|  |  |
| --- | --- |
|  | Library Catalogue System - CA2 |
|  | Group 2 |
|  | Michael O’Brien, Mariusz Ordon, Dariusz Piskorowski  Software Development  1/12/23 |

Table of Contents

[Introduction: 2](#_Toc150909249)

[System Overview: 2](#_Toc150909250)

[Minimum Class List: 3](#_Toc150909251)

[Book: 3](#_Toc150909252)

[Audio Book: 3](#_Toc150909253)

[Thesus: 3](#_Toc150909254)

[Dissertation: 3](#_Toc150909255)

[CD: 4](#_Toc150909256)

[DVD: 4](#_Toc150909257)

[Author: 4](#_Toc150909258)

[Library User: 4](#_Toc150909259)

[Abstract Classes: 5](#_Toc150909260)

[LibraryItem: 5](#_Toc150909261)

[Interface Classes: 5](#_Toc150909262)

[Printable: 5](#_Toc150909263)

[Convert to csv: 5](#_Toc150909264)

[Extra Classes required for functionality: 5](#_Toc150909265)

[Save To CSV File: 5](#_Toc150909266)

[Loan: 6](#_Toc150909267)

[Search: 6](#_Toc150909268)

[Starting Class Diagram 7](#_Toc150909269)

## Introduction:

The objective of this assignment will be for a team of three students to design and implement a Java based library catalogue system. The system will manage information about library resources, authors and users.

## System Overview:

The system will be able to store, retrieve and manipulate data objects. Using Java interfaces, abstract class and regular classes. The system will also make use of stacks, queues or array lists. Store data externally to CSV files. It will also make use of Maven dependencies for enhanced functionality.

The system will implement stacks, queues or linked lists. File handling and error handling. It will also implement sorting and searching algorithms.

The system will also be able to:  
Add an asset to the catalogue.  
Add an author to the catalogue.  
Add a library user to the system.  
Borrow an asset.  
Return an asset.  
List available books.  
List assets borrowed by a user.  
List assets authored by an author.  
List of overdue assets.

To operate the system, it will be through the command line as text-based menus.

## Minimum Class List:

### Book:

#### *Attribute:*

Title  
Author  
ISBN  
Availability status

#### *Methods:*

Constructor  
Get and set methods  
toString()

### Audio Book:

#### Attributes:

Title  
Author  
ISBN  
Availability status

#### Methods:

Constructor  
Get and set methods  
toString()

### Thesus:

#### Attributes:

Title  
Author  
Topic  
Abstract  
Date published  
Availability status

#### Methods:

Constructor  
Get and set methods  
toString()

### Dissertation:

#### Attributes:

Title  
Author  
Topic  
Abstract  
Date published  
Availability status

#### Methods:

Constructor  
Get and set methods  
toString()

### CD:

#### Attributes:

Title  
Producer  
Playtime  
Availability status

#### Methods:

Constructor  
Get and set methods  
toString()

### DVD:

#### Attributes:

Title  
Director  
Playtime  
Availability status

#### Methods:

Constructor  
Get and set methods  
toString()

### Author:

#### Attributes:

Name  
List of authored books

#### Methods:

Constructor  
Get and set methods  
toString()

### Library User:

#### Attributes:

ID  
Name  
List of borrowed assets

#### Methods:

Constructor  
Get and set methods  
toString()

## Abstract Classes:

Abstract classes will represent common attributes and functionality between classes. It acts as a base class that cannot be instantiated.

### Library Item:

#### Attributes:

ID  
Title  
Author  
Availability Status

#### Methods:

Constructor  
Get and set methods  
toString()

## Interface Classes:

Interface classes act as a way to provide common behaviour between classes. Some interfaces we have identified are as follow:

### Printable:

#### Attributes:

#### Methods:

Display all details  
Display summary details

### Convert to csv:

#### Attributes:

#### Methods:

Convert to csv

## Extra Classes required for functionality:

### Save To CSV File:

#### Attributes:

#### Methods:

Save to file

### Loan:

#### Attributes:

Member  
Library Item  
Date Borrowed  
Return Date

#### Methods:

Return Item

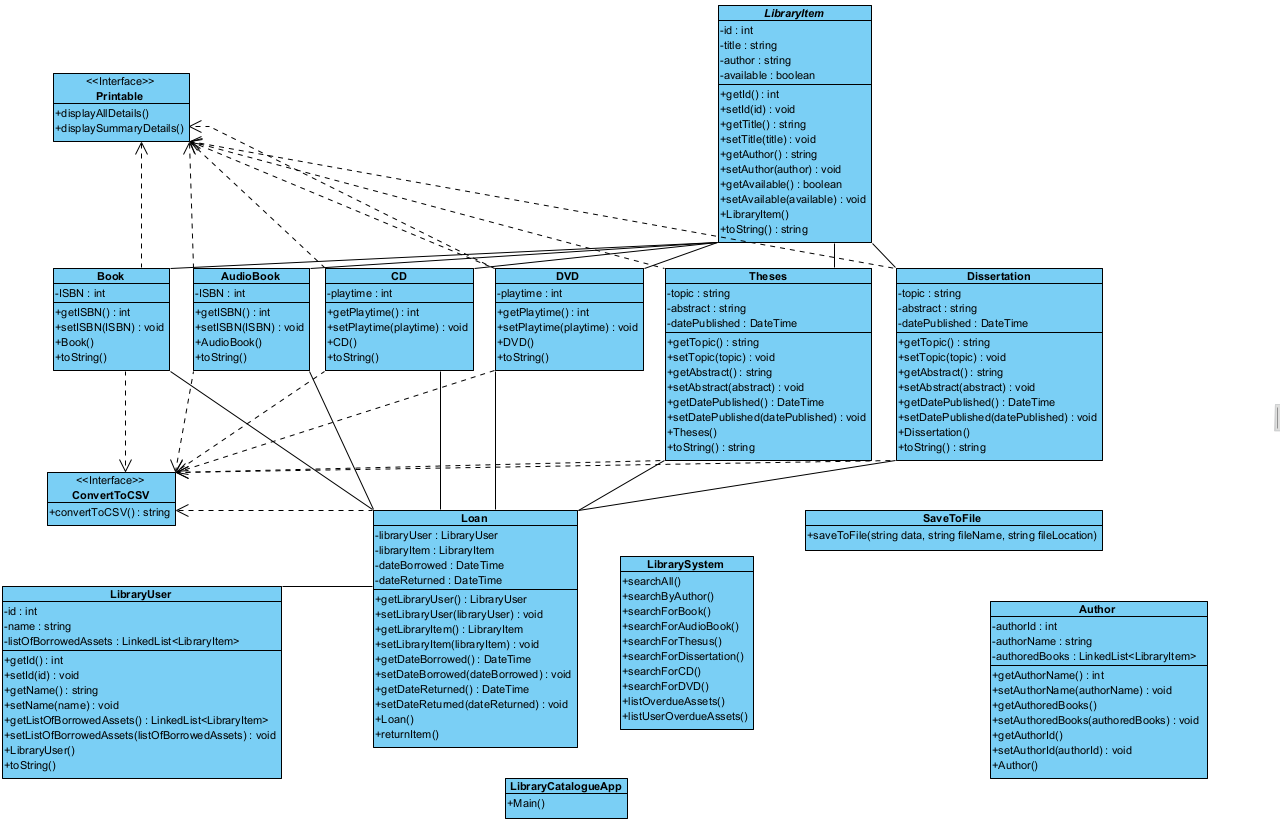
### Library System:

#### Attributes:

#### Methods:

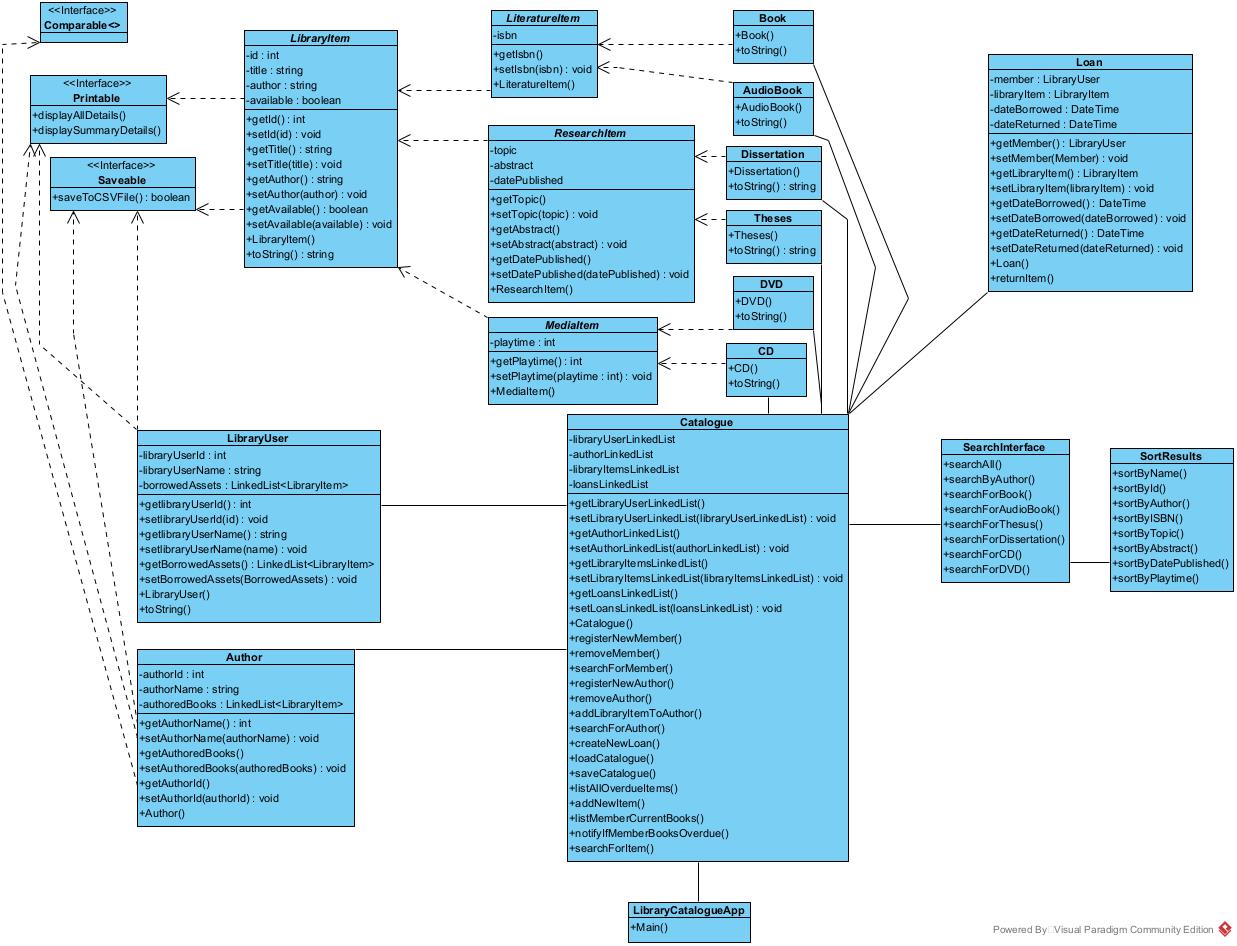
Search By Author  
Search All  
Search for Book  
Search for Audio Book  
Search for Thesus  
Search for Dissertation  
Search for CD  
Search for DVD  
List assets borrowed by user  
List all overdue assets

## Starting Class Diagram:



As the class diagram above does not use interfaces and enough abstract classes to create efficient design. We will have to refine the diagram.

## Refining Class Diagram:



Decide between Stack, Queue or LinkedList.

Decide between classes, interfaces, data structures.

Decide what searching/sorting algorithm to use.

Compare between them.