Library System

Submitted by: Yeong De Jong

T-00185309

Computing in Software Development – Stage 2

Date Submitted: 25/04/2017

**Table of Contents**

[1. Introduction/overview 4](#_Toc480993237)

[2. Functional Components 5](#_Toc480993238)

[3. User Requirements 6](#_Toc480993239)

[3.1. LibSYS will perform member administration 6](#_Toc480993240)

[3.2. LibSYS will perform book management 6](#_Toc480993241)

[3.3. LibSYS will manage loans 6](#_Toc480993242)

[3.4. LibSYS will perform financial administration 6](#_Toc480993243)

[4. System Requirements 7](#_Toc480993244)

[4.1. System Level Use Case Diagram 7](#_Toc480993245)

[4.2. Manage members 8](#_Toc480993246)

[4.2.1. Register member 8](#_Toc480993247)

[4.2.2. De-Register Member 11](#_Toc480993248)

[4.2.3. Update Member 14](#_Toc480993249)

[4.2.4. List Members 17](#_Toc480993250)

[4.2.5. View Members History 20](#_Toc480993251)

[4.3. Manage Books 24](#_Toc480993252)

[4.3.1. Register Book 24](#_Toc480993253)

[4.3.2. Remove Book 27](#_Toc480993254)

[4.3.3. Update Book 29](#_Toc480993255)

[4.3.4. Search Book 32](#_Toc480993256)

[4.4. Manage Book Loans 36](#_Toc480993257)

[4.4.1. Borrow Book 36](#_Toc480993258)

[4.4.2. Return Book 41](#_Toc480993259)

[4.5. Perform Finance 46](#_Toc480993260)

[4.5.1. Update Fine Balance 46](#_Toc480993261)

[4.5.2. Revenue Analysis 49](#_Toc480993262)

[5. System Model 53](#_Toc480993263)

[5.1. Level-0 DFD 54](#_Toc480993264)

[5.2. Level-1 DFD 55](#_Toc480993265)

[5.3. Level-2 DFD (Process P1: Manage Members) 56](#_Toc480993266)

[5.4. Level-2 DFD (Process P2: Manage Books) 57](#_Toc480993267)

[5.5. Level-2 DFD (Process P3: Manage Loans) 58](#_Toc480993268)

[5.6. Level-2 DFD (Process P4: Perform Finance) 59](#_Toc480993269)

[6. Data Model (Class Diagram) 60](#_Toc480993270)

[6.1. Class Diagram 61](#_Toc480993271)

[6.2. Relational Schema 63](#_Toc480993272)

[6.3. Database Schema 64](#_Toc480993273)

[7. Conclusion 66](#_Toc480993274)

[8. Appendices 67](#_Toc480993275)

[8.1. Appendix A – Listings 67](#_Toc480993276)

[8.2. Appendix B – Reports 70](#_Toc480993277)

[9. References 71](#_Toc480993278)

# Introduction/overview

A library can be defined as an organization, a room, or a building where books are kept and referenced. It is an area of various activity on book management. It acts as a collection of books and other materials for users to read or borrow without any payment needed.

Information Technology (IT) has transformed the life of human beings’ and has made the lives easier by various kinds of application or system. Nowadays all the businesses are changing to computer based system. The advantage of using computer based system is that it allowed data to be retrieved and processed much more quickly and it is very easy for users to use. This project is concerned with developing a Library System. In this system, the library management becomes much more well-organized and easier to handle (Priyantha, n.d.).

Librarian has the authority to add, delete or modify the details of a member available to/from the system. Librarian can also add, delete, or modify the details of a book to/from the system. Other than that, Librarian or Manager can also view the monthly fine collected report and monthly cost used for buying books report in the system.

# Functional Components

The following hierarchy chart models the function requirements as a set of functional components for Library System (LibSYS).

# User Requirements

The following requirements have been identified for Library System (LibSYS):

## LibSYS will perform member administration

* + 1. LibSYS will register a new member.
    2. LibSYS will de-register a member.
    3. LibSYS will allow a members’ profile to be updated.
    4. LibSYS will generate a member listing.
    5. LibSYS will allow members’ history to be viewed.

## LibSYS will perform book management

* + 1. LibSYS will register a new book.
    2. LibSYS will allow a book to be removed.
    3. LibSYS will allow books’ information to be updated.
    4. LibSYS will allow books to be searched.

## LibSYS will manage loans

* + 1. LibSYS will record a book borrowed.
    2. LibSYS will record a book returned.

## LibSYS will perform financial administration

* + 1. LibSYS will allow a member’s fine balance to be updated.
    2. LibSYS will generate a revenue analysis report.

# System Requirements

The following system level use case diagram defines the communications between actors and the system. The actors involve in the Library System (LibSYS) are Member, Librarian and Manager. Each actor involves in a single task to achieve a goal.

## System Level Use Case Diagram

System level UC diagram:

LibSYS

Librarian

Member

Manager

## Manage members

This module contains function for register member, de-register member, update member profile, list members and view member’s history.

### Register member

Description: This function registers a member profile on the system.

Use Case Diagram:

Member

Librarian

<<includes>>

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Register member | |
| **Use Case Id** | 1.1 | |
| **Priority** | High | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | Member | |
| **Description** | This function registers a member’s details on the system. | |
| **Preconditions** | Member must complete and sign a Member Registration form. | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Register Member function  **Step 4:** Librarian enters Member details:   * PPS No. * Surname * Forename * D.O.B * Gender * Street * Town * County * Phone * Email * Register Date | **Step 2:** Assign Member ID  **Step 3:** Display UI  **Step 5:** System validates member details:   * PPS No. and Email not already registered * All fields must be entered * PPS No. must be in the correct format, e.g. 1234567AB or 1234567K * Surname, Forename, Town, and County must not contain any numeric * Member must be over 16 * Email must be in the correct format, e.g. dejong@gmail.com * Phone must be in the correct format,  e.g. 085 4625788   **Step 6:** Assign member status a default value of ‘A’ (Active)  **Step 7:** Assign number of books borrowed a default value of zero  **Step 8:** Assign fine balance for a member a default value of zero.  **Step 9:** Save Member details in **Member File.**  **Step 10:** Display confirmation message.  **Step 11:** Clear the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | **Step 5:** Invalid data detected  **Step 6:** Display appropriate error message and return to step 4. |
| **Conclusions** | Member is registered and may now make loans. | |
| **Post conditions** |  | |
| **Business Rules** | Member must be over 16. | |
| **Implementation Constraints** |  | |

### De-Register Member

Description: This function de-register a member profile on the system.

Use Case Diagram:

Member

Librarian

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | De-Register Member | |
| **Use Case Id** | 1.2 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | Member | |
| **Description** | This function changes a member’s status from ‘Active’ to ‘De-registered’ | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes De-Register Member function  **Step 3:** Librarian enters member’s surname  **Step 5:** Librarian selects the member to be de-registered  **Step 7:** Librarian confirms member to be de-registered | **Step 2:** Display UI  **Step 4:** System retrieves ***summary details*** of all **active** members with matching surname from **Member File** and displays a list on the UI  **Step 6:** System retrieves ***all details*** for the **selected** member from **Member File** and displays on UI.  **Step 8:** System validates number of books borrow of the selected member in Member File if number of books borrow is equals to 0.  **Step 9:** System sets de-register date to current date (System Date)  **Step 10:** System sets member status to ‘D’ (Deregistered).  **Step 11:** System updates member details in **Member File**  **Step 12:** System displays confirmation message  **Step 13:** System clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid data entered.**  **Number of Books Borrowed is not equals to 0** |  | **Step 4:** Invalid data detected.  **Step 5:** Display appropriate error message and return to step 3.  **Step 8:** Total books borrow is not equals to 0.  **Step 9:** Display appropriate error message and clears UI. |
| **Conclusions** | Member is de-registered and may no longer make loans. | |
| **Post conditions** |  | |
| **Business Rules** | Only active members and number of books borrow of the selected member is equals to zero can be de-registered. | |
| **Implementation Constraints** |  | |

### Update Member

Description: This function updates a member’s profile on the system.

Use Case Diagram:

Member

Librarian

<<includes>>

<<excludes>>

.

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Member | |
| **Use Case Id** | 1.3 | |
| **Priority** | High | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | Member | |
| **Description** | This function updates a member’s details on the system. | |
| **Preconditions** | Member must complete and sign an update details form | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Update Member Profile function.  **Step 3:** Librarian enters member surname  **Step 5:** Librarian selects the member to be updated  **Step 7:** Librarian changes the details of the selected member.   * PPS No. * Surname * Forename * D.O.B * Gender * Street * Town * County * Phone * Email   **Step 8:** Librarian confirms update member profile | **Step 2:** Display UI  **Step 4:** System retrieves ***summary details*** of all **active** members with matching surname from **Member File** and displays a list on the UI  **Step 6:** System retrieves ***all details*** for the **selected** member from **Member File** and displays on UI for editing.  **Step 9:** System validates member details:   * All fields must be entered * PPS No. and Email not already registered. * PPS No. must be in the correct format, e.g. 1234567AB or 1234567K * Email must be in the correct format, e.g. dejong@gmail.com * Surname, Forename, Town, and County must not contain any numeric * Phone must be in the correct format, e.g. 085 4625788   **Step 10:** System updates member details in **Member File**  **Step 11:** System displays confirmation message  **Step 12:** System clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | **Step 4:** Invalid data detected  **Step 5:** Display appropriate error message and return to step 3.  **Step 9:** Invalid data detected  **Step 10:** Display appropriate error message and return to step 7. |
| **Conclusions** | Member details is now updated | |
| **Post conditions** |  | |
| **Business Rules** | Only active members can be updated | |
| **Implementation Constraints** |  | |

### List Members

Description: This function generates a member listing as shown in Appendix A on the system in alphabetical order of Member ID, Surname, and forename

Use Case Diagram:

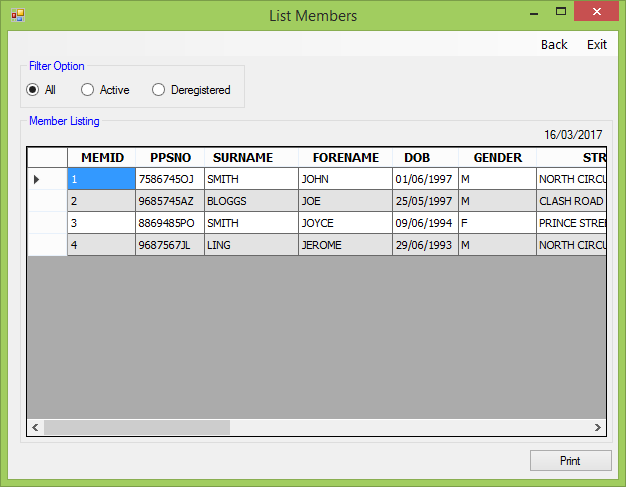
Librarian

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | List Members | |
| **Use Case Id** | 1.4 | |
| **Priority** | Medium | |
| **Source** | Manager | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | - | |
| **Description** | This function generates a member listing on the system in ascending order of member ID, surname, and forename. The list may be printed if required | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes List Members function  **Step 3:** Librarian specifies how the list is to be filtered:   * All * Active * Deregistered   **Step 5:** Librarian requests a print copy of the listing | **Step 2:** System retrieves details of all members from **Member File** and displays them in order of member ID, surname, and then forename on UI  **Step 4:** System retrieves **all details** of **only** the **specified** members from the **Member File** and displays them in order of surname and forename  **Step 6:** System generates a print file based on the displayed data and sends to the default print device  **Step 7:** System confirms that the print job has been sent. |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **No Member details found** |  | **Step 2:** System fails to retrieve member details from the **Member File.**  **Step 3:** System displays appropriate error message |
| **Conclusions** | A listing of all members at the library is generated and printed | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

***Sample of Member Listing:***



### View Members History

Description: This function generates a listing of books borrowed by a member as shown in Appendix A in ascending order of ISBN.

Use Case Diagram:

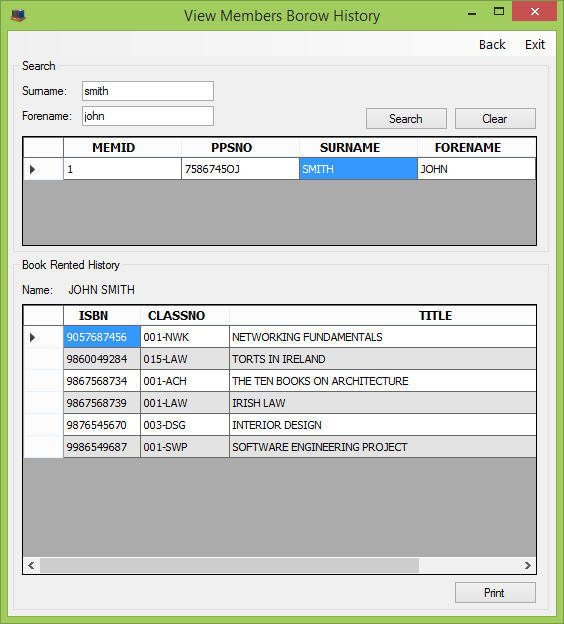
Librarian

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | View Members History | |
| **Use Case Id** | 1.5 | |
| **Priority** | Low | |
| **Source** | Member | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | - | |
| **Description** | The Member requires a list of books that has been borrowed. This function generates a listing of books borrowed by a member in ascending order of ISBN. The list may be printed if required. | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes View Members History function  **Step 3:** Librarian enters member’s surname and forename  **Step 5:** Librarian selects the member to be viewed  **Step 7:** Librarian requests a print copy of the listing | **Step 2:** Display UI  **Step 4:** System retrieves summary details of all members with matching surname and forename from Member File and displays a list in order of Member ID on UI.  **Step 6:** System retrieves **summary details** of **all rented books** of the selected member from **Member File**, **Book File**, **Loan File**, and **Loan Items File** and displays a list of books borrowed by the member in order of ISBN on UI.  **Step 8:** System generates a print file based on the displayed data and sends to the default print device.  **Step 9:** System confirms that the print job has been sent.  **Step 10:** System clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **No Member details found**  **No Book details found** |  | **Step 4:** No record found  **Step 5:** Display appropriate error message and return to step 3  **Step 6:** No record found  **Step 7:** Display appropriate error message and return to step 3 |
| **Conclusions** | A listing of all books that a member has borrowed is generated or printed | |
| **Post conditions** |  | |
| **Business Rules** | Book listing is displayed and cannot be amended or deleted | |
| **Implementation Constraints** |  | |

***Sample of Members History Listing:***



## Manage Books

This module contains function for register book, remove book, update book and search book.

### Register Book

Description: This function registers a book on the system.

Use Case Diagram:

Librarian

<<includes>>

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Register Book | |
| **Use Case Id** | 2.1 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | - | |
| **Description** | This function registers a book’s details on the system. | |
| **Preconditions** | The book must be physically on the library before it can be registered. | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Register Book function  **Step 4:** Librarian enters book details:   * ISBN * Class No * Subject * Title * Author * Edition * Price * No. Of Pages * Publisher’s Name * Publisher’s Country * Year Published | **Step 2:** Assign Book ID  **Step 3:** Display UI  **Step 5:** System validates book details:   * All fields must be entered * ISBN must be in either 10 digits or 13 digits, e.g. 1292018194 or 9783161484100 * Price, No. of Pages, and Year Published must not contain any alphabets. * Class No must be in the correct format, e.g. 007-CAY * Price must be in the correct format, e.g. 53 or 34.50   **Step 6:** Assign Purchase Date a default value of current date (System Date)  **Step 7:** Assign book status a default value of ‘A’ (Available)  **Step 8:** Saves book details in **Book File**  **Step 9:** System displays confirmation message  **Step 10:** System clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered** |  | **Step 5:** Invalid date detected  **Step 6:** Display appropriate error message and return to step 4 |
| **Conclusions** | Book is registered and may now be borrowed by Member. | |
| **Post conditions** |  | |
| **Business Rules** |  | |
| **Implementation Constraints** |  | |

### Remove Book

Description: This function removes a book on the system.

Use Case Diagram:

Librarian

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Remove Book | |
| **Use Case Id** | 2.2 | |
| **Priority** | High | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | - | |
| **Description** | This function changes a book’s status from ‘A’ (Available) to ‘R’ (Removed) | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Remove Book function  **Step 3:** Librarian enters book’s title  **Step 5:** Librarian selects the book to be removed.  **Step 7:** Librarian confirms the book to be remove. | **Step 2:** Display UI  **Step 4:** System retrieves **summary details** of ***all available*** books with matching book title from **Book File** and displays a list on UI.  **Step 6:** System retrieves **all details** for the **selected** book from **Book File** and displays on UI.  **Step 8:** System sets book status to ‘R’ (removed).  **Step 9:** System updates book details in **Book File**  **Step 10:** System displays confirmation message  **Step 11:** System clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid data entered** |  | **Step 4:** Invalid data detected  **Step 5:** Display appropriate error message and return to step 3. |
| **Conclusions** | Book is removed and may no longer available to be borrowed | |
| **Post conditions** |  | |
| **Business Rules** | Only available books can be removed | |
| **Implementation Constraints** |  | |

### Update Book

Description: This function updates a book’s detail on the system.

Use Case Diagram:

Librarian

<<includes>>

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Book | |
| **Use Case Id** | 2.3 | |
| **Priority** | High | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | - | |
| **Description** | This function updates a book’s details on the system | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Update Book function  **Step 3:** Librarian enters book’s title  **Step 5:** Librarian selects the book to be updated.  **Step 7:** Librarian changes the details of the selected book   * ISBN * Class No * Subject * Title * Author * Edition * Price * No. Of Pages * Publisher’s Name * Publisher’s Country * Year Published     **Step 8:** Librarian confirms the book to be update. | **Step 2:** Display UI  **Step 4:** System retrieves ***summary details*** of all **available** books with matching title from **Book File** and displays a list on UI.  **Step 6:** System retrieves **all details** for the **selected** book from Book File and displays on UI for editing.  **Step 9:** System validates book details:   * All fields must be entered * ISBN must be in either 10 digits or 13 digits, e.g. 1292018194 or 9783161484100 * Price, No. of Pages, and Year Published must not contain any alphabets. * Class No must be in the correct format, e.g. 007-CAY * Price must be in the correct format, e.g. 53 or 34.50   **Step 10:** System updates book details in **Book File**  **Step 11:** System displays confirmation message  **Step 12:** System clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid data entered** |  | **Step 4:** Invalid data detected  **Step 5:** Display appropriate error message and return to step 3  **Step 9:** Invalid data detected  **Step 10:** Display appropriate error message and return to step 7 |
| **Conclusions** | Book details is now updated | |
| **Post conditions** |  | |
| **Business Rules** | Only books with status ‘A’ (Available) may be updated. | |
| **Implementation Constraints** |  | |

### Search Book

Description: This function generates a book listing as shown in Appendix A on the system in ascending order of Book ID.

Use Case Diagram:

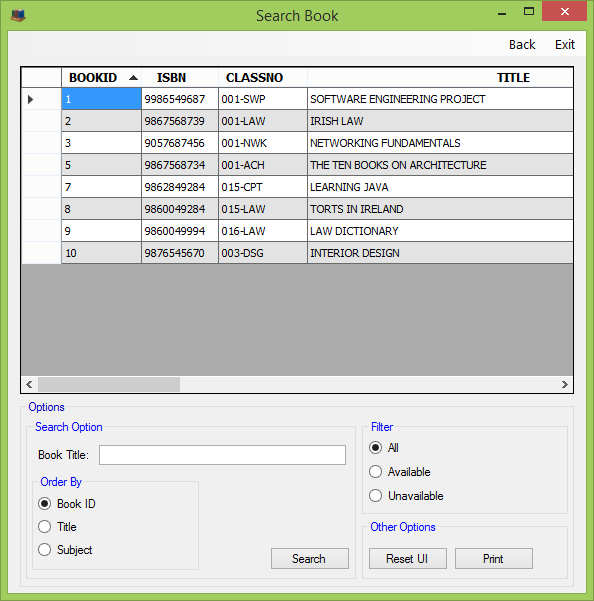
Librarian

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Search Book | |
| **Use Case Id** | 3.2.4 | |
| **Priority** | Medium | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | - | |
| **Description** | The Librarian request to search a book in the library. This function generates a book listing on the system in ascending order of Book ID. The list may be printed if required | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Search Book function  **Step 3:** Librarian searches books by entering book’s title  **Step 4:** Librarian specifies how the list is to be ordered:   * Book ID * Title * Subject Code   **Step 6:** Librarian specifies how the list is to be filtered:   * All * Available * Unavailable   **Step 8:** Librarian requests a print copy of the listing  **Step 11:** Librarian requests to reset the UI | **Step 2:** System retrieves **all details** of all **‘A’ (Available)** and **‘U’ (Unavailable)** books from **Book File** and displays them in ascending order of Book ID on UI  **Step 5:** System retrieves **all details** of books with ***matching title*** from **Book File** and displays them in ascending order of Librarian’s specification on UI.  **Step 7:** System retrieves **all details** of **only** the **specified** books from **Book File** and displays them on UI.  **Step 9:** System generates a print file based on the displayed data and sends to the default print device.  **Step 10:** System confirms that the print job has been sent.  **Step 12:** System resets the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **No Book Details Found**  **Invalid data entered** |  | **Step 2:** System fails to retrieve member details from the Member File.  **Step 3:** Systems displays appropriate error message  **Step 4:** Invalid data detected  **Step 5:** System displays appropriate error message |
| **Conclusions** | A listing of all books is generated or printed and Librarian can now search for books. | |
| **Post conditions** |  | |
| **Business Rules** | Book listing is displayed and cannot be amended or deleted. | |
| **Implementation Constraints** |  | |

***Example of Book Listing:***



## Manage Book Loans

This module contains function for borrow book and return book. When Member requests to make loans, Librarian invokes borrow book function to record loan details to the Database. When a book is returned, Librarian invokes return book function to update the loan details which includes calculating fine if a book is overdue in the Database.

### Borrow Book

Description: This function registers a loan on the system. Assume that each Member can borrow a book at least three days from the current date.

Use Case Diagram:

Member

Librarian

<<excludes>>

<<includes>>

Activity Diagram:

|  |  |
| --- | --- |
| **Librarian** | **System** |
| Valid?  Invoke Borrow Book  Assign Loan ID  Enter Member Surname  Valid?  Error Message  N  Y  Enter Book ISBN  Valid?  Error Message  Select Book  N  Y  Validate Loan Details  Y  Error Message  N  Assign Due Date  Display UI  Retrieves Book Summary Details and Display on UI  Retrieves Member Summary Details and Display on UI  Select Member | Save Loan Item Details |

Exceed?

Error Message

Y

N

Assign Borrow Date

Update Book Status

Update No. of Books Borrowed in Member File

Display confirmation message

Reset UI

Reset UI

Save Loan Details

Validate No. of Books Borrowed

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Borrow Book | |
| **Use Case Id** | 3.1 | |
| **Priority** | High | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | Member | |
| **Description** | This function registers a loan’s details on the system. Assume that each member can borrow a book at least three days from current date. | |
| **Preconditions** | Member must present a valid PPS Card and bring the book physically to the information desk. | |
| **Trigger** | - | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Borrow Book function  **Step 4:** Librarian enters Member Surname  **Step 6:** Librarian selects Member from list  **Step 8:** Librarian enters book ISBN  **Step 10:** Librarian selects the book to be borrowed from Member  **Step 12:** Librarian assigns due date | **Step 2:** Assign Loan ID  **Step 3:** Display UI  **Step 5:** System retrieves **summary details** of the member with matching surname from **Member File** and displays a list on UI  **Step 7:** System retrieves summary details of the **selected** member from **Member File** and displays on UI  **Step 9:** System retrieves **summary details** of all **available** book with matching ISBN from **Book File** and displays a list on UI  **Step 11:** System retrieves summary details of the **selected** book from **Book File** and displays on UI  **Step 13:** System validates loan details that value of Due Date must be at least three days from the current date (System Date).  **Step 14:** System validates the number of books borrowed in **Member File** that a Member does not allow more than three books to be borrowed.  **Step 15:** Assign Borrow Date a value of current date (System Date)  **Step 16:** System saves loan details including due date in **Loan File**  **Step 17:** System saves loan items details in **Loan Items File**  **Step 18:** System updates book status of the selected book in **Book File** to ‘U’ (Unavailable)  **Step 19:** System updates number of books borrowed in **Member File**  **Step 20:** System displays confirmation message  **Step 21:** Clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid Data Entered**  **Invalid Date Entered**  **Number of books borrowed exceed** |  | **Step 5:** Invalid data detected  **Step 6:** Display appropriate error message and return to step 4  **Step 9:** Invalid data detected  **Step 10:** Display appropriate error message and return to step 8  **Step 13**: Invalid date entered.  **Step 14**: Display appropriate error message and return to step 12  **Step 14:** Display appropriate error message  **Step 15:** Reset UI |
| **Conclusions** | Member has successfully borrowed the book | |
| **Post conditions** |  | |
| **Business Rules** | A Member does not allow more than three books to be borrowed. | |
| **Implementation Constraints** |  | |

### Return Book

Description: This function updates a loan item details of a member on the system and calculates fine upon book returned if book returned is overdue.

Use Case Diagram:

Member

Librarian

<<excludes>>

<<excludes>>

Activity Diagram:

|  |  |
| --- | --- |
| **Librarian** | **System** |
| Retrieve Summary Loan Item Details and Display on UI  Assign return date  Calculate Fine  Display Error Message  Invoke Return Book  Display UI  Enter Member PPS  Valid?  Error Message  N  Y  Y  N  Select Book  Confirm Book Returned  Check Overdue  Overdue?  Update Fine Amount  Display confirmation message  Update Book Status  Display confirmation message  Reset UI  Update Fine Balance  Display Fine Details  Specifies Payment Made  Paid?  N  Y  Update number of books borrowed  Ask User if more books to be return for same Member  More Book to be returned?  Y  N | Retrieve Summary Loan Details and Display on UI |

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Return Book | |
| **Use Case Id** | 3.2 | |
| **Priority** | High | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | Member | |
| **Description** | This function updates a loan item details on the system and calculate fine upon book returned if book returned is overdue. Fine may be issued and recorded if required. | |
| **Preconditions** | Member must present a valid PPS Card and return the book physically to the information desk. | |
| **Trigger** | - | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Librarian invokes Return Book function  **Step 3:** Librarian enters member PPS  **Step 5:** Librarian selects the book to be returned  **Step 7:** Librarian confirms the book to be returned  **Step 14:** Librarian selects either ‘Yes’ or ‘No’ option. | **Step 2:** Display UI  **Step 4:** System retrieves **summary details** of loan items with matching PPS from **Loan Items File, Loan File, Books File, and Member File** and displays a list on UI in ascending order of Loan ID  **Step 6:** System retrieves **summary** loan details of the **selected** book in **Loan File** and displays on UI  **Step 8:** Assign return date a value of current date (System Date)  **Step 9:** System checks if book returned is overdue.  **Step 10:** System updates book status of the returned book in **Book File** to ‘A’ (Available)  **Step 11:** System updates number of books borrowed in **Member File**  **Step 12:** System displays confirmation message  **Step 13:** System prompts User if more books to be return for the same Member  **Step 15:** If Librarian selects ‘Yes’, return to step 5 else if Librarian selects ‘No’, proceed to step 16  **Step 16:** Reset UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **Invalid data entered**  **Book Overdue** | **Step 13:** Librarian specifies payment is either Paid or Owed | **Step 4:** Invalid data detected  **Step 5:** Display appropriate error message and return to step 3  **Step 9:** System displays appropriate error message.  **Step 10:** System calculates fine  **Step 11:** System updates fine amount of the selected loan items in **Loan Items File**  **Step 12:** Display Fine Details on UI  **Step 14:** System updates fine balance of the member in **Member File** if the payment specified by librarian is owed.  **Step 15:** System displays confirmation message and return to Step 10. |
| **Conclusions** | Member has successfully returned the book and fine is calculated upon book returned if book returned is overdue. | |
| **Post conditions** |  | |
| **Business Rules** | A penalty of €0.20 for every day late return from due date | |
| **Implementation Constraints** |  | |

## Perform Finance

This module contains function that allows a Member’s fine balance to be updated and perform revenue analysis which analysed the total amount of book bought and the total amount of fine issued for each month.

### Update Fine Balance

Description: This function updates fine balance of a Member if the Member does not make fine payment upon book returned if fine is issued to the Member.

Use Case Diagram:

Librarian

<<excludes>>

<<includes>>

Member

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Update Fine Balance | |
| **Use Case Id** | 4.2 | |
| **Priority** | High | |
| **Source** | Librarian | |
| **Primary Business Actor** | Librarian | |
| **Other Participating Actors** | Member | |
| **Description** | This function updates fine balance of a Member if a Member does not make the fine payment upon book returned if fine is issued to the Member. | |
| **Preconditions** | Member must pay the amount of fine to the Librarian at the information desk. | |
| **Trigger** | - | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step** 1: Librarian invokes Update Fine Balance function.  **Step 3:** Librarian enters member surname.  **Step 5:** Librarian selects the member to be updated  **Step 7:** Librarian enters the amount of fine paid  **Step 8:** Librarian confirms update fine balance of the selected member. | **Step 2:** Display UI  **Step 4:** System retrieves **summary details** of all **active** members with matching surname from **Member File** and display a list on UI in order of Member ID, Surname, and Forename.  **Step 6:** System retrieves **summary details** for the **selected** member from **Member File** and displays on UI for editing  **Step 9:** System validates fine update details:   * The amount of fine paid must be less than or equal to the fine balance recorded for that member.   **Step 10:** System updates member details in **Member File**  **Step 11:** System displays confirmation message  **Step 12:** System clears the UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **No record found**  **No Fine Balance**  **Invalid data entered** |  | **Step 4:** No such record found.  **Step 5:** System displays appropriate error message and return to step 3  **Step 6:** Fine balance of the selected member is equal to 0  **Step 7:** System displays an appropriate information message and return to step 3  **Step 9:** Invalid data entered  **Step 10:** System displays an appropriate error message and return to step 7 |
| **Conclusions** | Fine Balance of a Member is now updated. | |
| **Post conditions** | - | |
| **Business Rules** | - | |
| **Implementation Constraints** | - | |

### Revenue Analysis

Description: This function generates a report as shown in Appendix B on the system that analysed the total cost used for buying books and the total amount of fine issued to member for each month. The report may be printed if required.

Use Case Diagram:

Librarian

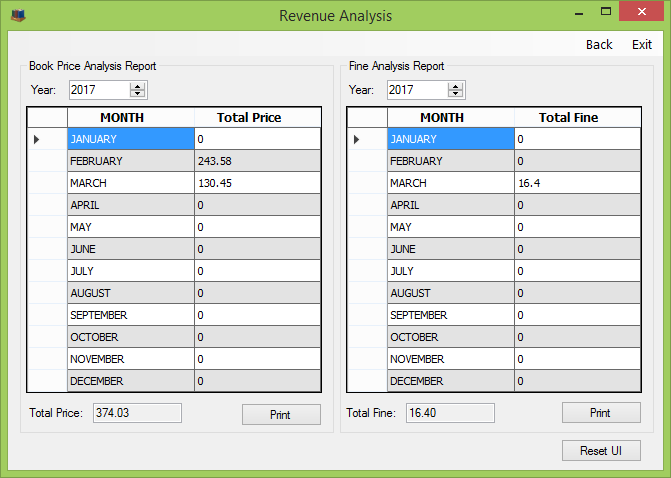
Manager

<<excludes>>

Use Case Narrative:

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | Revenue Analysis | |
| **Use Case Id** | 4.2 | |
| **Priority** | High | |
| **Source** | Manager | |
| **Primary Business Actor** | Manager, Librarian | |
| **Other Participating Actors** | - | |
| **Description** | This function generates a revenue analysis report on the system which analysed the total cost used for buying books and the total amount of fine issued to member for each month. The report may be printed if required. | |
| **Preconditions** |  | |
| **Trigger** | None | |
| **Expected Scenario** | **Actor Action** | **System Response** |
|  | **Step 1:** Manager or Librarian invokes Revenue Analysis function  **Step 4:** Librarian or Manager specifies the year to be viewed on monthly cost report  **Step 6:** Librarian or Manager specifies the year to be viewed on month fine report  **Step 8:** Librarian or Manager requests a print copy of the report  **Step 11:** Librarian or Manage requests to reset UI | **Step 2:** System retrieves book price details from **Book File** and generates a monthly cost report on current year and display them in order of month on UI  **Step 3:** System retrieves fine details from **Loan Items File** and generates a monthly fine report on current year and display them in order of month on UI  **Step 5:** System retrieves details of **only** the **specified** year from **Book File** and displays them in order of month  **Step 7:** System retrieves details of **only** the **specified** year from **Loan Items File** and displays them in order of month  **Step 9:** System generates a print file based on the displayed data and sends to the default print device  **Step 10:** System confirms that the print job has been sent  **Step 12:** System resets UI |
| **Alternate Scenarios** | **Actor Action** | **System Response** |
| **No Record Found** |  | **Step 2:** System fails to retrieve book price details from **Book Files.**  **Step 3:** System displays appropriate error message  **Step 3:** System fails to retrieve monthly fine details in **Loan Items File.**  **Step 4:** System displays appropriate error message. |
| **Conclusions** | Revenue analysis report is generated or printed. | |
| **Post conditions** |  | |
| **Business Rules** | Report is displayed and cannot be amended or deleted. | |
| **Implementation Constraints** |  | |

***Sample of Revenue Analysis***



# System Model

*The following dataflow diagrams have been produced for the system*:

**DFD Elements:**

**External Entities:**

Member

**Data Stores:**

D1 Member File

D2 Book File

D3 Loan File

D4 Loan Items File

**Processes:**

**P1 Manage Members** P1.1 Register Member  
 P1.2 De-register Member  
 P1.3 Update Member  
 P1.4 List Members  
 P1.5 View Member History

**P2 Manage Books** P2.1 Register Book  
 P2.2 Remove Book  
 P2.3 Update Book  
 P2.4 Search Book

**P3 Manage Loans** P3.1 Borrow Book  
 P3.2 Return Book

**P4 Process Fines** P4.1 Record Fine  
 P4.2 Revenue Analysis

## Level-0 DFD

Library System (LibSYS)

Member

Books

Loan Request

## Level-1 DFD

Loan Details

Member Details

Member Details

P1

Manage Members

D1

Member File

P3

Manage Loans

Member

Member Details

D3

Loan File

Book Details

Loan Items Details

D2

Book File

Book Details

P2

Manage Books

Fine Balance

D4

Loan Items File

Member Details

D1

Member File

P4

Perform Finance

Fine Balance

Payment

Member

## Level-2 DFD (Process P1: Manage Members)

P1.4

List Members

P1.1

Register Member

Member Details

Member Details

Member Details

Member

New Member Details

Member Details

Member

Member Details

Member Details

Member Status

Member Details

D1

Member File

New Member Details

P1.2

Deregister Member

P1.3

Update Member

Member Loan Details

P1.3

View Member History

D3

Loan File

## Level-2 DFD (Process P2: Manage Books)

P1.4

Search Book

P2.1

Register Book

Book Details

Book Details

Book Details

Book Details

Book Status

D2

Book File

New Book Details

P1.2

Remove Book

P1.3

Update Book

## Level-2 DFD (Process P3: Manage Loans)

Member

Loan Details

Loan Details

Book Details

Book Status

D2

Book File

D1

Member File

P3.1

Borrow Book

D3

Loan File

Member Details

Loan Details

Book Status

P3.2

Return Book

D2

Book File

New Total Books Borrowed

D1

Member File

Fine Balance

Member

Return Request

Borrow Request

Total Books Borrowed

Loan Details

D4

Loan Items File

Return Details

## Level-2 DFD (Process P4: Perform Finance)

Payment Request

New Fine Balance

P4.1

Update Fine Balance

P4.2

Revenue Analysis

Member

D1

Member File

Member Details

Analysis Request

Analysis Request

Librarian

Manager

Fine Amount Details

D4

Loan Items File

Book Price Details

D2

Book File

Fine Balance Details

# Data Model (Class Diagram)

Data Modelling is fundamental entities to introduce abstraction in a DBMS. Data Models define how data is connected to each other and how they are processed and stored inside the system (Tutorialspoint, n.d.). For this project, Entity-Relationship (ER) Model is used. ER Model refers to a collection of data requirements. The ER model contains Entities, Relationships, and attributes. The notation used for this project to produce the ER diagrams is the Unified Modelling Language (UML) Class Diagram.

Entity is a ‘thing’ which the organisation recognises as being capable of an independent existence, uniquely identified and is an abstraction for some domain and is an aspect of the real world that have properties called attributes. Each entity has a logical association called relationship. Relationships are mapped with entities in various ways. Mapping cardinalities defined the number of association between two entities. It is either one to one, one to many, many to one or many to many. Entities in this project are: Member, Book, Rental, Fine

Normalisation is the process of testing the correctness of a logical data model to reduce redundancy of data (Techopedia, n.d.) and remove the possibility of update anomalies. There are three main types of normalization, which is 1NF, 2NF and 3NF.

**1NF** where the relation contains no repeating groups and all non-key attributes are functionally dependent on the primary key.

**2NF** where the relation is in 1NF and all non-key attributes are fully functionally dependent on the primary key.

**3NF** where the relation is in 2NF and the pairs of non-key attributes are not transitively dependent on the primary key.

## Class Diagram

Library System UML Diagram

Month\_Reference

Months {pk}  
Value

0 … \*

1

Refers to

Member

MemID {pk}  
PPSNo {pk}  
Surname  
Forename  
DOB  
Gender  
Street  
Town  
County  
Phone  
Email  
Reg\_Date  
Mem\_Status  
Books\_Borrow  
Fine\_Balance  
Dereg\_Date

Book

BookID {pk}  
ISBN  
ClassNo  
Title  
Author  
Edition  
Price  
NoPages  
PubName  
YearPublish  
PurchaseDate  
Book\_Status

Rental

LoanID {pk}  
BorrowDate  
DueDate

BookRentalItems

ReturnDate  
FineAmount

🡪

🡨

Has

Has

0 … \*

1

🡫

1

1…3

Subject

SubjectCode {pk}  
SubjectName

Country

CountryCode {pk}  
CountryName

1

1

0 … \*

0 … \*

🡨

🡨

Refers to

Refers to

**Where:**

Gender is either ‘M’ (Male) or ‘F’ (Female)

Member Status is either ‘A’ (Active) or ‘D’ (Deregistered)

Book Status is either ‘A’ (Available), ‘U’ (Unavailable), or ‘R’ (Removed)

Dates are in the format dd-MM-yyyy (e.g. 12-DEC-2016)

Only Dereg\_Date and Return Date can be NULL

## Relational Schema

**Members** (MemID, PPSNo, Surname, Forename, DOB, Gender, Street, Town, County, Phone, Email, Reg\_Date, Mem\_Status, Books\_Borrow, Fine\_Balance, Dereg\_Date)

**Books** (BookID, ISBN, ClassNo, Title, SubjectCode, Author, Edition, Price, NoPages, PubName, CountryCode, YearPublish, PurchaseDate, Book\_Status)

**Rentals** (LoanID, MemID, BorrowDate, DueDate)

**BookRentalItems** (LoanID, BookID, ReturnDate, FineAmount)

**Countries** (CountryCode, CountryName)

**Subjects** (SubjectCode, SubjectName)

**Month\_Reference** (Months, Value)

## Database Schema

**Relation:** Members

Attributes:

MemberID numeric(3)

PPSNo char(9) NOT NULL UNIQUE

Surname char(25) NOT NULL

Forename char(25) NOT NULL

DOB Date NOT NULL

Gender char(1) NOT NULL

Street char(30) NOT NULL

Town char(30) NOT NULL

County char(20) NOT NULL

Phone char(14) NOT NULL

Email char(40) NOT NULL UNIQUE

Reg\_Date DATE NOT NULL

Mem\_Status char(1) NOT NULL

Books\_Borrow numeric(1) DEFAULT 0

Fine\_Balance numeric(5,2) DEFAULT 0

Dereg\_Date DATE

**Primary Key:** MemberID

**Relation:** Books

Attributes:

BookID numeric(4)

ISBN numeric(13) NOT NULL

ClassNo char(6) NOT NULL

Title char(110) NOT NULL

SubjectCode char(3) NOT NULL

Author char(35) NOT NULL

Edition numeric(2) NOT NULL

Price numeric(5,2) NOT NULL

NoPages numeric(3) NOT NULL

PubName char(35) NOT NULL

CountryCode char(3) NOT NULL

YearPublish char(4) NOT NULL

PurchaseDate DATE NOT NULL

Book\_Status char(1) NOT NULL

**Primary Key:** BookID

**Foreign Key:** SubjectCode References Subjects

**Foreign Key:** CountryCode References Countries

**Relation:** Rentals

Attributes:

LoanID numeric(5)

MemID numeric(4) NOT NULL

BorrowDate DATE NOT NULL

DueDate DATE NOT NULL

**Primary Key:** LoanID

**Foreign Key:** MemID References Members

**Relation:** BookRentalItems

Attributes:

LoanID numeric(5)   
 BookID numeric(4)

ReturnDate DATE

FineAmount numeric(4,2) DEFAULT 0

**Primary Key:** LoanID, BookID  
**Foreign Key:** LoanID References Rentals  
**Foreign Key:** BookID References Books (BookID)

**Relation:** Countries

Attributes:

CountryCode char(3)

CountryName char(45) NOT NULL

**Primary Key:** CountryCode

**Relation:** Subjects

Attributes:

SubjectCode char(3)

SubjectName char(110) NOT NULL

**Primary Key:** SubjectCode

**Relation:** Month\_Reference

**Attributes:**

Months char(9)

Value numeric(6,2) DEFAULT 0

**Primary Key:** Months

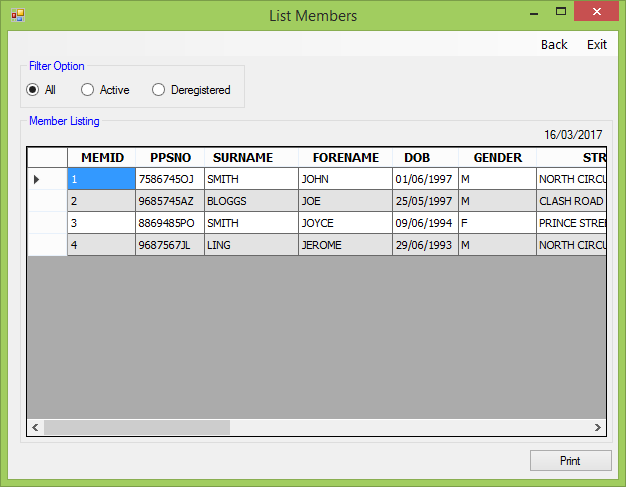
# Conclusion

In a nutshell, the Library system allows librarian to register a member, de-register a member, update member’s profile, list members and view members borrow history. The system also allows librarian to register a book, remove a book, update a book and search for a book in the system which improve the efficient of book management. This system can also manage loans by recording member’s details and book’s details into the system. Lastly, the system can also update fine balance due to late return and performs revenue analysis which analysed the monthly fine collected and total cost used for buying books that are recorded in the system.

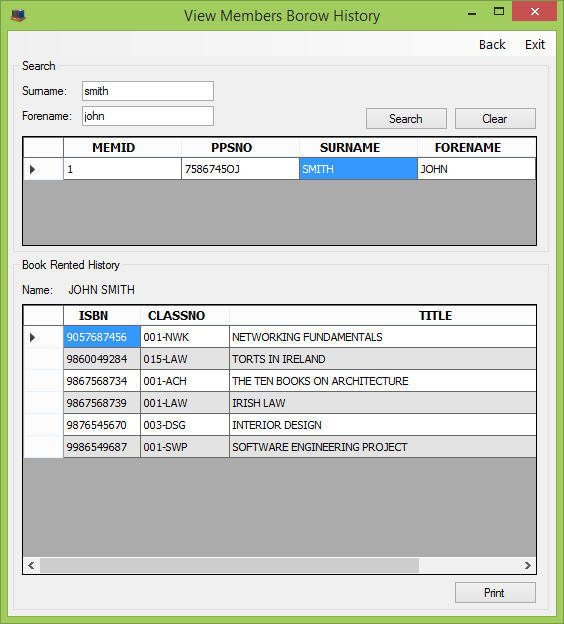
# Appendices

## Appendix A – Listings

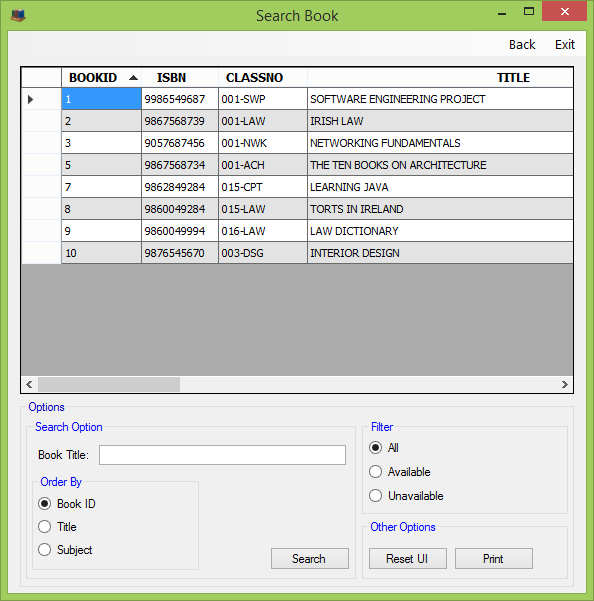
***Member Listing***



***Member’s History Listing***

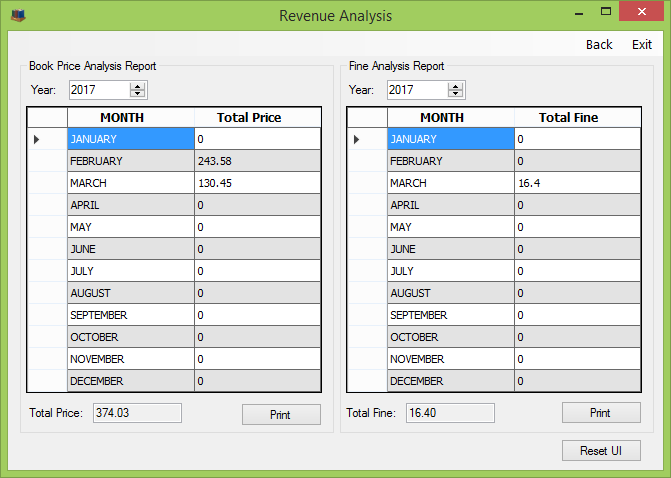


***Book Listing***



## Appendix B – Reports

***Revenue Analysis Report***



# References

Priyantha, I., n.d. *Library Management System - proposal.* [Online]   
Available at: https://www.scribd.com/doc/32660810/Library-Management-System-proposal  
[Accessed 24 11 2016].

Techopedia, n.d. *Normalization.* [Online]   
Available at: https://www.techopedia.com/definition/1221/normalization  
[Accessed 24 11 2016].

Tutorialspoint, n.d. *DBMS - Data Models.* [Online]   
Available at: https://www.tutorialspoint.com/dbms/dbms\_data\_models.htm  
[Accessed 24 11 2016].