

Enhanced Library Management System
Analysis and Design Document

Student: Boros Hanniel

Group: 30432

Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	

Revision History

Date	Version	Description	Author
04/04/2018	1.1	Writing Elaboration – Iteration 1	Boros Hanniel
25/04/2018	1.2	Writing – Iteration 1 all	Boros Hanniel
21/05/2018	2.0	Final update	Boros Hanniel

Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	

Table of Contents

I.	Project Specification	4
II.	Elaboration – Iteration 1.1	4
1.	Domain Model	4
2.	Architectural Design	4
2.1	Conceptual Architecture	4
2.2	Package Design	5
2.3	Component and Deployment Diagrams	6
III.	Elaboration – Iteration 1.2	7
1.	Design Model	7
1.1	Dynamic Behavior	7
1.2	Class Design	8
2.	Data Model	10
3.	Unit Testing	10
IV.	Construction and Transition	10
1.	System Testing	10
2.	Future improvements	10
V.	Bibliography	10

Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	

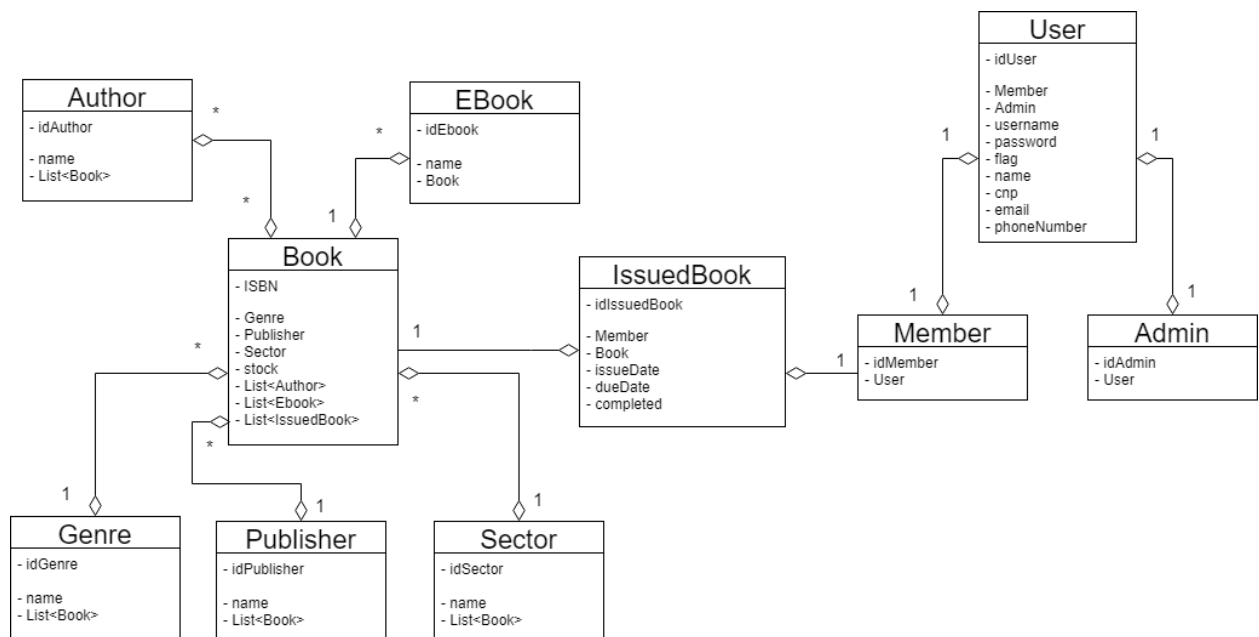
I. Project Specification

This software project is a library management software system with all the basic as well as some innovative features for managing a library). It consists of a large database of various books available in the library. It also lists various books issued to respective readers. The system keeps track of all the books readily available and also the books that have been issued to various readers the time period for which the books have been issued. The system also handles ebooks/pdf database. If the reader needs an ebook/pdf, the ebook can be directly mailed to the client email through the system through a single click. Readers usually tend to forget the date to resubmit their library books so for these books that have been issued to the reader and have reached their expiry date, an email notification is sent to the reader automatically by the system to remind him about the book return date. Thus this innovative library management system provides an enhanced library functionality for this modern day world.

II. Elaboration – Iteration 1.1

1. Domain Model

The Domain Model showing the main classes and the relationships between them.



2. Architectural Design

2.1 Conceptual Architecture

The architectural patterns used for this project is the Layers architecture pattern (otherwise known as the n-tier architecture pattern), together with the MVC and Client-Server patterns. These architectural patterns helps us to structure applications that can be decomposed into groups of subtasks in which each group of subtasks is at a particular level of abstraction.

For this project I consider that the combination of these patterns leads to the most convenient architectural design. In the followings I will present shortly each of them applied to the project.

Layers Architectural Pattern

Controller layer

This layer contains the user oriented functionality responsible for managing user interaction with the system, and generally consists of components that provide a common bridge into the core business logic

Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	

encapsulated in the business layer.

Business layer

This layer implements the core functionality of the system, and encapsulates the relevant business logic. It generally consists of components, some of which may expose service interfaces that other callers can use.

Persistence layer

This layer provides access to data hosted within the boundaries of the system, and data exposed by other networked systems; perhaps accessed through services. The data layer exposes generic interfaces that the components in the business layer can consume.

MVC Architectural Pattern

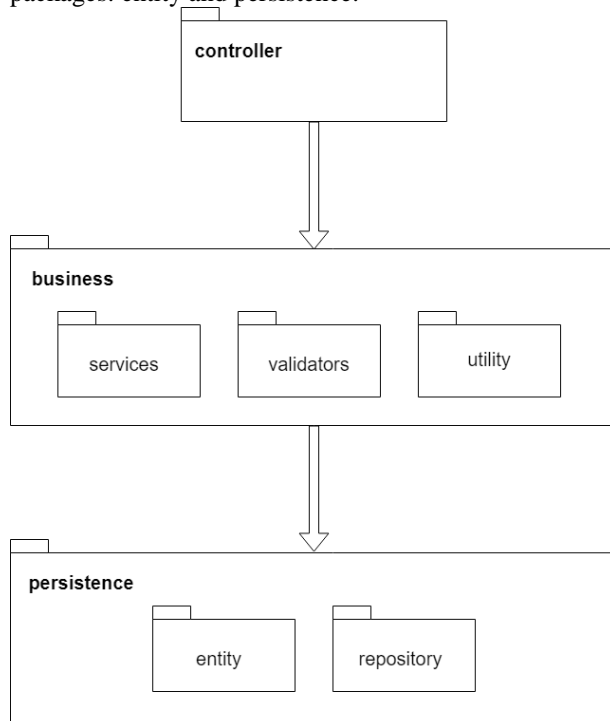
The major (model, controller, view) are decoupled, allowing for efficient code reuse and parallel development. Model-View-Controller patterns have a wide usage and forms. In this project I used Spring MVC which provides a good front-to-back-end data model mapping to UI and back to services, invoking actions.

Client-Server Architectural Pattern

The client/server architectural style describes distributed systems that involve a separate client and server system, and a connecting network. The simplest form of client/server system involves a server application that is accessed directly by multiple clients, referred to as a 2-Tier architectural style. The current assignment was developed in Spring Framework which has a Client-Server design structure handling requests from clients (represented by the html files) in the controller, which then calls the services to perform the required actions.

2.2 Package Design

Below I present the package diagram. As it can be seen on the package diagram as well, the layered design is respected. The controller package is at the top level, as it can use everything it is below it (business and persistence). The business logic is in the middle, it uses only the persistence package. It is divided in three sub-packages, namely: services, validators and utility. The Persistence package contains the two sub-packages: entity and persistence.



Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	

2.3 Component and Deployment Diagrams

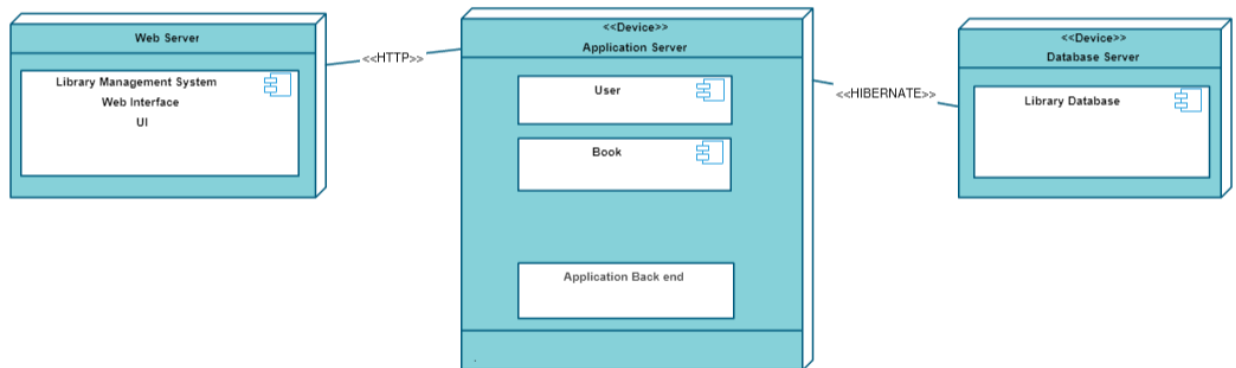
Deployment diagram

The application is composed of three (3) tiers: the application business logic (core part), the user interface on the web server and the database storage server.

The *http* protocol is used for communication between the web-server and the application server.

The *hibernate* integrated with Spring is used for communication between the application server and the database.

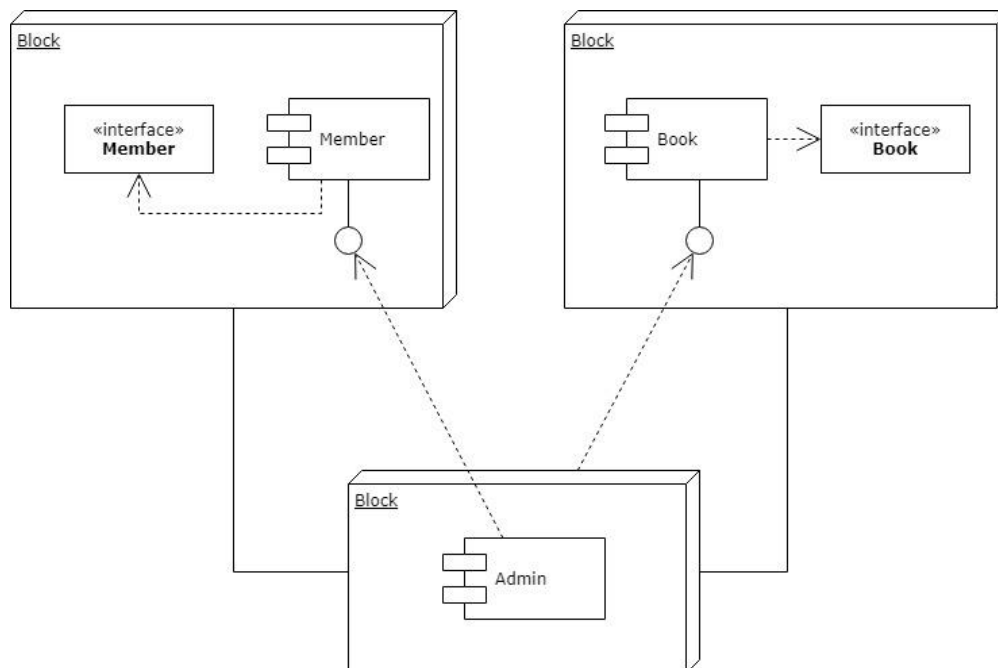
Deployment Diagram For the Library Administration System



Component Diagram

There are three main components of the application: the administrator, the member and the book.

The administrator can operate on library members and books as well, while the member cannot operate on books, just to check current state, and information.



Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	

III. Elaboration – Iteration 1.2

1. Design Model

1.1 Dynamic Behavior

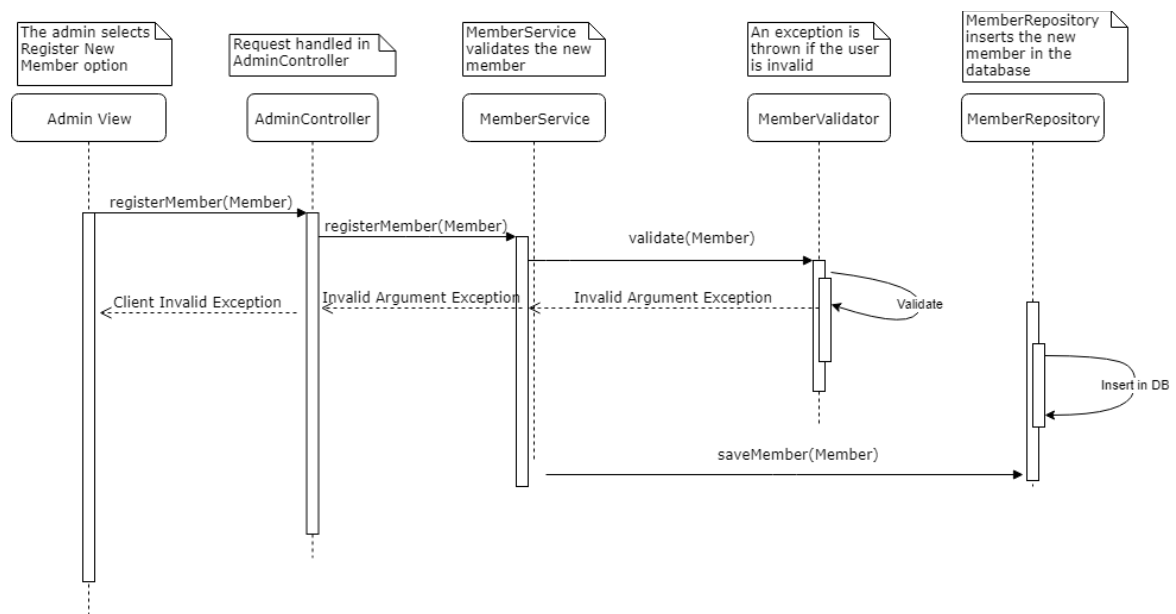
Interaction Diagrams

The interactive behavior is represented in UML by two diagrams known as Sequence diagram and Collaboration (communication) diagram.

Sequence diagram emphasizes on time sequence of messages and collaboration diagram emphasizes on the structural organization of the objects that send and receive messages.

Sequence diagram

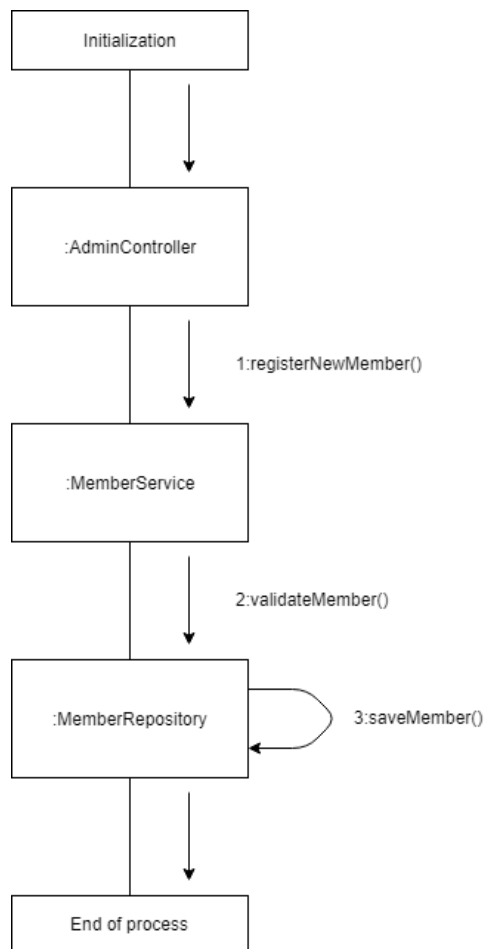
Scenario: Registering a new member to the library member



Collaboration diagram

Scenario: Registering a new member to the library

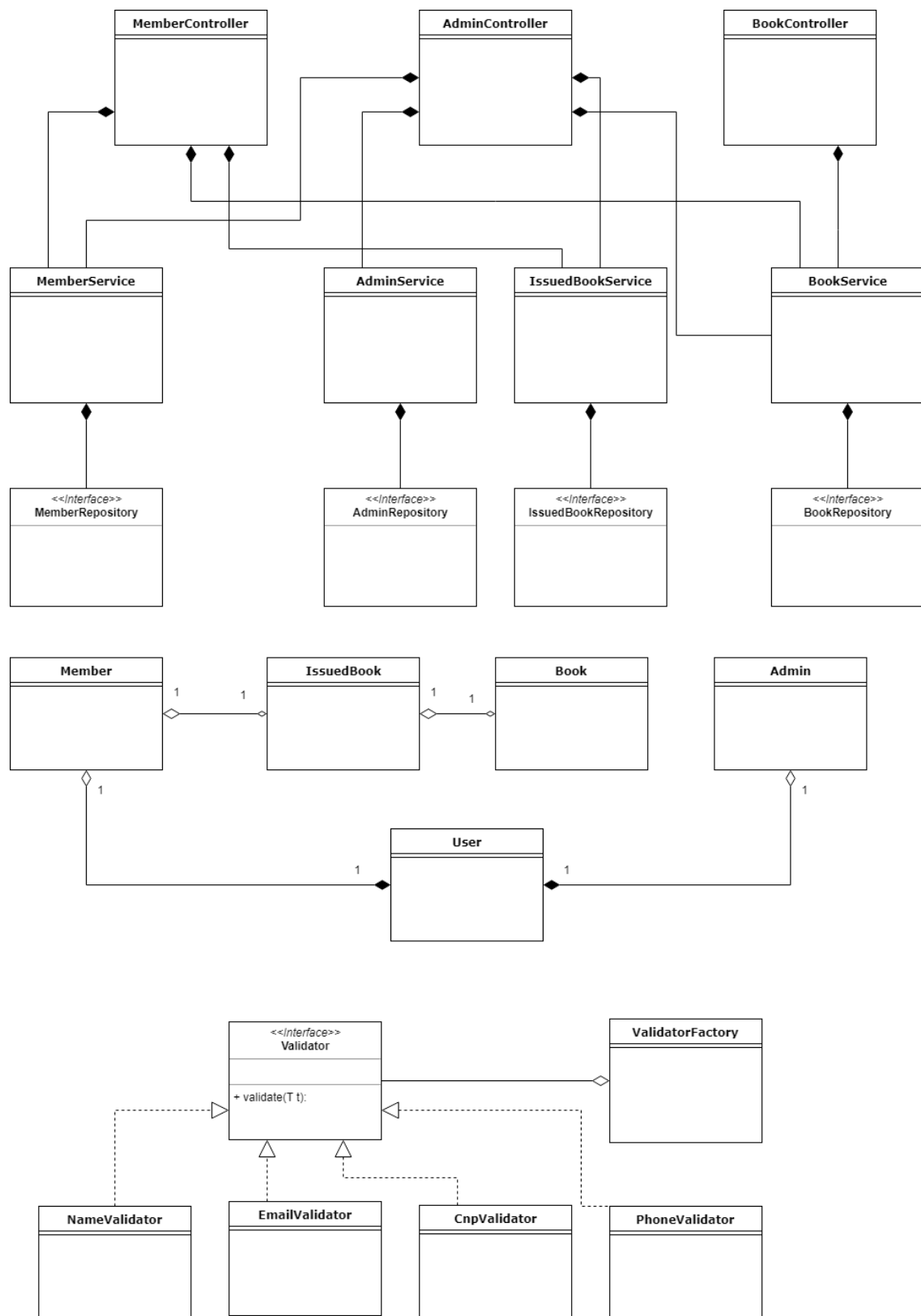
Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	



1.2 Class Design

Below I present the UML class diagram. It is not the final, as changes may arise. Important to mention that here are not all the entities represented, just the ones I considered that have a more important meaning for the understanding of the project. Other entities (mostly those which model the characteristics of a book such as Author, Publisher, Sector, etc) are not represented here, as the diagram would be too complex. However they are modeled in a way which is easy to understand and follows common sense. Being Book characteristics, they are composed by the Book entity (composition).

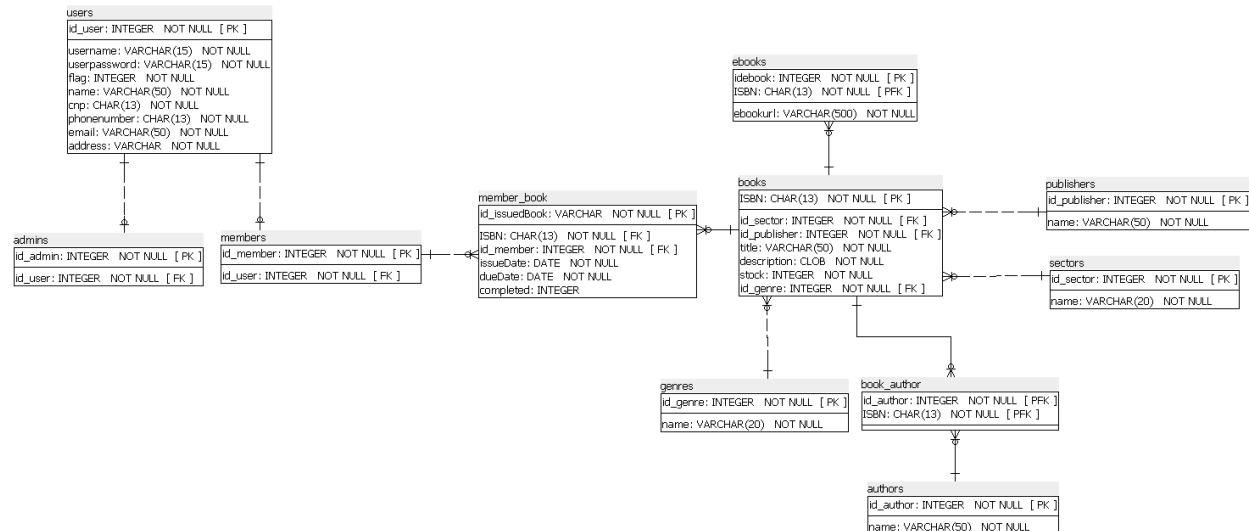
Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	



Library Management System	Version: 1.3
	Date: <25/04/2018>
<document identifier>	

2. Data Model

Below I present the Data Model IR diagram. This was used to represent and model the data of the application. There are eleven (11) tables: nine (9) of them for direct data storage (users, admins, members, books, authors, publishers, sectors, genres, ebooks) and the other two (2) for relationships (they model a many-to-many relationship by a one-to-many ⇔ many-to-one relationship: member_book and book_author).



3. Unit Testing

Unit testing was used in order to test the application.

Implementation is made to be favorable for and sustain testing.

Different parts of the projects were tested as the implementation was carried.

Testing not complete yet!

IV. Construction and Transition

1. System Testing

Endorsment:

Implementation not finished yet!

2. Future improvements

There are several possible future improvements. Considering that nowadays everyone is attached to mobile phones, the notification system could be done via SMS-sending, instead (or with) email notification.

As well, there could be implemented a messaging service, in order to make communication easier between the library members and the library administration. These are the improvements which I consider to be the most relevant ones, but there could be others as well.

V. Bibliography