

Library Management System

Vision Document 0.10

Práctica Profesional

Victor Grycuk

Contents

[Revision 3](#_Toc15157239)

[Introduction 4](#_Toc15157240)

[Purpose 4](#_Toc15157241)

[Scope 4](#_Toc15157242)

[Definitions, Acronyms, and Abbreviations 4](#_Toc15157243)

[Stakeholders and Users description 5](#_Toc15157244)

[Stakeholders summary 5](#_Toc15157245)

[Key Users 5](#_Toc15157246)

[User Environment 5](#_Toc15157247)

[Stakeholders Profile Librarian 5](#_Toc15157248)

[Member 5](#_Toc15157249)

[Global Product Description 6](#_Toc15157250)

[Product Perspective 6](#_Toc15157251)

[Characteristics Summary 6](#_Toc15157252)

[Assumptions and Dependencies 6](#_Toc15157253)

[Detailed Product Description 7](#_Toc15157254)

[Context Diagram 7](#_Toc15157255)

[Deployment Diagram 7](#_Toc15157256)

[Domain Model 8](#_Toc15157257)

[Class Diagram 9](#_Toc15157258)

[Primary Use Cases 10](#_Toc15157259)

[Activity Diagrams 11](#_Toc15157260)

[Registering New Book 11](#_Toc15157261)

[Registering New Author 12](#_Toc15157262)

[Registering New Member 13](#_Toc15157263)

[Checking out Book 14](#_Toc15157264)

[Returning a Book 15](#_Toc15157265)

[Alerting expiring due date 16](#_Toc15157266)

[Sequence Diagram 17](#_Toc15157267)

[Checking out Book 17](#_Toc15157268)

[Restrictions 18](#_Toc15157269)

[Functional Requirements 18](#_Toc15157270)

[Register new book manually 18](#_Toc15157271)

[Register new membership 18](#_Toc15157272)

[Update Book 18](#_Toc15157273)

[Searching for a book in the data base 19](#_Toc15157274)

[Checking out a book 19](#_Toc15157275)

[Other Product Requirements 19](#_Toc15157276)

[Standards 19](#_Toc15157277)

[System Requirements 19](#_Toc15157278)

[Performance Requirements 20](#_Toc15157279)

[Environment Requirements 20](#_Toc15157280)

[Documentation Requirements 21](#_Toc15157281)

[User Manual 21](#_Toc15157282)

[Online Help 21](#_Toc15157283)

[Installation Guides, Configuration, Readme File 21](#_Toc15157284)

[Use Cases Specifications 22](#_Toc15157285)

[Use Cases 22](#_Toc15157286)

[User Roles Management 26](#_Toc15157287)

[Log In / Log Out Management 26](#_Toc15157288)

[Multi Language Management 26](#_Toc15157289)

[Log Management 26](#_Toc15157290)

[Back Up Management 26](#_Toc15157291)

[Digit Verification Management 26](#_Toc15157292)

[Encryption Management 26](#_Toc15157293)

[Additional Specifications 27](#_Toc15157294)

[Navigation Map 28](#_Toc15157295)

[User Interface Prototypes 29](#_Toc15157296)

[Domain Model 30](#_Toc15157297)

[Components Diagram 31](#_Toc15157298)

[Test Cases Specifications 32](#_Toc15157299)

[Unit Tests 33](#_Toc15157300)

# Revision

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Author | Description |
| 15/07/2019 | 0.1 | Victor Grycuk | Initial creation of the document  Addition of Functional Requirements |
| 16/07/2019 | 0.2 | Victor Grycuk | Creation of Use Cases |
| 17/07/2019 | 0.3 | Victor Grycuk | Addition of Use Cases Diagram  Addition of Activity Diagram  Addition of Key Users |
| 18/07/2019 | 0.4 | Victor Grycuk | Addition of Domain Model  Addition of more Activity Diagrams |
| 19/07/2019 | 0.5 | Victor Grycuk | Addition of more Activity Diagrams |
| 20/07/2019 | 0.6 | Victor Grycuk | Addition of Stakeholders and User description section  Addition of Global Product Description  Update of Activity Diagrams  Update of Primary Use Cases |
| 22/07/2019 | 0.7 | Victor Grycuk | Addition of Context Diagram |
| 23/07/2019 | 0.8 | Victor Grycuk | Addition of Deployment Diagram  Addition of Member Stakeholder Profile |
| 24/07/2019 | 0.8.1 | Victor Grycuk | Organize structure of the document |
| 25/07/2019 | 0.9 | Victor Grycuk | Addition of Class Diagram  Addition of Domain Model |
| 27/07/2019 | 0.10 | Victor Grycuk | Addition of Sequence Diagram |

# Introduction

## Purpose

This document will gather, analyze, and define the high-level necessities and characteristics of the management system. This document also focuses in the required functionality by the participants in the project and the end users. The details regarding how the system covers the requirements can be found in the specification of the use cases and other additional documents.

## Scope

This Vision Document applies to the Library Management System (LMS), which will be developed by Victor Grycuk. Victor Grycuk will develop the LMS to work on the Windows based computers.

The purpose of this system is to allow librarians to manage their library using a digital library. It will allow the librarian to keep track of multiple editions of the same book, download a book information from internet when adding a new book, keep track of checked out books, and send reminders to members when the due date for returning a book is about to expire.

It will also allow them to manage the database of members, which will assign a Member ID when creating a new member.

## Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| Acronym | Definition |
| API | Application Programming Interface |
| DB | Data Base |
| DoB | Date of Birth |
| LMS | Library Management System |

# Stakeholders and Users description

## Stakeholders summary

### Key Users

|  |  |  |
| --- | --- | --- |
| Name | Description | Responsibilities |
| Librarians | Main user | * Adding, updating, and removing book entries in the system. * Creates account for new members * Keep track of checked out books |
| Members | Passive user | * Checks out and returns books |

## User Environment

* The Library Management System (*LMS*) will be accessible in appointed computers in the library that meet the following minimal requirements:
  + Windows 7 or higher
  + At last 4gb of RAM
  + Microsoft .NET Framework 4.3 or higher
  + Internet access (optional)
* The *LMS* will be used by authorized librarians. For this, an account for each authorized librarian will be created, along with its user and password.
* In order to access the *LMS* each librarian will have to login using their respective personal credentials. After logging in, the main menu will be displayed.
* Once logged, the users will be able to change their preferred language.

## Stakeholders ProfileLibrarian

|  |  |
| --- | --- |
| Representant | None |
| Description | Person in charge of attending the general public and managing the library’s book stock. |
| Type | Key User |
| Responsibility | Person in charge of managing the catalogue of books, lending books to members, receiving returned books, and keeping tracked of checked-out books. |
| Exit Criteria | None |
| Participation Level | High |

## Member

|  |  |
| --- | --- |
| Representant | None |
| Description | End customer of the library |
| Type | Passive User |
| Responsibility | Almost none. No direct contact with the system, except for receiving emails from it, and checking out/returning books. |
| Exit Criteria | None |
| Participation Level | Low |

# Global Product Description

## Product Perspective

The *LMS* was developed to help librarians manage their library catalogue in a digital platform.

*LMS* can help them to quickly add new books and update existing ones in their digital catalogue. Additionally, they have the option to download the book information from an online source instead of typing them manually.

*LMS* will also have a digital database of Members of the library, that will help to keep track of lent books, alerting both the librarian and members if the return date of a book is about to expire, apply a fine for returning a book after the specified return date of the book, and notifying a member of new available books according to their preferences.

## Characteristics Summary

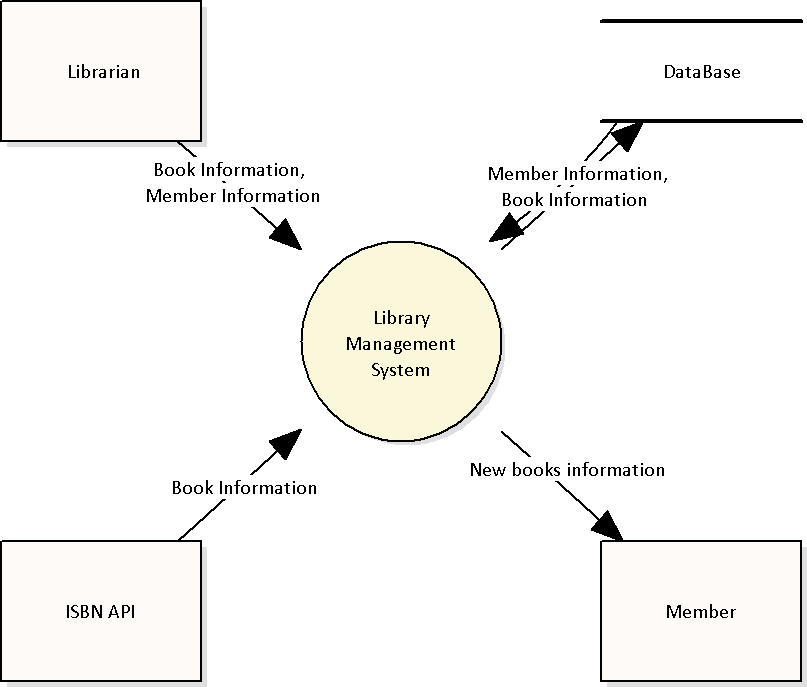
|  |  |
| --- | --- |
| Characteristic | Benefit for the Client |
| General | * Multi-language * Notifications of updates |
| Catalogue Management | * Add, remove, and update entries * Download book information from internet * Search of books in catalogue by their attributes (ISBN, Author, Release Date, etc.) |
| Member Management | * Create, update, and deactivate members account * Search for member by Name, Member ID, or any of their attribute * Notification of new books |
| Book check-out management | * Keep track of let books * Notifications of approaching returning date * Listing and search of current checked out books and the associated member * Application of fines for books returned after their specified date |
| Database | * Back up * Encryption |

## Assumptions and Dependencies

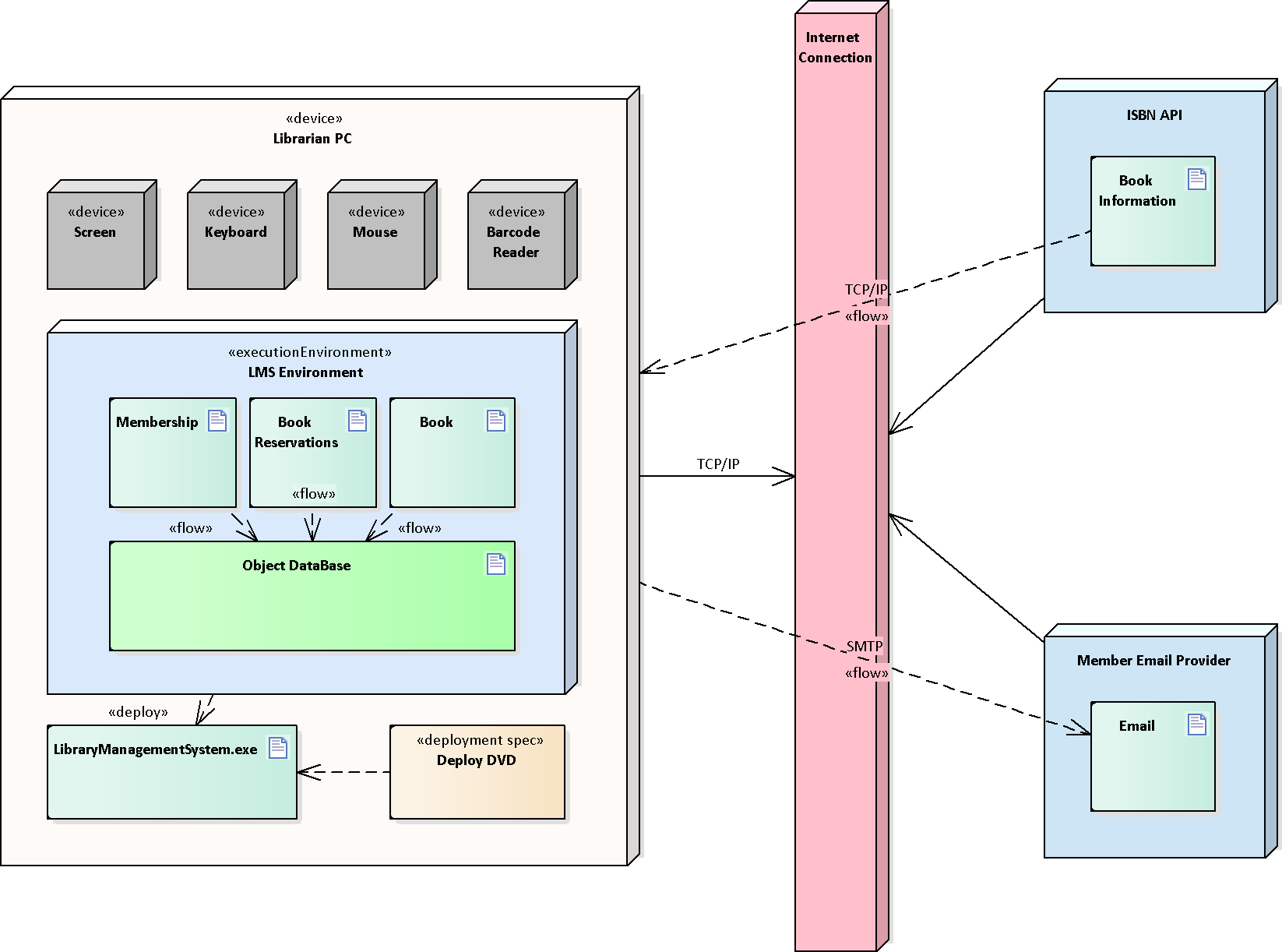
* It is assumed that the user has basic understanding of the Windows environment.
* The *LMS* default language is English. It is assumed the user knows enough English to be able to change the default language to the language of their preference.
* When trying to download a book information from an API, it is assumed that the computer will have access to the Internet.
* After the initial configuration, the user will oversee the creation of new accounts for librarians and users

# Detailed Product Description

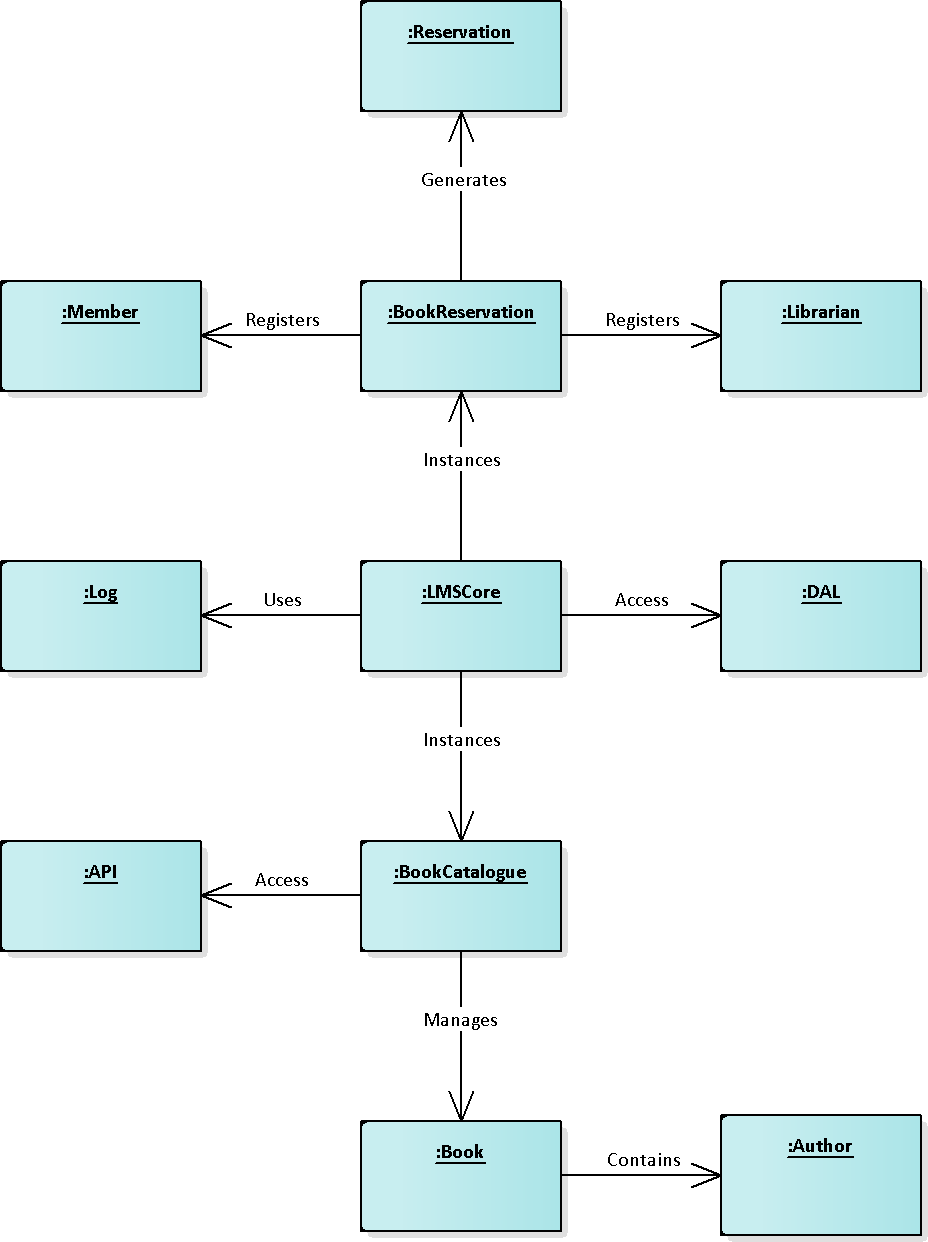
## Context Diagram



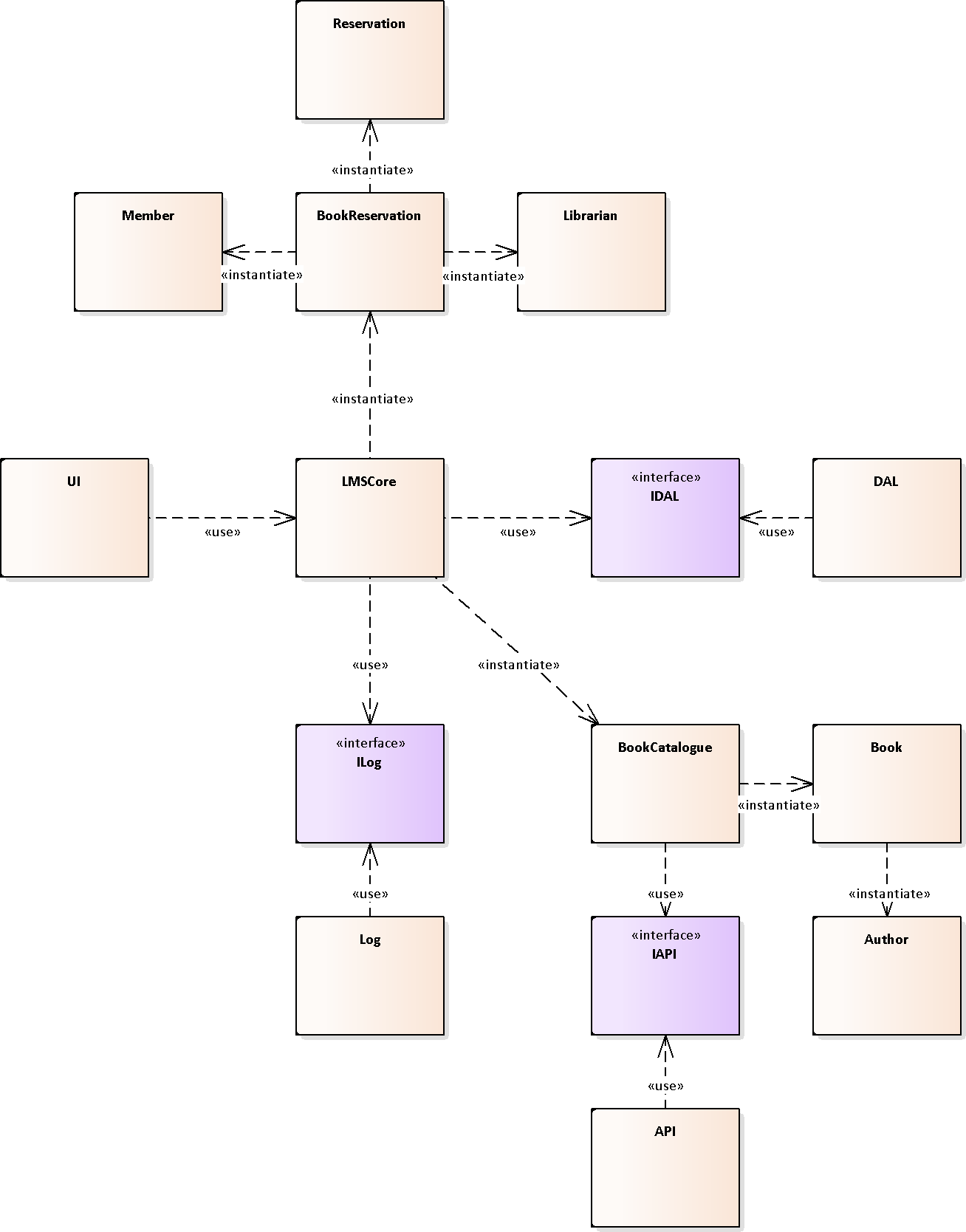
## Deployment Diagram



## Domain Model

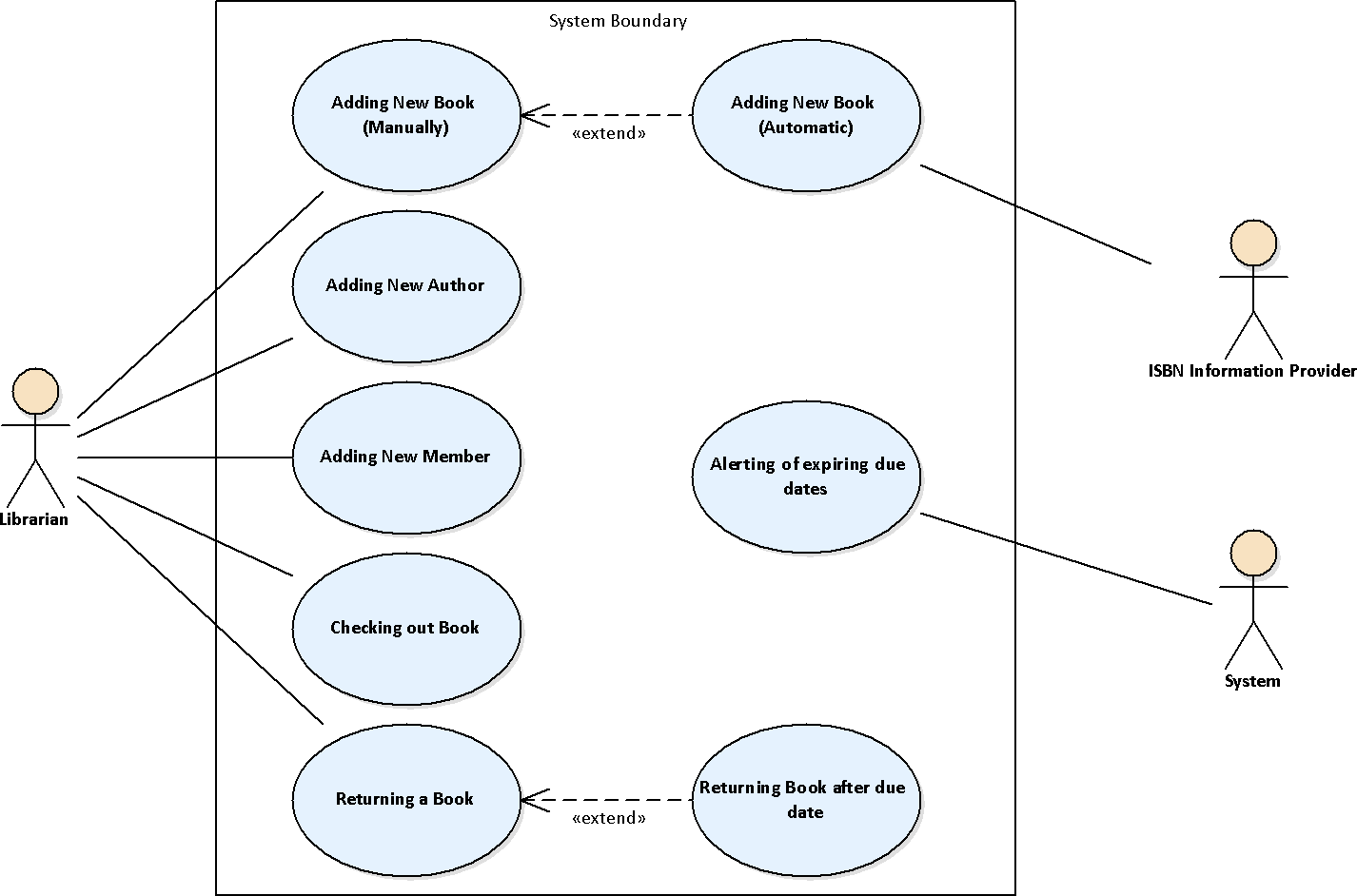


## Class Diagram



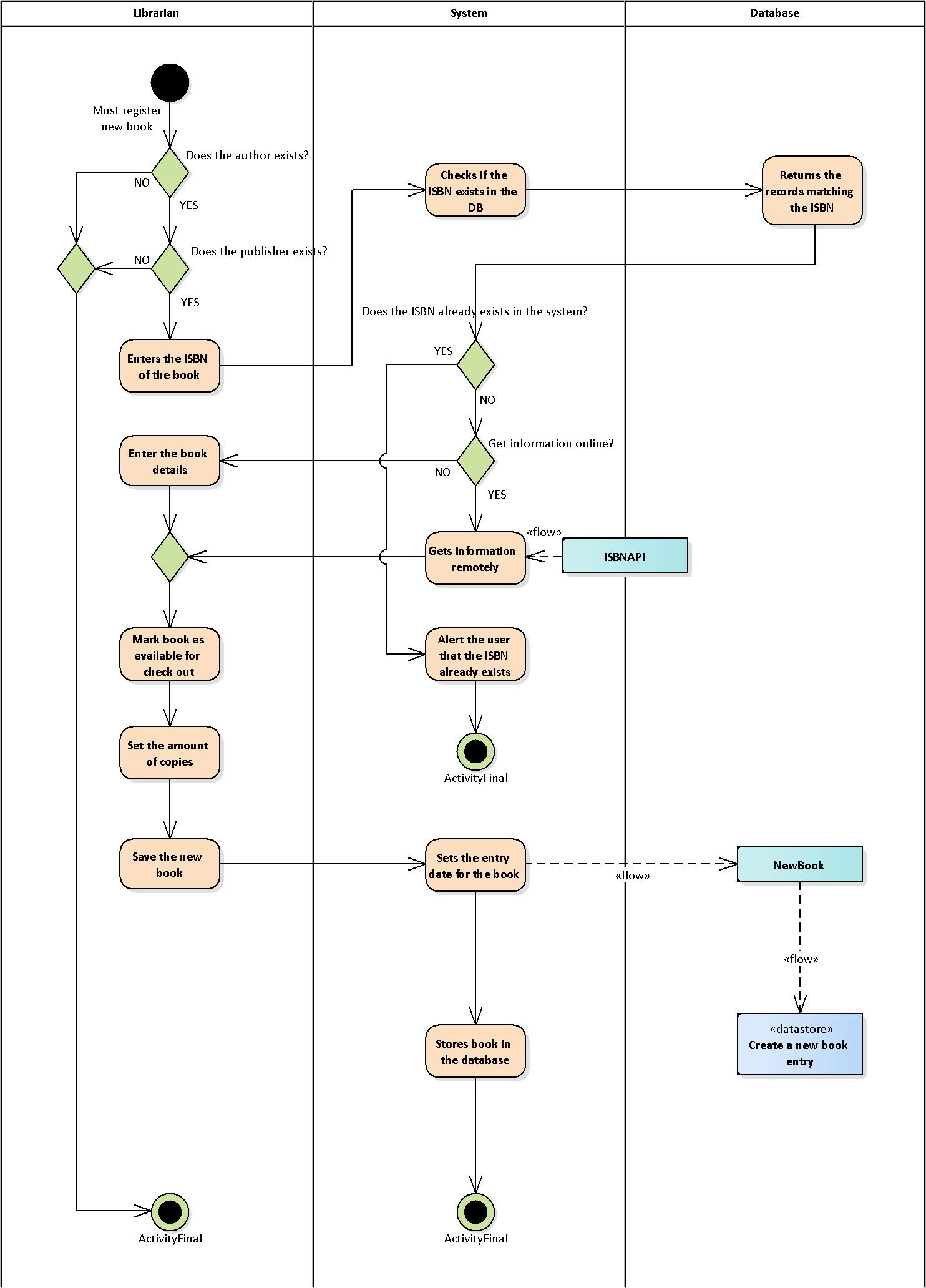
***Note:*** *This diagram offers a high-level view of the system, it might change during development.*

## Primary Use Cases

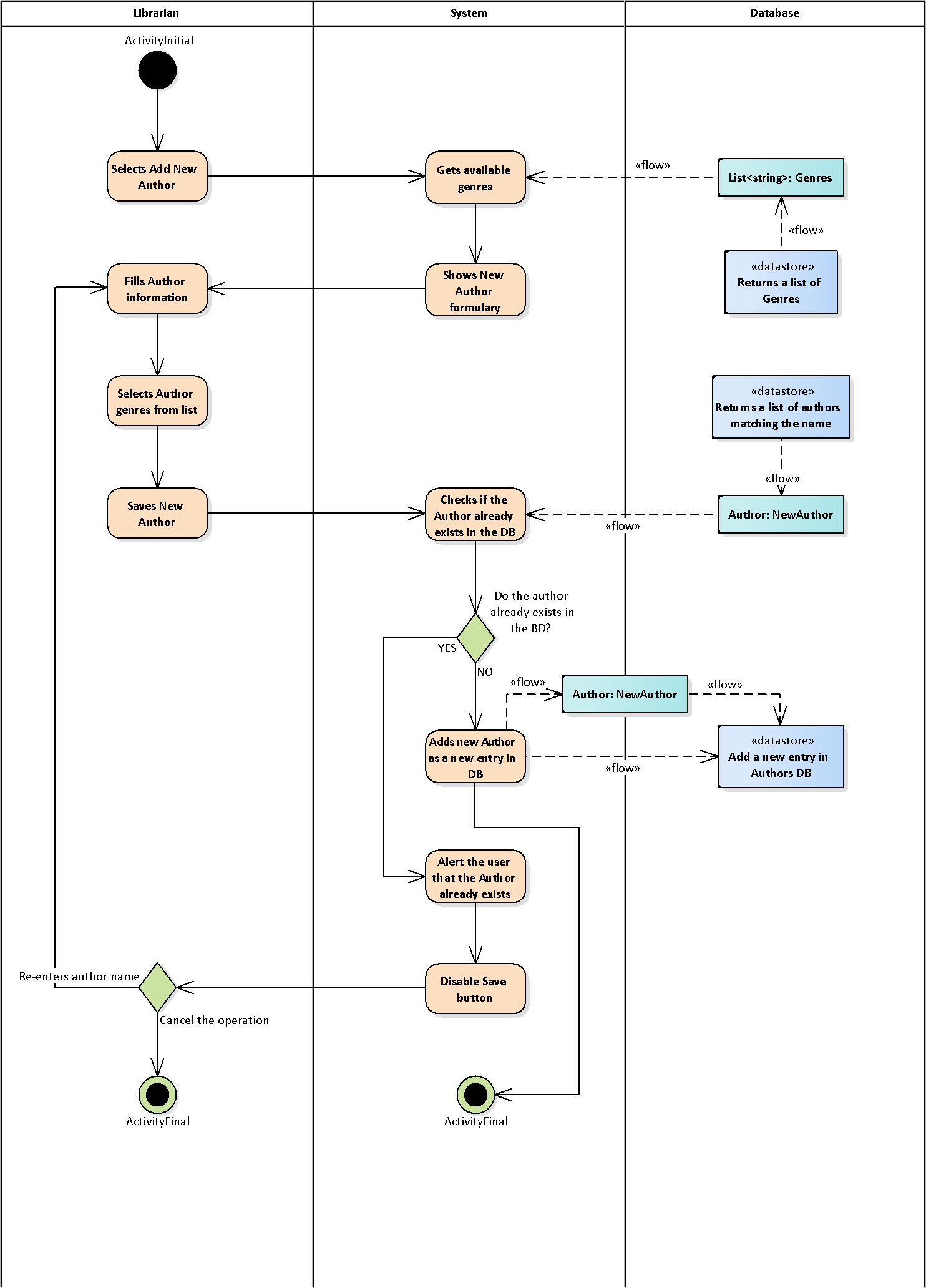


## Activity Diagrams

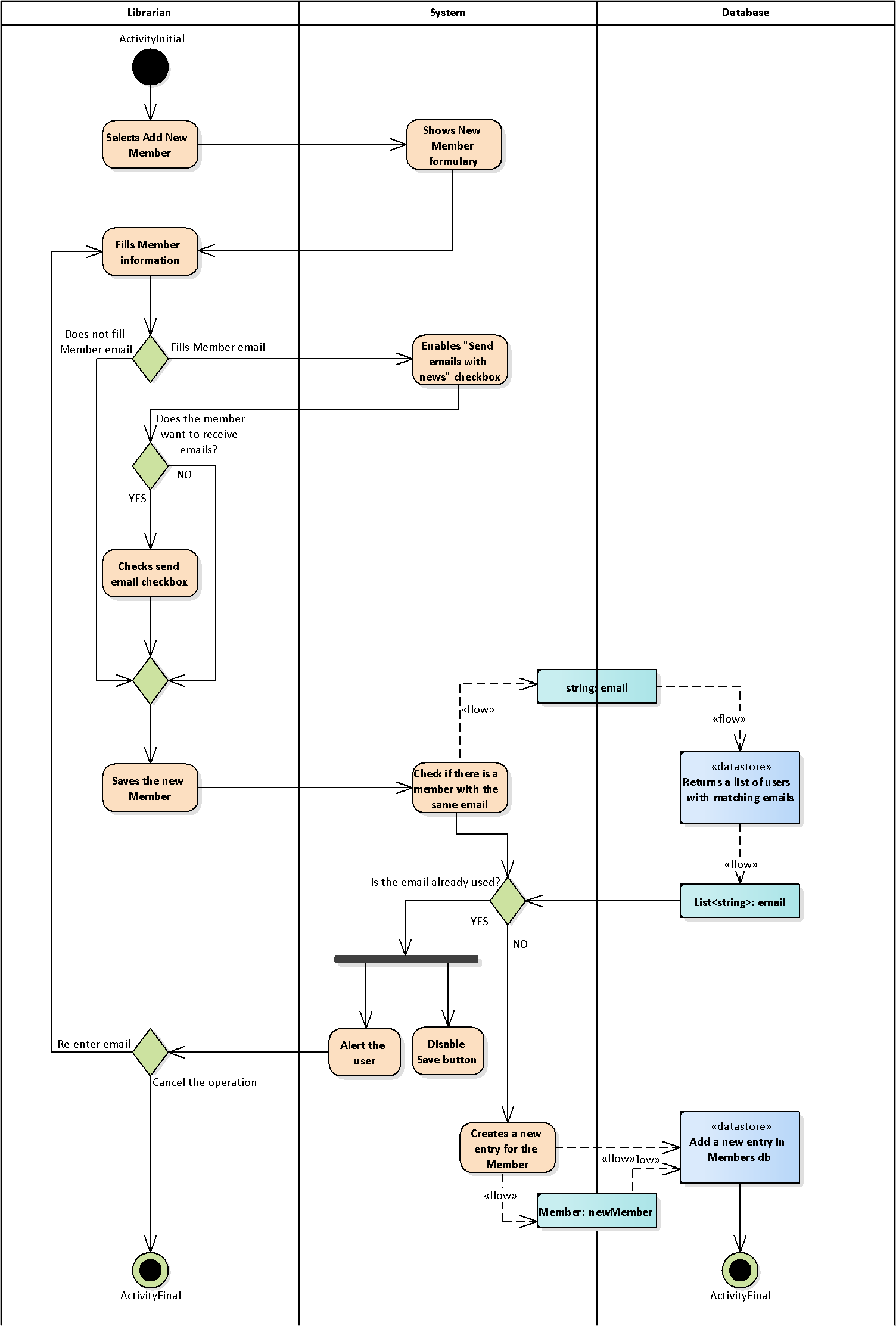
### Registering New Book



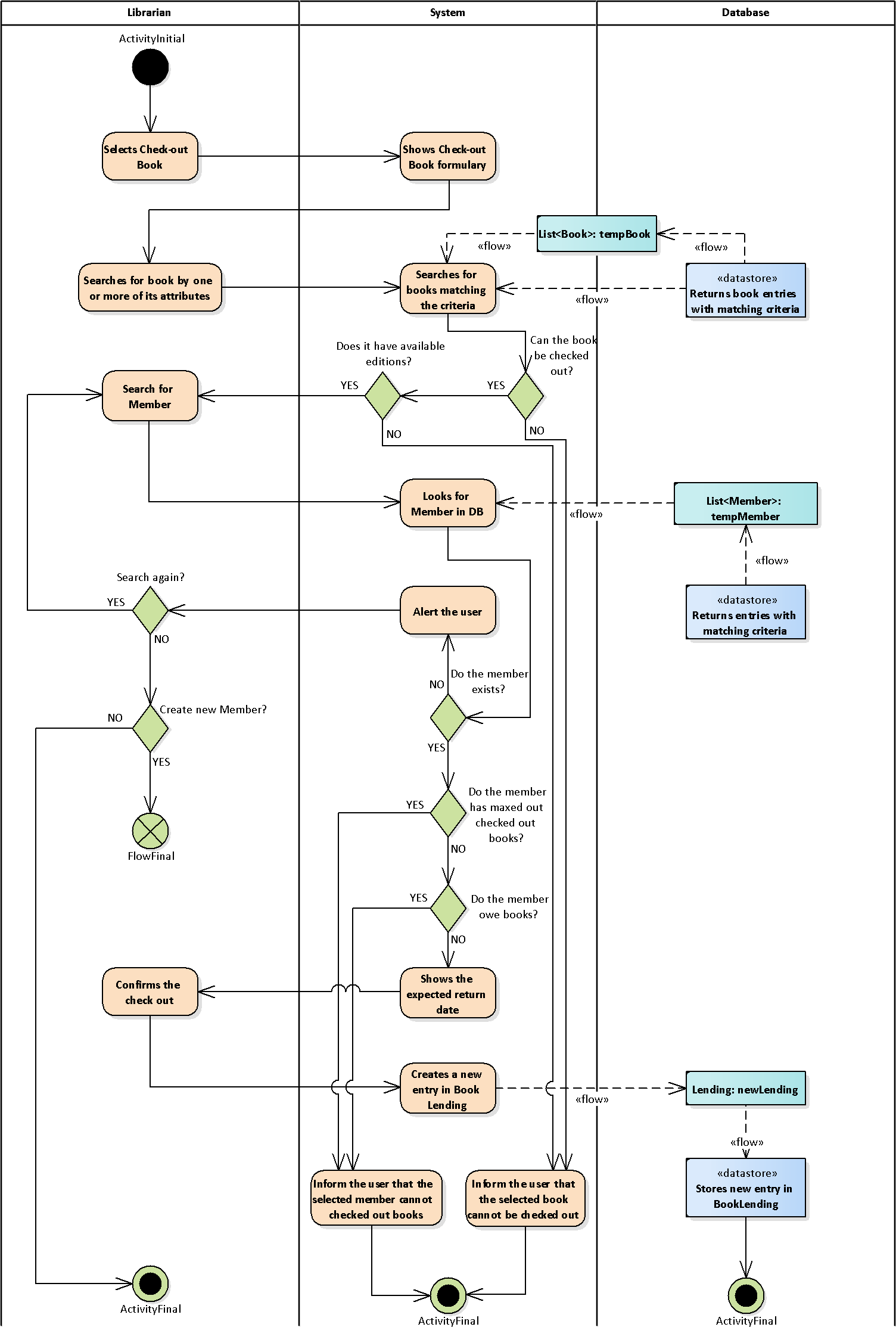
### Registering New Author



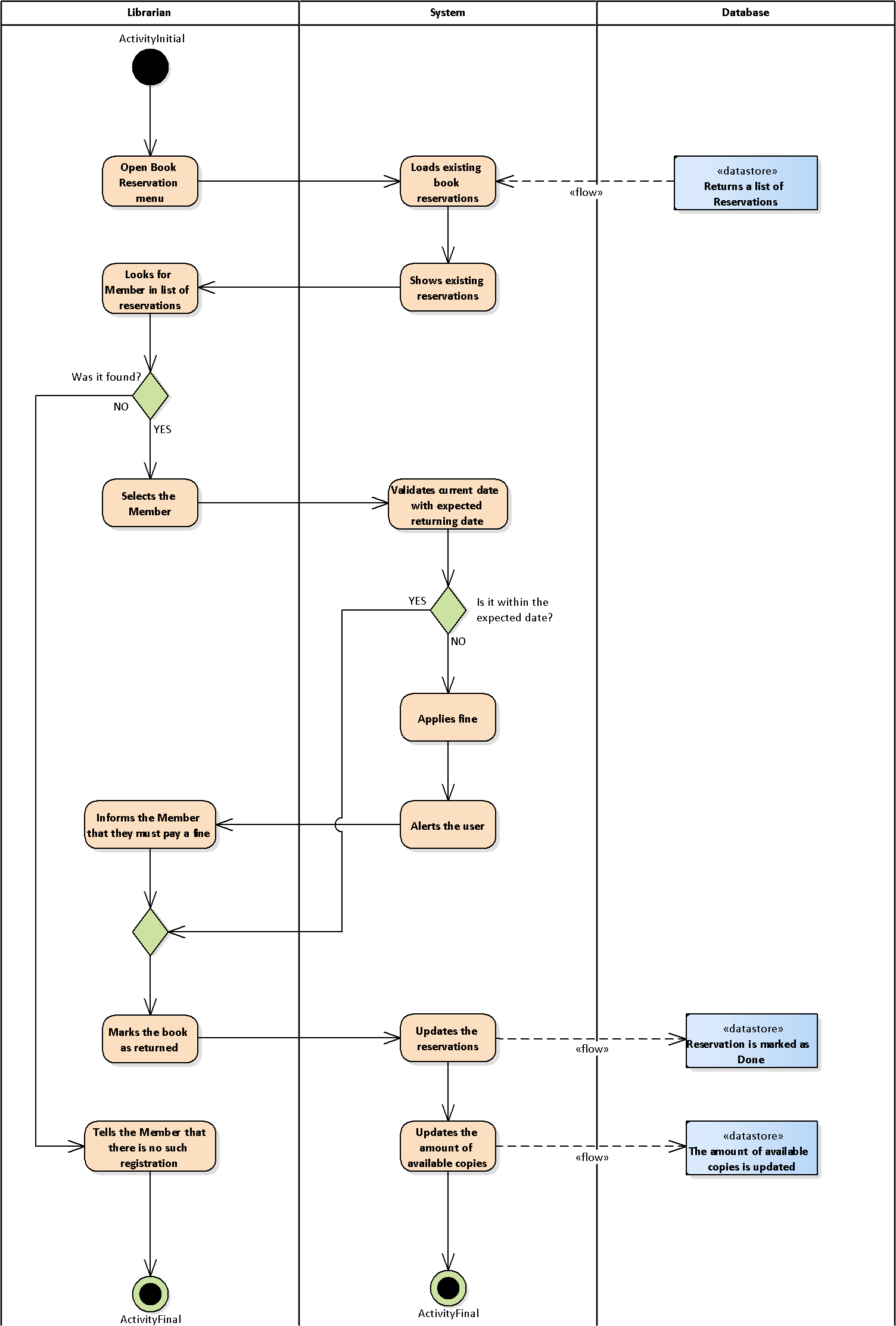
### Registering New Member



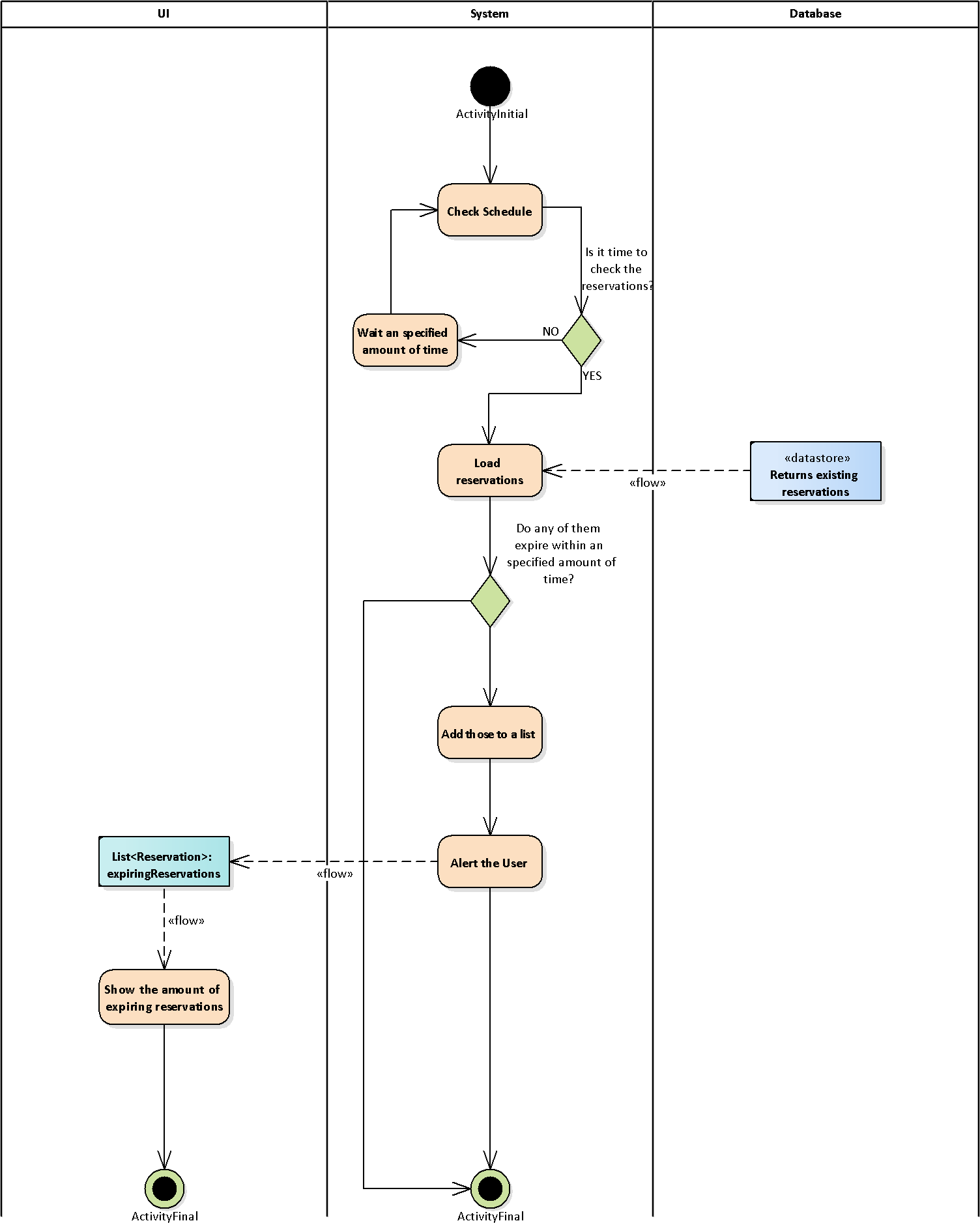
### Checking out Book



### Returning a Book

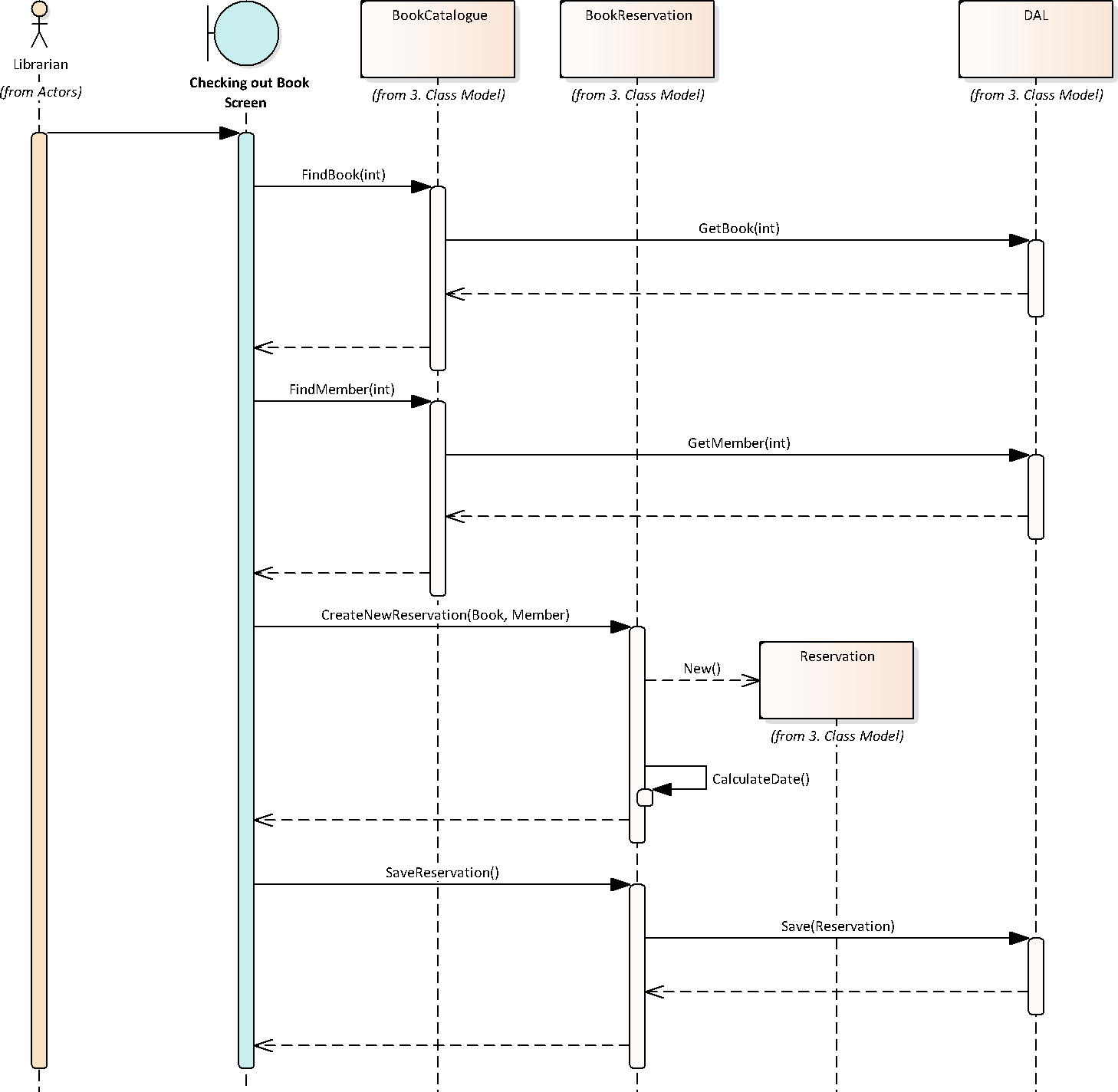


### Alerting expiring due date

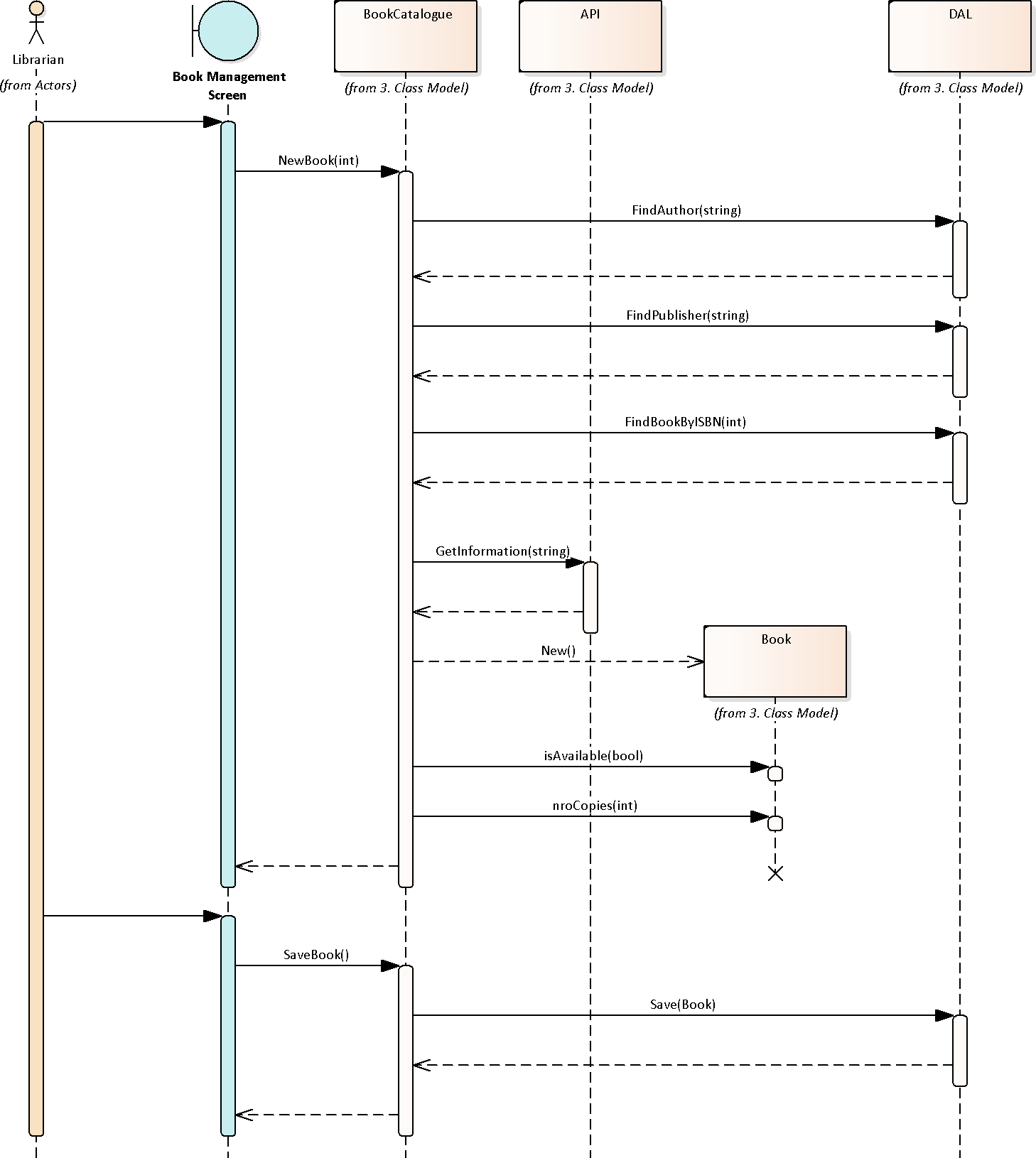


## Sequence Diagram

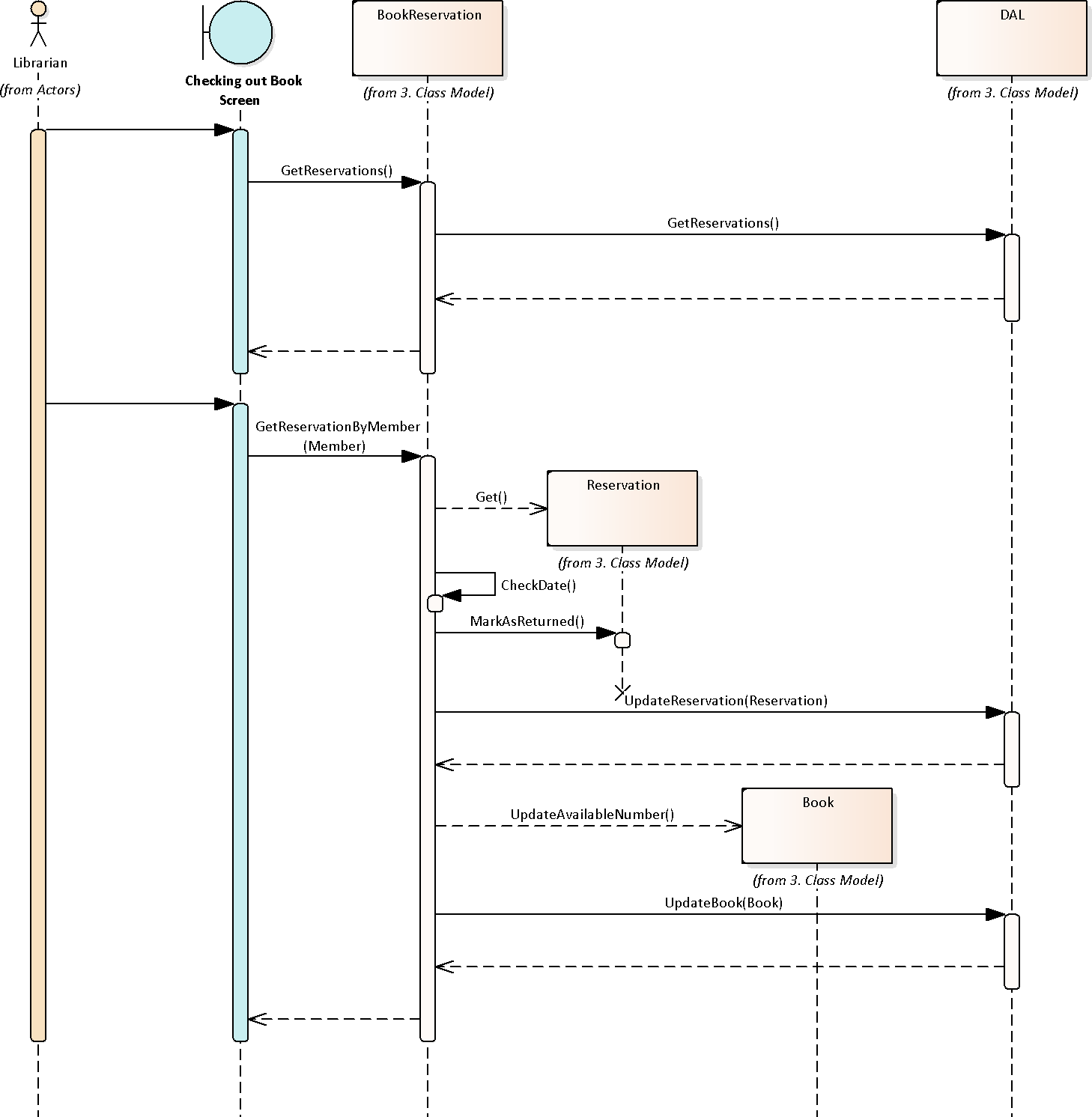
### Checking out Book



### Adding New Book



## Returning Book



# Restrictions

## Functional Requirements

### Register new book manually

|  |  |
| --- | --- |
| Introduction | The System must allow to register new books manually or atumatically. |
| Required Information | * Details of the book * Date of registration |
| Process | * The author of the book must already exist in the system * The publisher of the book must already exist in the system * Fill the ISBN of the book * Verify that the ISBN does not already exists in the system * Choose to either enter information manually or download it from internet   + Fill the name of the book   + Fill the author or authors of the book   + Fill the date of acquisition   + Fill the optional fields * Mark if the book will be available for check out * Fill the amount of copies |
| Goal | The book must exist in the system database with its required information |

### Register new membership

|  |  |
| --- | --- |
| Introduction | The System must allow to register new memberships |
| Required Information | * Member Information |
| Process | * The member must not exist in the DB * The Librarian fills the members information * The System assign a *Member ID* to the Member * The System creates a new Member in the DB |
| Goal | The new member must exist in the system database with its required information |

### Update Book

|  |  |
| --- | --- |
| Introduction | The System must allow to update a book details with online information. |
| Required Information | * ISBN of the book |
| Process | * Search for the required book to update * Update the book details by getting online information * Accept the changes |
| Goal | The book details must be updated with online information |

### Searching for a book in the data base

|  |  |
| --- | --- |
| Introduction | The System must allow to search and list the books registered by using different types of criteria. |
| Required Information | * Any information regarding the book * Any information regarding an author * Any information regarding a publisher |
| Process | * Fill any information regarding a book, author, or publisher * A list of books with matching details should be displayed |
| Goal | A list of books with the desired criteria should be shown |

### Checking out a book

|  |  |
| --- | --- |
| Introduction | The system must allow to check out books to customers and keep track of the time. |
| Required Information | * Customer information * ISBN of the book |
| Process | * Verify that the book exists in the system * Verify that the book is available for checking out * Search for the book in the system * Mark the copy of the book as checked out * Update the remaining available copies of the book * Keep track of remaining days until getting the book back |
| Goal | * The available copes for checking out for the book must be updated * The book and customer information must appear in the list of books to be returned |

## Other Product Requirements

None specified.

## Standards

None specified.

## System Requirements

|  |  |
| --- | --- |
| Operating System: (32- and 64-bit) | * Windows Server 2008 R2 with Service Pack 1 * Windows 7 SP 1 * Windows 8 * Windows 8.1 Microsoft .NET framework 3.5 with Service Pack 1 installed * Windows 10 |
| Memory | * 1 GB RAM (1.5 GB Recommended) for Windows Windows 7 Professional, Windows 8, Windows 10 Windows Server 2008 R2, and 2012 |
| Processor | * 1.7 GHz processor or better |
| Hard Disk | * Minimum free space is approximately NOT KNOWN MB, where 4.5 GB of that is for the Microsoft .NET 4.6.1 framework software, if not previously installed. |
| Connectivity | * Port 80 must be open for ISBN API connection and software updates |
| Screen Resolution | * The minimum supported resolution is 1024x768 |

## Performance Requirements

None specified.

## Environment Requirements

None specified.

# Documentation Requirements

## User Manual

## Online Help

## Installation Guides, Configuration, Readme File

# Use Cases Specifications

## Use Cases

|  |  |
| --- | --- |
| ID | UC-01-01 |
| Name | **Adding New Book (Manually)** |
| Description | Adding a new book edition entry to the DB |
| Actors | Librarian |
| Triggers | A new book edition is acquired by the library |
| Pre-Condition | The book edition must not exist in the DB already |
| Post-Condition | The new book edition must be accessible from the System search |
| Main Course | 1. The librarian selects the option *Add new Book* 2. A formulary with the book details fields will be shown 3. The librarian must enter the ISBN of the book    1. For downloading book information, see **UC-01-02** 4. The System checks if the ISBN exists in the DB    1. If the ISBN already exists, see **EX-01-01** 5. The librarian must select the authors of the book    1. If the author/s does not exist, see **UC-02-01** 6. The librarian must enter the date of acquisition of the book 7. The librarian must complete the rest of the required fields 8. The librarian enters the amount of available editions 9. The librarian selects *Save* 10. The System will create a new entry in the DB with the book information |
| Exceptions | * **EX-01-01**: The System will alert the librarian that a book with the same ISBN already exists in the DB, and won’t allow to continue. |

|  |  |
| --- | --- |
| ID | UC-01-02 |
| Name | **Adding New Book (Automatic)** |
| Description | Add a new book edition entry to the DB, and getting the book information automatically from Internet |
| Actors | Librarian |
| Triggers | * A new book edition is acquired by the library * UC-01-01 |
| Pre-Condition | The book edition must not exist in the DB already  The System must have access to Internet |
| Post-Condition | The new book edition must be accessible from the System search |
| Main Course | 1. After filling the ISBN of the book, the librarian selects *Get information from online data base* 2. The System checks if the ISBN exists in the DB    1. If the ISBN already exists, see **EX-01-01** 3. The system will search for the ISBN online and bring back the information of the book    1. If no matching ISBN is found, see **EX-01-02** 4. The System will autocomplete the book information fields 5. The librarian enters the amount of available editions 6. The librarian selects *Save* 7. The System will create a new entry in the DB with the book information |
| Exceptions | * **EX-01-01**: The System will alert the librarian that a book with the same ISBN already exists in the DB, and won’t allow to continue. * **EX-01-02**: The user will be alerted and will have to complete the fields manually. |

|  |  |
| --- | --- |
| ID | UC-02-01 |
| Name | **Adding New Author** |
| Description | Adding a new author entry to the DB |
| Actors | Librarian |
| Triggers | A new book is acquired whose author does not yet exist in the DB |
| Pre-Condition | The author must not exist in the DB |
| Post-Condition | The new author must be accessible from the System search |
| Main Course | 1. The librarian selects the option *Add new Author* 2. A formulary with the author details fields will be shown 3. The librarian must enter the Author names 4. The librarian must select the associated genres 5. The librarian selects *Save* 6. The System will create a new entry in the DB with the author information |

|  |  |
| --- | --- |
| ID | UC-03-01 |
| Name | **Adding New Member** |
| Description | Adding a new library member to the DB |
| Actors | Librarian |
| Triggers | A person applying for a library membership |
| Pre-Condition | The person must not be an existing member |
| Post-Condition | The new member must exist in the DB |
| Main Course | 1. The librarian selects the option *Create new Membership* 2. A formulary with the member information fields will be shown 3. The librarian must complete the personal information of the new Member 4. The librarian must check if the new Member wishes to receive emails with offers and news about books 5. The librarian selects *Save* 6. The System will inform the new *Member ID* |

|  |  |
| --- | --- |
| ID | UC-04-01 |
| Name | **Checking out Book** |
| Description | A Member wish to check out a book |
| Actors | Librarian |
| Triggers | A Member checking out a book |
| Pre-Condition | * The book must exist in the library catalogue * The person must be an existing Member |
| Post-Condition | * The amount of available numbers if the checked-out book must be decreased by 1 * A new registry must be created in the *Book Reservation* detailing the member, the book, and when the checked-out book must be returned |
| Main Course | 1. The librarian selects the option *Check-out Book* 2. A formulary will be shown for the librarian to choose the book to check out and the member checking out the book 3. The librarian searches for the book to check out 4. The System checks if the book can be checked out    1. If the book cannot be check out, see **EX-04-01** 5. The System checks if there are available editions    1. If there no amount available to check out, see **EX-04-02** 6. The librarian must select the member checking out the book    1. If the member exceeds the amount of checked out books, see **EX-04-03**    2. If the member has checked out books that exceed certain threshold, see **EX-04-04**    3. If the member does not exist, see **UC-03-01** 7. The System will inform the return date 8. The librarian confirms the information with the Member 9. The librarian selects *Check out* 10. The System creates a new entry in *Book Reservation* |
| Exceptions | * **EX-04-01**: Some books are not allowed to be checked out of the library, these books will be marked as un check-out-ables * **EX-04-02**: If there is no amount available left of book to check out, the System will alert the librarian * **EX-04-03**: If the Member exceeded the amount of allowed checked out books, the System will alert the librarian * **EX-04-04**: If the Member has unreturned books that exceed certain amount of days beyond the return date, the System will alert the librarian |

|  |  |
| --- | --- |
| ID | UC-05-01 |
| Name | **Returning a Book** |
| Description | A Member returning a book |
| Actors | Librarian |
| Triggers | A Member returning a book |
| Pre-Condition | * The book must exist in the library catalogue * The person must be an existing Member |
| Post-Condition | * The amount of available numbers if the checked-out book must be increased by 1 * The registry of the checked-out book must be marked as *Done* |
| Main Course | 1. The librarian searches for the Member in the Book Reservation    1. If the Member is not found, see **EX-05-01** 2. The librarian selects the Member returning the book, and mark the returning book as *Returned*    1. If the returning date is beyond the due date, see **UC-05-02** 3. The librarian selects *Done* |
| Exceptions | * **EX-05-01**: The System will alert the Librarian that the Member returning a book is not found |

|  |  |
| --- | --- |
| ID | UC-05-02 |
| Name | **Returning a Book after due date** |
| Description | A Member returns a book after its due date |
| Actors | Librarian |
| Triggers | UC-05-01 |
| Pre-Condition | * The book must exist in the library catalogue * The person must be an existing Member |
| Post-Condition | * The amount of available numbers if the checked-out book must be increased by 1 * The registry of the checked-out book must be marked as *Done* |
| Main Course | 1. The System informs the librarian that the book being returned is past its due date 2. The System shows the fine that the Member must pay 3. The librarian informs the Member the fine they must pay 4. The librarian selects *Done* |

|  |  |
| --- | --- |
| ID | UC-06-01 |
| Name | **Alerting of expiring due dates** |
| Description | The System must alert daily the librarian of books with expiring due dates |
| Actors | System |
| Triggers | Daily automatic trigger |
| Pre-Condition | At least one book in *Book Reservation* |
| Post-Condition | The System must show an alert showing the number of books to be returned that day |
| Main Course | 1. The System analyze the entries in *Book Reservation* 2. If an entry contains a book that should be returned the same day, the entry is added to *Expiring Today* 3. In the System main menu, a counter with the number of books to be returned will be shown |

## User Roles Management

## Log In / Log Out Management

Permite verificar la identidad del usuario a través del ingreso de su nombre de usuario y su clave, asignándole el perfil que tenga asignado en el sistema. Se debe describir como será la política de ‘log-in’ / ‘log-out‘. También deberán diferenciarse los procesos que se correrán en el arranque del sistema, el log in, el log out y el apagado de sistema.

## Multi Language Management

Debe permitir el cambio de idioma de todas las leyendas y títulos que se lean en las interfaces de usuario. El cambio debe ser dinámico. Este concepto implica que desde el sistema se puedan incorporar nuevos idiomas y las leyendas que estén afectadas al mismo.

## Log Management

En ella deben quedar registradas todas las operaciones que realicen los usuarios durante la utilización del sistema. Esto permitirá hacer un trazado de las actividades desarrolladas por el usuario dentro de la aplicación. Los datos mínimos que la bitácora debe incluir son fecha, hora, usuario, actividad, información asociada con la actividad. El subsistema de bitácora deberá prever la posibilidad de realizar búsquedas por los datos almacenados de manera combinada.

## Back Up Management

Esta gestión se utiliza para administrar las copias de seguridad. Esto implica gestionar el catálogo de backups así como los archivos físicos que contienen la información resguardada.

## Digit Verification Management

La función de los dígitos verificadores es la de permitir comprobar la integridad de los datos almacenados en la base de datos. Se desea poder detectar dos cosas. La primera es si se han agregado o quitado datos de la base de datos por fuera del sistema y la segunda es si se han intercambiado datos de posición. Para esto último es importante, al momento de determinar el algoritmo de cálculo a emplear, que en el cálculo no sólo participe el contenido del atributo sino también la posición del carácter y la posición del atributo dentro de la entidad.

Al iniciar la aplicación, y antes de dar acceso a la ventana de log-in, se debe realizar el proceso de verificación de integridad de la base de datos. En caso de error, se deberá informar al administrador para que tome las medidas adecuadas. Los dígitos verificadores horizontales se guardan en un atributo de las entidades bajo análisis mientras que los verticales se pueden guardar en una entidad adicional creada para ese fin, la cual deberá formar parte del DER.

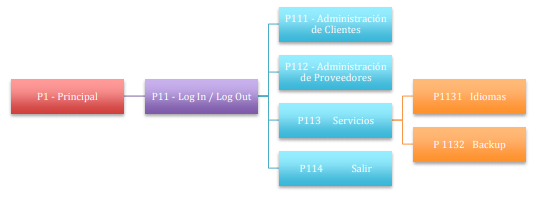
## Encryption Management

La gestión de encriptado es la responsable implementar los algoritmos de encriptación para proteger los datos sensibles del sistema.

# Additional Specifications

Este documento capturará todos los requisitos que no han sido incluidos como parte de los casos de uso y se refieren requisitos no-funcionales globales. Dichos requisitos incluyen: requisitos legales o normas, aplicación de estándares, requisitos de calidad del producto, tales como: confiabilidad, desempeño, etc., u otros requisitos de ambiente, tales como: sistema operativo, requisitos de compatibilidad, etc.

# Navigation Map

En el mapa de navegación deben estar representadas todas las GUI del sistema con las correspondientes rutas de acceso a ellas. Es importante utilizar una nomenclatura jerárquica que permita organizar la forma en que se relacionan las GUI del sistema. Cada GUI lleva un nombre y un código que representa su ubicación en el mapa. Las GUI deben implementar conceptos de ergometría y usabilidad para su diseño. A continuación se presenta un ejemplo. 

# User Interface Prototypes

Se trata de prototipos que le permiten al usuario hacerse una idea más o menos precisa de las interfaces que proveerá el sistema y así, conseguir retroalimentación de su parte respecto a los requisitos del sistema. Estos prototipos se realizarán como prototipos ejecutables interactivos.

# Components Diagram

El diagrama de componentes representa cómo un sistema de software es dividido y muestra las dependencias entre los componentes. Los componentes físicos incluyen archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables, o paquetes. Los diagramas de componentes prevalecen en el campo de la arquitectura de software pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema.

# Test Cases Specifications

Cada prueba es especificada mediante un documento que establece las condiciones de ejecución, las entradas de la prueba, y los resultados esperados. Estos casos de prueba son aplicados como pruebas de regresión en cada iteración. Cada caso de prueba llevará asociado un procedimiento de prueba con las instrucciones para realizar la prueba, y dependiendo del tipo de prueba dicho procedimiento podrá ser automatizable mediante un script de prueba.

# Unit Tests

En desarrollo del software, una prueba unitaria es una forma de probar la corrección de un módulo de código. La idea es escribir casos de prueba para cada función no trivial o método en el módulo de forma que cada caso sea independiente del resto. Para que una prueba unitaria sea buena se deben cumplir los siguientes requisitos:

* Automatizable: no debería requerirse una intervención manual. Esto es especialmente útil para la integración continua.
* Completas: deben cubrir la mayor cantidad de código.
* Repetibles: no se deben crear pruebas que sólo puedan ser ejecutadas una sola vez. También es útil para la integración continua y para las pruebas de regresión.
* Independientes: la ejecución de una prueba no debe afectar a la ejecución de otra.
* Profesionales: las pruebas deben ser consideradas igual que el código, con la misma profesionalidad, documentación, etc.