TOSHKENT SHAHRIDAGI INHA UNIVERSITETI
INHA UNIVERSITY IN TASHKENT

Module Name:	Application Programming in Java
Module Code:	SOC2030
Project Type:	Group Project (4-6 students)
Assigment Link:	https://classroom.github.com/g/MrMMp3f_
Date Set:	26.11.2020
Deadline:	10.01.2021 (midnight)
Weighting:	20%

Instructor:	Dr. Sarvar Abdullaev
Email:	s.abdullaev@inha.uz
Zoom:	https://us02web.zoom.us/j/88429816745

Learning Objectives :

1. To build strong programming skills using Java from theories of computer science.
2. To build design/implementation skills of advanced applications under Java environments.
3. To apply object-oriented concepts to the development of desktop applications.
4. To find elegant solutions to complex technical problems.
5. To obtain skills necessary to coordinate technical tasks within the team.

Things to Submit:

1. **ALL SOURCE CODE** and **REPORT** pushed to your GitHub repository.
2. **ALL SOURCE CODE** and **REPORT** submitted to group assignment in http://online.inha.uz.

Project Overview

At present, IUT's library is using either manual journal-based system or a spreadsheet like Excel to manage their day to day activity. Most of the things such as record of students who've issued the book, no. of books in the library, fine of students, etc. are recorded using pen-paper system. This traditional system requires a lot of time and manpower for performing simple operations in the library.

In this project, you are required to develop a desktop application using Java technologies and Object-Orient Programming concepts to help librarians accomplish their daily job. ***Library Management System*** (LMS) is a software solution that helps both students and library manager to keep a constant track of all the books available in the library. It allows both the admin and the student to search for the desired book. The main feature of this system is that all the books available in the library can be displayed in a list so that students need not roam through the entire library to find a book. Additionally, the application effectively maintains the details of users/students to whom books have been issued; it also records the issued date and return date.

Below are common features of a typical LMS:

- Add, modify and delete users with 3 roles (Student, Librarian and Administrator) into the database.
- Authenticate and authorize users to accomplish certain actions based on their role
- Add, modify and delete book details into the database.
- Search available books by their title, subject, author, ISBN and publish date.
- Issue book loans for a certain period (e.g. 1 semester, 1 month, 1 day) if they are available.
- Reserve borrowed books beforehand.
- Issue fines for book loan overdue.

As mentioned above, every user must be assigned to a role. For simplicity sake, assume that one user can be assigned to a single role only. Below is the list of roles and corresponding actions they can do:

- **Administrators –** users that are responsible for creating accounts for librarians and students as well as adding information about books.
  - Login/Logout from the system
  - View/Add/Modify/Delete Librarian
  - View/Add/Modify/Delete Student
  - View/Add/Modify/Delete Books
- **Librarians –** users that are responsible for adding and modifying books, book items, and users. The Librarian can also issue and return book items. For books returned after due date, the Librarian issues a fine to a responsible student.
  - Login/Logout from the system
  - View/Add/Modify/Delete Students
  - View/Add/Modify/Delete Books
  - View books borrowed by a student
  - View students with an overdue

- View all books and filter them by borrowed status, title, subject, author, ISBN and publish date.
    - View monthly report on books borrowed from the library
    - Issue a book loan for limited period
    - Return a borrowed book.
    - Issue a fine to a student who returned book after due date.
    - Block a student from borrowing a book.
- **Students –** users that can search the catalog, as well as check-out, reserve and return a book.
    - Login/Logout from the system
    - View his/her personal details as well as books he/she borrowed
    - View his/her current fine
    - View all books and filter them by borrowed status, title, subject, author, ISBN and publish date.
    - Reserve borrowed books beforehand.

LMS should contain a user-friendly GUI where users login first and depending on their role, a corresponding window is shown. In order to maintain data persistency, LMS should be connected to a database and manage its data via DBMS. Some key operations working with collections such as sorting and filtering records should also be supported by the LMS. Moreover, LMS may obtain useful information about a book from Internet (e.g. https://isbnsearch.org/isbn/[ISBN_number]) by providing its ISBN number. You can also add extra functionality of your own choice to LMS beyond its basic features.

In order to implement above tasks, you can use only following technologies and nothing more:

- Java SDK 11 or higher
- Java DB and JDBC for Database Connectivity
- Apache Derby for Database Management System
- Java Swing or JavaFX for GUI
- Git and Github.com for Version Control

You can use an IDE (e.g. IntelliJ, NetBeans, Eclipse, etc) of your own choice.

You are NOT ALLOWED:

- You are NOT ALLOWED to submit existing LMS software available on Internet.
- You are NOT ALLOWED to submit code written by someone else.
- You are NOT ALLOWED to collaborate between groups and use each other's code.
- You are NOT ALLOWED to use other languages or technologies, but Java.
- You are NOT ALLOWED to use third-party components/controls/frameworks other than covered in the lecture materials.

If you break any of the above requirements, your project will be marked 0 and you will be reported to Academic Affairs for further disciplinary action.

You are ALLOWED:

- You are ALLOWED to use architectural patterns, design patterns and idioms or other template solutions for recurring problems which were not covered in lecture materials.
- You are ALLOWED to use code snippets, workarounds or tricks written in public forums such as StackOverflow, StackExchange, GitHub Gists, etc, **BUT** you should cite the exact URL to a corresponding code snippet in a comment section of your code.

Disclaimer:

Generally, I do not need a presentation about the project. However,

- I reserve the right to call the group for a short viva in case if I have suspicion about the quality of work submitted. If group fails to explain how Java application has been implemented, all of its members **receive 0 for the project**.
- I reserve the right to grade each member of the group differently if I have a suspicion on his/her contribution to the project. I may review the history of commits in project's Github. If I find a member who has not contributed at all to the project, I will call this member for a short viva regarding his/her contribution to the project. If member fails to show up, he/she will **receive 0 for the project**.

Evaluation Criteria:

Section 1: Java Application [80 points]

Your Java application will be evaluated on how well key functionality of LMS has been implemented and if it works as expected. In particular, it will be evaluated according to following criteria:

1. **Main Functionality (30 points)**
   LMS must support all of the above mentioned functionality for each type of user. It has to be bug-free and require no additional dependencies other than JVM and Java DB for its execution.

2. **Use of GUI (10 points)**
   LMS must have an intuitive and usable GUI with all necessary functionality implemented through graphical controls provided by Swing or JavaFX. User input should be validated before being sent to database.

3. **Use of Database (10 points)**
   LMS must manage all of its data via DBMS and only Java DB can be used for this purpose. All necessary SQL for creating the database and populating it with data should be included in the project folder.

4. **Use of Best Practices (10 points)**
   LMS must apply design patterns, SOLID principles, coding conventions and appropriate commenting and indentation to make software reusable, extendable and maintainable. You are free to use any of 23 design patterns described by Gang of Four, or other best practices described in relevant literature as many times as it requires.

5. **Use of Java Collections/Streams/Lambdas (10 points)**
   LMS must perform various operations with collections using Java streams, generics and/or lambda functions. For example, records can be transformed, filtered, sorted or aggregated. You are free to find more applications of Java streams in this project.

6. **Use of Multithreading OR Network Programming (10 points)**
   LMS may use Java's multithreading and networking capabilities to implement some of the additional functions such as obtaining more information about given book from Internet, using Producer/Consumer relationship to update parallel threads, using concurrent collections and thread synchronization to prevent duplicate borrowing of same book items. You are free to find more applications Java's multithreading and networking capabilities in this project.

Section 2: Report [20 points]

Along with your Java application, you are expected to submit a report describing how you implemented the project with the team. It **MUST** be **PRINTED OUT**, **BOUND and SUBMITTED** to the instructor the first lecture **AFTER** the deadline. Your report should consist of following parts:

1. **Cover Page (2 points)**
   An appropriately filled cover page **MUST** include following information:

   - Team name
   - Section number, student ID and full name of all team members
   - Project's GitHub URL
   - Submission date

   Important: Report without an appropriate cover page will not be evaluated.

2. **User Accounts (2 points)**
   You must enclose login and password for at least 3 user accounts in your report: Administrator's Account, Librarian's Account and Student's Account.

3. **Description of Implemented Functionality (8 points)**
   This section must contain brief description of each function that is implemented in your project including screenshots for each use case. Below each screenshot, provide user instructions that should be carried out in order to complete a particular task in your LMS. You should also describe the contribution of each member to the project in terms of which features were implemented by whom. Reflect on your overall experience of developing this solution within your team and discuss about challenges you have faced in the process.

4. **Entity-Relationship Diagram (3 points)**
   You must include Entity-Relationship diagram which describes your database schema. You should discuss benefits and limitations of your current database model in terms of what data can be easily inserted/updated/deleted/retrieved from this database model and what cannot be done so.

5. **UML Diagrams (5 points)**
   You must include UML diagrams such as Class Diagram and Interaction Diagram which describe the conceptual design of your application. You should explain your implementation of design patterns in these diagrams and discuss their benefits and limitations to your project's reusability, maintainability, extensibility, and readability.