-

Assessment Details and Submission Guidelines	
Trimester	T1, 2024
Unit Code	BN231
Unit Title	Software Development Skills and Tools
Assessment Type	Assignment 2 (Group)
Assessment Title	Object Oriented Programming
Purpose of the assessment (with ULO Mapping)	a. Demonstrate the skills and knowledge in object-oriented programming (OOP) in solving a contemporary industry problem. b. Design and develop an OOP solution to solve a complex industry problem using appropriate tools and platforms. c. Implement, deploy, and troubleshoot OOP solution in an ICT application with sound knowledge of programming tools and technologies. d. Evaluate and recommend appropriate OOP solutions to solve real life problems with good understanding of industry best practices and technological landscapes. e. Communicate and contribute to a professional software development team with good understanding of teamwork and best ethical practices.
Weight	25%
Total Marks	60 marks
Word Limit	N/A
Due Date	Week 11 – Sunday, 2nd May 2024 at 11:59pm
Submission Guidelines	• All work must be submitted on Moodle by the due date along with a completed Assignment Cover Page. • The assignment must be in MS Word format, 1.5 spacing, 11-pt Calibri (Body) font and 2 cm margins on all four sides of your page with appropriate section headings. • Reference sources must be cited in the text of the report, and listed appropriately at the end in a reference list using IEEE referencing style.
Extension	• If an extension of time to submit work is required, a Special Consideration Application must be submitted directly on AMS. You must submit this application three working days prior to the due date of the assignment. Further information is available at: • https://www.mit.edu.au/about-us/governance/institute-rules-policies-and-plans/policies-procedures-and-guidelines/assessment-policy
Academic Misconduct	• Academic Misconduct is a serious offence. Depending on the seriousness of the case, penalties can vary from a written warning or zero marks to exclusion from the course or rescinding the degree. Students should make themselves familiar with the full policy and procedure available at: https://www.mit.edu.au/about-mit/institute-publications/policies-procedures-and-guidelines/AcademicIntegrityPolicyAndProcedure. For further information, please refer to the Academic Integrity Section in your Unit Description.

Assignment Instructions

Due: End of week 11. Check the exact date and time in the submission area on the unit website.
Weighting: 25%

To be Submitted:

1. A zip file of the NetBeans Project with your assignment 2 application.
2. A Word document with your report (that includes testing)

Note that:

1. Your program must be submitted as a NetBeans project compressed in one file or the marker will not be able to test your work and no marks will be awarded.
2. **No marks will be awarded if the assignment does not compile and run.** Markers must be able to run and test your code to mark it. It is therefore essential that you develop the code **incrementally** and maintain a backup of your last working phase. If your last phase does not compile and run, then you must submit the previous phase that does compile and run. In addition, always check that you have submitted the correct work. (A good test is to download your submission and check it on a different machine.)

1. Introduction

In this assignment, you are to develop GUI-based library system in Java. **A phased implementation approach is recommended.** The application is to be conformant with the class diagram provided in Figure 1.

2. Case Study: *University Library System*

This case is a simplified version of a new system for the University Library. Of course, the library system must keep track of books. Information is maintained both about book titles and the individual book copies. Book titles maintain information about title, author, publisher, and catalog number. Individual copies maintain copy number, edition, publication year, ISBN, book status (whether it is on the shelf or loaned out), and date due back in.

The library also keeps track of its patrons. Because it is a university library, there are several types of patrons, each with different privileges. There are faculty patrons, graduate student patrons, and undergraduate student patrons. Basic information about all patrons is name, address, and telephone number. For faculty patrons, additional information is office address and telephone number. For graduate students, information such as graduate program and advisor information is maintained. For undergraduate students, program and total credit hours are maintained.

The library also keeps information about library loans. A library loan is a somewhat abstract object. A loan occurs when a patron approaches the circulation desk with a stack of books to check out. Over time a patron can have many loans. A loan can have many physical books associated with it. (And a physical book can be on many loans over a period of time. Information about past loans is kept in the database.) So, in this case, an association class should probably be created for loaned books.

If a patron wants a book that is already checked out, the librarian can put that title on reserve for the patron. This is another class that does not represent a concrete object. Each reservation is for only one title and one patron. Information such as date reserved, priority, and date fulfilled is maintained. When a book is fulfilled, the system associates it with the loan on which it was checked out.

Librarians (and the clerks in the library) can search for book titles to see whether a book is available. They can also reserve a title if all copies are checked out. When patrons bring books to the circulation desk, a clerk checks out the books on a loan. Clerks also check books in. When books are dropped in the return slot, clerks check in the books. Stocking clerks keep track of the arrival of new books.

Librarians have their own activities. They will print reports of book titles by category. They also like to see all overdue books. When books get damaged or destroyed, managers delete information about book copies. Librarian also like to see what books are on reserve.
shown in the UML Diagram below:

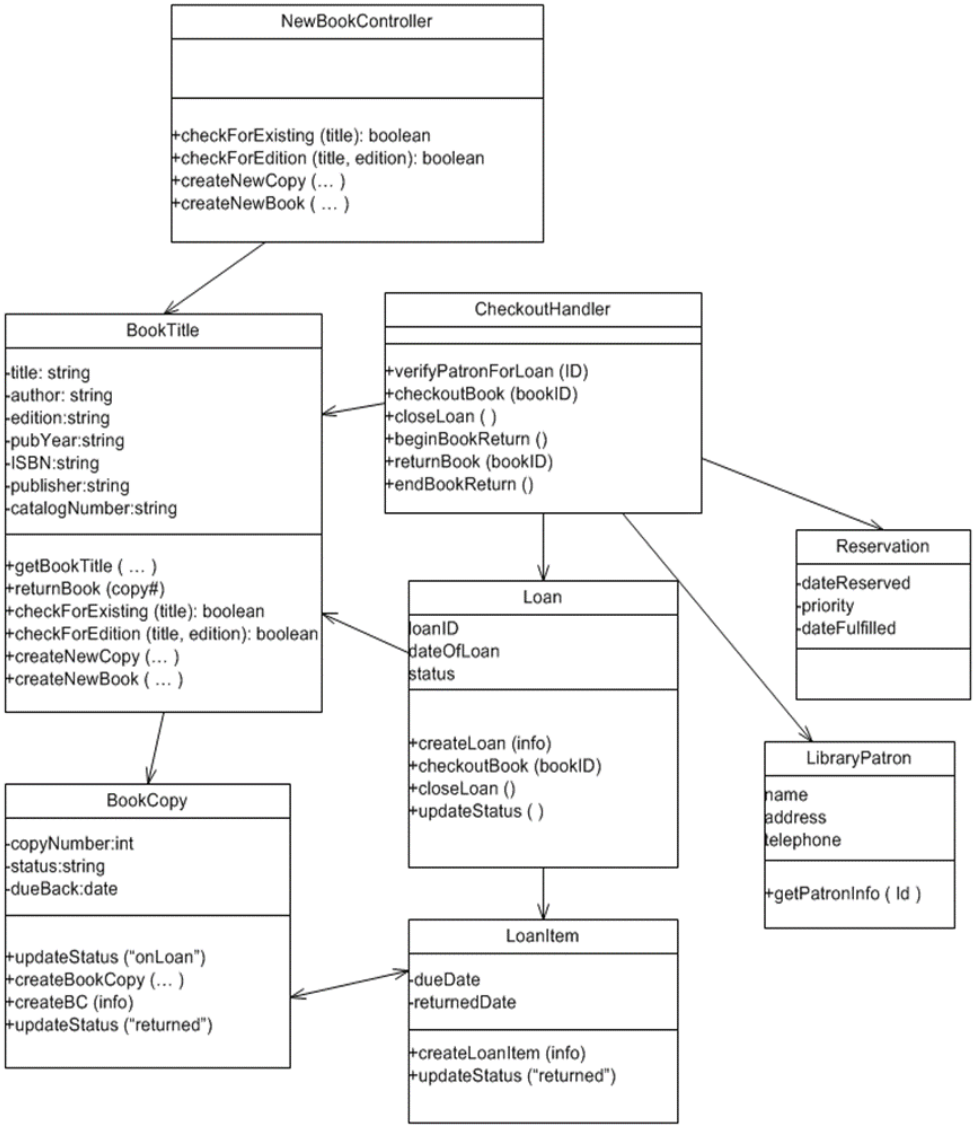

Figure 1. Class Diagram (you must add the subclasses of LibraryPatron)

Your task is to develop a Java application (GUI) that conform to the design of the UML class diagram with the following requirements:

a) Must implement MVC design pattern by creating 3 packages:
   - model for Data, e.g. BookTitle, LibraryPatron, …
   - view for the GUI's (e.g. BookFrame, PatronFrame, ..)
   - controller, e.g. NewBookController, CheckOutHandler, …

b) Data are saved inside the program using (Java) ArrayLists. It is an advantage to connect your application to a DBMS like MySQL to manage your data. But you are not required to do this in this assignment.

c) The application must start with a main switchboard that contains a group of Buttons for all the functionalities and users can perform one functionality by clicking the corresponding button.

3. Submission

You are to submit two separate files:

1. Your zipped NetBeans project folder.
2. Report.docx. You must complete the report template supplied on the unit website. As you will see, this file is to contain the following Sections:
    a. Limitations
    b. Known bugs
    c. Test plan
    d. Test results

    Student name, student ID number, unit name and unit code are to be included on the title page.
    The limitations section is to specify any limitations that your program has in terms of calculations and data validation.
    The test plan is to contain a comprehensive list of program functionality to be tested, the input values (and patient record data) to be used to test each item of functionality, the expected output from the test and the actual output from the test.
    The test results section is to contain screenshots to demonstrate that the program generates the actual outputs shown in the test plan.

4. Marking Criteria

	Functionality (and correct use of programming constructs) Follows Design specified	**Code Documentation & Layout**	**Good coding practices (Naming conventions, meaningful names, efficient code**	**Total**
Model Classes	11	2	2	**15**
View Classes	8	1	1	**10**
Controller Classes	8	1	1	**10**
System Integration	7			**7**
Testing (3 Junit Tests)	9		1	**10**
Report	Including Assumptions, Limitations, bugs and screenshots of the testing outcome.			**8**
Total				**60**

Note:

1. **The final mark will be converted to a mark out of 25.** This assessment item is worth 25% of your marks.
2. It is your responsibility to ensure that source code files are included in your NetBeans project submission as well as your report.doc. **Note: the report file must not be in the zip file. It must be uploaded as a separate .doc file to go through Turnitin.**
3. Marks will be based on working code. **No marks will be awarded for code that does not compile.**
4. **To be awarded marks you must also follow the design specified (i.e. adhere to the design in the UML class diagram).**
5. If you are aware of a bug in your code that you don't fix before submission, document this in the bugs section of the report. If the marker finds an undocumented bug you will lose the mark for the bug and also for poor testing and documentation. If it is documented you will only lose the mark for the bug.
6. Marks will be deducted for late submission. (-5% of the total possible marks for the assignment is deducted for each day late.)

Remember to:

1. Document the code – see the guide about code documentation on the unit website. You must include a comment before every method and class header as well as the code in the body of methods when appropriate. The start of each class should include a description of the class and the author. Before each method there should be a comment to describe what the method does (not how), the parameters and any return values**. You must use Javadoc comments to add this information to your code in this assignment.**
2. Test the code thoroughly as you develop each phase.
3. Complete the report with screenshots of your test results.
4. Follow good programming practices – see the coding guidelines document on the unit website.