JAGANNATH INTERNATIONAL MANAGEMENT SCHOOL

(Department of Information Technology)

SE Lab

Bachelor of Computer Application (BCA)

Semester- IV

Software Engineering Lab File

Subject Code: **BCA 274**

Batch: 2023-26

Submitted To:

Mr. Kunal Anand

Assistant Professor– IT Department

JIMS, Vasant Kunj, New Delhi

Submitted By:

Name of students

Enrollment Number

Class: BCA-IV (Shift-I)

INDEX

S.No.	Experiment	Signature
1.	Select and Write down the problem statement for a real time system of relevance.	
2	To perform the user‘s view analysis for the suggested system: Use case diagram.	
3	To create the function oriented diagram: Data Flow Diagram (DFD)	
4	To draw the structural view diagram for the system: Class diagram	
5	To perform the behavioral view diagram for the suggested system : Sequence diagram	
6	Analyze requirement for a system and develop Software Requirement Specification Sheet (SRS) for suggested system.	
7	To draw the behavioral view diagram: State-chart diagram or Activity diagram.	
8	Draw the component diagram.	
9	Draw the Deployment diagram.	
10	Perform Measurement of complexity with Halstead Metrics.	

PRACTICAL 1

Select and Write down the problem statement for a real time system of relevance.

Problem statement:

This study will support and manage library Management and services in JIMS – VASANT KUNJ Campus, Also, the computer technology students (BCA) will be able to develop a system program using Visual Basic 6.0 or Java netbeans 8.2 software for a design of library monitoring system in response to the problems in the current manual library operations, services and transactions. The system should address the challenges faced by the college library in effectively managing its resources, providing seamless access to educational materials, and streamlining administrative processes. The solution should automate various library tasks, including book cataloging, inventory management, patron registration, circulation, and book reservation.

Inefficient Resource Management: Without a proper system in place, it can be challenging to manage and track the college library's resources effectively. This can lead to difficulties in locating books, misplacement of materials, and inaccurate inventory records.

Manual Administrative Processes: Traditional manual methods of book cataloging, issuing, and returning books can be time-consuming and error-prone. This can result in delays in serving students, increased administrative workload, and potential mistakes in managing due dates, fines, and reservations.

Lack of Real-time Updates: Without a real-time system, it becomes difficult to provide accurate and up-to-date information about the availability of books and the status of borrowings. This can lead to frustration among students and faculty members who may face difficulties in finding the required resources.

Inconvenient Book Reservation Process: In the absence of an online reservation system, students may face challenges in reserving books in advance. This can result in missed opportunities to access popular materials and an inefficient allocation of library resources.

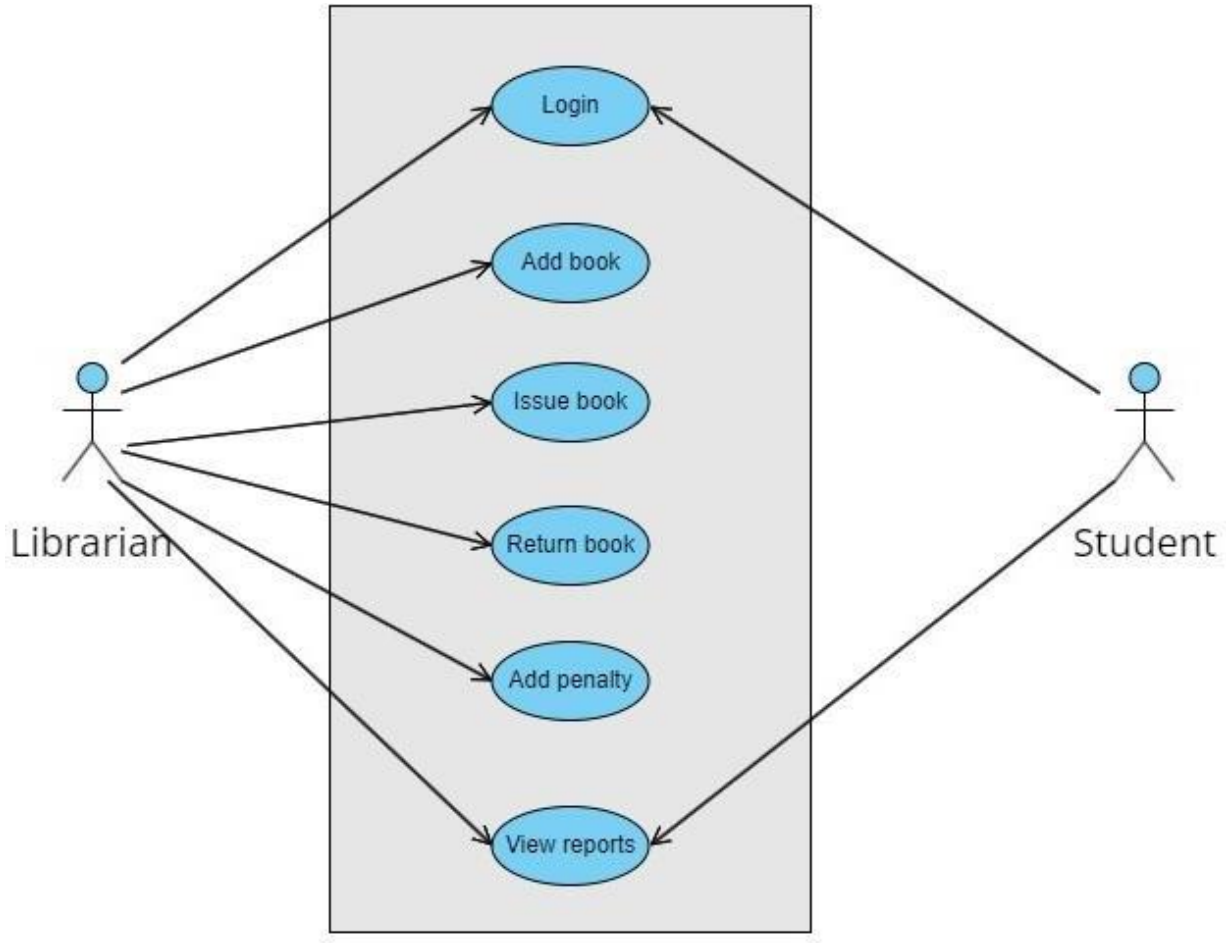
Limited Reporting and Analysis: Manual record-keeping makes it difficult to generate comprehensive reports and analyze library usage patterns. Without this data, it becomes challenging to make informed decisions regarding resource allocation, collection development, and identifying areas for improvement.

Lack of Integration with Student Information System: In the absence of integration with the college's student information system, there can be discrepancies in patron information, resulting in delays or errors during the registration and borrowing processes.

PRACTICAL 2

To perform the user‘s view analysis for the suggested system: Use case diagram.

USE CASE DIAGRAM FOR LIBRARY MANAGEMENT



USE CASE DESCRIPTION

1. Login

- **Description**

  This use case describes the process of logging into the library management system for students (Member) , Librarian,.

- **Actors**

  Student and Librarian

- **Basic Flow**

  a. The user navigates to the login page and enters their credentials.

  b. The system verifies the user's credentials and checks their user type (Student, or Librarian).

  c. The system redirects the user to their respective dashboard.

- **Alternative Flow**

  a. If the user enters an incorrect username or password ,the system displays an error message, prompting the user to enter valid credentials.

  b. If the user's account is inactive, the system displays a message stating that the user's account is inactive and they should contact the administrator.

- **Precondition**

  The user must have a valid username and password.

- **Postcondition**

  The user will be redirected to their respective dashboard.

- **Extension points**
  None

---

2. Add Book

Description:

This use case represents the process of adding a new book to the library's collection. Library staff and administrators can input the book's details, such as title, author, publication information, and categorization, into the system for proper cataloging and tracking.

Actors:

- Librarian

Preconditions:

- The user must be logged into the library management system.

Basic Flow:

1. The user navigates to the "Add Book" section within the system.

2. The user enters the required details of the book, such as title, author, ISBN, publication date, and other relevant information.

3. The system validates the entered information, ensuring that mandatory fields are filled correctly.

4. If the information is valid, the system adds the book to the library's collection and assigns a unique identifier.

5. The system updates the inventory to reflect the addition of the new book.

Alternative Flows:

- If the entered information is incomplete or invalid:
- The system displays an error message, prompting the user to correct the erroneous fields.

Postconditions:

- The new book is successfully added to the library's collection and is available for borrowing.

---

3. Issue Book

Description: This use case represents the process of issuing a book to a student or library patron. Library staff and administrators can select a book from the library's collection and assign it to a specific student for a designated borrowing period.

Actors:

- Librarian

Preconditions:

- The user must be logged into the library management system.
- The student must be registered in the system.
- The book must be available in the library's inventory.

Basic Flow:

1. The user searches for the desired book within the library management system.

2. The system checks the availability status of the book. If the book is available, the user selects the student to whom the book will be issued.

3. The system records the issuance details, including the book, student, and due date. The system updates the availability status of the book in the inventory.

Alternative Flows:

- If the book is not available:
- The system displays a notification or suggests alternative options, such as placing a reservation for the book or selecting a similar book.

Postconditions:

- The book is successfully issued to the student, and the corresponding records are updated in the library management system.

4. Return Book

Description:

This use case represents the process of returning a borrowed book to the library. Students or library patrons can return the book to the library staff or through a designated book drop-off point.

Actors:

- Librarian

Preconditions:

- The book must have been issued to the student or library patron.
- The user must have borrowed the book from the library.
- The user must be logged into the library management system (if returning through staff).

Basic Flow:

1. The user navigates to the "Return Book" section within the library management system (if returning through staff) or physically visits the library.
2. The book return is valid, the system updates the return date and marks the book as returned in the system.
3. The library staff verifies the returned book and updates the system accordingly.

Alternative Flows:

- If the book return is not valid:
- The system displays an error message, indicating the reason for the invalid return (e.g., overdue book, incorrect borrower).
- The library staff may need to handle any associated fines or penalties.

Postconditions:

- The book is successfully returned to the library, and the corresponding records are updated in the library management system.

---

5. Add Penalty

Description:

This use case represents the process of adding penalties or fines to a user's account for overdue books, damaged materials, or other library-related offenses. Library staff and administrators can apply penalties and track the associated fines within the library management system.

Actors:

- Librarian

Preconditions:

- The user must be logged into the library management system.
- The user must have access and authorization to apply penalties.

Basic Flow:

1. The user selects the user account to which the penalty will be applied.
2. The user identifies the reason for the penalty (e.g., overdue book, damaged material).
3. The user enters the necessary details, such as the penalty type, fine amount, and duration.
4. The system updates the user's account with the penalty information, including the fine amount and due date.
5. The system may send a notification to the user regarding the penalty and the associated fine.

Alternative Flows:

- If the user account cannot be located:
- The system displays an error message, prompting the user to re-enter the user information.

Postconditions:

- The penalty is successfully applied to the user's account, and the associated fines are recorded in the library management system.

---

6. View Record

Description:

This use case represents the process of viewing records within the library management system. Users, including library staff and administrators, can access and review various records, such as borrowing history, fine records, and inventory details, for managing library operations effectively.

Actors:

- Student, Librarian

Preconditions:

- The user must be logged into the library management system.

Basic Flow:

1. The user selects the specific type of record they want to view (e.g., borrowing history, fine records, inventory details).
2. The system retrieves the requested records based on the user's selection.
3. The system presents the records to the user in a readable and organized format, such as a list, table, or report.
4. The user can browse and navigate through the records as needed, filtering or sorting the data based on specific criteria.

Alternative Flows:

- If there are no records available for the selected type:
- The system displays a message indicating the absence of relevant records.
- The user may need to perform a different search or adjust the search criteria.

Postconditions:

- The user successfully views the desired records, enabling them to gain insights and make informed decisions related to library management.

PRACTICAL 3

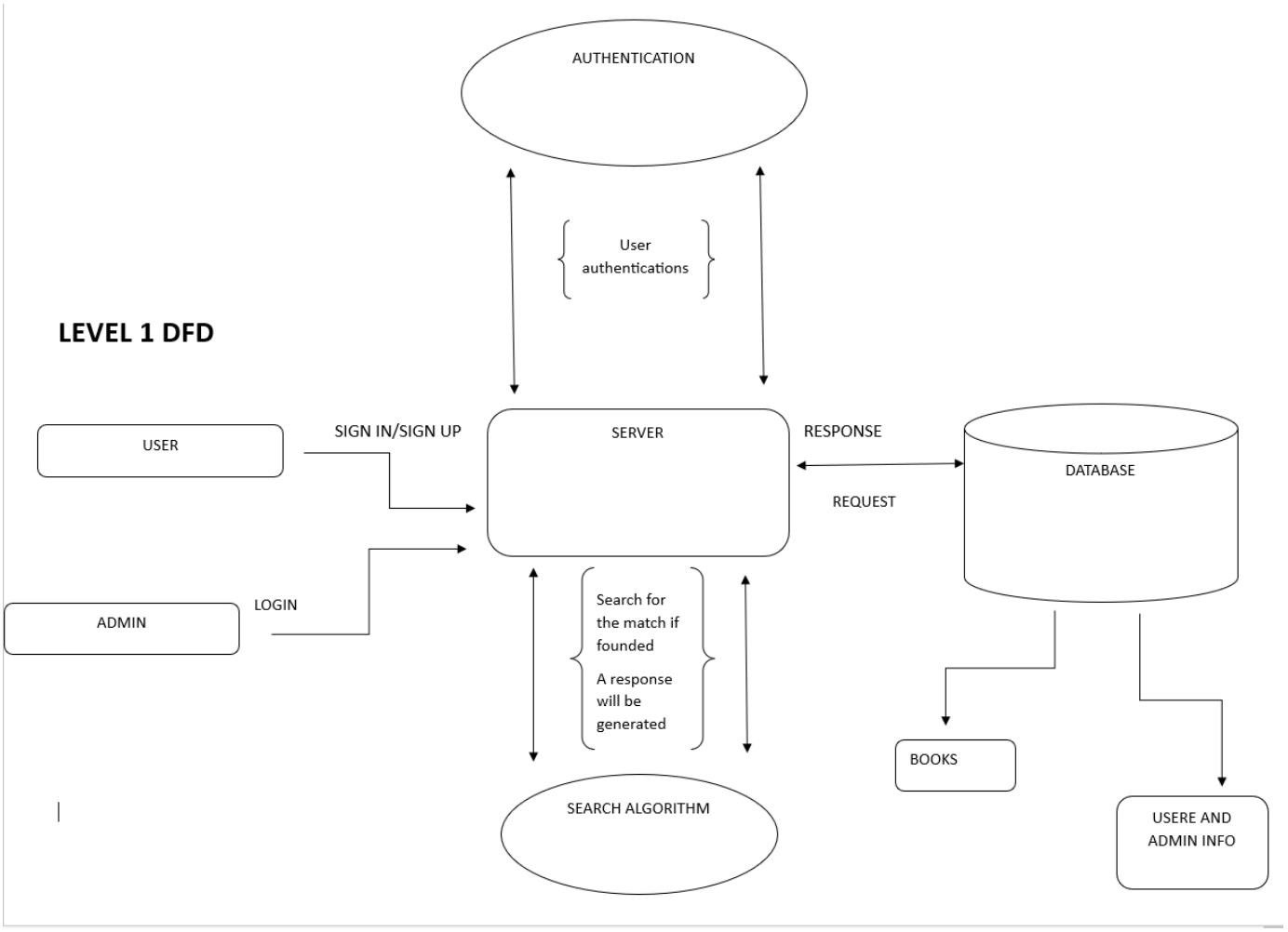
To create the function oriented diagram: Data Flow Diagram (DFD)

O level DFD

The Level 0 DFD Diagram for Library Management System is also known as the context diagram. It is composed of the main process, users and data flows. The concept is shown in a single process visualization.

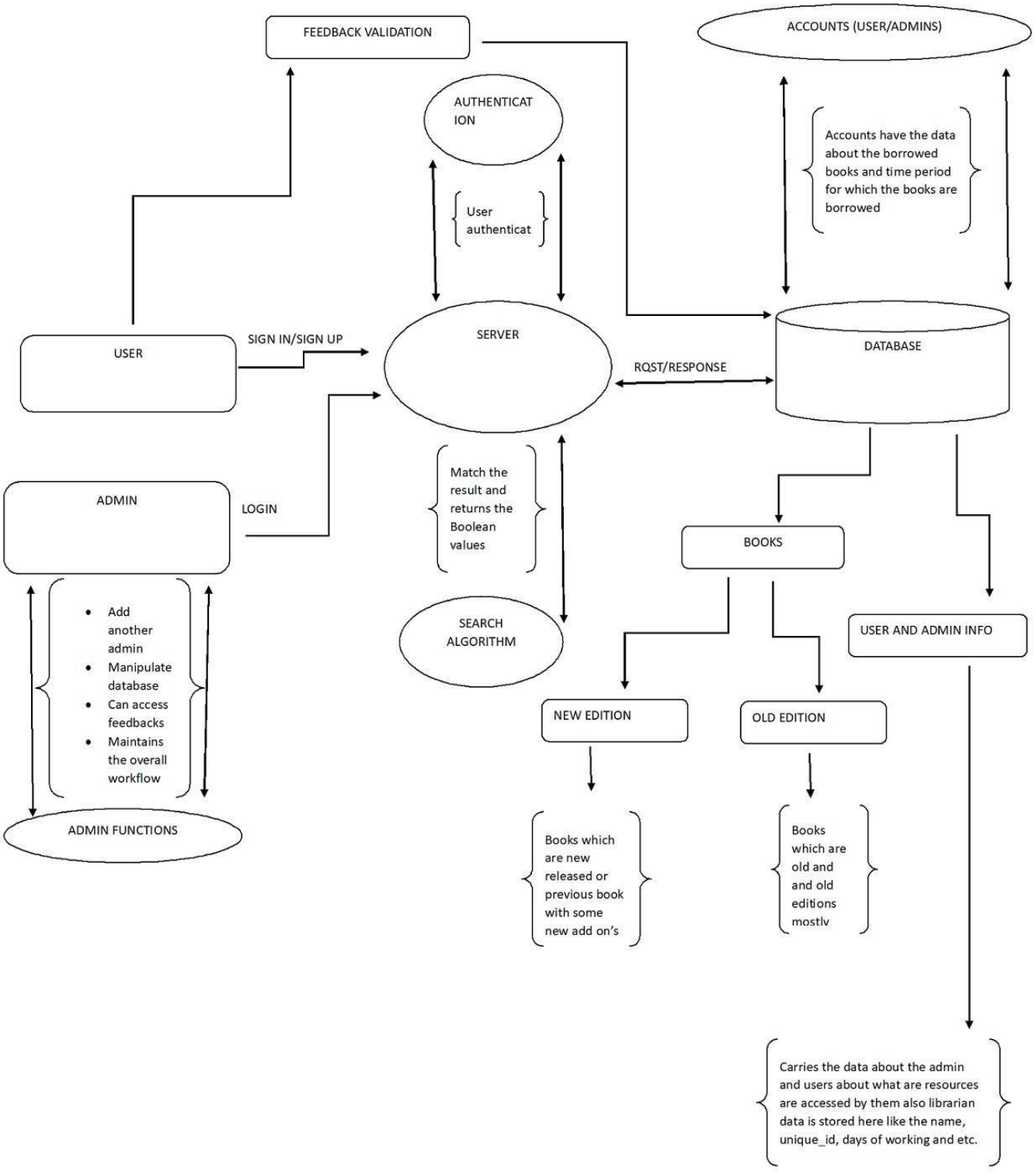
LEVEL 1 DFD

DFD Level 1 Diagram for Library management system is considered as the detonated view of context diagram. Its function is to deepen the concept derive from the DFD Level 0.



LEVEL 2 DFD

Level 2 DFD for Library managemnet is also called as the highest abstraction of data flow diagram. This level also broadens the idea from the DFD level 1. It includes the sub-processes from level 1 as well as the data that flows



PRACTICAL 4

To draw the structural view diagram for the system: Class diagram

Class Diagram:

A class diagram is a type of static structure diagram in UML (Unified Modeling Language) that represents the structure and relationships of classes, interfaces, and their associations in a system. It provides an overview of the classes and their attributes, methods, and associations in the system.

Components of a Class Diagram:

Class: A class represents a blueprint or template for creating objects.

Attributes: Attributes represent the characteristics or properties of a class.

Methods: Methods represent the behaviors or operations that a class can perform.

Associations: Associations represent the relationships between classes.

Multiplicity: Multiplicity defines the number of instances or objects that participate in an association.

Inheritance: Inheritance represents the "is-a" relationship between classes.

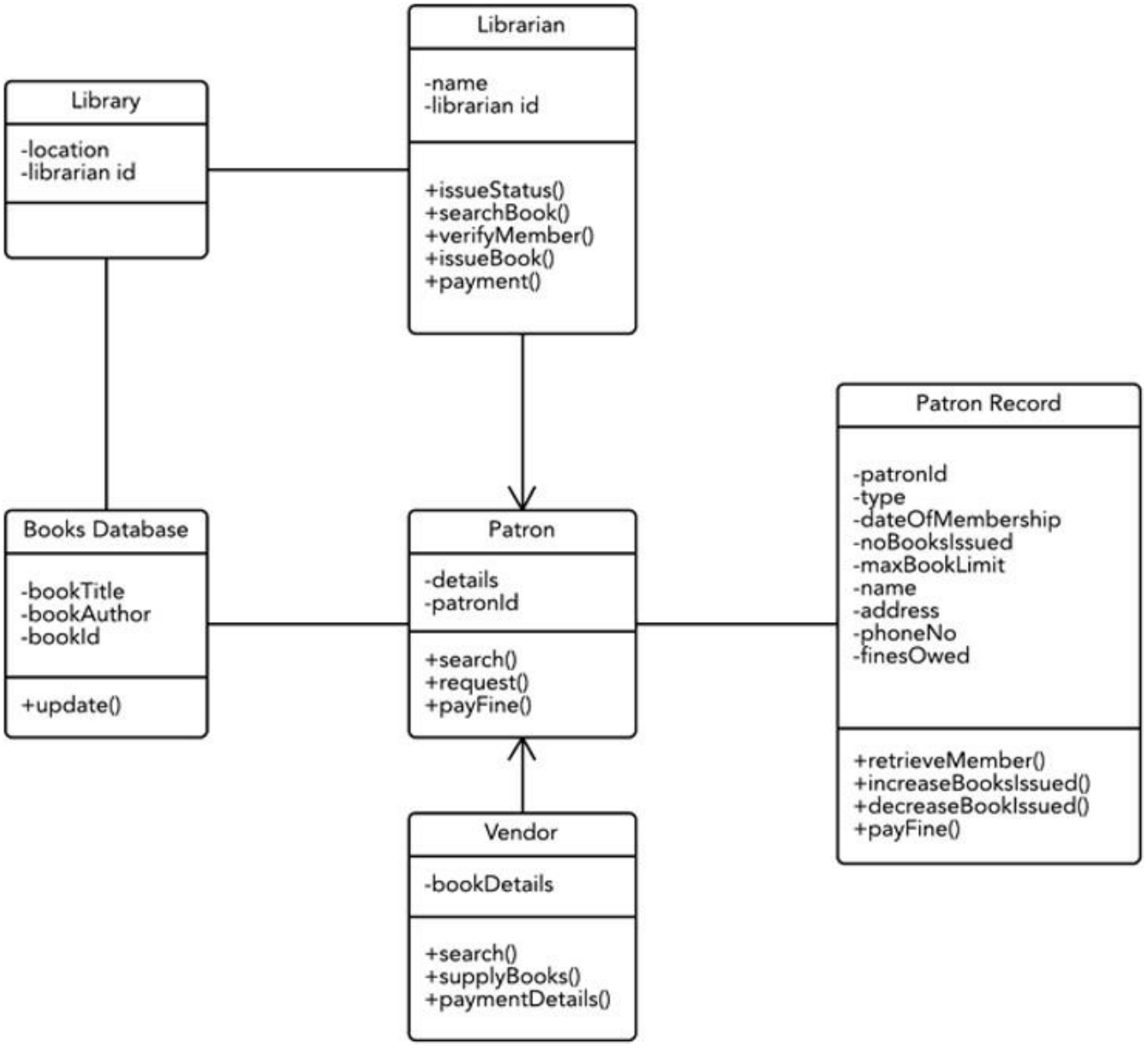
Interfaces: Interfaces define a contract or set of methods that a class must implement.

Dependency: Dependency represents a relationship where one class depends on another class.

Aggregation and Composition: Aggregation and composition represent the whole-part relationships between classes.

Stereotypes and Notes: Stereotypes and notes provide additional information or annotations to enhance the understanding of the class diagram.

CLASS DIAGRAM FOR LIBRARY MANAGEMENT



PRACTICAL 5

To perform the behavioral view diagram for the suggested system: Sequence diagram

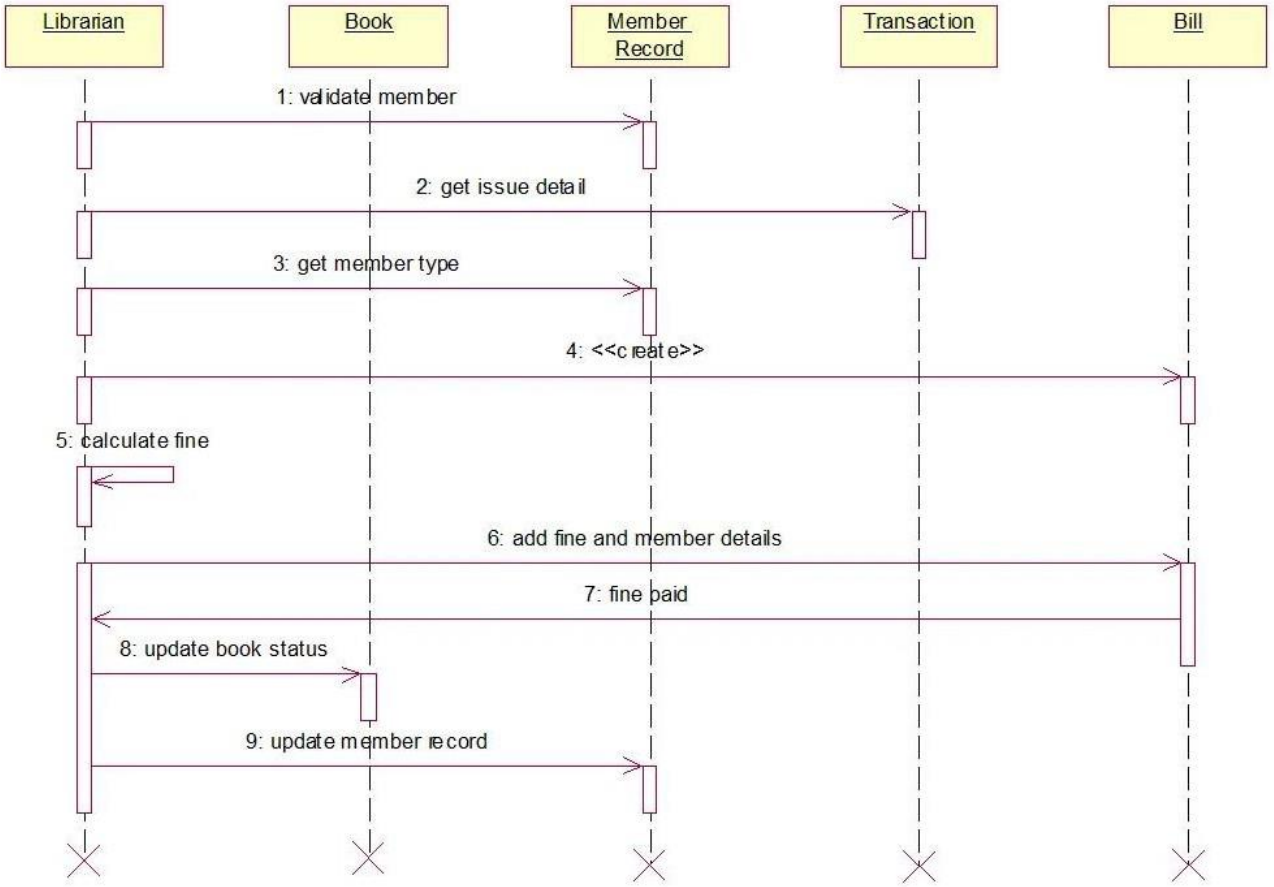
SEQUENCE DIAGRAM:

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram.

COMPONENTS OF SEQUENCE DIAGRAM:

1. **Actors:** Actors represent the external entities that interact with the system.
2. **Objects:** Objects are the specific instances of classes or components within the system.
3. **Lifelines:** Lifelines represent the lifespan of an object during the execution of a sequence diagram..
4. **Messages:** Messages depict the communication or method calls between objects.
5. **Activation Boxes:** Activation boxes, also known as execution specifications, represent the period of time during which an object is executing a method or performing an action.
6. **Combined Fragments:** Combined fragments are used to represent conditional or looping behavior within a sequence diagram.
7. **Notes:** Notes provide additional explanations, comments, or annotations to enhance the understanding of the sequence diagram.

SEQUENCE DIAGRAM FOR LIBRARY MANAGEMENT



PRACTICAL 6

Analyze requirement for a system and develop Software Requirement Specification Sheet (SRS) for suggested system.

1. Introduction

1.1 Purpose

The purpose of this document is to provide a detailed description of the requirements for the Library Management System. It outlines the functionality, constraints, and interfaces of the system.

1.2 Scope

The Library Management System is designed to automate and streamline the operations of a library. It includes features such as book management, student management, book issuance, return, penalty calculation, and user authentication.

1.3 Definitions, Acronyms, and Abbreviations

- SRS: Software Requirement Specification
- LMS: Library Management System
- ISBN: International Standard Book Number 1.4 References
- Use case diagram for the Library Management System

2. OVERALL DESCRIPTION

2.1 PRODUCT PRESPECTIVE

The proposed Library Management System will take care of the current book detail at any point oftime. The book issue, book return will update the current book details automatically so that user willget the update current book details.

2.2 SOFTWARE REQUIREMENT

Front end:
• Android developer tool
• Advance java

Back end:
• MySQL

2.3 HARDWARE REQUIREMENT

- Android version 2.3 ginger bread(minimum, android user's)
- 2GB ram
- 1.2 GHz processor
- Intel i5
- Windows 7/8/8.1/10
-

2.4.1 FUNCTIONAL REQUIREMENT

R.1:Register

Description :
First the user will have to register/sign up. There are two different type of users.

The library manager/head :
The manager have to provide details about the name of library, address, phone number, email id.

Regular person/student :
The user have to provide details about his/her name of address, phone number, email id.

R.1.1 : Sign up

Input: Detail about the user as mentioned in the description.

Output: Confirmation of registration status and a membership number and password will be generated and mailed to the user.

Processing: All details will be checked and if any error are found then an error message is displayed else a membership number and password will be generated.

R.1.2 : Login

Input: Enter the membership number and password provided.

Output : User will be able to use the features of software.

R.2 : Manage books by user.

R.2.1 : Books issued.

Description : List of books will be displaced along with data of return.

R.2.2 : Search

Input : Enter the name of author's name of the books to be issued.

Output : List of books related to the keyword.

R.2.3 : Issues book

State : Searched the book user wants to issues.

Input : click the book user wants.

Output : conformation for book issue and apology for failure in issue.

Processing : if selected book is available then book will be issued else error will be displayed.

R.2.4 : Renew book

State : Book is issued and is about to reach the date of return.

Input : Select the book to be renewed.

Output : conformation message.

Processing: If the issued book is already reserved by another user then an error message will be sent and if not then a confirmation message will be displayed.

R.2.5 : Return

Input ; Return the book to the library.

Output : The issued list will be updated and the returned book will be listed out.

R.2.6 ; Reserve book

Input ; Enter the details of the book.

Output : Book successfully reserved.

Description : If a book is issued by someone then the user can reserve it ,so that later theuser can issue it.

R.2.6 Fine

Input : check for the fines.

Output : Details about fines on different books issued by the user.

Processing : The fine will be calculated, if it crossed the date of return and the user did notrenewed if then fine will be applied by Rs 10 per day.

R.3 Manage book by librarian

R.3.1 Update details of books

R.3.1.1 Add books

Input : Enter the details of the books such as names ,author ,edition, quantity.

Output : confirmation of addition.

R.3.1.2 Remove books

Input : Enter the name of the book and quantity of books.

Output : Update the list of the books available.

2.4.2 Non Functional Requirements

- **Usability Requirement**

The system shall allow the users to access the system from the phone using android application. The system uses a android application as an interface. Since all users are familiar with the general usage of mobile app, no special training is required. The system is user friendly which makes the system easy.

- **Availability Requirement**

The system is available 100% for the user and is used 24 hrs a day and 365 days a year. The system shall be operational 24 hours a day and 7 days a week.

- **Efficiency Requirement**

Mean Time to Repair (MTTR) - Even if the system fails, the system will be recovered back up within an hour or less.

• **Accuracy**
The system should accurately provide real time information taking into consideration various concurrencyissues. The system shall provide 100% access reliability.

• **Performance Requirement**
The information is refreshed depending upon whether some updates have occurred or not in the application. The system shall respond to the member in not less than two seconds from the time of the request submittal. The system shall be allowed to take more time when doing large processing jobs. Responses to view information shall take no longer than 5 seconds to appear on the screen.

• **Reliability Requirement**
The system has to be 100% reliable due to the importance of data and the damages that can be caused by incorrect or incomplete data. The system will run 7 days a week, 24 hours a day.

2.5 USER CHARACTERSTICS

We have 3 levels of users :

User module: In the user module, user will check the availability of the books.

- Issue book
- Reserve book
- Return book
- Fine details

Library module:

- Add new book
- Remove books
- Update details of book

Administration module: The following are the sub module in the administration module :

- Register user
- Entry book details
- Book issue

2.6 CONSTRAINTS

Any update regarding the book from the library is to be recorded to have update & correct values, and anyfine on a member should be notified as soon as possible and should be correctly calculated.

PRACTICAL 7

To draw the behavioral view diagram: State-chart diagram or Activity diagram.

A state diagram, also known as a state machine diagram or state chart diagram, is a type of behavioral diagram in UML (Unified Modeling Language) used to model the behavior of a system or object over time. It represents the various states that an object or system can be in, as well as the transitions between those states based on events and conditions.

The components of a state diagram include:

1. State: A state represents a specific condition or mode of an object or system. It describes the behavior and attributes of the object or system at a particular point in time.

2. Initial State: The initial state indicates the starting point of the object or system when it is first created or initialized.

3. Final State: The final state represents the end point or termination of the object or system's behavior. Once the object or system reaches the final state, it cannot transition to any other state.

4. Transitions: Transitions depict the movement or change of an object or system from one state to another. They are triggered by events or conditions and may have associated actions or guards (conditions that must be met for the transition to occur).

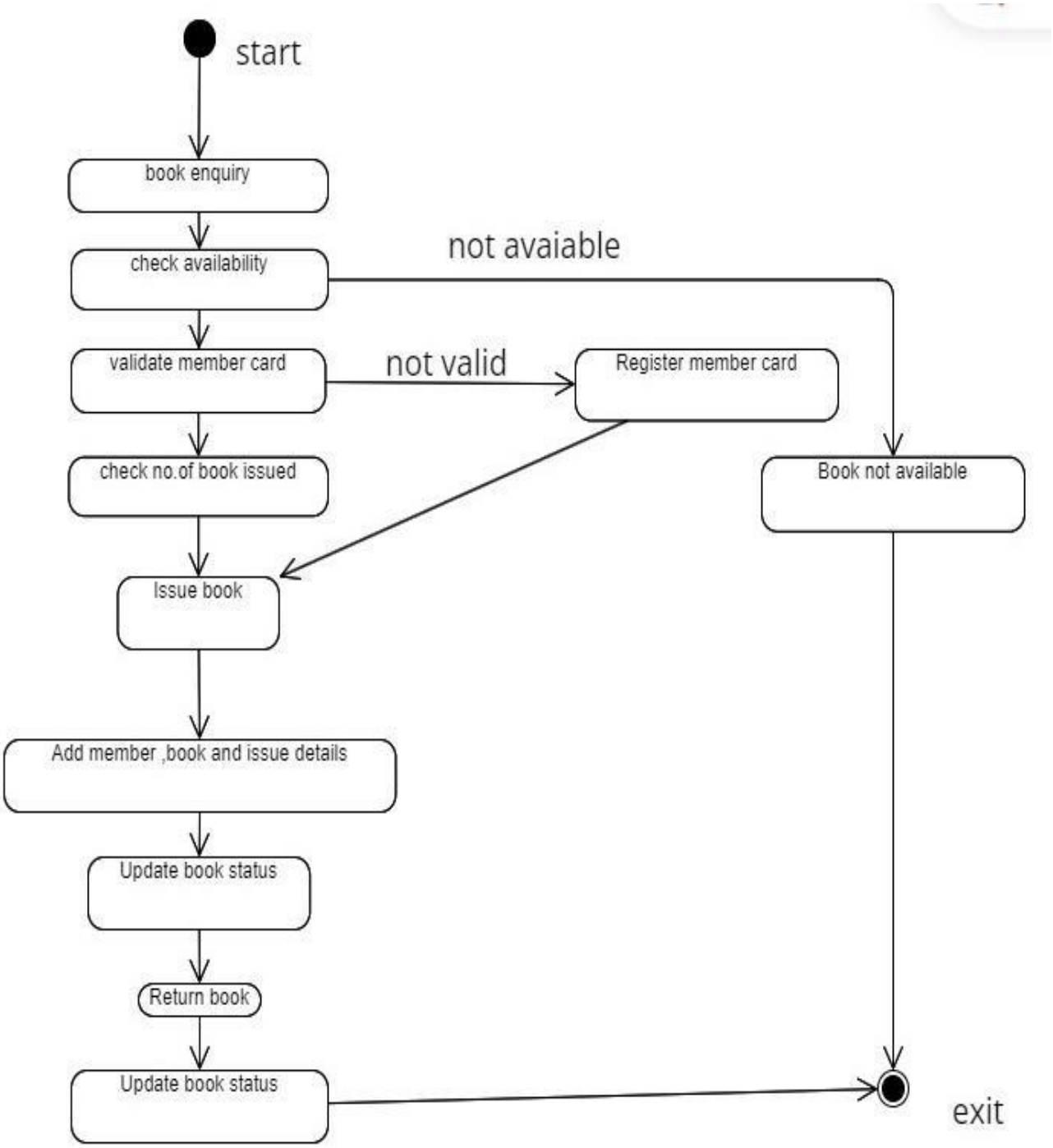
5. Events: Events are stimuli or occurrences that trigger transitions in the state diagram. They can be external events (e.g., user input, system signals) or internal events (e.g., completion of an action).

6. Actions: Actions represent the behavior or activities associated with a state or transition. They can be actions performed upon entering or exiting a state, as well as actions performed during a transition.

7. Guards: Guards (also known as conditions or guards) are conditions that must be true for a transition to occur. They define the constraints or criteria that determine if a transition is allowed to happen.

8. Composite State: A composite state is a state that contains nested states within it. It allows for the hierarchical modeling of complex systems or objects, where each nested state can have its own set of sub-states, transitions, and behaviors.

State diagrams provide a visual representation of the dynamic behavior of a system, illustrating the possible states and transitions that can occur. They are useful for modeling complex systems, specifying the life cycle of an object, and understanding the behavior of a system in response to events and conditions.


start
book enquiry
check availability
not avaiable
validate member card
not valid
Register member card
check no.of book issued
Book not available
Issue book
Add member ,book and issue details
Update book status
Return book
Update book status
exit

PRACTICAL 8

Draw the component diagram.

A component diagram is a structural diagram in UML that represents the organization and dependencies among components in a system. Its main components include:

1. Component: Represents a modular unit of software or system with defined functionality.

2. Interface: Defines the contract for communication and interactions between components.

3. Dependency: Indicates that one component depends on another for implementation or functionality.

4. Association: Represents a structural connection or link between components.

5. Aggregation: Describes a "has-a" relationship where one component contains other components.

6. Composition: Depicts a stronger "part-of" relationship where parts are exclusive to the whole component.

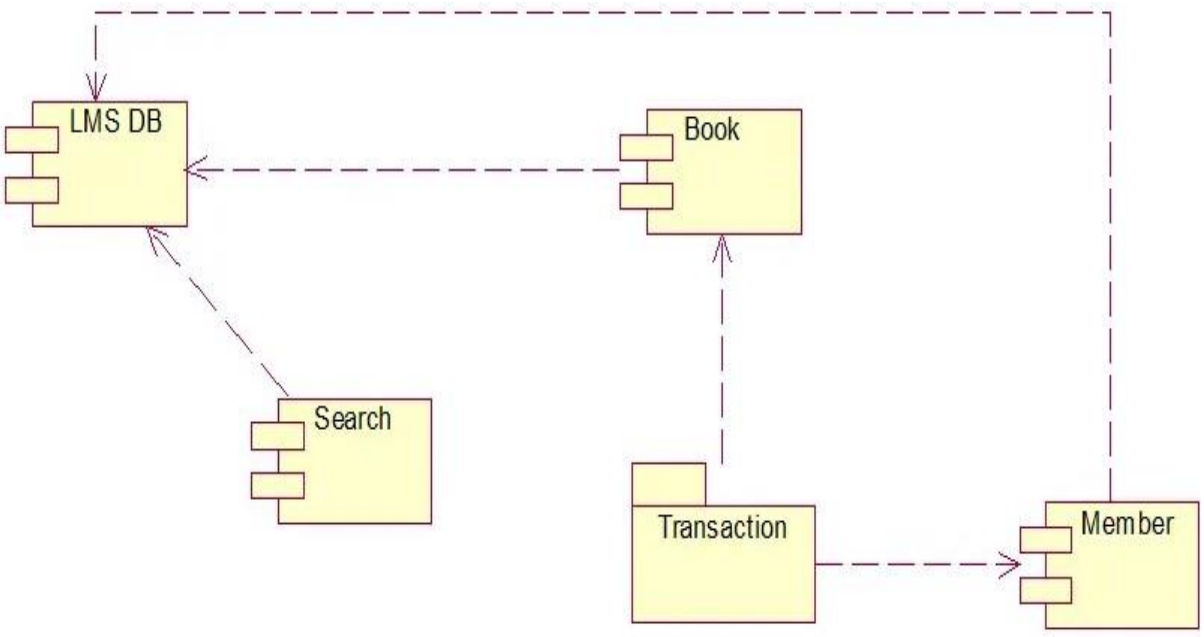
7. Realization: Shows the implementation of an interface by a component.

8. Generalization: Represents inheritance between components, where one extends the functionality of another.

9. Package: A grouping mechanism for organizing related components.

Component diagrams provide a visual overview of the structure and relationships of components, aiding in system design and development.

LIBRARY MANAGEMENT SYSTEM



PRACTICAL 9

Draw the Deployment diagram

A deployment diagram in UML represents the physical deployment of software components onto hardware nodes. Its components include:

1. Node: Represents a physical computing resource or device.

2. Component: Represents a modular unit of software or system functionality.

3. Deployment Relationship: Shows how components are deployed to nodes.

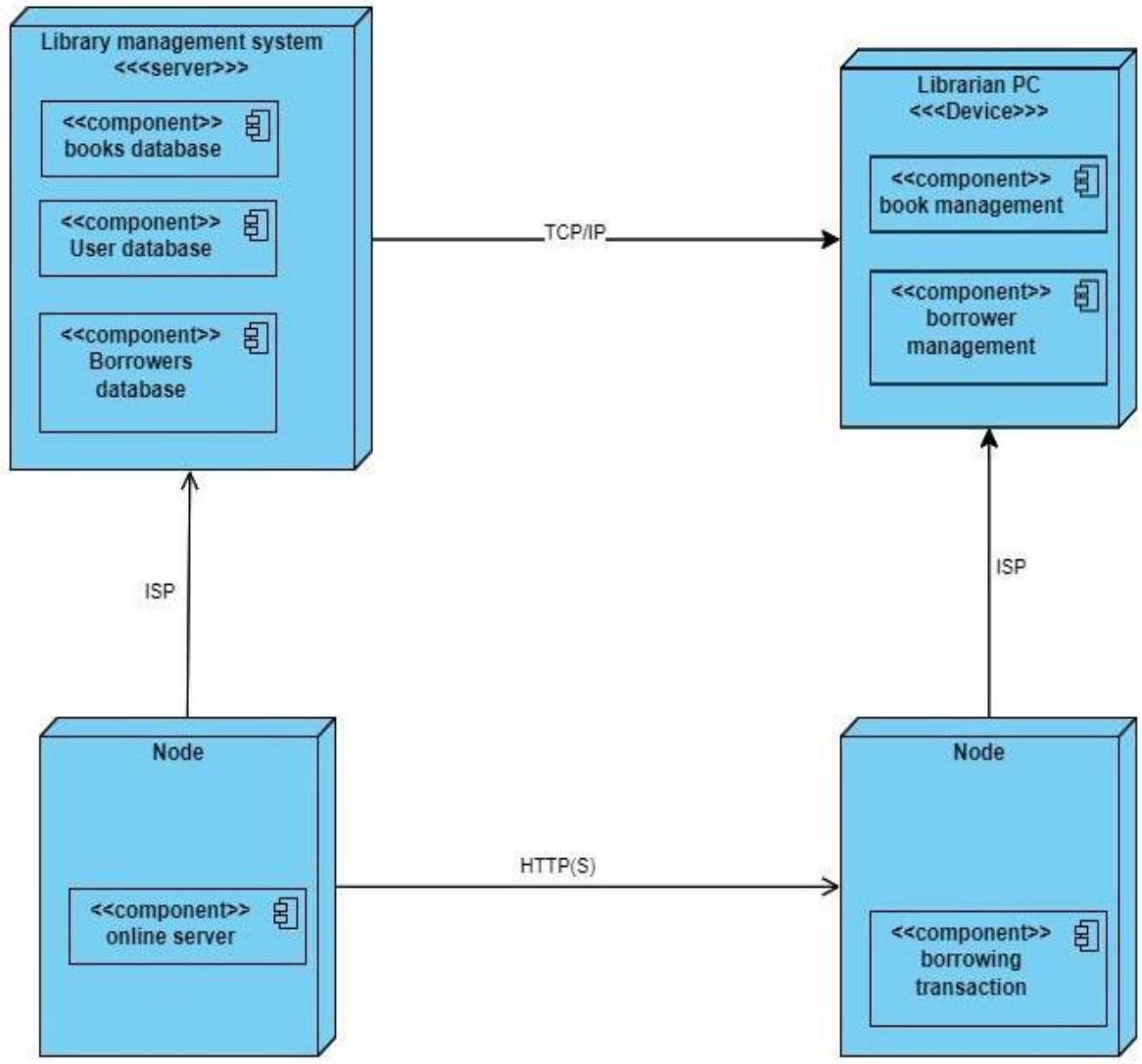
4. Communication Path: Depicts the channels or networks between nodes.

5. Artifact: Represents physical files or data resources used or produced by components.

6. Stereotype: Notations to extend or modify UML elements.

Deployment diagrams illustrate the physical architecture and distribution of components, aiding in discussions about system deployment, scalability, and hardware requirements.

LIBRARY MANAGEMENT SYSTEM



PRACTICAL 10

Perform Measurement of complexity with Halstead Metrics.

Halstead's software metrics is a set of measures proposed by Maurice Halstead to evaluate the complexity of a software program.

These metrics are based on the number of distinct operators and operands in the program, and are used to estimate the effort required to develop and maintain the program.

PROGRAM:

```
#include <stdio.h>

int main() {

  int r, c, a[100][100], b[100][100], sum[100][100], i, j;

  printf("Enter the number of rows (between 1 and 100): ");

  scanf("%d", &r);

  printf("Enter the number of columns (between 1 and 100): ");

  scanf("%d", &c);


  printf("\nEnter elements of 1st matrix:\n");

  for (i = 0; i < r; ++i)

    for (j = 0; j < c; ++j) {

      printf("Enter element a%d%d: ", i + 1, j + 1);

      scanf("%d", &a[i][j]);

    }


  printf("Enter elements of 2nd matrix:\n");

  for (i = 0; i < r; ++i)

    for (j = 0; j < c; ++j) {

      printf("Enter element b%d%d: ", i + 1, j + 1);

      scanf("%d", &b[i][j]);

    }


  // adding two matrices

  for (i = 0; i < r; ++i)

  for (j = 0; j < c; ++j) {

      sum[i][j] = a[i][j] + b[i][j];

    }
```

```
  // printing the result
  printf("\nSum of two matrices: \n");
  for (i = 0; i < r; ++i)
    for (j = 0; j < c; ++j) {
      printf("%d   ", sum[i][j]);
      if (j == c - 1) {
        printf("\n\n");
      }
    }

  return 0;
}
```

Halstead metrics Operands and operators table

operators	occurrence	operands	ocurrence
Int	2	r	6
()	23	c	7
{}	6	a	3
,	15	b	3
;	32	sum	3
&	4	i	21
=	9	j	22
<	8	0	9
++	8	1	5
[]	18		
+	5		
==	1		
-	1		
return	1		
for	8		
if	1		
n1=16	**N1=142**	**n2=9**	**N2=79**

- Calculate the program length (N),
  N= N1+N2= 142 + 79 = 221

- Calculate the program vocabulary (n),
  n = n1 +n2 = 16 + 9 = 25

- Calculate the program volume (V),
  V= N * $\log_2$ n = 221 * $\log_2$ (25) = 1026.29222

- Calculate the program difficulty (D),
  D = (N1 / 2) * (n2 / N2) = (221 / 2) * (9 / 79) = 12.588

- Calculate the program effort (E),
  E = D * V= 12.588 * 1026.29222= 12918.9385

The Halstead metrics for the given addition program are as follows:

- Program length (N): 221
- Program vocabulary (n): 25
- Program volume (V): 1026.29222
- Program difficulty (D): 12.588
- Program effort (E): 12918.9385

These metrics provide insights into the complexity and effort required to understand and maintain the program.