

Part I
BOOK BANK SYSTEM

A Book Bank lends books and magazines to member, who is registered in the system. Also it handles the purchase of new titles for the Book Bank. Popular titles are brought into multiple copies. Old books and magazines are removed when they are out of date or poor in condition. A member can reserve a book or magazine that is not currently available in the book bank, so that when it is returned or purchased by the book bank, that person is notified. The book bank can easily create, replace and delete information about the titles, members, loans and reservations from the system.

Task 1

Design of problem statement, perform a detailed feasibility study and finalize the process model to be used.

Aim:

The aim is to design a problem statement, perform a detailed feasibility study and finalize the process model to be used.

PROBLEM STATEMENT:

In an Electronic Cash Counter system, the user enters the secret code being verified with the database of the customers. If a match is found, then the transaction can be performed. If the secret code does not match, the user is requested to re-enter the code and try again. The users must be displayed transaction options such as withdrawal, balance enquiry, mini statement, and pin change. The users can withdraw the amount from their account if the amount is available in their account. The users can also deposit the amount in their account either by cash or cheque. The users can check their balance in their account through balance enquiry and also they can get the mini statement for recent transaction. The pin change if needed by the user can also be done. After the completion of transaction process, the system should display the balance amount to the user and provided the facility to print the receipt of the same.

SCOPE:

The Scope of Electronic Cash Counter system is to allow users to securely withdraw, deposit, check balances, view mini statements, and change PINs. It ensures user-friendly navigation, data security, and compliance with financial regulations.

FEASIBILITY STUDY:

Feasibility Study is used to evaluate a project's potential, including the technical, financial, and economic aspects, and to determine whether it should proceed.

- **Technical Feasibility:** The technical feasibility study always focuses on the existing computer hardware, software and personal. This also includes need for more hardware, software or personal and possibility of procuring or installing such facilities. Electronic Cash Counter is a system that can work on single stand alone Pentium machine with 128 MB RAM, Hard disk drive size of 80 GB, mouse, monitor and keyboard & it also require internet connection to corresponding computer. The equipments are easily available in the market, so technically the system is very much feasible
- **Economical Feasibility:** This feasibility is useful to find the system development cost and checks whether it is justifiable. The cost overheads include software and hardware maintenance cost, training costs that includes cost required for manpower, electricity, stationary etc. The Electronic Cash Counter system will provide the right type of information at right time, and in the required format. This will save time required for decision-making and routine operations. Considering all these advantages, the cost overheads of the system are negligible. So the system is economically feasible.
- **Operational Feasibility :** It is also known as resource feasibility. The operation users of the system are expected to have minimum knowledge of computer. The developed system is simple to use, so that the user will be ready to operate the system. The Electronic Cash Counter system can be developed using JAVA programming language & Mysql database which is platform independent and user friendly. So the system is operationally feasible.

PROCESS MODEL

Agile Model



The Agile process model is preferred for developing an Electronic Cash Counter due to several reasons,

1. Iterative and Incremental Development:

Agile promotes an iterative and incremental development approach. This allows for continuous refinement and improvement of the system based on user feedback.

2. Flexibility to Changing Requirements:

Agile allows for flexibility in adapting to changes during the development process, ensuring the system aligns with the evolving requirements and market conditions.

3. Frequent User Involvement:

Agile emphasizes regular collaboration with end-users. In the context of an Electronic Cash Counter system, involving users throughout the development process ensures that the system meets their needs, enhances usability, and aligns with regulatory requirements.

4. Continuous Integration and Testing:

Agile encourages continuous integration and testing, allowing developers to identify and address issues early in the development cycle. This helps in ensuring the reliability and security of the software.

5. Rapid Delivery of Value:

Agile focuses on delivering a minimum viable product (MVP) quickly. In the case of an Electronic Cash Counter, this means that essential functionalities can be delivered and deployed rapidly, allowing users to start benefiting from the system sooner.

6. Adaptability to Risks:

The financial sector comes with inherent risks and uncertainties. Agile provides a framework for adapting to changes and managing risks effectively during the development process, ensuring that the system can respond to unforeseen challenges.

7. Close Collaboration Between Teams:

Agile encourages close collaboration among cross-functional teams. For a complex system like an Electronic Cash Counter, this collaboration is essential as it involves integrating hardware, software, security measures, and compliance considerations.

8. Customer Satisfaction:

Agile focus on customer satisfaction aligns with the goal of delivering a reliable and user-friendly Electronic Cash Counter system. Regular feedback and involvement of end-users contribute to a product that better meets their expectations.

Result:

Thus the problem statement has been defined, feasibility study has been explained in detail and the process model is finalized.

Task 2

Software Requirement Specification Document

Aim:

To prepare Software Requirement Specification Document for Book Bank system.

1	Introduction
1.1	Purpose
1.2	Scope
1.3	Definition, Acronyms and Abbreviation
1.4	Reference
1.5	Overview
2	System Description
2.1	Product Perspective
2.2	Product Functions
2.3	User Functions
2.4	System Constraints
2.5	System Dependencies
2.6	Requirements Subdomain
3	Specific System Requirements
3.1	Function Requirements
3.2	Non-Functional Requirements
3.3	External Interfaces
4	Appendices

1 Introduction

The Book Bank System version 1.0 is a sophisticated solution, aiming to revolutionize the way we borrow and lend books, fostering a culture of learning and literary exploration. the Book Bank System is a comprehensive platform designed to streamline the borrowing process, ensuring that books are readily available to all enthusiasts. This system goes beyond traditional library models, incorporating modern technology to enhance accessibility and user experience. The Book Bank System serves as a guiding light, opening the world of books to everyone.

1.1 Purpose

The purpose of this document is to illustrate the requirements of the project Book Bank Management System. The document gives the detailed description of both functional and non functional requirements proposed by the client. The document is developed after a number of consultants with the client and considering the complete requirements specifications of the given project. The final product of the team will be meeting the requirements if this document.

1.2 Scope

The Book Bank System will include features such as a digital catalog, online reservation, automated check-in/check-out, personalized recommendations, and resource management. It will be accessible to both users and administrators through a user-friendly interface.

1.3 Definitions, Acronyms, and Abbreviations

Member	The one who registers himself and purchase books from the bank.
Database	Database is used to store the details of members and books
Administrator	The one who verifies the availability of book and issue the

1.4 References

The references for the above software are as follows:-

- i. www.google.co.in
- ii. www.wikipedia.com

iii. IEEE. Software Requirements Specification Std. 30-1993.

1.5 Overview

- Chapter 1.0 discusses the purpose and scope of the software.
- Chapter 2.0 describes the overall functionalities and constraints of the software and user characteristics.
- Chapter 3.0 details all the requirements needed to design the software.

2. System description

2.1 Product Perspective

This book bank system replaces the traditional, manual book bank management by which lot of paper work will be reduced. This system will provide a search functionality to facilitate the resources. This search will be based on various categories viz book name. Also advanced search features are provided in order to search various categories. In this system the Admin should have a computer to view the details of the members. Admin should be able to see the number of books that the particular person has taken from the book bank, details of the book, returned or taken. This is the primary feature. Another feature is that the book bank admin is able to edit the details. The admin look after long time pending of books.

2.2. Product Function

There are two different users who will be using this product.

1) User (Member)

2) Admin

The Book Bank System serves as a comprehensive platform, encompassing key functions to revolutionize book lending and borrowing processes. With robust catalog management, it enables administrators to efficiently organize and maintain a digital collection of books, ensuring accurate information on titles, authors, and availability. The system facilitates user interaction through seamless account creation, authentication, and personalized profiles. Online reservation functionality streamlines the process, allowing users to reserve books conveniently with timely notifications. Personalized recommendations based on user profiles and resource management features, including usage reports for administrators, contribute to an optimized library collection. The system further allows administrators to configure settings, ensuring flexibility and customization.

2.3 User Function

a) Admin:

The admin can view the different categories of books available in the book bank. Also can view the list of books available in each category. The admin can take the book returned from member. Add books and their information of the books to the database. Edit the information of the existing books in the database. The admin can check the report of the issued book. Admin can access all the accounts of the students.

b) User (Member):

The member can view the list of books available in each category. Member can view the different categories of books available in the book bank. They can own a ID by registering in the book bank system. User can view the books issued. Member can put a request for new book and Can search for particular book. Also can view the history of books issued to him previously.

2.4 System constraints

System constraints outline limitations or restrictions that may impact the development or operation of the Book Bank System.

- **Compatibility Constraints:** The system must be compatible with modern web browsers such as Google Chrome, Mozilla Firefox, and Safari.
- **Budgetary Constraints:** The development and maintenance budget for the Book Bank System is limited, requiring cost-effective solutions and efficient resource utilization.
- **Infrastructure Requirements:** The system is dependent on a reliable internet connection for users to access online features and services.
- **User Access Constraints:** Users must have internet access to benefit from the online reservation and personalized recommendation features.
- **Hardware Limitations:** The system should be designed to run on standard hardware configurations, taking into account potential limitations in processing power and memory.

- **Data Privacy and Security Compliance:** The system must adhere to data privacy and security regulations, imposing constraints on the collection, storage, and processing of user data.
- **Software Dependency Constraints:** The system may rely on third-party software or libraries, and any dependencies should be clearly defined.
- **Accessibility Standards:** The Book Bank System must comply with accessibility standards to ensure usability for individuals with disabilities.
- **Backup and Recovery Constraints:** The system must implement regular backup procedures, and recovery mechanisms should be in place in case of data loss or system failures.

2.5 System dependencies

Book bank system depends on a designated database system for the storage and management of book information and user records. Network infrastructure is crucial for online features and communication between system components, while barcode scanning technology is essential for automated check-in/check-out processes.

The system may use third-party libraries or APIs for functionalities such as personalized book recommendations. Web technologies, including HTML, CSS, and JavaScript, are integral to the user interface and online features. Integration with an existing user authentication system, stable server infrastructure, and compliance with geographical data hosting and accessibility standards are additional dependencies. Backup and recovery mechanisms safeguard against data loss, and the choice of programming language and framework, along with potential integration with a payment gateway, further contribute to the system's functionality.

3. Specific system requirement

3.1 Functional Requirement

3.1.1 User Management- Users should be able to create accounts with unique identifiers. The system must authenticate users securely for access to personalized features.

3.1.2 Catalog Management - Administrators should have the ability to add, modify, or remove books from the digital catalog. The system must store and display comprehensive information about each book, including titles, authors, genres, and availability status.

3.1.3 Search and Navigation - Users should easily search for books based on titles, authors, genres, or keywords. Users should be able to filter search results based on availability, genre, or other relevant criteria.

3.1.4 Online Reservation System - Users must log in to reserve books. Users should be able to reserve books online, and the system should notify them when reserved books are available for pickup.

3.1.5 Automated Check-in/Check-out - Automated check-in and check-out processes should be enabled using barcode scanning technology.

3.1.6 Personalized Recommendations - Users should be able to create and manage profiles with reading preferences. The system should suggest relevant book titles based on user profiles and preferences.

3.1.7 Resource Management- Administrators should generate reports on the usage patterns of books. Admin should be able to manage the book collection based on demand and usage reports.

3.2 Non Functional requirement

3.2.1 Performance- This system should work concurrently in multiple processors between the working hours in book bank. The system should support at least 500 users.

3.2.2 Safety- The database may get crashed at any certain time due to virus or operating system failure. Therefore, it is required to take the database backup.

3.2.3 Usability- The user interface shall be intuitive and user-friendly. The system should be accessible to users with disabilities.

3.2.4 Security- User login and access to sensitive data should be secure. All communication between the user's browser and the server should be encrypted.

3.3 External Interface

3.3.1 User Interface

The homepage should provide quick access to key functionalities such as book search, reservation, and user login. The search interface must allow users to search for books using various criteria, and the results should be presented in a clear and organized manner. Each book detail page

should display comprehensive information, including the book cover, author, and availability status. Users should be able to easily navigate to their profile, where they can view their borrowing history and personalized book recommendations.

3.3.2 Hardware Interface

- **User Devices-**The system should be compatible with standard personal computers, laptops, and mobile devices for user access.
- **Barcode Scanners-** Self-service kiosks require compatible barcode scanners for automated check-in and check-out processes.
- **Network Compatibility-** A stable network infrastructure is essential to facilitate communication between the server and user devices.

3.3.3 Software interface

The software interfaces are specific to the target book bank software systems.

- Languages supported: java (Front end)
- Database: SQL Server (Back end)
- MS-Office
- ArgoUml /StarUml

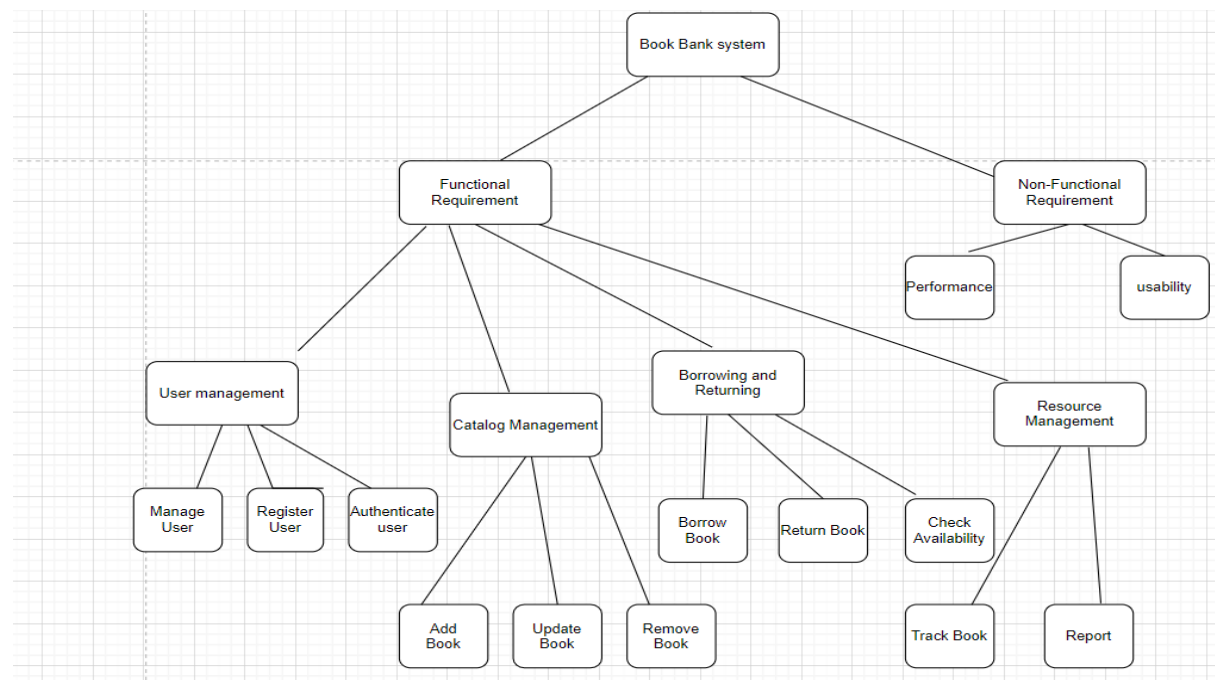
3.3.4 Communication Interface

These are protocols that are needed to directly interact with the customers. Apart from these protocols, to maintain a healthy relationship with the customer, both formal and informal meetings, group discussions and technical meetings will be conducted frequently

4. Appendices

Appendix A: Definition

Appendix B: Database Schema



Result:

Thus the Software Requirement Specification Document for Book Bank system has been Prepared.

Task 3

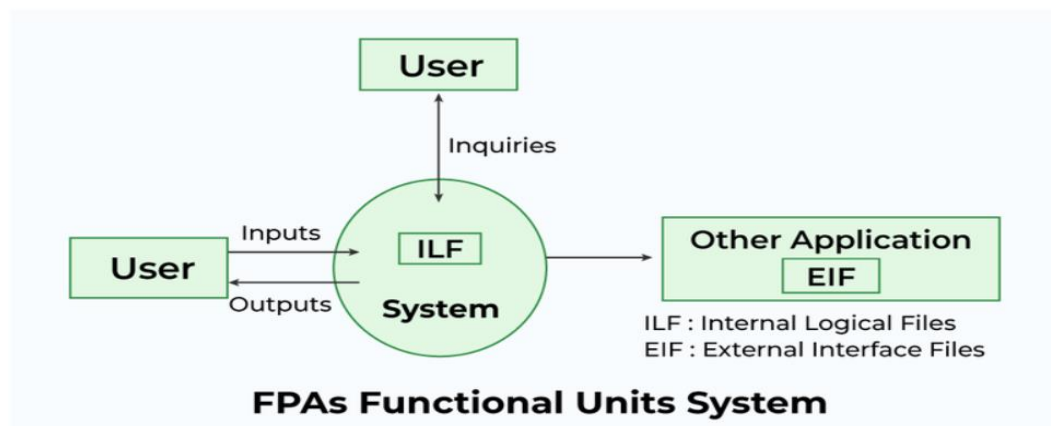
To Prepare a detailed estimate using the FP and COCOMO Model. Also prepare a detailed schedule of the project.

Aim:

To perform the estimation (FP and COCOMO) and to develop a project schedule.

FP Model:

The functional size of the product is measured in terms of the function point, which is a standard of measurement to measure the software application.



- **Step-1:**
 $F = 14 * \text{scale}$
Scale varies from 0 to 5 according to character of Complexity Adjustment Factor (CAF).
Below table shows scale:
0 - No Influence
1 - Incidental
2 - Moderate
3 - Average
4 - Significant
5 - Essential
- **Step-2:** Calculate Complexity Adjustment Factor (CAF).
 $CAF = 0.65 + (0.01 * F)$
- **Step-3:** Calculate Unadjusted Function Point (UFP).
TABLE (Required)

Function Units	Low	Avg	High
EI	3	4	6
EO	4	5	7
EQ	3	4	6
ILF	7	10	15
EIF	5	7	10

Multiply each individual function point to corresponding values in TABLE.

- **Step-4:** Calculate Function Point.

$$FP = UFP * CAF$$

Computing the function point when all complexity adjustment factor (CAF) and weighting factors are average, Also computing the productivity, documentation, cost per function for the following data:

1. Number of user inputs = 24
2. Number of user outputs = 46
3. Number of inquiries = 8
4. Number of files = 4
5. Number of external interfaces = 2
6. Effort = 36.9 p-m
7. Technical documents = 265 pages
8. User documents = 122 pages
9. Cost = \$7744/ month

Solution:

Solution :

Step 1 : Scale = 3

$$\begin{aligned} F &= 14 * \text{scale} \\ &= 14 * 3 \\ &= 42 \end{aligned}$$

Step 2 :

$$\begin{aligned} CAF &= 0.65 + (0.01 * F) \\ &= 0.65 + (0.01 * 42) \\ &= ~~0.65~~ 1.07 \end{aligned}$$

Step 3 :

$$\begin{aligned} UFP &= 24 \times 4 + 46 \times 5 + 8 \times 4 + 4 \times 10 + \\ &\quad 2 \times 3 \\ &= 96 + 230 + 32 + 40 + 14 \\ &= 412 \end{aligned}$$

Step 4 :

$$\begin{aligned} FP &= UFP * CAF \\ &= 412 * 1.07 \\ &= 440.84 \end{aligned}$$

$$\text{productivity} = \frac{FP}{\text{effort}} = \frac{440.84}{36.9} = 11.9$$

Functional point (FP) Analysis

$$\begin{aligned}
 \text{Total pages of documentation} &= \text{technical docum} \\
 &\quad \text{- ent + user document} \\
 &= 265 + 122 = 387 \text{ pages} \\
 \text{Documentation} &= \text{pages of documentation} / \text{FP} \\
 &= 387 / 440.84 = 0.877 \\
 \text{Cost per function} &= \frac{\text{cost}}{\text{productivity}} = \frac{7744}{11.9} \\
 &= \$650.75
 \end{aligned}$$

COCOMO Model

A project was estimated to be 400 KLOC. The effort and development time for each of the three model i.e., organic, semi-detached & embedded is calculated by basic COCOMO.

The basic COCOMO equation takes the form:

$$\text{Effort} = a * (\text{KLOC})^b \text{ PM}$$

$$\text{Tdev} = c * (\text{efforts})^d \text{ Months}$$

Project	a_b	b_b	c_b	d_b
Organic	2.4	1.05	2.5	0.38
Semi detected	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Estimated Size of project = 400 KLOC

(i) Organic Mode

$$E = 2.4 * (400)^{1.05} = 1295.31 \text{ PM}$$

$$D = 2.5 * (1295.31)^{0.38} = 38.07 \text{ PM}$$

(ii) Semidetached Mode

$$E = 3.0 * (400)^{1.12} = 2462.79 \text{ PM}$$

$$D = 2.5 * (2462.79)^{0.35} = 38.45 \text{ PM}$$

(iii) Embedded Mode

$$E = 3.6 * (400)1.20 = 4772.81 \text{ PM}$$

$$D = 2.5 * (4772.8)0.32 = 38 \text{ PM}$$

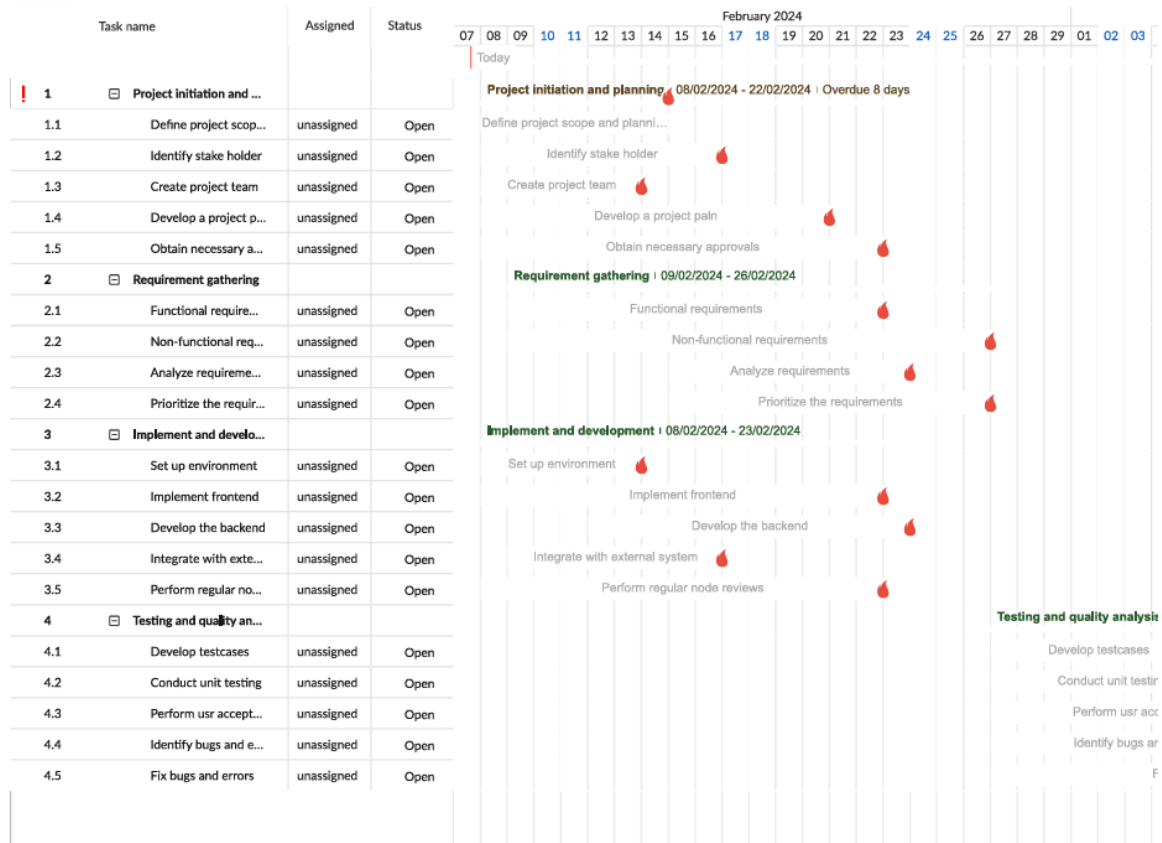
Project scheduling:

A project schedule is a timetable that organizes tasks, resources and due dates in an ideal sequence so that a project can be completed on time.

2/7/24, 2:55 PM

GanttPRO

GANTT EXPORT



Result:

Thus, the estimation (FP and COCOMO) has been performed project schedule is prepared.

Task 4

Identify Use Cases and develop the Use Case model for the given scenario Draw the relevant Entity Relationship Diagram and class diagram.

Aim:

The aim is to Identify Use Cases and develop the Use Case diagram, Entity Relationship Diagram and class diagram for the given scenario.

Use case diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

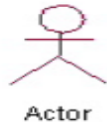
Use case

A use case describes the sequence of actions a system performs yielding visible results



Actor

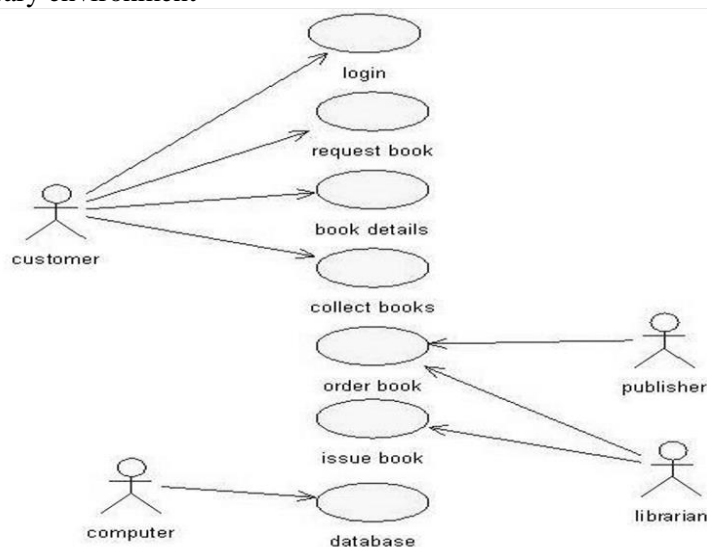
An actor represents the roles that the users of the use cases play. An actor may be a person (e.g. student, customer), a device (e.g. workstation), or another system (e.g. bank, institution).



The actor identified is

- Librarian
- member
- System

The use case diagram for the Book Bank System depicts various actors and their interactions with the system. The primary actors include "Librarian" who can perform actions such as adding, deleting, and updating book records, issuing and returning books, and managing user accounts. Another actor is the "Member" who can search for books, borrow and return them, and manage their account information. Additionally, there is the "Admin" who oversees system configurations, user management, and generating reports. The system facilitates functions like book management, user account management, book borrowing, and returning processes. These interactions encapsulate the core functionalities of the Book Bank System, ensuring efficient management of books and user activities within the library environment

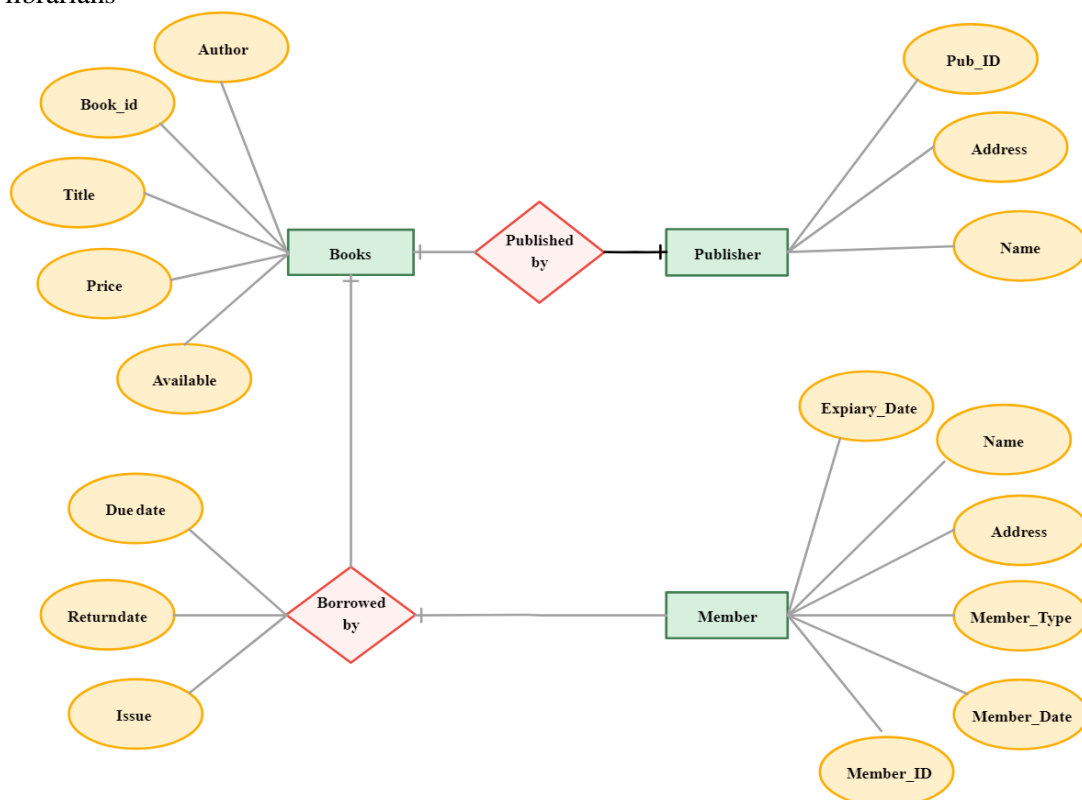


E-R Diagram

- The Entity-Relation model represents real-world entities and the relationship between them. An ER diagram is used to design database data models.
- Components of a ER Diagram
 - ✓ Entities
 - ✓ Attributes
 - ✓ Relationship

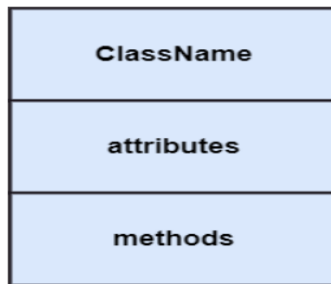


The ER diagram for the Book Bank System outlines how books, members, librarians, and transactions are interlinked within the system's database. Books are uniquely identified and contain details like title and author, while members are identified by memberID and store personal information. Librarians manage transactions, which record borrowing and returning activities, including dates and statuses. The diagram shows how members borrow multiple books through transactions, managed by librarians



Class Diagram

Class Diagram a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects



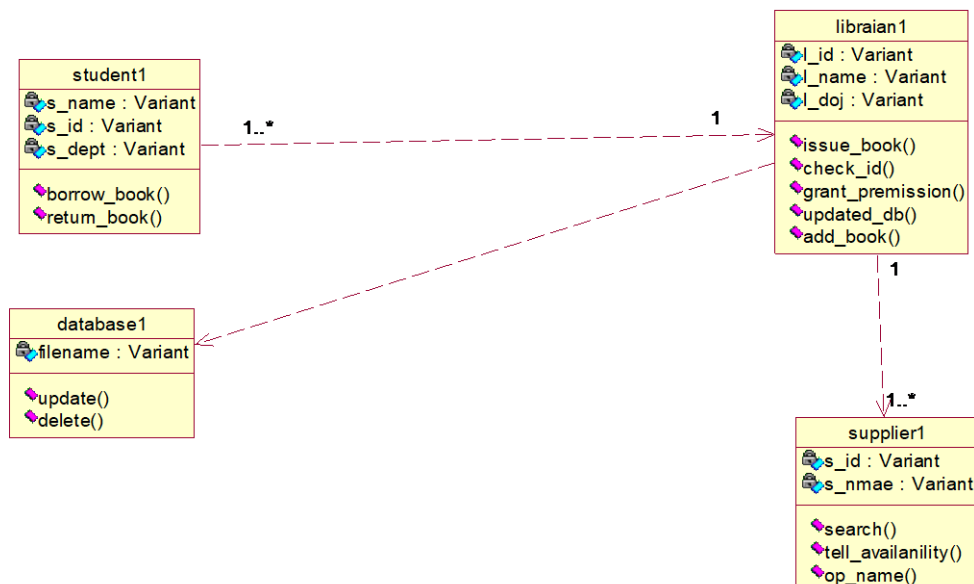
The purpose of the class diagram can be summarized as:

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering.

Class diagrams commonly contain the following things

- Classes
- Interfaces
- Collaborations
- Dependency, generalization and association relationships

The class diagram includes three main classes: Book, Member, and Librarian. The Book class represents individual books with attributes such as bookID, title, author, ISBN, and availability. It also includes methods for checking out, returning, updating availability status, and displaying book details. The Member class represents library members with attributes like memberID, name, contact, and a list of borrowed books. It includes methods for borrowing, returning, and displaying borrowed books. The Librarian class represents the librarian with attributes such as librarianID and name. It includes methods for adding, removing, and updating book details, as well as issuing and returning books for members.



Result

Thus the Use Cases has been identified and the Use Case diagram, Entity Relationship Diagram and class diagram for the given scenario has been designed

Task 5

Interaction Diagram (UML Sequence & Communication diagram).

Aim:

The aim is to find the interaction between objects and represent them using UML Sequence & Communication diagrams.

Interaction diagrams

Interaction diagram depict interactions of objects and their relationships. They also include the messages passed between them. There are two types of interaction diagrams –

- Sequence Diagram
- Collaboration Diagram

Interaction diagrams are used for modeling

- the control flow by time ordering using sequence diagrams.
- the control flow of organization using collaboration diagrams.
-

Sequence diagram

A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur.

Drawing a sequence diagram helps to

- Represent the details of a UML use case.
- Model the logic of a sophisticated procedure, function, or operation.
- See how objects and components interact with each other to complete a process.
- Plan and understand the detailed functionality of an existing or future scenario.

Notations:

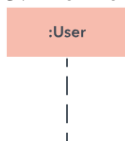
1.Object



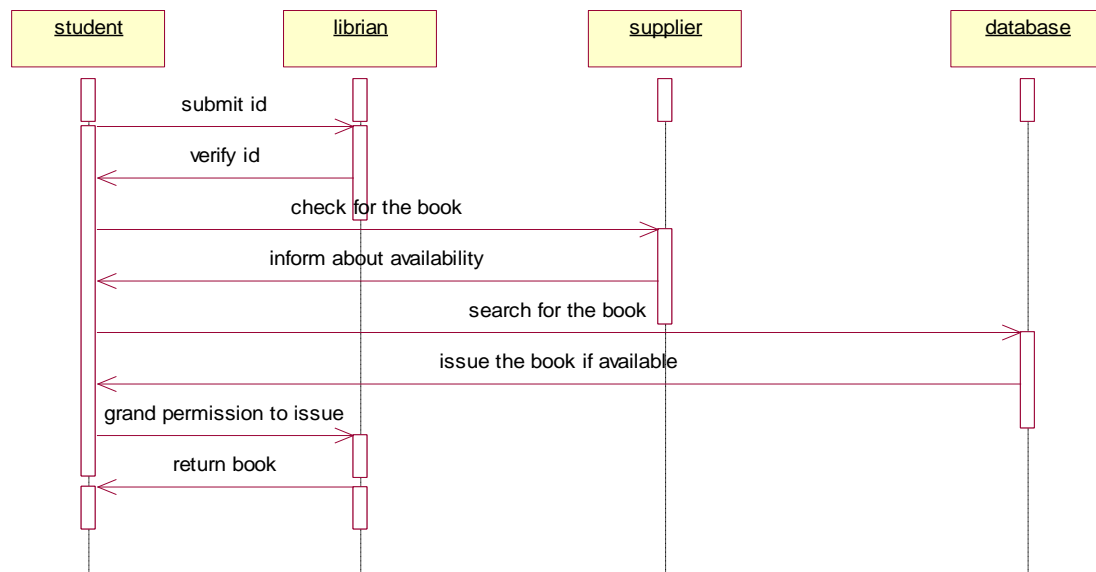
2.Actor



3.Lifeline



Book Bank System, the process of a member borrowing a book from the library can be outlined. Initially, the member sends a request to borrow a specific book, triggering a search for the book's availability within the system. Once the book is found to be available, the librarian is notified to proceed with the transaction. The librarian then verifies the member's credentials and checks out the book for the member. This action updates the book's availability status and generates a transaction record. Finally, the member receives confirmation of the successful borrowing transaction and can proceed with reading the book. This sequential diagram illustrates the step-by-step process of how a member interacts with the system to borrow a book from the library.



Collaboration Diagram

The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.

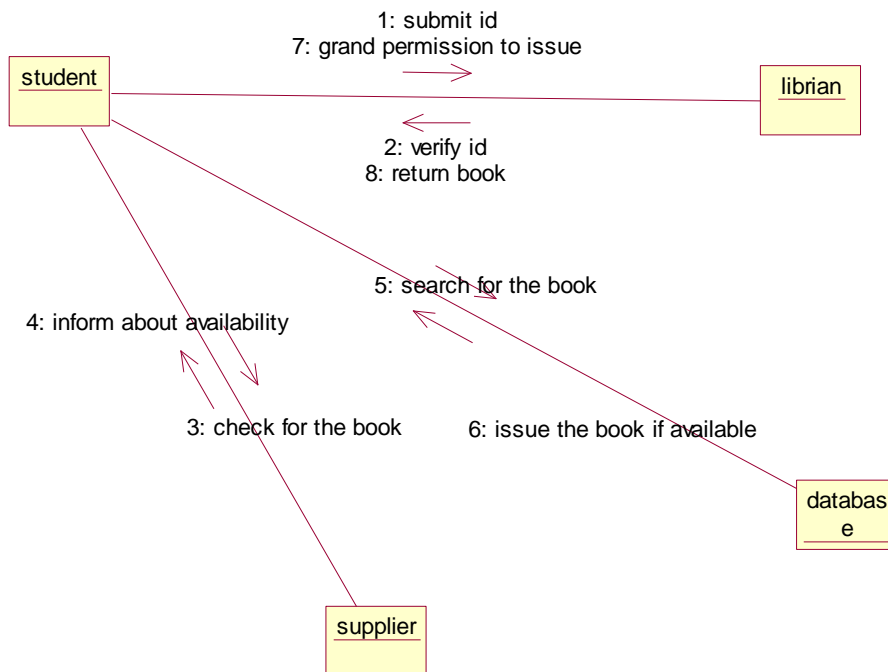
An object consists of several features. Multiple objects present in the system are connected to each other. The collaboration diagram, which is also known as a communication diagram, is used to portray the object's architecture in the system.

Notations

Notations used in communication diagrams are the same notations for sequence diagrams.

- Rectangles represent objects that make up the application.
- Lines between class instances represent the relationships between different parts of the application.
- Arrows represent the messages that are sent between objects.
- Numbering lets you know in what order the messages are sent and how many messages are required to finish a process.

In a communication diagram for the Book Bank System, the Member requests to borrow a book, directly communicating with the Librarian. The Librarian checks the book's availability and informs the Member. If available, the Librarian updates the book's status and records the transaction. Finally, confirmation is sent back to the Librarian, who notifies the Member of the successful transaction. This simple communication diagram illustrates the exchange of information and interactions between the various components of the Book Bank System during the book borrowing process.



Result

Thus, interaction between objects and UML Sequence & Communication diagrams for the given scenario has been designed.

Task 6

User Interface diagrams, State Transition diagram and Activity diagrams

Aim:

The aim is to model the User Interface diagrams, State Transition diagram and Activity diagram for the given scenario.

User Interface Diagram

User interface (UI) design is the process designers use to build interfaces in software or computerized devices, focusing on looks or style.

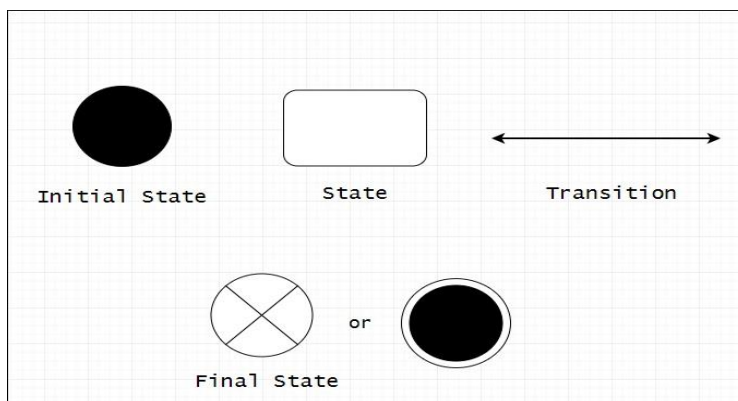
User interface design for a book bank system aims to make it easy and straightforward for users to find, borrow, and return books. It includes clear menus, search bars, and buttons for navigation, ensuring users can quickly locate books they need. The design prioritizes simplicity and ease of use, ensuring a pleasant experience for all users interacting with the system. Visual elements like buttons, icons, and color schemes are chosen to enhance usability and guide users through the process seamlessly. Additionally, the design ensures accessibility for users of all abilities, making it simple for anyone to navigate and utilize the book bank system effectively.



State Transition diagram

The state machine diagram is also called the Statechart or State Transition diagram, which shows the order of states underwent by an object within the system.

Notations



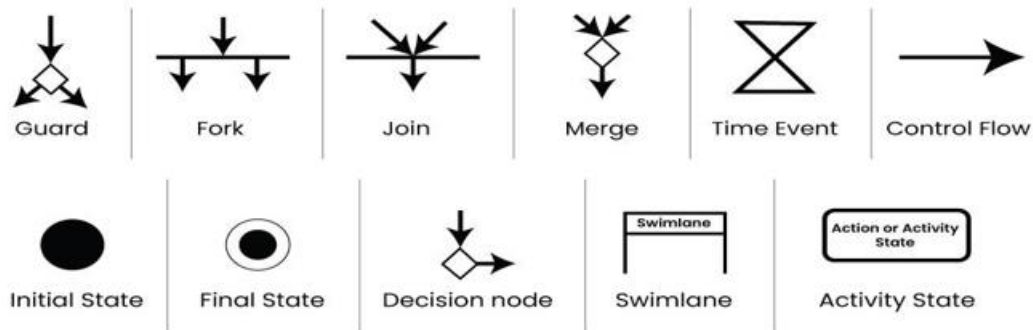
A state chart diagram for a book bank system illustrates the various states and transitions that the system and books can undergo. It typically starts with an "Idle" state where the system is ready to handle requests. When a user logs in, the system transitions to a "Logged In" state. From there, it may move to states like "Searching" when the user is looking for books, "Borrowing" when a book is selected for loan, and "Renewing" if the user extends the loan period. Once a book is returned, it reverts to the

"Available" state. The state chart simplifies understanding the system's behaviour and possible states during user interactions, ensuring smooth management of book borrowing and returning processes.

Activity diagram

Activity diagrams show the flow of one activity to another within a system or process

Notations



An activity diagram for a book bank system outlines the sequence of actions involved in borrowing and returning books. It typically begins with a user accessing the system and logging in. From there, the diagram branches into activities such as searching for books, selecting items for borrowing, and checking out. Subsequently, it may depict activities like renewing loans or returning books. Each action is represented by a sequential flow of steps, simplifying the process of how users interact with the book bank system from start to finish.

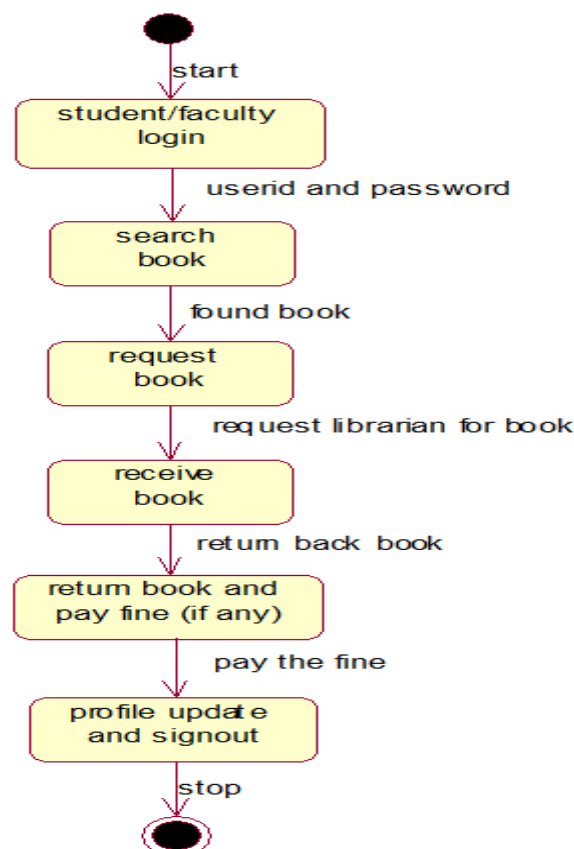


Figure: State Chart Diagram

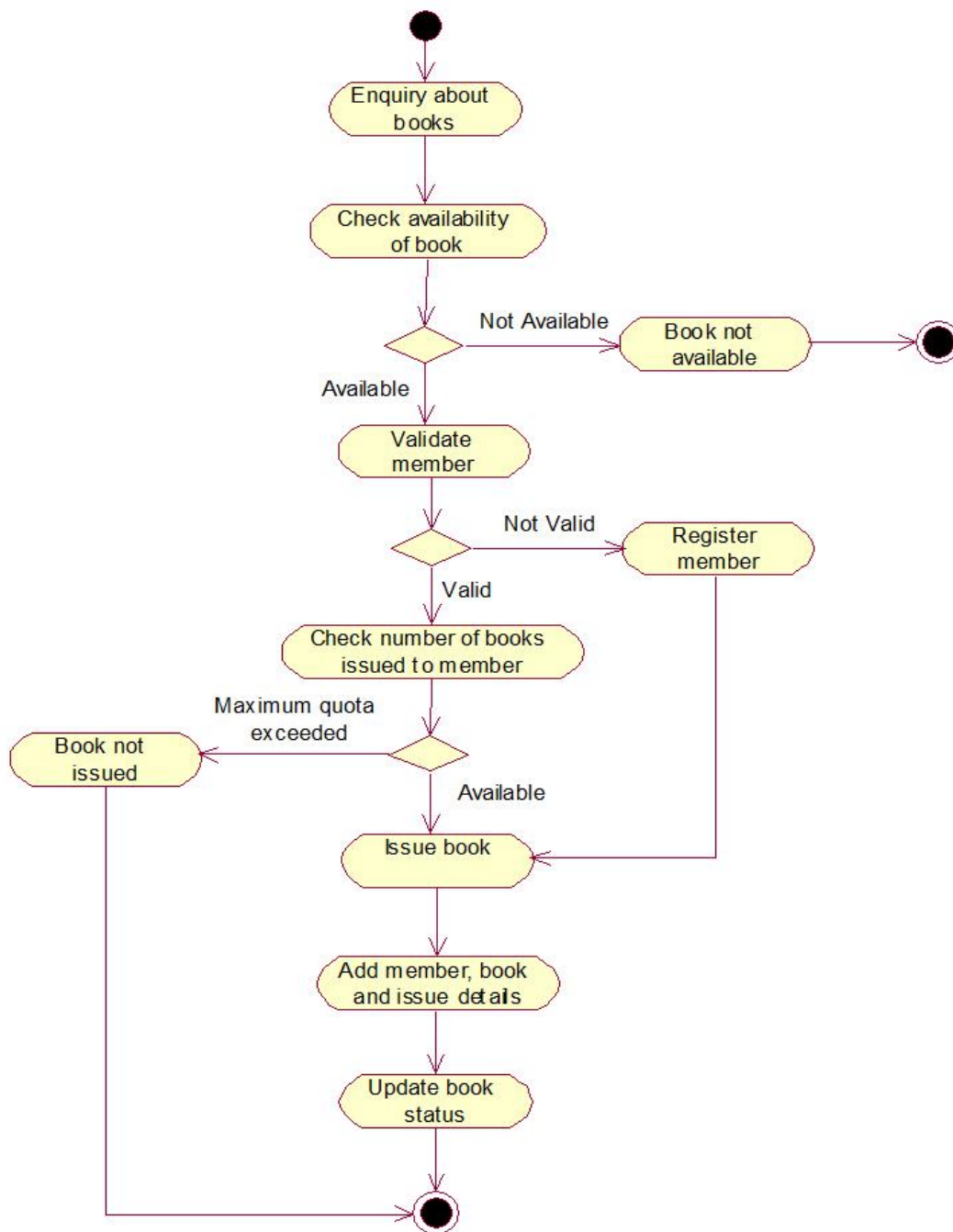


Figure: Activity Diagram

Result

Thus the User Interface diagrams, State Transition diagram and Activity diagram for the given scenario has been designed.

Task 7

Object Diagram and Software Architecture Diagram

Aim:

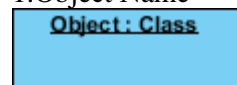
The aim is to implement partial layered, Object Oriented and logical software architecture diagram to visualize the high level design.

Object Diagram:

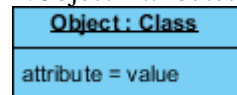
UML object diagram is an instance of a class diagram. An object diagram encompasses objects and their relationships which may be considered a special case of a class diagram or a communication diagram.

Object Diagram Symbols and Notations

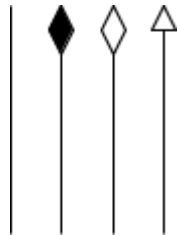
1.Object Name



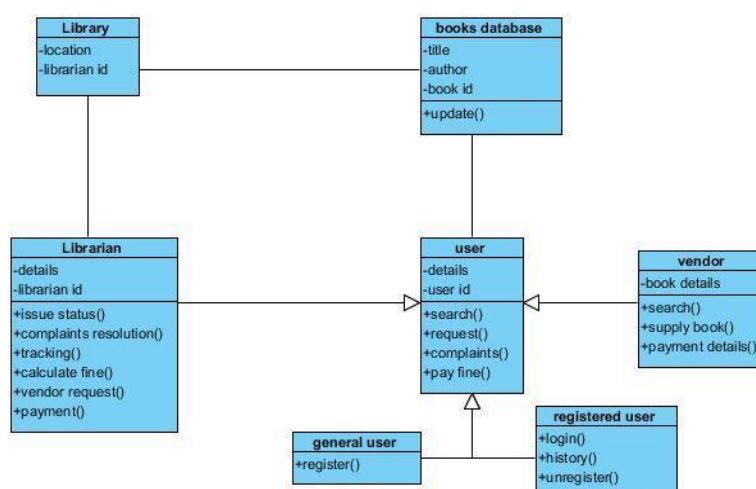
2.Object Attributes



3.Links



Diagram



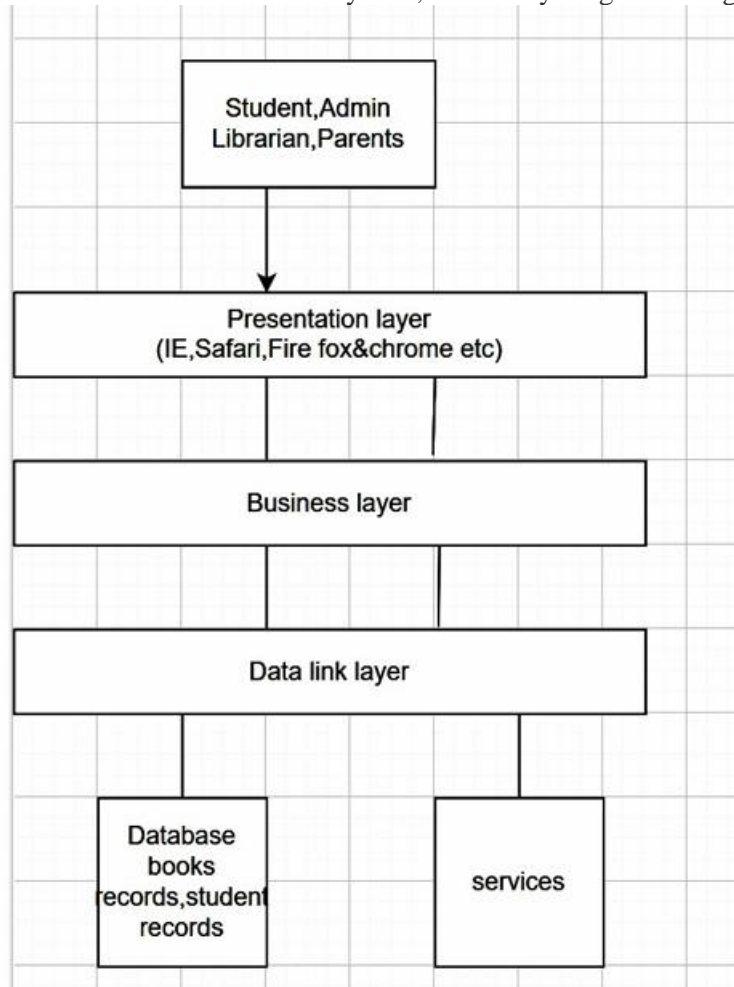
The Library class manages a collection of Book objects and Member objects. The Library class has methods for adding/removing books and members, and for searching for books and members.

Each Book object has attributes such as title, author, ISBN, and availability. The Book class has methods to get its title, author, ISBN, check availability, and set availability.

Each Member object has attributes such as name, member ID, and a list of books borrowed. The Member class has methods to get name, member ID, list of books borrowed, and methods to borrow and return books.

Architecture diagram

Software architecture is all about how a software system is built at its highest level. Software architecture provides a basic design of a complete software system. It defines the elements included in the system, the functions each element has, and how each element relates to one another. It is a big picture or overall structure of the whole system, how everything works together.



Result

Thus the Object Oriented and logical software architecture diagram to visualize the high level design for the given scenario has been designed.

Task 8

Data Flow Diagrams


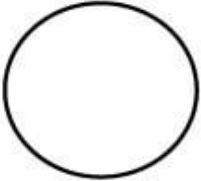

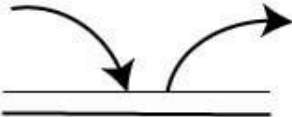
Aim:

The aim is to implement the Data Flow Diagrams to perform the data modelling with Level 0, Level 1, and Level 2 analysis.

Data Flow Diagram

A data flow diagram (DFD) maps out the flow of information for any process or system. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present.

Notations in DFD

Symbol	Name	Function
	Data flow	Used to Connect Processes to each other, to sources or Sinks; the arrow head indicates direction of data flow.
	Process	Performs Some transformation of Input data to yield output data.
	Source of Sink (External Entity)	A Source of System inputs or Sink of System outputs.
	Data Store	A repository of data; the arrow heads indicate net inputs and net outputs to store.

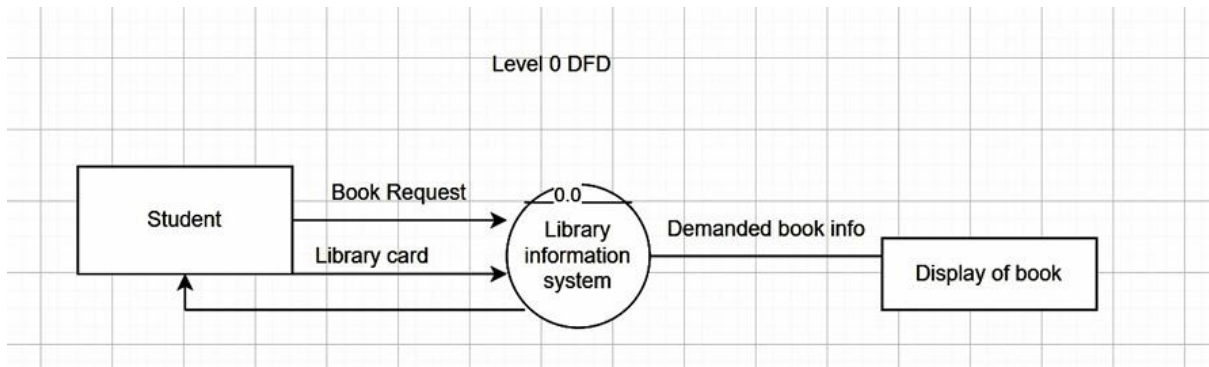
Symbols for Data Flow Diagrams

Levels of DFD

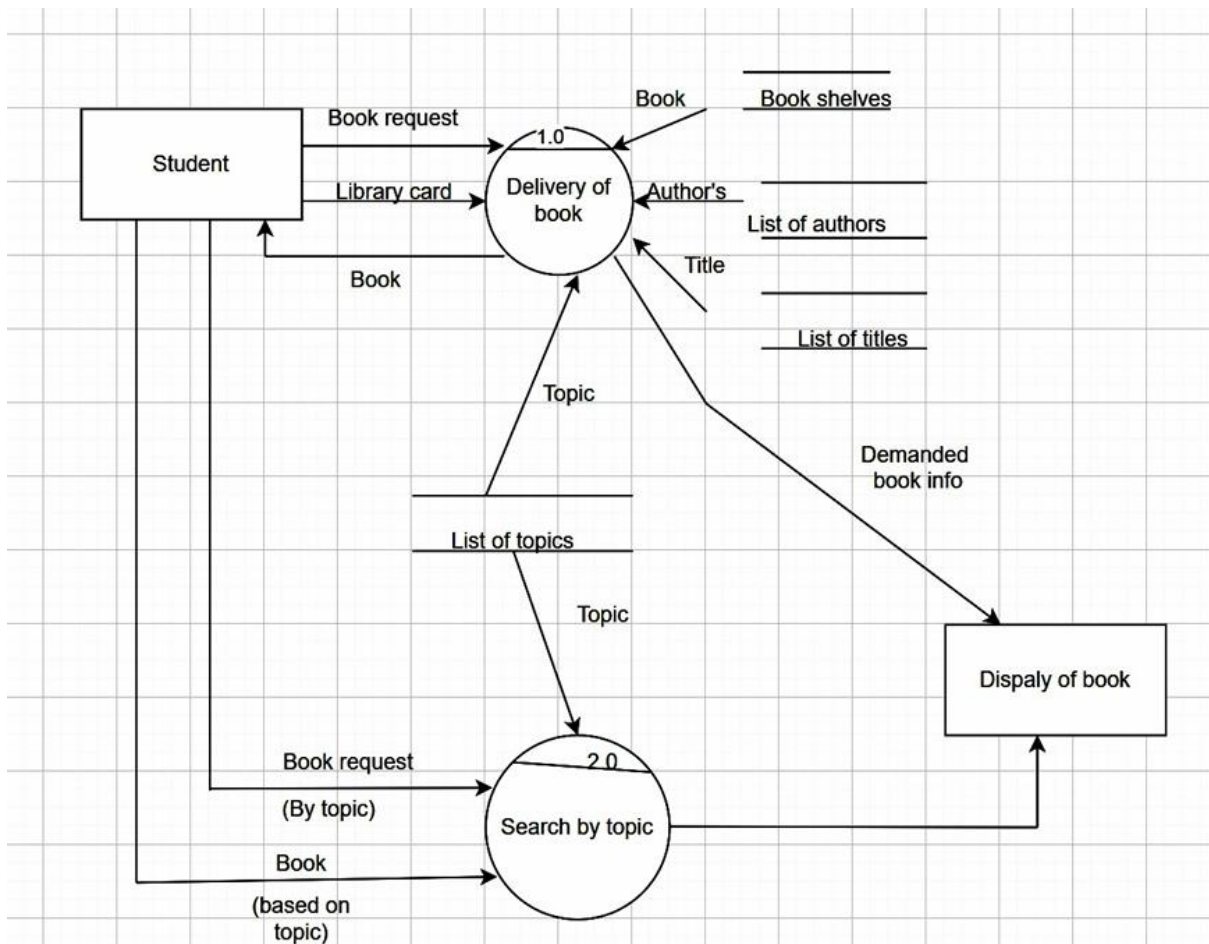
DFD uses hierarchy to maintain transparency thus multilevel DFD's can be created. Levels of DFD are as follows:

- 0-level DFD: It represents the entire system as a single bubble and provides an overall picture of the system.
- 1-level DFD: It represents the main functions of the system and how they interact with each other.
- 2-level DFD: It represents the processes within each function of the system and how they interact with each other.
- 3-level DFD: It represents the data flow within each process and how the data is transformed and stored.

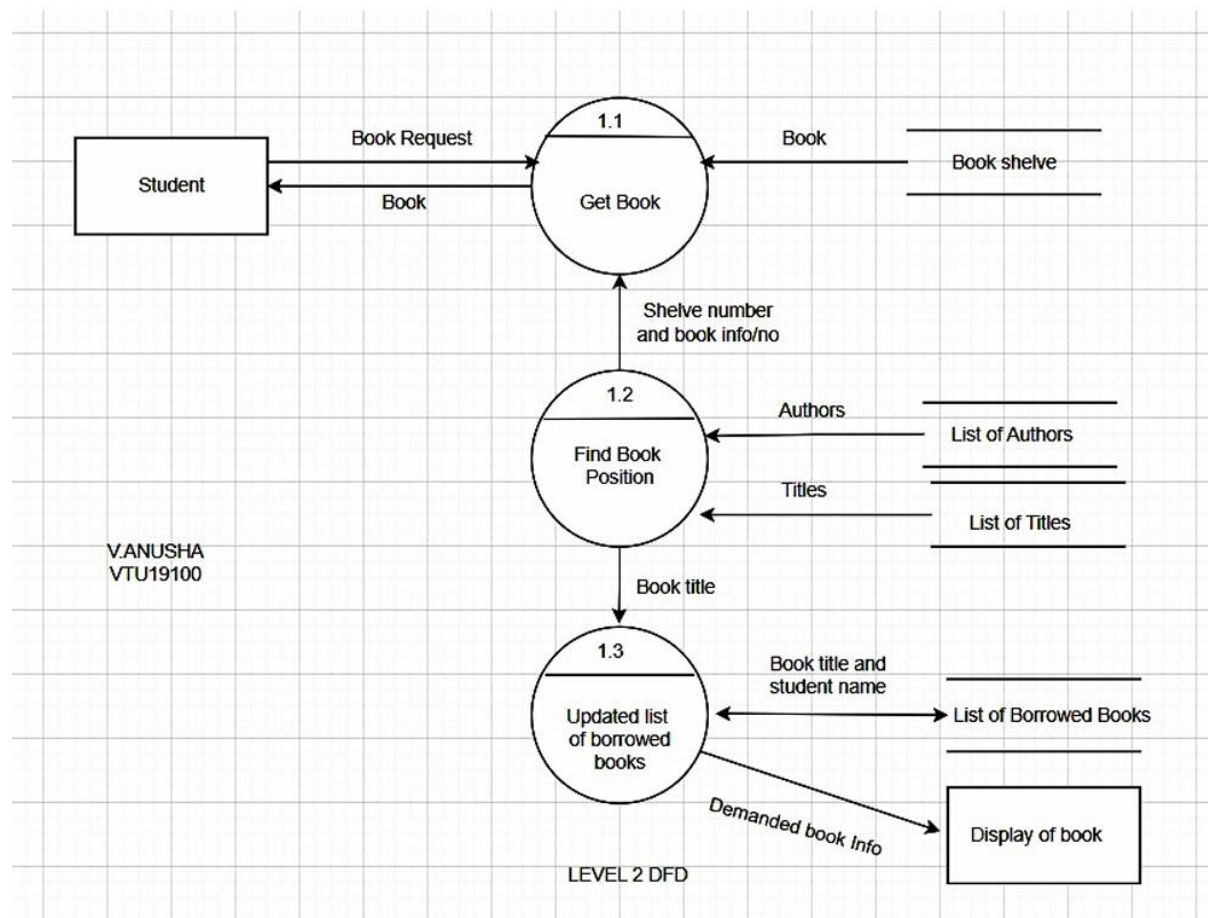
Level 0 DFD



Level 1 DFD



Level 2 DFD



Result

Thus the the Data Flow Diagrams to perform the data modelling with Level 0, Level 1 and Level 2 analysis for the given scenario has been designed.

Task 9

Test cases for both Black box Testing and White Box Test Testing

Aim:

The aim is to write the test cases for both Black box Testing and White Box Test Testing

Black Box Testing:

Black box testing for a book bank system involves testing the system's functionalities without knowing its internal code or structure. Here are some use cases for black box testing of a book bank system.

1.	User Registration:
	<ul style="list-style-type: none">• Test Case: Attempt to register a new user with valid information.<ul style="list-style-type: none">• Expected Result: Registration successful.• Pass/Fail: Pass• Test Case: Attempt to register a new user with invalid information (e.g., incomplete form).<ul style="list-style-type: none">• Expected Result: Registration fails with appropriate error message.• Pass/Fail: Pass
2.	User Login:
	<ul style="list-style-type: none">• Test Case: Attempt to log in with valid credentials.<ul style="list-style-type: none">• Expected Result: User successfully logged in.• Pass/Fail: Pass• Test Case: Attempt to log in with invalid credentials (e.g., wrong password).<ul style="list-style-type: none">• Expected Result: Login fails with appropriate error message.• Pass/Fail: Pass
3.	Book Search:
	<ul style="list-style-type: none">• Test Case: Search for a book by title.<ul style="list-style-type: none">• Expected Result: Relevant books are displayed.• Pass/Fail: Pass• Test Case: Search for a book by author.<ul style="list-style-type: none">• Expected Result: Books written by the specified author are displayed.• Pass/Fail: Pass
4.	Book Borrowing:
	<ul style="list-style-type: none">• Test Case: Attempt to borrow an available book.<ul style="list-style-type: none">• Expected Result: Book is successfully borrowed, and availability status is updated.• Pass/Fail: Pass• Test Case: Attempt to borrow an already borrowed book.<ul style="list-style-type: none">• Expected Result: Borrowing fails with appropriate error message.• Pass/Fail: Pass
5.	Book Return:
	<ul style="list-style-type: none">• Test Case: Return a borrowed book.<ul style="list-style-type: none">• Expected Result: Book is successfully returned, and availability status is updated.• Pass/Fail: Pass• Test Case: Attempt to return a book that hasn't been borrowed.<ul style="list-style-type: none">• Expected Result: Return fails with appropriate error message.• Pass/Fail: Pass
6.	Reservation:
	<ul style="list-style-type: none">• Test Case: Reserve a book that is currently unavailable.<ul style="list-style-type: none">• Expected Result: Reservation is successful, and user is notified when the book becomes available.

	<ul style="list-style-type: none"> • Pass/Fail: Pass
	<ul style="list-style-type: none"> • Test Case: Attempt to reserve a book that is already reserved by another user.
	<ul style="list-style-type: none"> • Expected Result: Reservation fails with appropriate error message. • Pass/Fail: Pass
7.	Fine Calculation:
	<ul style="list-style-type: none"> • Test Case: Simulate overdue book return and verify fine calculation.
	<ul style="list-style-type: none"> • Expected Result: Fine is calculated accurately based on the overdue duration. • Pass/Fail: Pass
	<ul style="list-style-type: none"> • Test Case: Return a book on time and verify no fine is applied.
	<ul style="list-style-type: none"> • Expected Result: No fine is applied. • Pass/Fail: Pass
8.	Admin Functionality:
	<ul style="list-style-type: none"> • Test Case: Add a new book to the system.
	<ul style="list-style-type: none"> • Expected Result: Book is successfully added to the inventory. • Pass/Fail: Pass
	<ul style="list-style-type: none"> • Test Case: Update book information.
	<ul style="list-style-type: none"> • Expected Result: Book details are updated correctly. • Pass/Fail: Pass

White Box Testing:

White box testing, also known as structural testing, involves examining the internal logic and structure of the software system. For a book bank system, white box testing ensures that the internal components, algorithms, and code paths function correctly. Here's how white box testing can be applied to a book bank system, along with use cases:

Here are some white box test cases for a Book Bank System along with the expected results:

1.	Login Functionality:
	<ul style="list-style-type: none"> • Test Case 1: Verify that valid credentials allow users to log in successfully.
	<ul style="list-style-type: none"> • Expected Result: User should be logged into the system and directed to the dashboard.
	<ul style="list-style-type: none"> • Test Case 2: Verify that invalid credentials result in appropriate error messages.
	<ul style="list-style-type: none"> • Expected Result: User should receive an error message indicating invalid credentials.
	<ul style="list-style-type: none"> • Test Case 3: Verify that the system prevents multiple failed login attempts within a short time frame.
	<ul style="list-style-type: none"> • Expected Result: After a certain number of failed attempts, the system should lock the user account or implement a delay before allowing further login attempts.
2.	Book Search Functionality:
	<ul style="list-style-type: none"> • Test Case 4: Verify that users can search for books by title, author, or ISBN.
	<ul style="list-style-type: none"> • Expected Result: Search results should include books matching the search criteria.
	<ul style="list-style-type: none"> • Test Case 5: Verify that the search results are accurate and relevant to the query.
	<ul style="list-style-type: none"> • Expected Result: Books returned in the search results should match the search query accurately.
	<ul style="list-style-type: none"> • Test Case 6: Verify that the system handles empty or invalid search queries gracefully.
	<ul style="list-style-type: none"> • Expected Result: The system should prompt the user to enter a valid search query or display a message indicating no results found.
3.	Book Checkout Process:
	<ul style="list-style-type: none"> • Test Case 7: Verify that users can add books to their checkout cart.
	<ul style="list-style-type: none"> • Expected Result: Selected books should be added to the user's checkout cart.

	<ul style="list-style-type: none"> Test Case 8: Verify that the system updates the availability status of books upon checkout.
	<ul style="list-style-type: none"> Expected Result: The availability status of checked-out books should be updated to reflect that they are no longer available for checkout.
	<ul style="list-style-type: none"> Test Case 9: Verify that users cannot checkout more books than their maximum allowed limit.
	<ul style="list-style-type: none"> Expected Result: The system should prevent users from checking out more books than their allowed limit and provide a relevant error message.
4. Book Return Process:	
	<ul style="list-style-type: none"> Test Case 10: Verify that users can return books they have checked out.
	<ul style="list-style-type: none"> Expected Result: Users should be able to return checked-out books, and the system should update their status accordingly.
	<ul style="list-style-type: none"> Test Case 11: Verify that the system updates the availability status of returned books.
	<ul style="list-style-type: none"> Expected Result: The availability status of returned books should be updated to indicate that they are available for checkout again.
	<ul style="list-style-type: none"> Test Case 12: Verify that overdue books are flagged appropriately upon return.
	<ul style="list-style-type: none"> Expected Result: If a returned book was overdue, the system should flag it as overdue and calculate fines accordingly.
5. User Account Management:	
	<ul style="list-style-type: none"> Test Case 13: Verify that users can update their personal information (e.g., email, password).
	<ul style="list-style-type: none"> Expected Result: Users should be able to successfully update their personal information, and the changes should be reflected in the system.
	<ul style="list-style-type: none"> Test Case 14: Verify that administrators can add new users to the system.
	<ul style="list-style-type: none"> Expected Result: Administrators should be able to add new users, and the new users should be able to log in with the provided credentials.
	<ul style="list-style-type: none"> Test Case 15: Verify that administrators can deactivate or suspend user accounts.
	<ul style="list-style-type: none"> Expected Result: Administrators should be able to deactivate or suspend user accounts, and deactivated/suspended users should not be able to log in.

Result

Thus the test cases for both Black box Testing and White Box Test Testing has be written successfully.

Task 10

Prototype of the Project

Aim:

The aim is to Develop a prototype of the project with database management system and deploy the project.

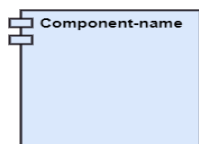
Component Diagram

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It visualizes the relationships as well as the organization between the components present in the system.

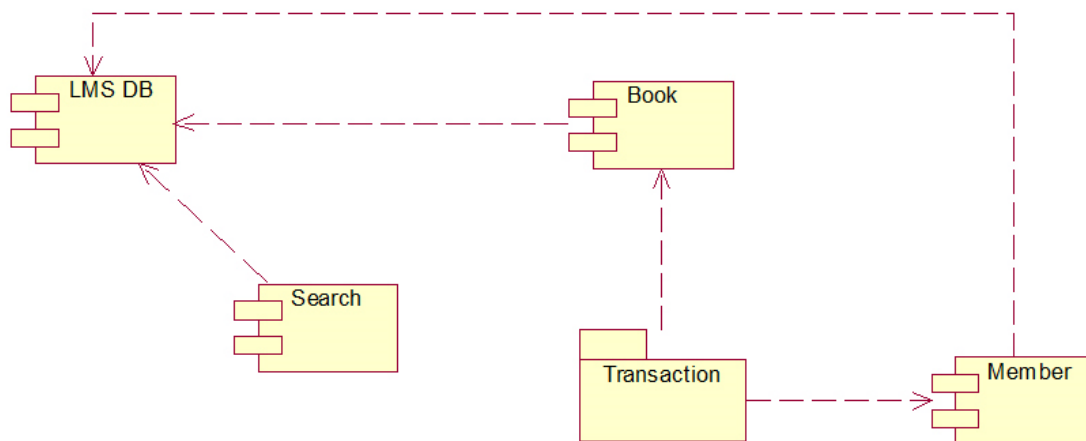
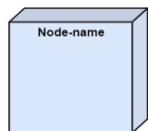
A component diagram for a book bank system illustrates the different components or modules that make up the system and how they interact with each other. The components identified are ECC Machine, card reader, PIN pad, cash dispenser, network interface.

Notations in component Diagram

a) A component



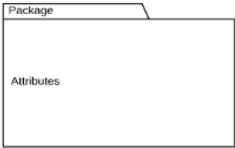

b) A node

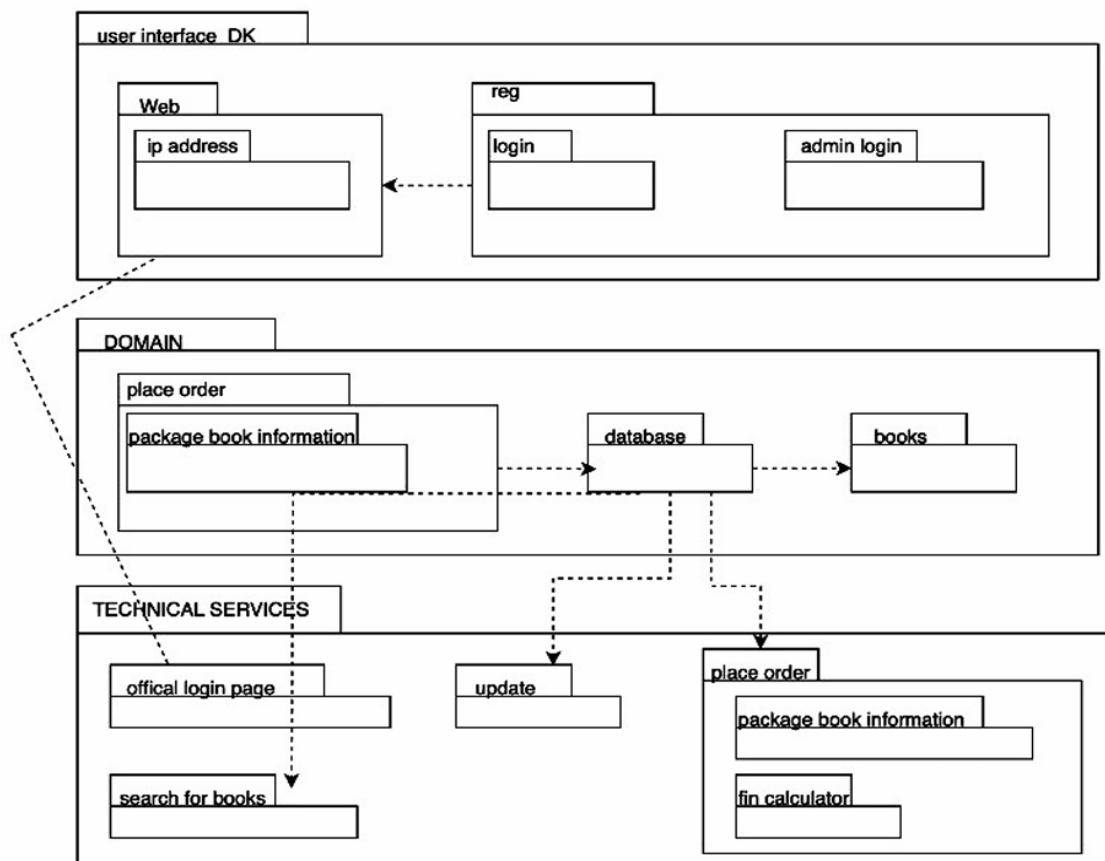


Package diagram

Package diagrams are structural diagrams used to show the organization and arrangement of various model elements in the form of packages.

Notation in package diagram

Symbol Image	Symbol Name	Description
	Package	Groups common elements based on data, behavior, or user interaction
	Dependency	Depicts the relationship between one element (package, named element, etc) and another



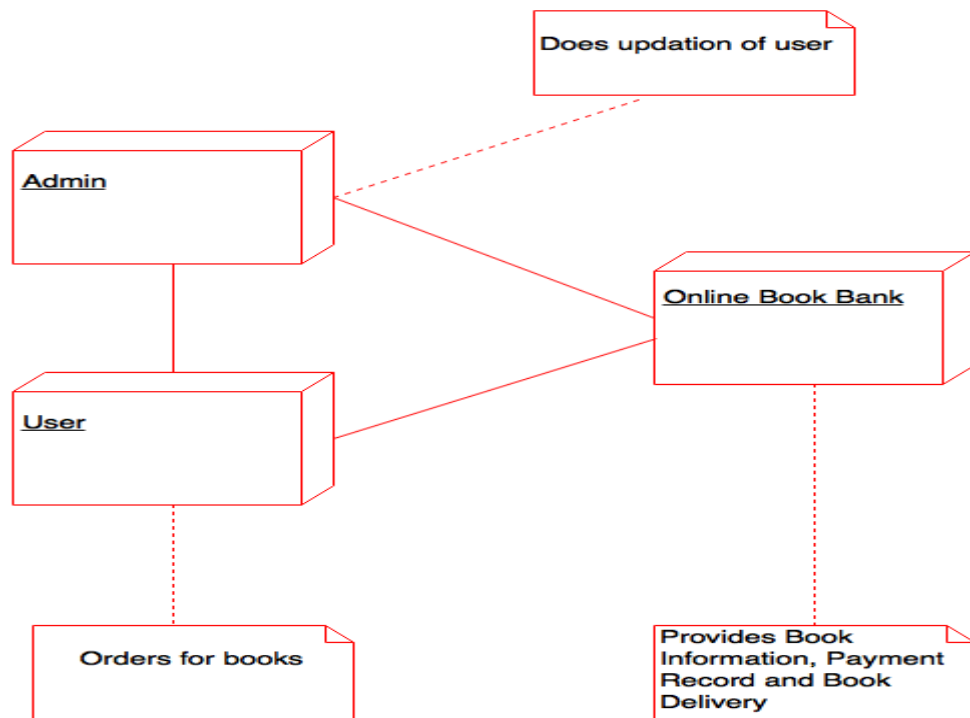
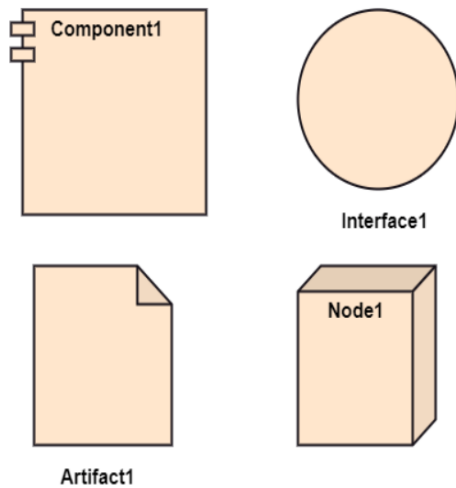
Deployment Diagram

The deployment diagram visualizes the physical hardware on which the software will be deployed. The main purpose of the deployment diagram is to represent how software is installed on the hardware component

Notations in Deployment Diagram

The deployment diagram consist of the following notations:

1. A component
2. An artifact
3. An interface
4. A node



Result

Thus the prototype of the project with database management system and deployment of the project has be designed successfully.

Part II Use Cases

Use Cases 1: Analyse and implement the **Employee management and Payroll Processing** application.

Develop an Employee Management System for a start up company. The company has multiple departments, each with its own team of employees. The goal is to create a centralized system that efficiently manages employee data, facilitates HR processes, and ensures smooth communication within the organization. Please outline the key features, functionalities, and considerations for the Employee Management System. Additionally, describe how the system would handle various scenarios, such as employee onboarding, leave requests, performance evaluations, and security measures to protect sensitive employee information. Consider both the HR team's perspective and the employees' experience.

Use Cases 2: Build the **Library Management system** application with project plans, schedule, estimation and design.

Construct a Library Management System (LMS) for a city library. The library serves a diverse community and houses a collection of books, journals, and multimedia resources. The goal is to create a user-friendly system that enhances the efficiency of library operations, improves user experience, and provides valuable insights for library administrators. Please outline the key features, functionalities, and considerations for the Library Management System. Additionally, describe how the system would handle various scenarios, such as user registrations, borrowing and returning items, inventory management, and overdue fines. Consider both the librarian's perspective for system administration and the user's experience in borrowing and accessing library resources.

Use Cases 3: Design, Analyse and implement the **Passport Automation** System application

Develop a Passport Automation System for a government agency. The agency processes a large volume of passport applications, renewals, and replacements. The goal is to create an efficient and secure system that enhances the passport application experience for citizens while ensuring accuracy and compliance with regulatory requirements. Please outline the key features, functionalities, and considerations for the Passport Automation System. Additionally, describe how the system would handle various scenarios, such as new passport applications, expedited processing, biometric data collection, and security measures to prevent identity theft and fraud. Consider both the applicant's perspective and the internal processes involved in passport issuance.

Use Cases 4: Identify, plan, analyse and implement the **E Voting** System application.

Sketch an E-Voting System for a national election. The government aims to modernize the voting process, increase accessibility, and enhance the overall efficiency of the election. Please outline the key features, functionalities, and considerations for the E-Voting System. Additionally, describe how the system would handle various scenarios, such as voter registration, ballot casting, result tabulation, and security measures to ensure the integrity of the election process. Consider both the voter's experience and the administrative aspects of managing the election.