Idea: Library Management System

Scenario: A basic version of a library management system with all the core functionalities implemented using the terminal.

Functionalities:

1. Member
   - Can log in
   - Can only reserve and borrow novels (if user does not have an unpaid late fee)
   - Can return novels
   - Can pay late fee
   - Can search for novels based on title, author (or both)
   - Can view all novels and available magazines
2. Librarian
   - Can log in
   - Access the library of publications as well as the list of borrowed publications and their corresponding students
   - Add a new publication to the library
   - Process a student's publication return
   - View all students with an unpaid late fee (sorted by fee amount)
3. Student
   - Can log in
   - Has access to reference books and can borrow borrowable publications for longer
   - Reserve + Borrow a publication (if student doesn't have a late fee)
   - Return a publication
   - Can pay late fee
   - View a list of all publications or filter based on type
   - Search for a specific publication based on the title or author (or both)

Expected Output: Students can search for different publications and borrow them for a maximum length (depending on the publication type) and librarian can process returned books as well as notify students that have passed their return limit and calculate the total price to pay. Publications each act as a file that is then given to students (not a real file but still a variable).

First hierarchy: Users can be a librarian (can process returns) or a student (can borrow publications for longer and has access to reference books) or member (regular library user)

Second hierarchy: Publications can be novels (accessible by all users, borrowable for 3 weeks), magazines (not borrowable, can only be consult in the library, but can be searched for through the library system), or reference books (only accessible to students, borrowable for 6 weeks)

Interface: Borrowable

- Abstract method borrow() which adds the book in the library system's borrowed books with the user's details and the date it was borrowed after checking if the user can borrow the book.
- Abstract method calculateFee() which is implemented by borrowable publications to calculate based on the publication type

The interface simplifies the process of borrowing books and directly sets the expected return date.

Methods applying runtime polymorphism:

- Students/members searching for a book based on title, author, or both
- Student overrides the inferface borrow() method and sets the return date to be twice as long (for novels)
- Novel and ReferenceBook override the calculateFee() method

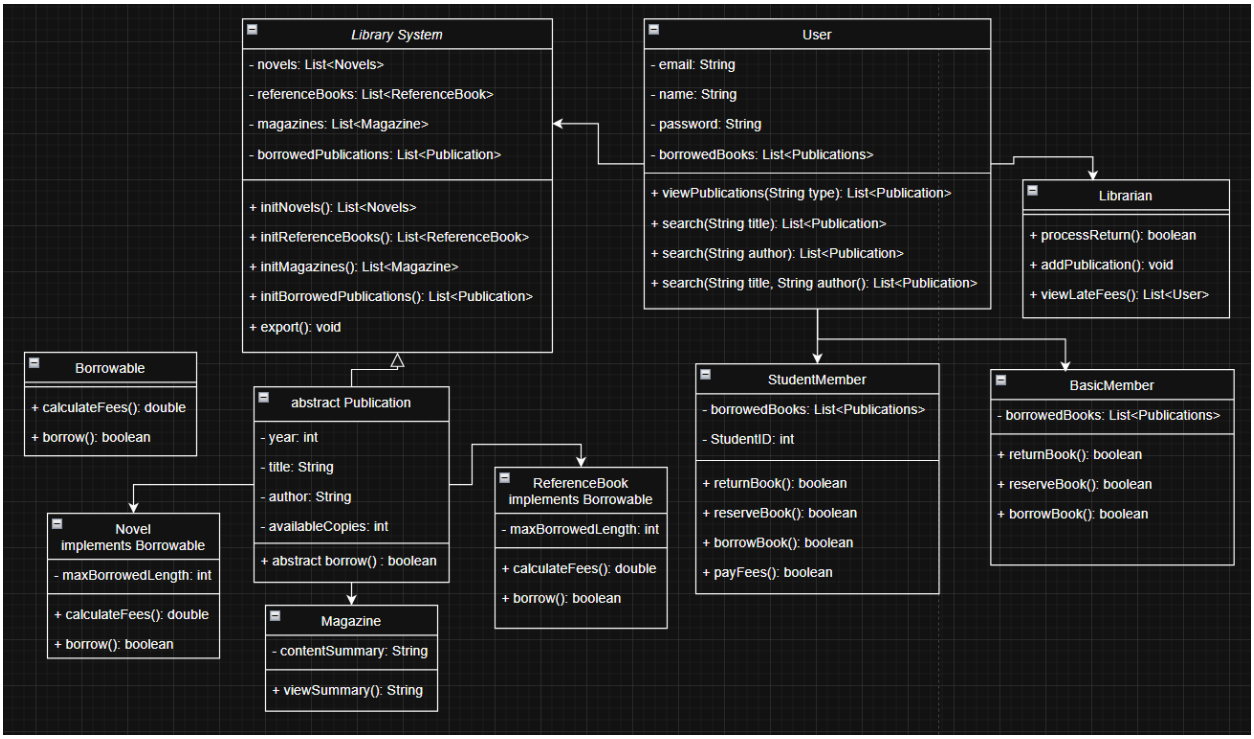
TextIO :

- Used to store a list of all publications of the library in different files based on their types
- Used to keep track of all librarians and all users (and students) that are members of the library

Comparable: implemented in the librarian class to sort all students with unpaid late fees by the amount of the fee (descending)

Comparator: Used to sort searched books by alphabetical order or date published

Class Diagram:



The implementation of the user class, librarian class, and library system will be finished for deliverable 2 while the student class will remain as a boilerplate (only contains variables and method names)

The payFees() method will not be entirely implemented during this project, it will simply mock the redirection to another website to complete the transaction, but the fees of users who return books late will be viewable and will prevent users from borrowing books.