Deadline: Wednesday, June 25, 2025, until 2pm (14:00)

Yaşar University
Spring, 2024 – 2025
SE 2232 - Software System Analysis
Asst. Prof. Dr. Deniz Özsoyeller

Final Project

NOTES:

- ❖ **Submit your Project to Moodle "Final Project" assignment link. Do not send your project via email!**
- ❖ **Late submissions (submissions after the deadline) will not be evaluated.**
- ❖ **You can do the Project individually or as a group that includes at most 2 students.**
- ❖ **The project requires the completion of three parts: 1- Report [55 points], Implementation (Coding) [30 points], 3- Demo [15 points].**

Requirements:

The project should be designed as a Desktop Application (with GUI) developed with Java programming language. Note that you are not designing a Web Application or a Mobile Application. The integrated development environment (IDE) that should be used is either *NetBeans* or *IntelliJ Idea*.

For the database creation, you should use *MySQL*.

Introduction:

In this project, you will develop a software application called "**MyLibrary**" which keeps a list of the books that a user has read or wishes to read.

Each *book* has the following attributes: *bookId* (int), *authorId* (int), *title* (string), *year* (int), *numberOfPages* (int), *cover* (string), *about* (string), *read* (int), *rating* (int), *comments* (string), *releaseDate* (date).

- "cover" attribute shows the path (directory) of the book cover image.
- "about" attribute briefly explains what the book is about.
- "read" attribute shows if a book has been read by the user or not, or is wished to be read by the user. "read=1" means the user has read the book, "read=2" means the user has not read the book, "read=3" means the user wishes to read the book.
- "rating" attribute should be from 1 to 5 and shows how much the user has liked a book. If the user has not read the book or wishes to read the book, then rating can be assigned to 0.
- "comments" attribute is the part where the user can enter extra comments about the book. If the user has no comments, this attribute can be null.
- "releaseDate" attribute shows the the release date of the book that user wishes to read (i.e. when "read=3"). "releaseDate" is null if "read=1" or "read=2".

Each *author* has the following attributes: *authorId* (int), *name* (string), *surname* (string), *website* (string).

- For simplicity, assign the website attribute for the authors as follows: "website-1" for author with "authorId=1", "website-2" for author with "authorId=2", and so on.

** Consider that a book is written by a single author, but an author can have many books.

Each *user* has the following attributes: *userId* (int), *username* (string), *password* (string), *userType* (int).

LOGIN:

The application starts with **a login JFrame** (labeled as LoginFrame) where the user should enter his/her username and password to access the functionalities. There are **two types of users**: *Type-1*: the users who has access to all functions (i.e. functions 1-10 given in "Functions" section below), and *Type-2*: the users who can access only to some functions (i.e. functions 3, 4, 6, 7, 8, 10 given in "Functions" section below).

When the user successfully logs in, the LoginFrame will disappear and instead anotherJFrame (labeled as Type1MainFrame/Type2MainFrame) will appear. MainFrame includes the main functionalities of the application that can be performed by the logged in user type. Depending on the username and password, you should determine the type of the user, and show the appropriate JFrame.

The application's functions are explained in detail below.

Database:

Keep at least 3 database tables:

1. The list of users should be stored in a database table called "*userinfo*".
2. The list of books for all users should be stored in a database table called "*books*".
3. The list of authors should be stored in a database table called "*authors*". The authors that have no books in the library should not be in the "authors" database table.

Functions:

1. **Add a book** to the database entering all attributes of a book from a JTextFields. While adding a book, the author name and surname should be also received by the user to check if the "authors" database table has the same author or not. If it has, then the authorId should be retrieved from the "authors" table. Otherwise, a new authorId should be assigned and a new author entry should be inserted to the "authors" table. Note that this new authorId should also be used when entering the new book information. Whenever a new book is added to the user's library, you should automatically increment the bookId by 1.
2. **Delete a book** from the database with the given bookId from a JTextField. This task should delete the book with the given id from the database. A message dialog box should be displayed to show that this task is completed. If the deleted book's author does not have any other books in the user's book list, then the corresponding author entry should also be deleted from the "authors" table.
3. **Display Book Information:** Display the related information of a book with the given bookId from a JTextField.
4. **Search an author by name** and display the author's information.
5. **Display-Edit-Update:** After displaying the related information of a book with the given bookId from a JTextField, edit this retrieved information and update the database.
6. **Display the list of favorite books** of a user. These are the books that the user has read and rated as either 4 or 5.
7. **Display the list of favorite authors** of a user. For an author to be favorite, at least 3 books of that author should be in the user's library.
8. **Display the list of books** that the user has not read yet. Display all of the related attributes of found books.

9. When the application is started, **notify the user about the books** that are on user's wish list (i.e. when "read=3") and will be released within 1 week.
10. **Display the image of a book's cover** with the given bookId from a JTextField. The directory of the image should be kept in the *cover* attribute of "*books*" database table.
    - For locations, create 5 sample image files (in jpeg format) in your Project folder. An example is shown on right:
    - You can create your own simple image files using a Paint program.
    - The file name for book-1 is Book1.jpg, for book-2 is Book2.jpg, and so on.

Note: How you design the GUI (Graphical User Interface) of your project is optional. Make use that it is easy to use and the components on the frames are organized well.

Report:

- You should use the project template in Moodle to prepare your project report. The report consists of eight sections to be completed.
- Note that you should use Visual Paradigm to draw the UML Diagrams required in the project.

Presentation:

- Prepare a demonstration (demo) video of your project in English.
- The video should only show the application (the codes, database tables and GUIs). You do not need to explain any section from your Report.
- Briefly introduce yourself and the goal of the project.
- Briefly explain each of your class and its purpose.
- Test each function/feature of your application and explain in which class the tested function is implemented in your project.
- **Duration:** 5-8 minutes.
- **Format:** mp4
- If the project is completed as a group, then each group member should present, and the presentation time should be divided equally amongst the group members.
- Check to see if your demo video is working or not before submitting it. The videos that cannot be opened will not be evaluated.

Project Submission:

Submit the parts below in a single zip/rar compressed folder to Moodle Assignment link "SE 2232 Final Project".

1. **Report file (Format: .pdf/.doc/.docx)**
2. **Demo file (Format: mp4)**
3. **Java codes (Format: .java)**
4. **Database files**