LIBRARY MANAGEMENT SYSTEM

PROJECT REPORT

18CSC202J/ 18AIC203J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY

(2018 Regulation)

II Year/ III Semester

Academic Year: 2022 -2023

By

MOHD TAUSEEN IQBAL (RA2111028010066)

KAUSTUBH SHATDHAR (RA2111028010093)

Under the guidance of

Dr. L.N.B.Srinivas

Assistant Professor

Department of Computational Intelligence

FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

NOVEMBER 2022

BONAFIDE

This is to certify that **18CSC202J - OBJECT ORIENTED DESIGN ANDPROGRAMMING LABORATORY project report** titled "**LIBRARY MANAGEMENT SYSTEM"** is the bonafide work of **MOHD TAUSEEN IQBAL(RA2111028010066) KAUSTUBH SHATDHAR(RA2111028010093)** who undertook the task of completing the project within the allotted time.

Signature of the Guide	**Signature of the II Year Academic Advisor**
Dr. L.N.B.Srinivas	----------
Assistant Professor	**Professor and Head**
Department of NWC,	Department of NWC
SRM Institute of Science and Technology	SRM Institute of Science and Technology

About the course:-

18CSC202J/ 8AIC203J - Object Oriented Design and Programming are 4 credit courses with **L T P C as 3-0-2-4** (Tutorial modified as Practical from 2018 Curriculum onwards)

Objectives:

The student should be made to:

- Learn the basics of OOP concepts in C++
- Learn the basics of OOP analysis and design skills.
- Be exposed to the UML design diagrams.
- Be familiar with the various testing techniques

Course Learning Rationale (CLR): The purpose of learning this course is to:

CLAP-1	5=(2(E-lab Completion) + 2(Simple Exercises)(from CodeZinger, and any other coding platform) + 1(HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-2	7.5=(2.0(E-lab Completion)+ 2.0 (Simple Exercises)(from CodeZinger, and any	Elab test

1. Utilize class and build domain model for real-time programs
2. Utilize method overloading and operator overloading for real-time application development programs
3. Utilize inline, friend and virtual functions and create application development programs
4. Utilize exceptional handling and collections for real-time object-oriented programming applications
5. Construct UML component diagram and deployment diagram for design of applications
6. Create programs using object-oriented approach and design methodologies for real-time application development

Course Learning Outcomes (CLO): At the end of this course, learners will be able to:

1. Identify the class and build domain model
2. Construct programs using method overloading and operator overloading
3. Create programs using inline, friend and virtual functions, construct programs using standard templates
4. Construct programs using exceptional handling and collections
5. Create UML component diagram and deployment diagram
6. Create programs using object oriented approach and design methodologies

Table 1: Rubrics for Laboratory Exercises

(Internal Mark Splitup:- As per Curriculum)

	other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	
CLAP-3	7.5=(2.0(E-lab Completion(80 Pgms)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	**2 Mark -** E-lab Completion **80 Program** Completion from 10 Session (Each session min 8 program) **2 Mark -** Code to UML conversion GCR Exercises **3.5 Mark - Hacker Rank** Coding challenge completion
CLAP-4	5= 3 (Model Practical) + 2(Oral Viva)	• **3 Mark** – Model Test • **2 Mark** – Oral Viva
Total	25	

COURSE ASSESSMENT PLAN FOR OODP LAB

S.No	List of Experiments	Course Learning Outcomes (CLO)	Blooms Level	PI	No of Programs in each session
1.	Implementation of I/O Operations in C++	CLO-1	Understand	2.8.1	10
2.	Implementation of Classes and Objects in C++	CLO-1	Apply	2.6.1	10
3,	To develop a problem statement. 1. From the problem statement, Identify Use Cases and develop the Use Case model. 2. From the problem statement, Identify	CLO-1	Analysis	4.6.1	Mini Project Given

	the conceptual classes and develop a domain model with a UML Class diagram.				
4.	Implementation of Constructor Overloading and Method Overloading in C++	CLO-2	Apply	2.6.1	10
5.	Implementation of Operator Overloading in C++	CLO-2	Apply	2.6.1	10
6.	Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams and Collaboration diagrams	CLO-2	Analysis	4.6.1	Mini Project Given
7.	Implementation of Inheritance concepts in C++	CLO-3	Apply	2.6.1	10
8.	Implementation of Virtual function & interface concepts in C++	CLO-3	Apply	2.6.1	10
9.	Using the identified scenarios in your project, draw relevant state charts and activity diagrams.	CLO-3	Analysis	4.6.1	Mini Project Given
10.	Implementation of Templates in C++	CLO-3	Apply	2.6.1	10
11.	Implementation of Exception of Handling in C++	CLO-4	Apply	2.6.1	10
12.	Identify the User Interface, Domain objects, and Technical Services. Draw the partial layered, logical architecture diagram with UML package diagram notation such as Component Diagram, Deployment Diagram.	CLO-5	Analysis	4.6.1	Mini Project Given
13.	Implementation of STL Containers in C++	CLO-6	Apply	2.6.1	10
14.	Implementation of STL associate containers and algorithms in C++	CLO-6	Apply	2.6.1	10
15.	Implementation of Streams and File Handling in C++	CLO-6	Apply	2.6.1	10

LIST OF EXPERIMNENTS FOR UML DESIGN AND MODELLING:

To develop a mini-project by following the exercises listed below.

1. To develop a problem statement.
2. Identify Use Cases and develop the Use Case model.
3. Identify the conceptual classes and develop a domain model with UML Class diagram.
4. Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams.
5. Draw relevant state charts and activity diagrams.
6. Identify the User Interface, Domain objects, and Technical services. Draw the partial layered, logical architecture diagram with UML package diagram notation.

Suggested Software Tools for UML:

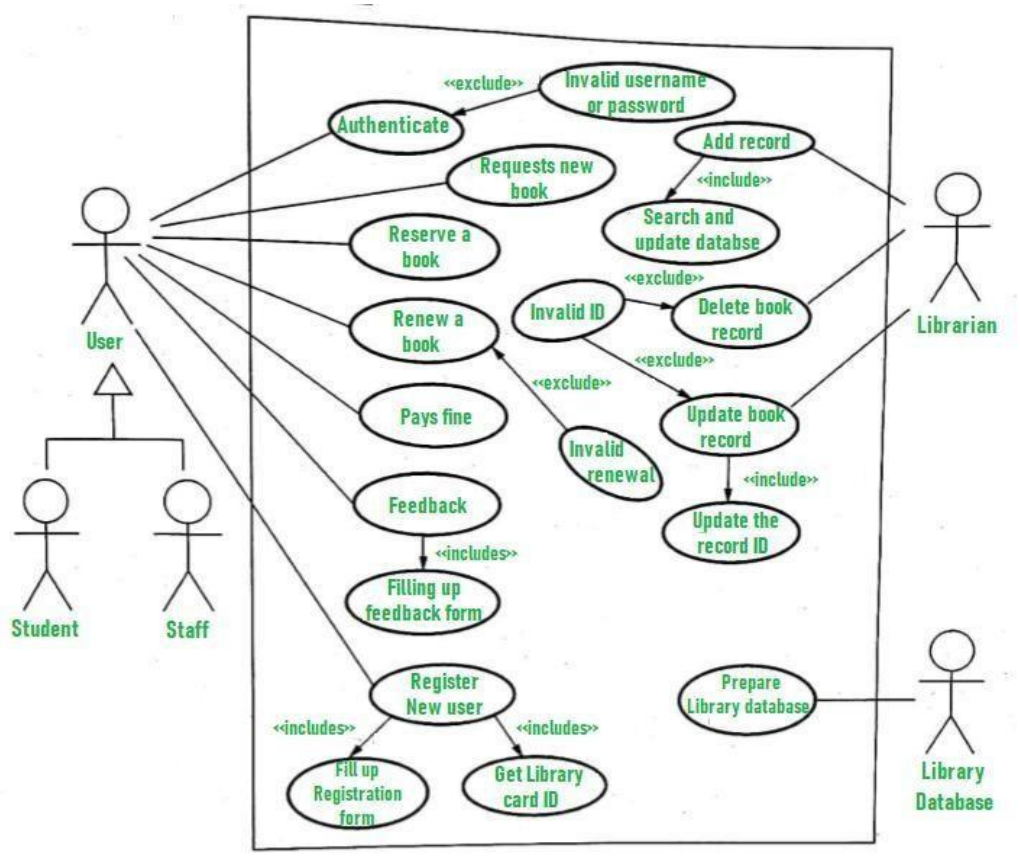
StarUML, Rational Suite, Argo UML (or) equivalent, Eclipse IDE and Junit

ABSTRACT

The UML Diagrams for Library Management System are used to represent the Library Management system as well as its primary users, roles, activities, artifacts, or classes. The UML Diagrams are created to easily understand, update, maintain, and document Library management system information. UML diagrams for Library Management system were used to visualize the project. It can be done before the development begins or to document its progress once it is completed. However, Library Management System UML Diagrams can be used in any sector, not only in software engineering. Its overall objective is to help teams or developers visualize what a project is or how it will work. In this project multiple UML diagram for Library Management system are represented which will help developers to develop link between different classes , and connect backend to frontend as well as for creating multiple web pages requires for different processes and time the occurrence of web pages as per the processes , interact with various people activities and represent how web pages will communicate with each other as a part of a system . These UML also represents and provide all necessary information for people to properly hide and encapsulate data and verify data as per needs. The UML Diagrams are created to easily understand, update, maintain, and document Library Management system information.

Table of contents

USE CASE DIAGRAM :



Purpose:

Use-case diagrams **describe the high-level functions and scope of a system**. These diagrams also identify the interactions between the system and its actors. The use cases and actors in use-case diagrams describe what the system does and how the actors use it, but not how the system operates internally.

ACTORS:

- **User:** It is the person who through internet is accessing library management system.
- **Student:** User who already registered himself.
- **Staff:** User who already registered himself.

- **Librarian:** Librarian adds the records in the library database about each student or user every time issuing the book or returning the book, or paying fine.

- **Library Database:** All the data is stored in library database such as date of issue of book, name of student/staff who issued the book, their registration number, fine amount etc.

Process:

User who registers himself as a new user initially is regarded as staff or student for the librarysystem.

- For the user to get registered as a new user, registration forms are available that is needed to be fulfilled by the user.
- After registration, a library card is issued to the user by the librarian. On the library card, an ID is assigned to cardholder or user.

After getting the library card, a new book is requested by the user as per there requirement.
After, requesting, the desired book or the requested book is reserved by the user that means no other user can request for that book.
Now, the user can renew a book that means the user can get a new due date for the desired book if the user has renewed them.
If the user somehow forgets to return the book before the due date, then the user pays fine. Or if the user forgets to renew the book till the due date, then the book will be overdue and the user pays fine.
User can fill the feedback form available if they want to.
Librarian has a key role in this system. Librarian adds the records in the library database about each student or user every time issuing the book or returning the book, or paying fine.
Librarian also deletes the record of a particular student if the student leaves the college or passed out from the college. If the book no longer exists in the library, then the record of the particular book is also deleted.

CLASS DIAGRAM :

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.
Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Purpose of Class Diagrams:

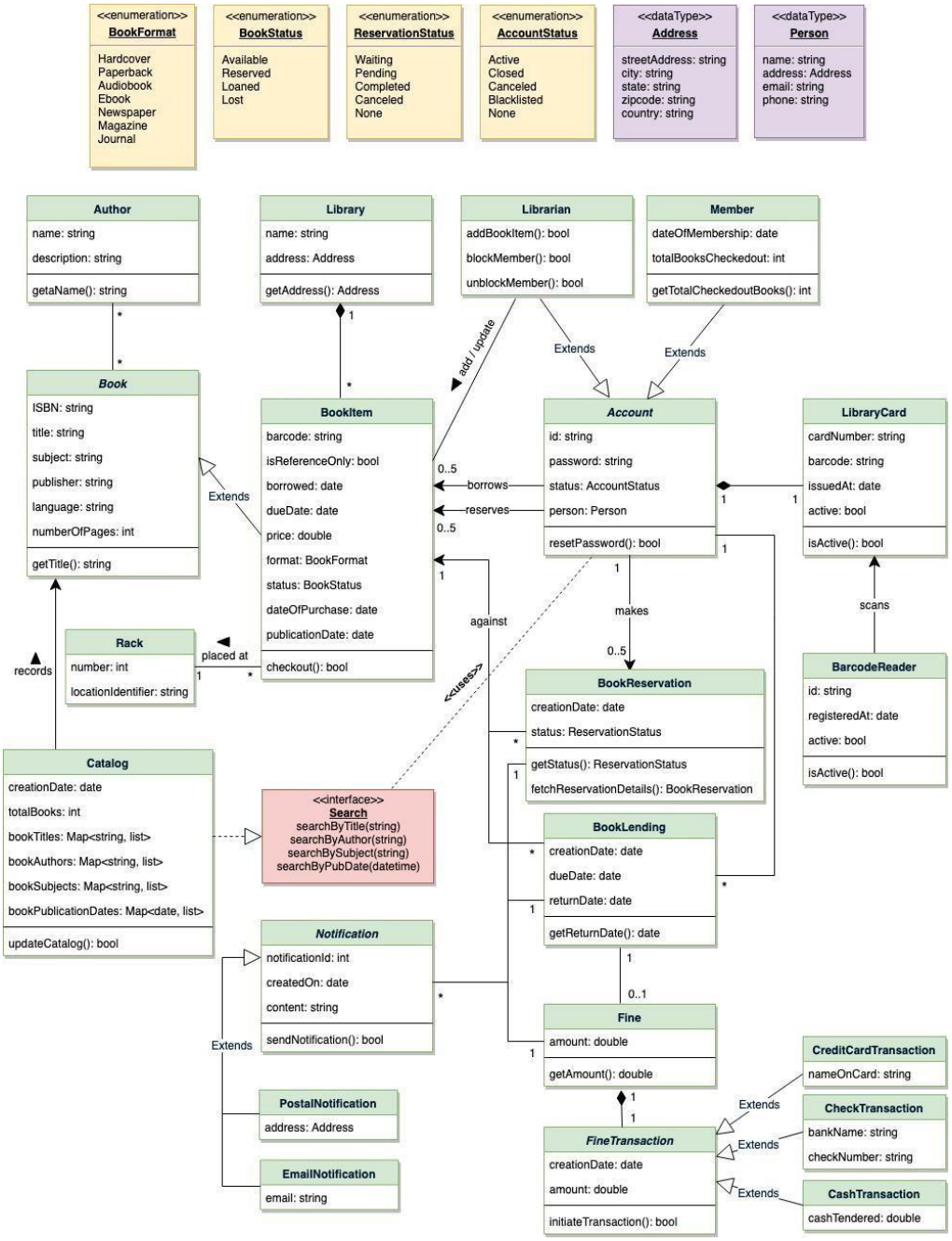
The main purpose of class diagrams is to build a static view of an application. It is the only diagram that is widely used for construction, and it can be mapped with object-oriented languages. It is one of the most popular UML diagrams. Following are the purpose of class diagrams given below:

1. It analyses and designs a static view of an application.
2. It describes the major responsibilities of a system.

3. It is a base for component and deployment diagrams.
4. It incorporates forward and reverse engineering.

Classes:

- **Library:** The central part of the organization for which this software has been designed. It has attributes like 'Name' to distinguish it from any other libraries and 'Address' to describe its location.
- **Book:** The basic building block of the system. Every book will have ISBN, Title, Subject, Publishers, etc.
- **BookItem:** Any book can have multiple copies, each copy will be considered a book item in our system. Each book item will have a unique barcode.
- **Account:** We will have two types of accounts in the system, one will be a general member, and the other will be a librarian.
- **LibraryCard:** Each library user will be issued a library card, which will be used to identify users while issuing or returning books.
- **BookReservation:** Responsible for managing reservations against book items.
- **BookLending:** Manage the checking-out of book items.
- **Catalog:** Catalogs contain list of books sorted on certain criteria. Our system will support searching through four catalogs: Title, Author, Subject, and Publish-date.
- **Fine:** This class will be responsible for calculating and collecting fines from library members.
- **Author:** This class will encapsulate a book author.
- **Rack:** Books will be placed on racks. Each rack will be identified by a rack number and will have a location identifier to describe the physical location of the rack in the library.
- **Notification:** This class will take care of sending notifications to library members.



Description:

Any library member should be able to search books by their title, author, subject category as well by the publication date. Each book will have a unique identification number and other details including a rack number which will help to physically locate the book. There could be more than one copy of a book, and library members should be able to check-out and reserve any copy. We will call each copy of a book, a book item. The system should be able to retrieve information like who took a particular book or what are the books checked-out by a specific library member. There should be a maximum limit (5) on how many books a member can check-out. There should be a maximum limit (10) on how many days a member can keep a book. The system should be able to collect fines for books returned after the due date. Members should be able to reserve books that are not currently available. The system should be able to send notifications whenever the reserved books become available, as well as when the book is not returned within the due date. Each book and member card will have a unique barcode. The system will be able to read barcodes from books and members' library cards.

SEQUENCE DIAGRAM :

Purpose of Sequence Diagrams:

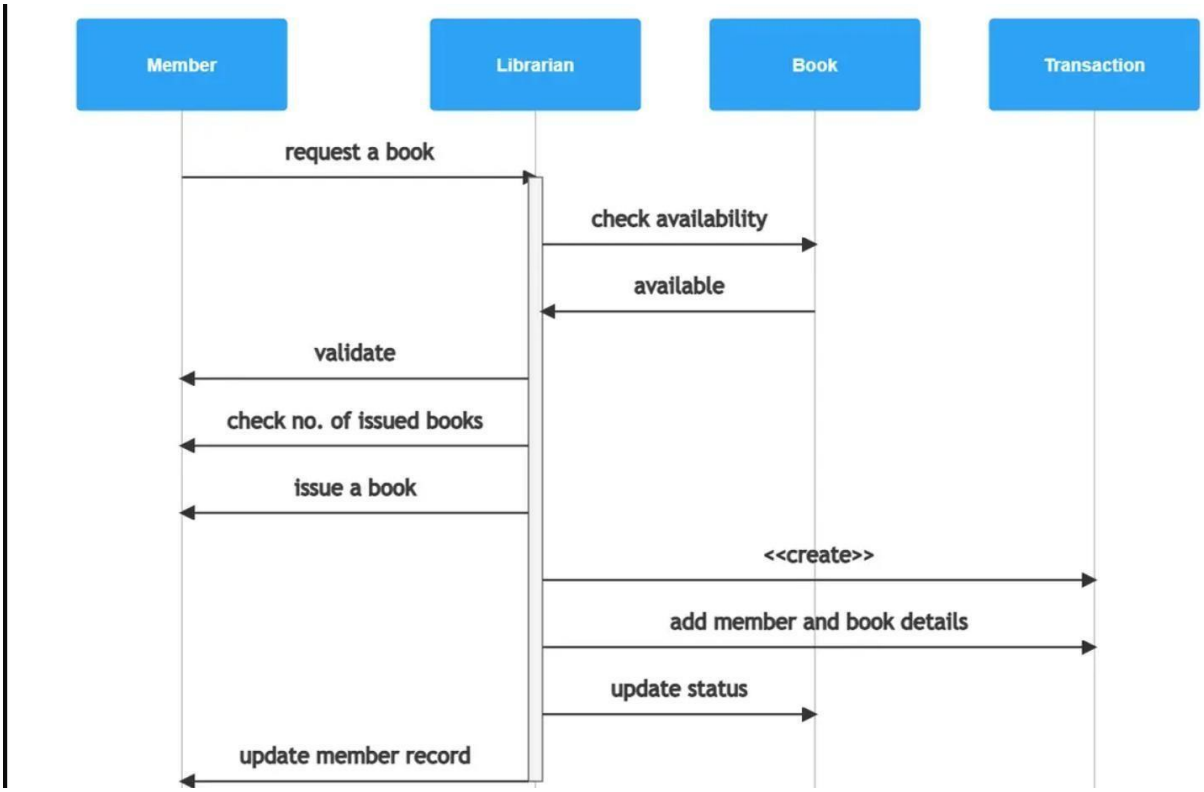
The **UML Sequence Diagram for Library Management System** is one of the methods used for project development. It is a useful tool for documenting the Library Management System's needs and fleshing out its architecture. The system's sequence diagram is designed to represent a timeline that shows the series of interactions. Each object has a column, and the arrows indicate the messages that are sent between them. Sequence diagrams help display collaborations between items, but not so much for defining behavior precisely. Developers frequently use sequence diagrams to model the interactions between items in a single-use case. They show how the various pieces of a system communicate with one another to perform a function, as well as the order in which the interactions take place when a specific use case is executed.

USERS OF COLLEGE LIBRARY MANAGEMENT SYSTEM SEQUENCE DIAGRAM:

- **School Librarian** : The school librarians will be the ones to use the system most of the time. They will monitor the books from time to time and will cater to the borrowing and returning of books. They were also responsible for all the activities related to the library.

- **Book Borrowers** : Book borrowers were not just the students but also the professors or instructors. They will also have access to the system and to do that, they will have to log into the system. This will help the librarian and the admin monitor the book borrowers.

- **School Admin** : The Library Management System can be a stand-alone project or a part of a bigger project. Nevertheless, it always has the admin which can access all of the library information. This is done when there are serious scenarios or problems.



PROCESS TIMING:

- First user will request the book from librarian which is show by a arrow from member to librarian.
- Next, the librarian will check the system to see if the book is available. This means an interaction with the Book object in the library management system. Again, we'll use a labeled arrow and the Book object will be created.
- If the book is available, the Book object sends a message back to the Librarian object.
- The librarian now needs to validate whether the member can borrow the book from the library. This is a two-step process of validating membership and also checking the number of books already issued.
- If the member is authorized to borrow the book, the librarian can now issue it.
- The librarian also needs to record the transaction in the library management system, and that brings us to our final object, Transaction. The Librarian sends a "create" message to Transaction. The message adds the member and book details to that transaction and it is stored in the database so that it can be used to check up on the member or book later on.
- The librarian next needs to update the book's status in the system, so it's clear that it has been borrowed and isn't available until returned. The librarian also needs to update the member's record, to indicate that they have borrowed an additional book. That gives us the final lines in our simple library management sequence diagram.

STATECHART DIAGRAM:

Purpose of State chart Diagrams:

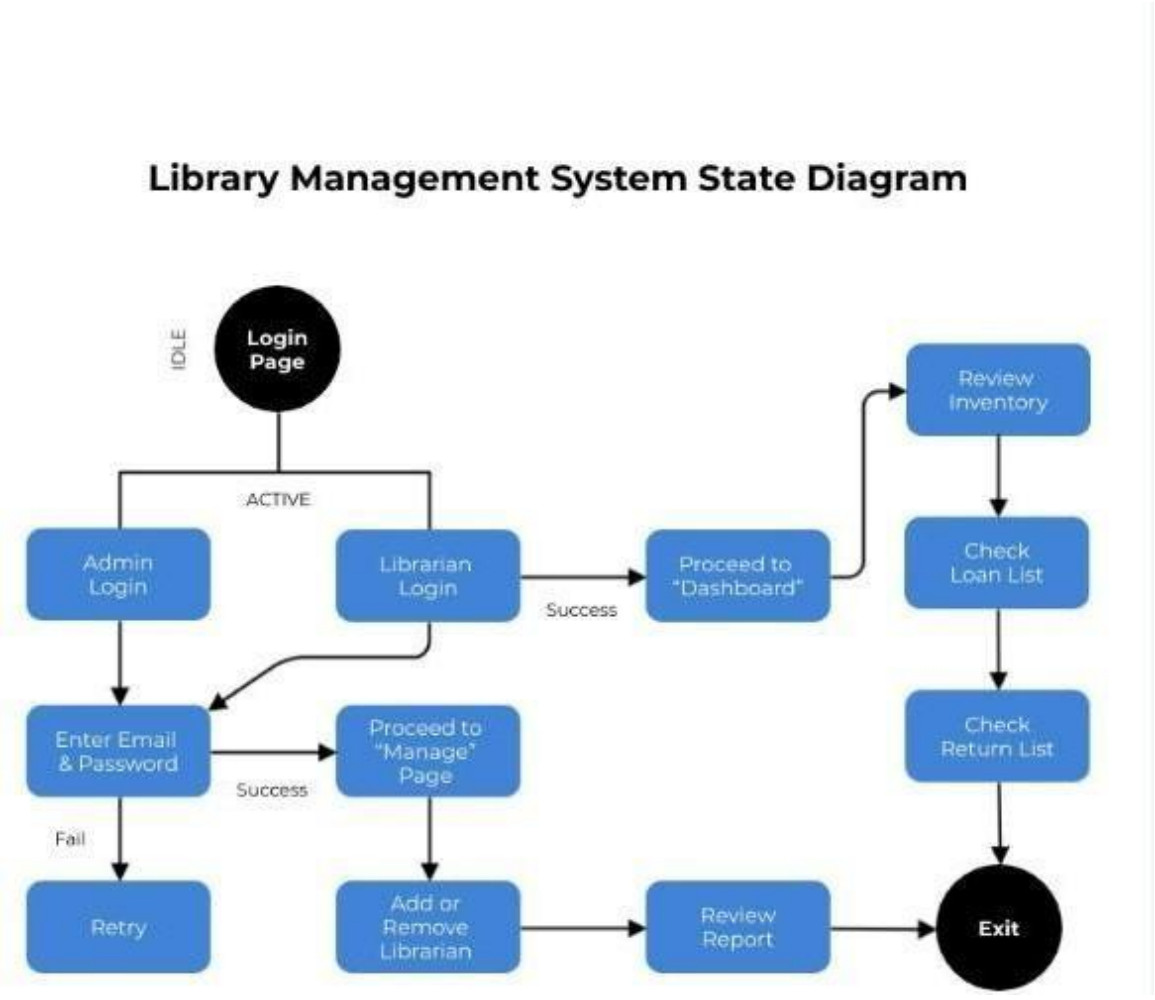
Statechart diagram is one of the five UML diagrams used to model the dynamic nature of a system. They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events.

Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.

Statechart diagrams are also used for forward and reverse engineering of a system. However, the main purpose is to model the reactive system.

Following are the main purposes of using Statechart diagrams −

- To model the dynamic aspect of a system.
- To model the life time of a reactive system.
- To describe different states of an object during its life time.
- Define a state machine to model the states of an object.

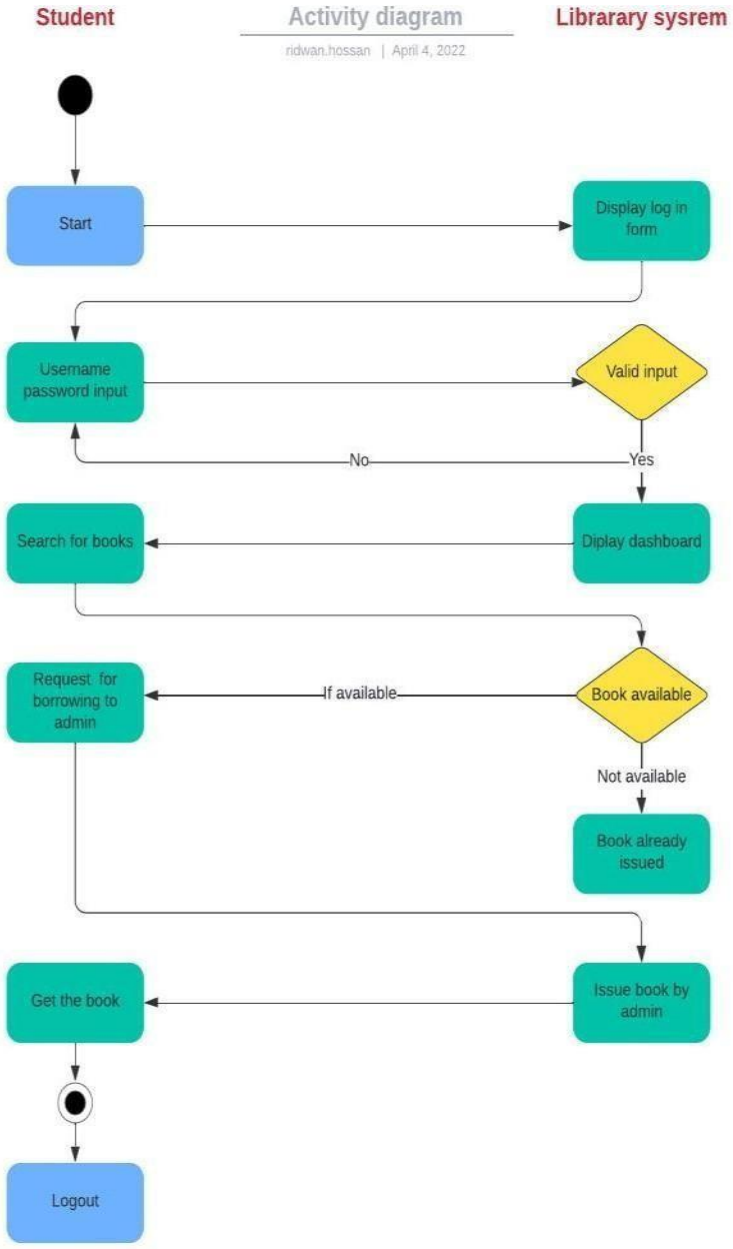


ACTIVITY DIAGRAM:

PURPOSE OF ACTIVITY DIAGRAM:

Activity diagrams are graphical presentation of processes that include choice, iteration, and concurrency. It specifies the target system's control flow, such as 12 complicated business rules and processes, as well as the use case and business process. Activity diagrams are used to represent both computing and organizational processes in the Unified Modelling Language (UML) (i.e., workflows)

According to the admin activity diagram when the admin enters to the library management webpage it displays the login form, where the admin has to put theusername and password and click enter. If the user credentials match to the database, it displays the admin dashboard and if admin credentials doesn't match the system display again the login page. From the admin dashboard the admin can add or delete a book and save the data in the database. The admin can viewstudent details and put them inactive mode if they have overdue books. The admin can issue books to the student and also can issue a fine if the student doesn't return there borrow books on time. After finishing all the transections, theadmin can log out from the system.



User activity diagram

The diagram illustrates about user activity on the library management system. According to the user activity diagram when students open the library management webpage, it shows the login form where the student have to enter personal credentials and if the credentials match with the database It displays the user dashboard otherwise the web page return to the login form state. From the user dashboard the user can search a book and if the book is available the usercan make a borrowing request to the admin, then the admin issues the book to the user. After finishing the transactions, the user has to log out from the system.

DEPLOYMENT DIAGRAM:

PURPOSE:

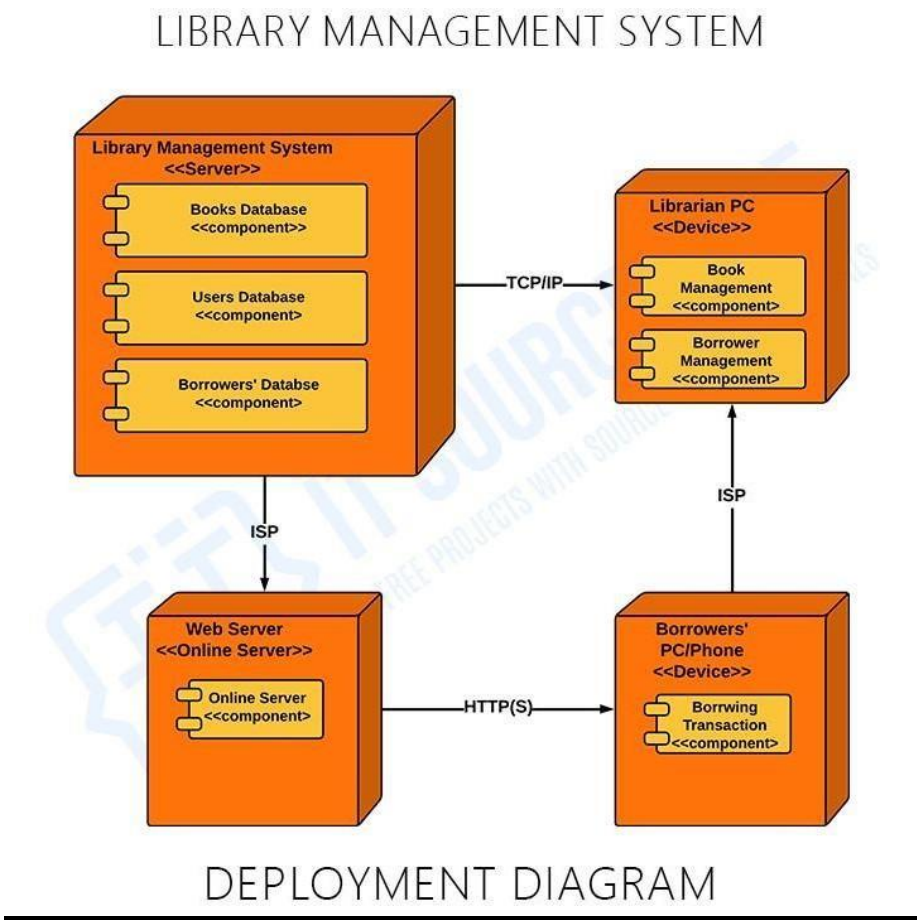
A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware.

Deployment diagrams help model the hardware topology of a system compared to other UML diagram types which mostly outline the logical components of a system.

Deployment Diagram for Online Library Management System :

The **Deployment Diagram for Online Library Management System** shows a detailed illustration of the system's software and hardware specification. Additionally, it gives the complete physical structure of the library management system that is needed in its deployment for its users.



The library management system UML deployment diagram explains the sketch of the relationship between software and hardware. These hardware and software also include their components to clarify their part in the system's operation. They were represented by nodes and the connections were represented by labeled arrows.
The deployment diagram shows the scenario when the system is deployed. It has 4 nodes represented with boxes and components within. The Library Management System node has the component of several databases such as books, borrowers, and users. Then the librarian must be connected to the network thru TCP/IP in order for them to access the system.

Additionally, the software is connected to an ISP which enables it to pass data to the online server and then will be accessed by the borrowers thru browsers also with the help of URLs. Lastly, the Librarian and the Borrowers can communicate with the use of ISP.

UML COLLABORATION DIAGRAM

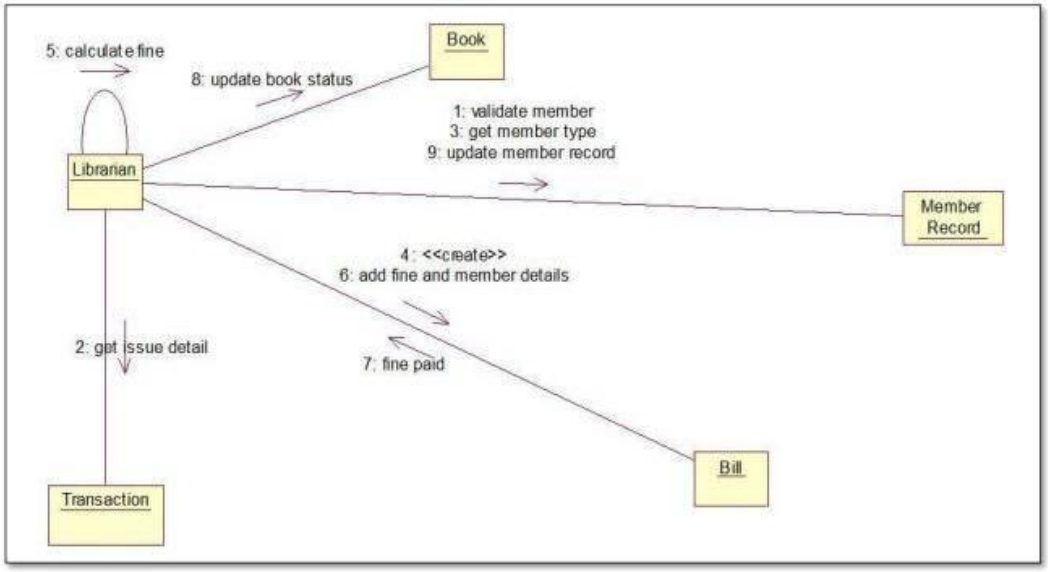
UML Collaboration diagram depicts the interactions between objects or parts in terms of sequenced messages and describes both the static structure and dynamic behaviour of a system

PURPOSE :

Collaboration Diagrams focus mainly on processes between classes rather that sequence of messages, thus depicting what will take place after/ before what. So thus it is helpful in designing web pages accordingly of what process is likely to be done first.

DIAGRAM :

COLLABORATION DIAGRAM:



A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. The four major components of a collaboration diagram are:

NOTATIONS OF COLLABORATION DIAGRAM

Objects- Objects are shown as rectangles with naming labels inside. The naming label follows the convention of object name: class name. If an object has a property or state that specifically influences the collaboration, this should also be noted.

Actors- Actors are instances that invoke the interaction in the diagram. Each actor has a name and a role, with one actor initiating the entire use case.

Links- Links connect objects with actors and are depicted using a solid line between two elements. Each link is an instance where messages can be sent.

messages- Messages between objects are shown as a labeled arrow placed near a link. These messages are communications between objects that convey information about the activity and can include the sequence number.

The most important objects are placed in the center of the diagram, with all other participating objects branching off. After all objects are placed, links and messages should be added in between.

CONCLUSIONS

Library Management system is developed by using a proper channel. The objectives of this are pre-defined on which the whole system work to achieve them by managing the details of all the library database and so on. This system helps in boosting efficient services.

All in all, the UML Diagrams work together to achieve the most desired functions of an Library Management System. All of these were designed to guide people and staff of what should be the behavior and structure of the Library Management System.

By completing all the given Diagrams, the Library Management System development would be much easier and more attainable. So those UML diagrams were given to assist developer and guide through the project development journey. All functions of the given UML diagrams can be used as reference for project development. The ideas presented in UML Diagrams were all based on Library management requirements.

UML diagrams can be used to envision the System before it begins or to document it once was completed. However, UML diagrams can be used in any sector, not only in software engineering. Its' overall objective is to help teams to visualize how the system works or will work.

The UML Diagrams are created to easily understand, update, maintain, and document Library Management system information. UML diagrams for Library Management system were used to visualize the project. It can be done before the development begins or to document its progress once it is completed.

REFRENCE

https://itsourcecode.com/uml/online-shopping-complete-uml-diagrams/

https://www.uml-diagrams.org/examples/online-shopping-example.html

https://www.edrawmax.com/templates/1007460/

https://www.freeprojectz.com/uml-diagram/shopping-management-system-uml- diagram

https://www.lucidchart.com/pages/templates/online-shopping-cart-uml-class-diagram- example

https://creately.com/diagram/example/gqz0nrua1/component-diagram-for-online- shopping-system-classic

https://meeraacademy.com/activity-diagram-for-online-shopping-website/

https://www.softwareideas.net/a/1582/e-shop--online-shopping-uml-communication- diagram-

https://www.geeksforgeeks.org/unified-modeling-language-uml-state-diagrams/