Software Requirements Specification

Public Library System

Retakers:

- Jasper Schinkel
- Lennert van Splunder

Table of Contents

1 Introduction
1.1 Motivation
1.2 Scope
1.3 Definitions and abbreviations
1.4 References
2 The overall description
2.1 Product overview and functionalities
3 System requirements
3.1 External interface requirements
3.2 Functional requirements
3.3 Non-functional requirements
3.4 Use Case
3.5 Class Diagram

Introduction

1.1 Motivation

This document is meant for stakeholders and future users. The purpose of this document is to clarify what the system will be and why it will be useful.

1.2 Scope

Future customers can loan books with our system. It will be as functional as possible, but it won't handle payments. Data traffic will be handled internally and not with a database. The system will function as an addition to the public library in a console-based manner. The deadline for this project is 16-04-2022.

1.3 Definitions and abbreviations

SRS > Software Requirement Specification
PLS > Public library system
PL -> Public Library

1.4 References

Assignment doc:
https://docs.google.com/document/d/198BOn1kX0Sw45HEIayfMBjdwGp_ql5-gpCaLA-_7dI0/edit#heading=h.sdsb4obv13zq

Overall description

2.1 Product overview and functionalities

The product is an independent product, but it could be used as a component of a larger system. It doesn't have too many dependencies; the only ones are hardware related. You'll need a functioning computer, hard disk, keyboard, mouse, RAM, and a monitor.

The product will manage login options, so that both admins and members can log in. As soon as you're logged in as a member, you should be able to; search for books, loan books, see their availability and search for them. This is all done in the library catalog which will be saved in data file(s). When you're an admin you can do the same as a member and more. The admins are the ones that register members into the system. They can see the list of members in the library system and are able to add, edit and delete members. They also manage the catalog, where they can check, add, edit, delete, and search for books. They can also load and add a list of books into the library-catalog. And finally, they're the ones to make backups of the system or to restore a specific backup.

The system itself must check the data files for the catalog, book items and members as soon as it starts. If there is no data file yet, the system should initialize and configure one.

System requirements

3.1 External Interface requirements

The user shall be able to see the system in a console-based manner. This will be possible with python and an IDE to work in.

3.2 Functional requirements

3.2.A. Users

REQ ID	DESCRIPTION	MOSCOW
3.2.A.1	A user can log in to the system by filling in a correct username and password.	Must have
3.2.A.2	A user can see the full list of books from the library system's catalog.	Must have
3.2.A.3	A user can search for books from the library system's catalog.	Must have
3.2.A.4	A user can see the full list of book items with their availability from the library system's library.	Must have
3.2.A.5	A user can search for book items with their availability from the library system's catalog.	Must have
	LIBRARY ADMIN	
3.2.A.6	A library admin should have a panel for managing members, books, book items, backups, and loans.	Should have
3.2.A.7	A library admin can check the status of members' current loan items.	Must have
3.2.A.8	A library admin can see the full list of library members.	Must have
3.2.A.9	A library admin can add a member to the list.	Must have
3.2.A.10	A library admin can edit a member from the list.	Must have
3.2.A.11	A library admin can delete a member from the list.	Must have
3.2.A.12	A library admin can add a list of members to the system by importing a CSV-file.	Should have

3.2.A.13	A library admin can add a book in the library system’s catalog.	Must have
3.2.A.14	A library admin can edit a book in the library system’s catalog.	Must have
3.2.A.15	A library admin can delete a book in the library system’s catalog.	Must have
3.2.A.16	A library admin can add a book item in the library system’s library.	Must have
3.2.A.17	A library admin can edit a book item in the library system’s library.	Must have
3.2.A.18	A library admin can delete a book item in the library system’s library.	Must have
3.2.A.19	A library admin can lend a book item to a library member.	Must have
3.2.A.20	A library admin can make a backup of the system.	Must have
3.2.A.21	A library admin can restore the system from a previously saved backup.	Must have
3.2.A.22	A library admin can delete a saved backup.	Should have
3.2.A.23	A library admin can add a list of books to the library system importing a JSON-file.	Should have
3.2.A.24	A user’s username and password should be lowercase.	Should have
3.2.A.25	A library member’s username must be unique.	Must have
	LIBRARY MEMBER	
3.2.A.26	A library member should have a dedicated panel so that he or she can navigate through available features and perform their desired activities with ease.	Should have
3.2.A.27	A library member can loan a book item.	Must have
3.2.A.28	A library member can return a book item.	Must have

3.2.B. Catalog and Library

REQ ID	DESCRIPTION	MOSCOW
3.2.B.1	The catalog consists of a list of books.	Must have
3.2.B.2	The library consists of a list of book items.	Must have
3.2.B.3	A book item in the library must have a given number of printed copies.	Must have
3.2.B.4	A book item in the library must initially have 5 printed copies.	Must have

3.2.D. Search function

REQ ID	DESCRIPTION	MOSCOW
3.2.D.1	The search function must accept partial keys.	Must have
3.2.D.2	The search function must not be case-sensitive.	Must have
3.2.D.3	The search function must be able to find books by an author.	Must have
3.2.D.4	The search function must be able to find books by a title.	Must have

3.2.E. Extra

REQ ID	DESCRIPTION	MOSCOW
3.2.E.1	The menu items should be represented by one digit.	Must have
3.2.E.2	A user should always be able to exit the system.	Should have
3.2.E.3	A user must be logged in to fully use the system.	Must have

3.2.F. Library system

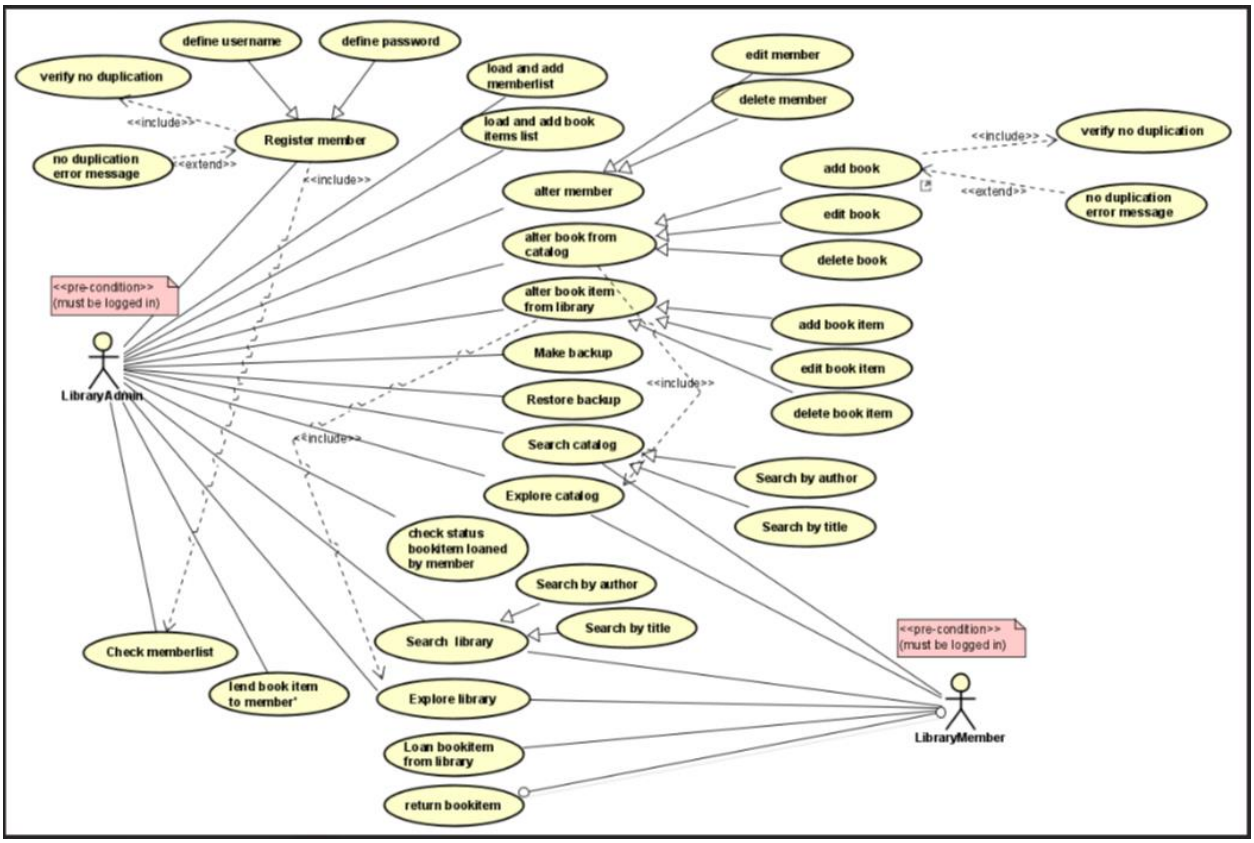
REQ ID	DESCRIPTION	MOSCOW
3.2.F.1	The system is able to load the catalog.	Must have
3.2.F.2	The system is able to load the library.	Must have
3.2.F.3	The system is able to load the members list.	Must have
3.2.F.4	The system has a login system.	Must have
3.2.F.5	The system has a comprehensive list of members, storing	Must have

	data that includes, at a minimum, their username and password.	
3.2.F.6	The system has a list of books, storing data that includes, at minimum, their ISBN, title, and author.	Must have
3.2.F.7	The system has a list of book items, storing data that include, at minimum, their ISBN and availability.	Must have

3.3 Non-functional requirements

1. The system must be built with python.
2. All data will be stored in RAM and files.
3. There must be 2 account types available for the user: admin & member.
4. The system interface must be console-based.
5. A member is not allowed to loan the same book item more than once.
6. A member is not allowed to borrow a book for a longer time than 60 days.
7. A member is not allowed to borrow more than 3 books simultaneously.
8. A member must not be able to loan the same book more than once.

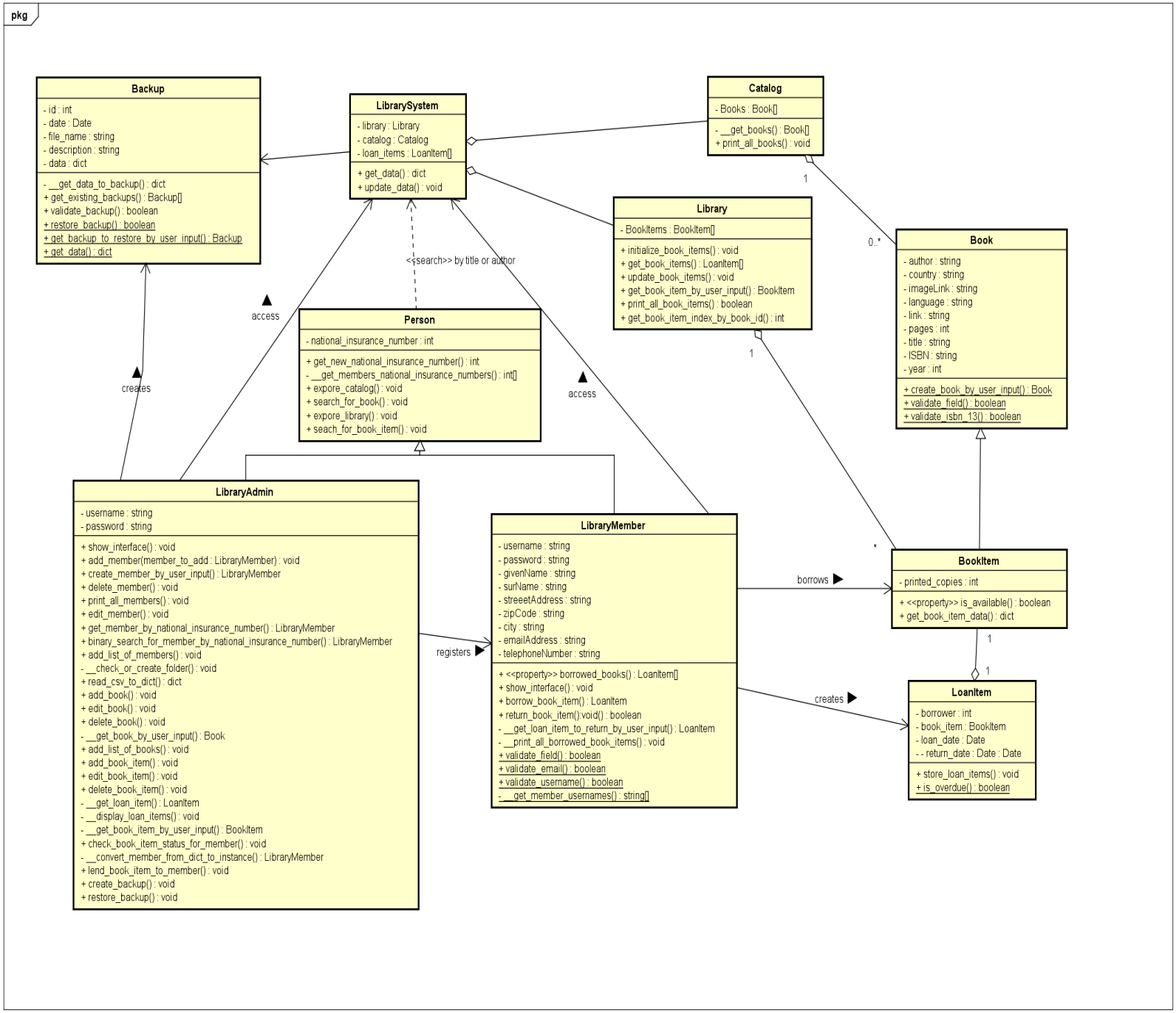
3.4 Use case



Description

There is a pre-condition that requires the actors to be logged in. Because you don't have access to any of these use cases when you're not logged in.
Any user that is logged in can search through the catalog/library by author or title.
The member can explore and search catalog, explore and search library, loan a book item and return a book item.
The librarian can register a member, load, and add a list of books and members, explore /search/alter a member, book, or book item, it can also make and restore backups. A librarian can also check the status of loan items and lend book items to members. Loan a book item can happen trough the member and through the librarian, but they both work different so that's why they are separated use cases. The librarian is responsible for registering members. He does that by defining a username and password. There is no other way to register a member. It also has a <<include>> dependency with check member list because it must see the member list to check for no duplication. And when a librarian wants to add a member to the system, the system checks for no duplication. The same goes for the adding a book. I also added an <<exclude>> with an error message for adding members and books. If there is a duplication it will not be allowed.

3.5 Class diagram



Description

A library admin is a type of user of PLS who is responsible for the administration of the system. There is only one admin account, and all admins (individuals) of the library system use the same account to perform administration tasks.

A library member is a type of user of PLS who is able to explore the library system and see and search the list of books in the catalog or in the library. He/she can perform some specific operations though the system, for example borrow a book, return a book, etc. Every member has his/her own specific account on the library system.

A catalog is a list of books. The books in this list might or might not be physically available in the library (to borrow). This list contains information of books.

A book means the information of a book, such as the title, author, publication year, etc. A book does not necessarily exist in the library, but it is an abstract concept of a book defined in the catalog. There might be some real physical copies (zero or more) of a book in the library. These physical copies are called book items. Note a book cannot be borrowed, because it is a concept, but a book item is a physical object which can be borrowed.

A book item is a physical copy of a book in the library, which can be borrowed by a member. This might be zero or more copies of a book in the library.

A library is a collection of physical book items. These book items which are listed in the library can be borrowed if there are available (printed paper) copies.

At the top of the class are two classes, the library system class, and the backup class. The library system class is the most generic form of a library. It is sort of used as an application gate way. It has an association relation with the backup class since backups are made of a library system instance and a library system uses the backups to save or restore data.

The person class inherits from the Library class since person instances need to interact with the library system in some way, for example to check out the catalog or library.

The library member and library admin classes are subclasses of the Person class. They'll inherit the national insurance number and operations that are accessible for both library members and library admins according to the requirements.

The library admin class only has the inherited national insurance number, username and password as fields. That's because it's used by each librarian to manage the library system, so having specific data such as a name does not really fit the class. It does however fit to have a bunch of administrator related operations encapsulated in it. Because these should be only accessible for administrator instances and library admin subclasses.

The library member class is a more detailed entity with a lot of fields provided for personal information.

The library has a composition relation with the library system because the library system will cease to exist once the library is gone and the other way around. Although the library could still exist, it won't make much sense because when a library system is not around there will not be any borrowing or returning book items.

The library has a composition relation with the book Item class, since the library can't exist without book items. The book items theoretically can exist without a library because it's a paper book, but in the context of a library system where everything is based on the book items, they'll need each other.

The loan item class inherits the book Item class since it's a book item that's loaned by a user. As soon as the user returns the book item it will transition back from being a loan item to being a book item again.

Since book items are physical copies of books, they inherit the book class. Books are stored in the catalog and a catalog can't exist without books. Therefore, it has a composition relationship, because a book is just information of a book, which would not make much sense when having books without a catalog.

The catalog only has an association relationship with the library system because the library system is not particularly dependent on a catalog. It can still lend out book items without a catalog, which is the main goal of a library system.