Distributed Systems

Assignment 1: Online Bookstore Project

Objective:
The aim of this assignment is to create an online bookstore application which utilizes client-server architecture and where users can browse, search, borrow and lend books. You're required to implement it **using socket programming and multithreading.**

The application will consist of a server component that manages book inventory and user requests, and a client component that allows users to interact with the bookstore as well as with each other. The application should provide the following list of features.

Features:

1. **Server-Client Communication:** Implement communication between the server and clients using Java SE sockets to handle user requests and responses.
2. **Book Inventory Management:** Allow the server to maintain some sort of a database of available books, including details such as title, author, genre, price, quantity, and list of clients who has this book.
3. **User Authentication:** Implement user authentication mechanisms to ensure secure access to the bookstore's features, such as login and registration. Existing users would need to login, while new users need to register first.
    - To login, the user needs to send his username and password.
    - To register, the user needs to send his name, username, and password.
    - Invalid login/registration scenarios should be handled as follows:
        - if password is wrong, 401 error should appear (unauthorized)
        - if username is not found, 404 error should appear (not found)
        - if a username cannot be used to register a new user because it is already reserved, some custom error should appear.
4. **Browse and Search Books:** Enable users to browse through the bookstore's catalog, search for specific books by title, author, genre, and view detailed information about each book.
5. **Add and Remove Books:** The users can add books which they wish to lend and specify the details of these books. Also, they can remove books which they don't want to lend anymore.

6. **Submit a Request:** A user (as a borrower) can submit a borrowing request to another user (as a lender) and once this request is accepted by the lender, they can start chatting together to see how they're going to communicate with each other.
7. **Accept / Reject a Request:** The user can check the incoming borrowing requests from other users and can accept or reject any of them. **Note:** A user can be both a borrower and a lender at the same time.
8. **Request History:** Provide users with access to their requests' history, allowing them to view and track their status (accepted, rejected or pending).
9. **Library overall statistics:** Provide the admin with the ability to view the overall status of the library, in terms of the current borrowed books, available books, accepted/rejected/pending requests so far.
10. **Error Handling:** Implement error handling mechanisms to deal with various scenarios, such as invalid user inputs.

Notes:

1) **Database/Files Setup:** Design and create a database schema or just files to store book inventory, users information, and requests details. In case of using a database, you need to specify its exact version, so that the TA can properly run your submission on his/her machine.
2) **Testing:**
   a) Test the online bookstore application by running at least 4 instances of the client and verifying that users can successfully do all of the required features.

Additional Features (For students working in teams of 3):

1) Add a review book feature from a specific client or based on specific user defined review format including various data.
2) Calculate on the server, the overall accumulated rating for that book, and any additional review information as per your user-defined review format.
3) Display to any reader the list of books from each genre, listed by their current reviews' calculation. This would be one recommendation, and other recommendations would be based on user's personal genre preferences, or those whom he mostly borrows from.

No graphical user interface is needed. The whole assignment can be run through the command line.

P.S. You must handle any exceptions that may occur. For example, you should handle the issues which may occur if a client disconnects at any moment for any reason.

Guidelines and Submission Details:

1) The project must:
   a) Be developed in the Java SE programming language only.
   b) Compiles against JDK 8 at least.
2) The assignment will be solved in a group of **2 or 3 students** from the same lab or with the same TA. In case you're forming a group of 3, then you **MUST** implement the additional features.
3) If more than 2 team members submit only the features for a team of two, all team members will get **ZERO**.
4) If more than 3 team members submit the assignment, all team members will get **ZERO**.
5) You should submit your assignment as **ONE zip file** with the below naming convention:
   Assign1_GroupNumber_ID1_ID2 (example: Assign1_S1_20116001_20116002)
6) You should submit your **source code along with a document** explaining any decisions or assumptions that you made. This document should also include any steps needed so that the TA can properly run your code. It is your duty to write down the exact steps needed to run the source code properly.
7) You SHOULD NOT copy any code from the internet or from your colleagues. It will be detected and considered as a cheating case.
8) Submission of the assignment will be on Google Classroom through a Google form link that will be shared later.
9) Deadline for the submission is **Tuesday 26th of March 2024**