National Technical University of Athens

School of Electrical and Computer Engineering

Semester 7

Final paper in the Multimedia Technology course

Application Description

As part of the work, an electronic library system will be implemented. The application will allow the administrator to configure and monitor the library material. Users of the system will be able to search for books, apply for their borrowing and add comments and ratings for the available material.

A.1. Logic design and implementation (30%)

The following describes the capabilities that the system administrator or any user may have.

Administrator Functions

- Add, delete and modify a book's information. Relevant information should include: title, author, publisher, ISBN, year of publication, category to which it belongs (eg novel) and number of copies. For each new book the initial score will be 0. In addition, it will be possible to delete a book. In this case, open borrowings for the deleted book must be removed from the system. In addition, the administrator will be able to modify the details of a book, but the user comments and rating are excluded from the modification.

- Add, delete and modify category. The administrator will be able to define new categories, giving the relevant title. In addition, he will be able to modify the title of a category. For simplicity we assume that there are no subcategories. He will also be able to delete a category along with the books that belong to it. In addition, open loans should be updated appropriately for the books that are written off.

- Delete and modify user data. In case of deletion of a user who has open loans, the number of available copies for the books that were borrowed should be renewed accordingly.
- Borrowing management: The administrator will be able to see all open book borrowings. It will also be possible to terminate the borrowing of a book. In this case, the number of available copies should be renewed accordingly.

User Functions

- Registration in the system. The user must enter the desired user name (username), password, first name, last name, TIN and email during registration. Registration is required so that the user can borrow material and add comments and ratings.

- Search for material. Users can search for books based on any combination of the following criteria: title, author, and year of publication.

- Loan of books. In case there are copies available for a book, the user is given the possibility to reserve the material in order to borrow it. The limit of books that can be borrowed is 2 and the borrowing time limit is 5 days. For each book borrowed the user can add comments and rate it on a scale of 1 to 5.

A.2. Saving and retrieving application information (20%)

A corresponding application would actually be supported by a specialized data management system, such as some database. However, as part of the work you will implement a simpler mechanism based on the use of a series of files to store and retrieve all application information.

In this process, the ability provided by Java will be used to convert objects into a series of bytes that can then, among other things, be stored in a permanent storage medium (eg a file in the filesystem). The process of converting objects into bytes is called *object serialization.*
Accordingly, when an object is converted to an array of bytes we can reverse the process to create a copy of the object in the JVM. The process of recovering an object from its "serialization" is called *object deserialization.* More information can be found in the lab slides in the *“Java I/O” section.*

First you should decide the set of files that will be used to store your application objects. We consider special files to be stored in a folder named ***"medialab".*** Then you must implement through the appropriate classes the methods that will allow you to retrieve the objects that the files have as well as refresh the files so that the overall state of the system can be maintained between successive executions of the application.

Next we describe the logic that must be implemented to retrieve and refresh the application data.

- Application initialization: You should retrieve all the information found in the special files and at the same time initialize the corresponding objects (desearilization) in your application.
- Application Execution: The application will use the state information retrieved from program memory during initialization. All operations related to reading, creating, modifying, searching and deleting application data (eg, books, users, loans, comments, rating) will be performed based on the information present in the program's memory.
- Application Termination: Refreshing of special files with system status information will be done only before application termination. The implementation should store in the special files the total state of the application at the time of termination through the serialization of the relevant objects.

Finally, to facilitate the grading process, the implementation of the application should support the connection of a default administrator with elements:

- ***Username:*** medialab
- ***Password:*** medialab_2024

A.3. Creating a GUI (30%)

You should design and implement the appropriate Graphical User Interface (GUI) using the JavaFX framework [1][2].

Note: Below are the basic specifications for the GUI, for all the details of the final implementation you can make any choices you want about the appearance and overall user interaction with the application, without any impact on the final score. For example, you can choose a simple rendering for the various elements or combine various features from JavaFX to create an effect that

corresponds to a modern application. In any case, there is no reason to complicate this particular part of the work.

The graphical interface that will appear at the start of the application should support the following functions:

- Show the 5 books with the highest average score: for each book, the title, author, ISBN, average score and the number of users who have rated it should be displayed.
- Registration of a new user: there should be the corresponding form that will allow the registration of a new user based on the specifications from section A.1. No users with the same email or ADT are allowed. If there is an error, the user should be informed with an appropriate message.
- User login: there should be the corresponding form for logging in a user based on username and password. If there is an error, the user should be informed with an appropriate message.

When a user successfully logs into the application the GUI should support the following features:

- Book search: there should be the corresponding form that will allow searching based on the specifications from section A.1. The results should include: title, author, ISBN, average rating and the number of users who have rated it.
- Book presentation and borrowing: for each result of a search, the user will be able to choose or see all available information (title, author, publishing house, ISBN, year of publication, category, number of copies available, average score, number of users who rated and user comments) or proceed to borrow it based on the specifications from section A.1.
- Overview of open user loans: the material that has been borrowed by the user and has not yet been closed by the administrator should be displayed. The information must include the title, ISBN, date of loan and maximum possible return date.
- Rating and commenting: for each open book loan the user should be able to add a short comment and rate the book on a scale from 1 to 5.

When an administrator successfully logs into the application the GUI should support the following features:

- Book management: the application must present the list of available books by category. It should also be possible for the administrator to define new books, to modify the information of a book
  and delete a book. The implementation must be done according to the corresponding logic from section A.1.
- Category management: the application must present the list of categories. The administrator should also be able to define new categories, change the name of a category and delete categories. The implementation must be done according to the corresponding logic from section A.1.
- User management: the application must present the list of available users. The administrator should also be able to modify user information and delete users from the system. The implementation must be done according to the corresponding logic from section A.1.
- Borrowing management: the application must present all open book borrowings. Also, the administrator should be able to define the end of borrowing a book. In this case we consider that the book has been returned and the number of available copies must be renewed accordingly.

Finally, the implementation of your graphical interface when terminating the application should ensure the renewal of the system information in the special files, according to the procedure described in section A.2.

A.4. Other requirements (20%)

- The implementation should follow the design principles of object-oriented programming (OOP design principles).
- In a class of your choice, every public method it contains must be documented according to the specifications of the javadoc tool [3].

Note: For anything that is not clear from the pronunciation you can make your own assumptions and assumptions. The statement describes the basic requirements for the application, but you can make your own design assumptions in an effort to make the application more realistic without overcomplicating the implementation.

Deliverable

- The project (of the IDE of your choice) with the implementation of the application.

- A short (maximum 2 pages) report containing a general description of the implementation design, mentioning any functionality you have not implemented and any additional assumptions you have made. Under no circumstances should you include code snippets.

References

[1] https://docs.oracle.com/javase/8/javafx/get-started-tutorial/jfx-overview.htm

[2] https://docs.oracle.com/javafx/2/get_started/jfxpub-get_started.htm

[3] https://www.oracle.com/technical-resources/articles/java/javadoc-tool.html