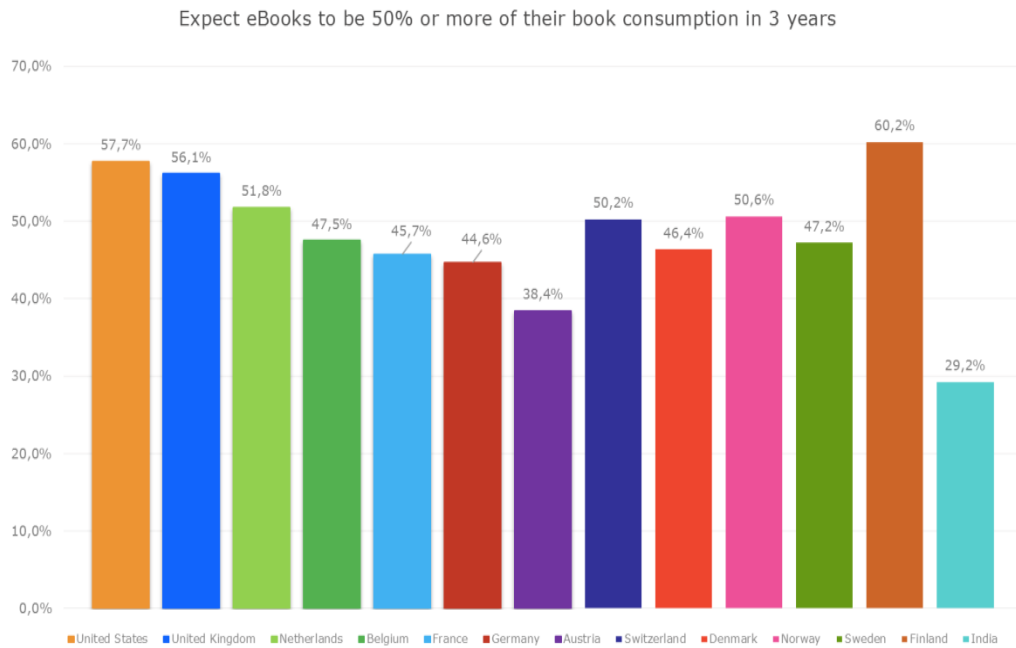


FIGURE 52. Expect e-books to be 50% or more of their book consumption in 3 years (Bookboon 2013).

As for the web application, a library information management system is a wide project and has many functions which can always be modified and developed further, depending on the stakeholders needs. A web application could also be created for the use of customers who prefer to use a web-based application as opposed to a mobile one. In this case the web application could also be extended to include other languages, such as Swedish and English, and possibly Russian and Estonian.

The web application and the central database can be expanded further to create a wide personnel management system, including, for example, attendance and absence information, salary payments, annual development discussion details, etc. This would allow the library to use a single portal for the management of the operations of the library, including HR.

The applications could also include the provision of paid services to customers, such as home delivery of reserved items, for example, which could benefit those with mobility restrictions or people in remote locations with a long distance to the nearest library. Such services could be partially or fully subsidized by the municipality as a part of their social programmes.

To make customer information more secure, data could be encrypted when sending and receiving it, especially when using the mobile application. Password security requirements can also be added so that simple passwords are not allowed, or that the customer is required to change his password in predetermined regular intervals by displaying a push-up notification to the customer when logging in and/or by an email informing the customer of the need to change the password.

When a big number of customers access the service simultaneously, there may be a need to improve system performance and to prevent excessive traffic to or from the server. This could be done by improving the hardware configuration of the server, especially by increasing the random access memory (RAM) capacity, since MySQL can use RAM to store frequently-accessed data. In addition, MySQL configurations file has special parameters to manage the load on the server and to improve the performance. These parameters include, for example, 'max\_connection' which, by increasing its own value, allows more users to access the database (Torres Gabriel 2013).

## 8 DISCUSSION

The thesis illustrates that it is possible to develop new service delivery methods for libraries using several different techniques and platforms, and that the Windows Phone 8 operating system can be used to implement an application to provide such new service delivery methods. It also shows that an application created for devices operating on Windows Phone 8 can easily and efficiently connect to and interact with a central database. Combining different techniques and utilizing a variety of tools allows for a wider range of functions and an enhanced user experience, both on the mobile application and the web application side.

Customers benefit from the project in several ways. Firstly, the mobile application gives the customer greater choice, as he or she is not limited to the selection of items in one library, only, but rather the whole library system. Secondly, it provides customers with more freedom, as they have many of the library network's services and information about them available to them at their fingertip at all times. Thirdly, the application allows the customer a greater option of services, namely the opportunity to independently loan from or borrow to a friend, or even borrow from a library without the need for a librarian's services.

Libraries also have many ways to benefit from the project. Arguably the biggest benefit is the opportunity for libraries to redirect existing limited human resources away from the routine tasks, such as check-outs, as the application provides for greater autonomy to the customer, allowing them, for example, to use self-service check-outs. There is consequently less need to acquire and maintain automated check-out machines, which leads to financial savings. Furthermore, libraries are able to easily advertise new services and events through the application, which may increase attendance and widen further their customer base, providing a justification to continue and further develop public library services. The web application allows libraries to use a single Internet-based platform to manage their own library's functions, reservations, and even human resources.

The architecture and design of both the mobile and the web applications allow them to be developed, expanded and customized further with relatively little effort. Functions and services which can be added in the future are too numerous to mention. They

include, for example, e-books, which would allow customers to either read a book online or download it for a limited period to read it in places without Internet access, for example in airplanes or remote areas. In addition, the application can include the collection of statistics of loans, which would allow the libraries to know customer preferences and to further develop the services to better meet the customer requirements.

The web application and the central database, especially the human resources component, can be expanded to include more HR-related functions, such as salary payments or professional development information, according to need. Needless to say, the mobile application can be made available in other operating systems, as well, and both the mobile and the web applications can be translated to other languages, such as Swedish, English, Estonian, or Russian, for example. The web application can be modified to be used not only for library management, but also by customers, so that a mobile device is not needed in order to have access to the same information as through the mobile application.

Paid services could also be made available through the applications. Such services could be, for example, home delivery of reserved items, relieving the customer from the need to visit the library physically. This would be especially beneficial to the elderly or people with otherwise limited mobility or those in remote locations. Such services could ultimately be selectively subsidized by the municipality.

## REFERENCES

- AddCMS. 2013. How is PHP used in Web Development. Read 15.04.2014.  
[http://www.contentmanagementsoftwares.net/php\\_web\\_development.htm](http://www.contentmanagementsoftwares.net/php_web_development.htm)
- Bookboon. 2013. Global eBook Survey 2013 by Bookboon. Read on 12.01.2014.  
<http://bookboon.com/blog/bookboon-coms-global-ebook-survey/>
- Center for Digital Government. 2014. Enterprise Mobility: Transforming Public Service and Citizen Engagement. Read on 01.08.2014.  
<http://www.nascio.org/events/sponsors/vrc/Enterprise%20Mobility-Transforming%20Public%20Service%20and%20Citizen%20Engagement.pdf>
- Hertrich, Stephanie. 2011. WP7 : Real-time video scan a barcode/QR code in your app using ZXing lib. Read on 26.07.2013.  
<http://blogs.msdn.com/b/stephe/archive/2011/11/07/wp7-real-time-video-scan-a-barcode-qr-code-in-your-app-using-zxing-lib.aspx>
- JQuery. 2014. What is JQuery?. Read on 15.04.2014. <http://jquery.com/>
- Kirjastot.fi. 2007. Kysy kirjastonhoitajalta –kysymys. Read 02.05.2014.  
<http://www.kirjastot.fi/kysy/arkistohaku/kysymys/?ID=c9dbc24e-f1f5-4d6f-9d69-670fc65a73ff>
- Libraries.fi. 2014. Public Libraries. Read 02.05.2014. [http://www.libraries.fi/en-GB/libraries/public\\_libraries/](http://www.libraries.fi/en-GB/libraries/public_libraries/)
- Microsoft Developer Network . 2014. What is XAML?. Read on 15.05.2014.  
<http://msdn.microsoft.com/en-us/library/cc295302.aspx>
- Ministry of Education and Culture. 2013. Statistics on municipal public libraries. Read 01.05.2014. <http://www.minedu.fi/OPM/Kirjastot/tilastot/?lang=en>
- Ministry of Education and Culture. 2014. Library network in Finland. Read 01.05.2014.  
<http://www.minedu.fi/OPM/Kirjastot/kirjastoverkosto/?lang=en>
- MySQL. 2014. MySQL Workbench. Read on 15.04.2014.  
<http://www.mysql.com/products/workbench/>
- Nygrén, Toni. 2013. Market Visio. Windows Phone noussut Suomen johtavaksi älypuhelinlustaksi. Read on 03.05.2014.  
<http://www.marketvisio.fi/fi/ajankohtaista/uutiset-marketvisio/1703-windows-phone-noussut-suomen-johtavaksi-lypuhelinlustaksi>
- Suomen virallinen tilasto (SVT): Väestön tieto- ja viestintätekniikan käyttö. ISSN=2341-8699. 2013. Helsinki: Tilastokeskus. Read 03.08.2014.  
[http://www.stat.fi/til/sutivi/2013/sutivi\\_2013\\_2013-11-07\\_tie\\_001\\_fi.html](http://www.stat.fi/til/sutivi/2013/sutivi_2013_2013-11-07_tie_001_fi.html)
- Tabor, B., Rutakas, C. & Lieberman, L. 2013. Windows Phone 8 development for absolute beginners. Microsoft: Channel 9.

Torres, Gabriel. 2013. How to optimize a MySQL server. Read on 20.08.2014.  
<http://www.hardwaresecrets.com/printpage/How-to-Optimize-a-MySQL-Server/1747>

## APPENDICES

### Appendix 1. Database SQL script

1(12)

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0;
SET@OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
```

```
DROP SCHEMA IF EXISTS `dbe3malmus1` ;
CREATE SCHEMA IF NOT EXISTS `dbe3malmus1` DEFAULT CHARACTER SET
utf8 COLLATE utf8_general_ci ;
USE `dbe3malmus1` ;
```

```
-----
-- Table `dbe3malmus1`.`Libraries`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`Libraries` ;
```

```
CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Libraries` (
  `idLibraries` TINYINT UNSIGNED NOT NULL ,
  `Name` VARCHAR(45) NOT NULL ,
  `Address` VARCHAR(50) NOT NULL ,
  `City` VARCHAR(20) NOT NULL ,
  `Email` VARCHAR(50) NOT NULL ,
  PRIMARY KEY (`idLibraries`));
ENGINE = InnoDB;
```

```
-----
-- Table `dbe3malmus1`.`Employee`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`Employee` ;
```

(continues)

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Employee` (
  `SSN` VARCHAR(11) NOT NULL ,
  `Fname` VARCHAR(25) NOT NULL ,
  `Lname` VARCHAR(25) NOT NULL ,
  `DateOfBirth` DATE NULL ,
  `Position` VARCHAR(45) NOT NULL ,
  `Salary` FLOAT NULL ,
  `Address` VARCHAR(50) NOT NULL ,
  `PostalCode` VARCHAR(10) NOT NULL ,
  `City` VARCHAR(25) NOT NULL ,
  `Email` VARCHAR(50) NULL ,
  `PrimaryPhoneNo` VARCHAR(20) NOT NULL ,
  `SecondPhoneNo` VARCHAR(20) NULL ,
  `AdditionalPhoneNo` VARCHAR(20) NULL ,
  `Libraries_idLibraries` TINYINT UNSIGNED NOT NULL ,
  `UserName` VARCHAR(20) NULL ,
  `Password` CHAR(4) NULL ,
  PRIMARY KEY (`SSN`) ,
  INDEX `fk_Employee_Libraries1_idx` (`Libraries_idLibraries` ASC) ,
  CONSTRAINT `fk_Employee_Libraries1`
    FOREIGN KEY (`Libraries_idLibraries`)
    REFERENCES `dbe3malmus1`.`Libraries` (`idLibraries`)
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`Service`
-----

```

```

DROP TABLE IF EXISTS `dbe3malmus1`.`Service` ;

```

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Service` (
  `Name` VARCHAR(45) NOT NULL ,

```

(continues)



```

`Libraries_idLibraries` TINYINT UNSIGNED NOT NULL ,
PRIMARY KEY (`Libraries_idLibraries`, `Name`),
CONSTRAINT `fk_Service_Libraries1`
FOREIGN KEY (`Libraries_idLibraries`)
REFERENCES `dbe3malmus1`.`Libraries` (`idLibraries`)
ON DELETE CASCADE
ON UPDATE CASCADE)
ENGINE = InnoDB;

-----
-- Table `dbe3malmus1`.`Event`
-----

DROP TABLE IF EXISTS `dbe3malmus1`.`Event` ;

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Event` (
  `event_id` TINYINT UNSIGNED NOT NULL AUTO_INCREMENT ,
  `Name` VARCHAR(100) NOT NULL ,
  `event_date` DATETIME NOT NULL ,
  `EndTime` TIME NOT NULL ,
  `Desicription` TEXT NOT NULL ,
  `Imagefile` VARCHAR(255) NULL ,
  `Libraries_idLibraries` TINYINT UNSIGNED NOT NULL ,
PRIMARY KEY (`event_id`, `Libraries_idLibraries`),
INDEX `fk_Event_Libraries1_idx` (`Libraries_idLibraries` ASC),
CONSTRAINT `fk_Event_Libraries1`
FOREIGN KEY (`Libraries_idLibraries`)
REFERENCES `dbe3malmus1`.`Libraries` (`idLibraries`)
ON DELETE CASCADE
ON UPDATE CASCADE)
ENGINE = InnoDB;

-----
-- Table `dbe3malmus1`.`Members`
-----

```

(continues)

```
DROP TABLE IF EXISTS `dbe3malmus1`.`Members` ;
```

```
CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Members` (
  `CardLibraryNumber` VARCHAR(15) NOT NULL ,
  `Password` CHAR(4) NOT NULL ,
  `SSN` VARCHAR(11) NOT NULL ,
  `Fname` VARCHAR(25) NOT NULL ,
  `Lname` VARCHAR(25) NOT NULL ,
  `DateOfBirth` DATE NULL ,
  `Address` VARCHAR(50) NOT NULL ,
  `PostalCode` VARCHAR(10) NOT NULL ,
  `City` VARCHAR(25) NOT NULL ,
  `Email` VARCHAR(50) NOT NULL ,
  `PhoneNumber` VARCHAR(20) NOT NULL ,
  `DateIssue` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON
UPDATE CURRENT_TIMESTAMP ,
  `Employee_SSN` VARCHAR(11) NOT NULL ,
  PRIMARY KEY (`CardLibraryNumber`),
  INDEX `fk_Members_Employee1_idx` (`Employee_SSN` ASC),
  CONSTRAINT `fk_Members_Employee1`
  FOREIGN KEY (`Employee_SSN` )
  REFERENCES `dbe3malmus1`.`Employee` (`SSN` )
  ON DELETE NO ACTION
  ON UPDATE CASCADE)
ENGINE = InnoDB;
```

```
-----
-- Table `dbe3malmus1`.`ChargedItems`
-----
```

```
DROP TABLE IF EXISTS `dbe3malmus1`.`ChargedItems` ;
```

```
CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`ChargedItems` (
  `ChargedItems_Id` INT NOT NULL AUTO_INCREMENT ,
```

(continues)

```

`Status` ENUM('Charged','Renewed') NOT NULL ,
`StartDate` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP ON
UPDATE CURRENT_TIMESTAMP ,
`DueDate` DATETIME NOT NULL ,
`RenewedNumber` TINYINT NULL DEFAULT 0 ,
`Fee` FLOAT(4,2) NULL DEFAULT 0 ,
`Members_CardLibraryNumber` VARCHAR(15) NOT NULL ,
`Librarybarcode` VARCHAR(20) NOT NULL ,
PRIMARY KEY (`ChargedItems_Id`, `Members_CardLibraryNumber`),
CONSTRAINT `fk_ChargedItems_Members1`
FOREIGN KEY (`Members_CardLibraryNumber`)
REFERENCES `dbe3malmus1`.`Members` (`CardLibraryNumber`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`Reservation`
-----

```

```

DROP TABLE IF EXISTS `dbe3malmus1`.`Reservation` ;

```

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Reservation` (
`Re_Id` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
`Members_CardLibraryNumber` VARCHAR(15) NOT NULL ,
`ISBN` VARCHAR(20) NOT NULL ,
`DateOfRequest` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
ON UPDATE CURRENT_TIMESTAMP ,
`To_library` TINYINT UNSIGNED NOT NULL ,
`Status` VARCHAR(15) NOT NULL ,
`librarycode` VARCHAR(20) NULL ,
PRIMARY KEY (`Re_Id`, `Members_CardLibraryNumber`),
UNIQUE INDEX `librarycode_UNIQUE` (`librarycode` ASC),
CONSTRAINT `fk_Reservation_Members1`

```

(continues)

```

FOREIGN KEY (`Members_CardLibraryNumber` )
  REFERENCES `dbe3malmus1`.`Members` (`CardLibraryNumber` )
  ON DELETE CASCADE
  ON UPDATE CASCADE)
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`Publisher`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`Publisher` ;

```

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Publisher` (
  `idPublisher` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
  `Name` VARCHAR(45) NOT NULL ,
  PRIMARY KEY (`idPublisher`))
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`Classification`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`Classification` ;

```

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Classification` (
  `idSubjects` INT UNSIGNED NOT NULL ,
  `Name` VARCHAR(45) NOT NULL ,
  PRIMARY KEY (`idSubjects`))
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`Collection`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`Collection` ;

```

(continues)

```
CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Collection` (
  `idCollection` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
  `name` VARCHAR(30) NOT NULL ,
  PRIMARY KEY (`idCollection`))
ENGINE = InnoDB;
```

```
-----
```

```
-- Table `dbe3malmus1`.`languages`
```

```
-----
```

```
DROP TABLE IF EXISTS `dbe3malmus1`.`languages` ;
```

```
CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`languages` (
  `idLanguages` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
  `name` VARCHAR(45) NOT NULL ,
  `short` VARCHAR(3) NULL ,
  PRIMARY KEY (`idLanguages`))
ENGINE = InnoDB;
```

```
-----
```

```
-- Table `dbe3malmus1`.`type`
```

```
-----
```

```
DROP TABLE IF EXISTS `dbe3malmus1`.`type` ;
```

```
CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`type` (
  `idtype` INT UNSIGNED NOT NULL AUTO_INCREMENT ,
  `name` VARCHAR(30) NOT NULL ,
  PRIMARY KEY (`idtype`))
ENGINE = InnoDB;
```

```
-----
```

```
-- Table `dbe3malmus1`.`MediaItems`
```

```
-----
```

```
DROP TABLE IF EXISTS `dbe3malmus1`.`MediaItems` ;
```

(continues)

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`MediaItems` (
  `ISBN` VARCHAR(20) NOT NULL ,
  `Title` VARCHAR(45) NOT NULL ,
  `Edition` VARCHAR(3) NOT NULL ,
  `Year` YEAR NOT NULL ,
  `Publisher_idPublisher` INT UNSIGNED NOT NULL ,
  `Classification_idSubjects` INT UNSIGNED NOT NULL ,
  `Collection_idCollection` INT UNSIGNED NOT NULL ,
  `Languages_idLanguages` INT UNSIGNED NOT NULL ,
  `type_idtype` INT UNSIGNED NOT NULL ,
  PRIMARY KEY (`ISBN`),
  INDEX `fk_MediaItems_Publisher1_idx` (`Publisher_idPublisher` ASC),
  INDEX `fk_MediaItems_Classification1_idx` (`Classification_idSubjects` ASC),
  INDEX `fk_MediaItems_Collection1_idx` (`Collection_idCollection` ASC),
  INDEX `fk_MediaItems_Languages1_idx` (`Languages_idLanguages` ASC),
  INDEX `fk_MediaItems_type1_idx` (`type_idtype` ASC),
  CONSTRAINT `fk_MediaItems_Publisher1`
    FOREIGN KEY (`Publisher_idPublisher` )
    REFERENCES `dbe3malmus1`.`Publisher` (`idPublisher` )
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_MediaItems_Classification1`
    FOREIGN KEY (`Classification_idSubjects` )
    REFERENCES `dbe3malmus1`.`Classification` (`idSubjects` )
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_MediaItems_Collection1`
    FOREIGN KEY (`Collection_idCollection` )
    REFERENCES `dbe3malmus1`.`Collection` (`idCollection` )
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_MediaItems_Languages1`
    FOREIGN KEY (`Languages_idLanguages` )

```

(continues)

```

REFERENCES `dbe3malmus1`.`languages` (`idLanguages` )
  ON DELETE RESTRICT
  ON UPDATE CASCADE,
CONSTRAINT `fk_MediaItems_type1`
  FOREIGN KEY (`type_idtype` )
  REFERENCES `dbe3malmus1`.`type` (`idtype` )
  ON DELETE RESTRICT
  ON UPDATE CASCADE)
ENGINE = InnoDB;

-----
-- Table `dbe3malmus1`.`Librarybarcode`
-----

DROP TABLE IF EXISTS `dbe3malmus1`.`Librarybarcode` ;

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Librarybarcode` (
  `Librarybarcode` VARCHAR(20) NOT NULL ,
  `Stutas` ENUM('Available','Not Available','Not on shelf') NOT NULL ,
  `CellNumber` VARCHAR(10) NOT NULL ,
  `Employee_SSN` VARCHAR(11) NOT NULL ,
  `Libraries_idLibraries` TINYINT UNSIGNED NOT NULL ,
  `Items_ISBN` VARCHAR(20) NOT NULL ,
  PRIMARY KEY (`Librarybarcode`, `Items_ISBN`),
  INDEX `fk_Librarybarcode_Employee1_idx` (`Employee_SSN` ASC),
  INDEX `fk_Librarybarcode_Libraries1_idx` (`Libraries_idLibraries` ASC),
  INDEX `fk_Librarybarcode_MediaItems1_idx` (`Items_ISBN` ASC),
  CONSTRAINT `fk_Librarybarcode_Employee1`
    FOREIGN KEY (`Employee_SSN` )
    REFERENCES `dbe3malmus1`.`Employee` (`SSN` )
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
  CONSTRAINT `fk_Librarybarcode_Libraries1`
    FOREIGN KEY (`Libraries_idLibraries` )

```

(continues)

```

REFERENCES `dbe3malmus1`.`Libraries` (`idLibraries` )
  ON DELETE RESTRICT
  ON UPDATE CASCADE,
CONSTRAINT `fk_Librarybarcode_MediaItems1`
  FOREIGN KEY (`Items_ISBN` )
  REFERENCES `dbe3malmus1`.`MediaItems` (`ISBN` )
  ON DELETE NO ACTION
  ON UPDATE CASCADE)
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`Author`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`Author` ;

```

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Author` (
  `Fname` VARCHAR(25) NOT NULL ,
  `Lname` VARCHAR(25) NOT NULL ,
  PRIMARY KEY (`Fname`, `Lname`))
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`MediaItems_has_Author`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`MediaItems_has_Author` ;

```

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`MediaItems_has_Author` (
  `MediaItems_ISBN` VARCHAR(20) NOT NULL ,
  `Author_Fname` VARCHAR(25) NOT NULL ,
  `Author_Lname` VARCHAR(25) NOT NULL ,
  PRIMARY KEY (`MediaItems_ISBN`, `Author_Fname`, `Author_Lname`),
  INDEX `fk_MediaItems_has_Author_Author1_idx` (`Author_Fname` ASC, `Author_Lname` ASC),

```

(continues)



```

INDEX `fk_MediaItems_has_Author_MediaItems1_idx` (`MediaItems_ISBN` ASC),
  CONSTRAINT `fk_MediaItems_has_Author_MediaItems1`
    FOREIGN KEY (`MediaItems_ISBN` )
      REFERENCES `dbe3malmus1`.`MediaItems` (`ISBN` )
    ON DELETE RESTRICT
    ON UPDATE CASCADE,
  CONSTRAINT `fk_MediaItems_has_Author_Author1`
    FOREIGN KEY (`Author_Fname` , `Author_Lname` )
      REFERENCES `dbe3malmus1`.`Author` (`Fname` , `Lname` )
    ON DELETE NO ACTION
    ON UPDATE CASCADE)
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`LibraryNumber`
-----
DROP TABLE IF EXISTS `dbe3malmus1`.`LibraryNumber` ;

```

```

CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`LibraryNumber` (
  `PNumber` VARCHAR(20) NOT NULL ,
  `Description` VARCHAR(45) NOT NULL ,
  `Libraries_idLibraries` TINYINT UNSIGNED NOT NULL ,
  PRIMARY KEY (`Libraries_idLibraries` , `Description`),
  CONSTRAINT `fk_LibraryNumber_Libraries1`
    FOREIGN KEY (`Libraries_idLibraries` )
      REFERENCES `dbe3malmus1`.`Libraries` (`idLibraries` )
    ON DELETE CASCADE
    ON UPDATE CASCADE)
ENGINE = InnoDB;

```

```

-----
-- Table `dbe3malmus1`.`Image`
-----

```

(continues)

```
DROP TABLE IF EXISTS `dbe3malmus1`.`Image` ;
```

```
CREATE TABLE IF NOT EXISTS `dbe3malmus1`.`Image` (  
  `Image_id` INT UNSIGNED NOT NULL AUTO_INCREMENT ,  
  `Image_name` VARCHAR(45) NOT NULL ,  
  `Libraries_idLibraries` TINYINT UNSIGNED NOT NULL ,  
  PRIMARY KEY (`Image_id`, `Libraries_idLibraries`),  
  INDEX `fk_table1_Libraries1_idx` (`Libraries_idLibraries` ASC),  
  CONSTRAINT `fk_table1_Libraries1`  
    FOREIGN KEY (`Libraries_idLibraries` )  
    REFERENCES `dbe3malmus1`.`Libraries` (`idLibraries` )  
    ON DELETE CASCADE  
    ON UPDATE CASCADE)  
ENGINE = InnoDB;
```

```
USE `dbe3malmus1` ;
```

```
SET SQL_MODE=@OLD_SQL_MODE;  
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;  
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```